

Kodolányi János Egyetem
Újmédia- és Kreatívipari Kar

SZAKDOLGOZAT

VÁRADI VIKTOR
ÜZEMMÉRNÖK-INFORMATIKUS
ALAPKÉPZÉSI SZAK

Budapest

2026.

Kodolányi János Egyetem
Újmédia- és Kreatívipari Kar
Informatika Tanszék

„NewsCast”

Hírgyűjtő, hírelemző és hírolvasó alkalmazás

Konzulens: Dr. Pitlik László

Készítette: Váradi Viktor

ÜZEMMÉRNÖK-INFORMATIKUS

ALAPKÉPZÉSI SZAK

Budapest

2026.

Absztrakt

A jelen szakdolgozat a NewsCast elnevezésű, hat mikroszolgáltatásból álló szoftverrendszert mutatja be, amelynek célja a rádiós hírgyártás teljes munkafolyamatának automatizálása. A rendszer a magyar nyelvű online hírportálok tartalmát valós időben gyűjti, természetes nyelvfeldolgozás és mesterséges intelligencia segítségével elemzi, a közösségi média trendjeit integrálja, az időjárás-előrejelzéseket feldolgozza, a kiválasztott híreket rádiós narratívává formázza és szintetizált magyar nyelvű hangkimenetet állít elő. A hírforrások objektivitásának vizsgálata a COCO matematikai modell OAM elemzésével valósul meg. A dolgozat ismerteti a rendszer követelmény-specifikációját, a Docker-alapú konténerizált architektúrát, az adatbázis- és API-tervezést, a hat backend modul megvalósítását, a biztonsági és monitorozási megoldásokat, valamint a tesztelési eredményeket. A tesztelés részeként három nagy nyelvi modell (GPT-5.4-mini, Claude Haiku 4.5, Gemini 3.1 Flash Lite) összehasonlító benchmark vizsgálata is elkészült a hírelemzés, a szövegnormalizálás és a szövegösszegzés területén. A rendszer a teljes feldolgozási láncot – az RSS letöltéstől a szintetizált hangkimenetig (TTS) – automatikusan, minimális emberi beavatkozással képes végrehajtani.

Abstract

This thesis presents NewsCast, a software system composed of six microservices designed to automate the entire radio news production workflow. The system collects content from Hungarian online news portals in real time, analyses it using natural language processing and artificial intelligence, integrates social media trend signals, processes weather forecasts, formats selected news items into a radio narrative and produces synthesised Hungarian language audio output. Source objectivity is assessed through OAM analysis based on the COCO mathematical model. The thesis describes the requirements specification, the containerised architecture, database and API design, the implementation of all six backend modules, security and monitoring solutions, and testing results. As part of the evaluation, a comparative benchmark of three large language models (GPT-5.4-mini, Claude Haiku 4.5, Gemini 3.1 Flash Lite) was conducted in the areas of news analysis, text normalisation and text summarisation. The system is capable of executing the full processing chain – from RSS retrieval to synthesised audio output (TTS) – automatically with minimal human intervention.

Köszönetnyilvánítás

Szeretném kifejezni köszönetemet konzulensemnek, Dr. Pitlik Lászlónak, aki a szakdolgozat teljes készítési folyamata során szakmai iránymutatásával, konstruktív kritikáival, professzionális hozzáállásával és a COCO modell alkalmazásában nyújtott módszertani segítségével támogatta a munkámat. A miau.my-x.hu API hozzáférés biztosítása lehetővé tette az OAM elemzés integrálását a rendszerbe.

Köszönöm a Kodolányi János Egyetem Informatika Tanszék oktatóinak, hogy a képzés során átadott elméleti és gyakorlati ismeretek megalapozták a szakdolgozatban bemutatott fejlesztés megvalósítását.

Köszönettel tartozom továbbá családomnak a szakdolgozat készítése alatt tanúsított türelemért és támogatásért.

Tartalomjegyzék

1. BEVEZETÉS.....	13
1.1. Kutatási és fejlesztési célok.....	13
1.2. Problémafelvetés és indoklás	14
1.3. Motiváció	17
1.3.1. Személyes motiváció.....	17
1.3.2. Piaci motiváció.....	18
1.4. Célcsoportok.....	19
1.5. Hasznosság	20
1.5.1. Társadalmi és szakmai hasznosság	20
1.5.2. Gazdasági hasznosság	21
1.6. A dolgozat hatóköre és korlátai.....	22
1.7. A dolgozat szerkezete.....	23
2. SZAKIRODALMI ÁTTEKINTÉS ÉS TECHNOLÓGIAI HÁTTÉR . 25	
2.1. A digitális híripiac és a rádiós munkafolyamatok	25
2.1.1. A magyar digitális hírökoszisztéma	25
2.1.2. A rádiós hírszerkesztés munkafolyamata	26
2.2. Alkalmazott technológiák	27
2.2.1. Python backend és a FastAPI keretrendszer	27
2.2.2. Természetes nyelvfeldolgozás (NLP).....	28
2.2.3. Szövegfelolvasás (Text-to-Speech)	31
2.2.4. Közösségi média platform korlátok és a Google Trends integráció.....	33
2.2.4.1. A Cambridge Analytica hatás (2018–)	33
2.2.4.2. Platform-specifikus korlátozások elemzése	34
2.2.4.3. A GDPR és az MI-modellek „adatéhségének” hatása.....	36
2.2.4.4. Scraping mint alternatíva – lehetőségek és korlátok	36
2.2.4.5. A választott megoldás indoklása – Google News/Trends RSS	37
2.2.5. RESTful architektúra és mikroszolgáltatások	38
2.2.6. Adatbázis-kezelés: SQLAlchemy és MariaDB	40
2.2.7. COCO modell és az OAM elemzés.....	40
2.2.8. Konténerizáció és monitorozás	42
2.3. Kapcsolódás a tanulmányokhoz (tantárgyi integráció)	43

2.3.1.	Szoftverarchitektúrák, Rendszertervezés és Rendszermodellezés	43
2.3.2.	Programozás, Programozási alapelvek és módszertanok	44
2.3.3.	Adatbázisok	45
2.3.4.	Adatszerkezetek és algoritmusok, Matematikai alapok	46
2.3.5.	Mesterséges intelligenciák az IT-biztonság területén	47
2.3.6.	Hálózatok és számítógép architektúrák, Operációs rendszerek	48
2.3.7.	Informatikai védelem és biztonság	49
2.3.8.	Szoftvertesztelés és Szoftverüzemeltetés	50
2.3.9.	Felhasználói interfészek és vizualizáció	50
2.3.10.	Az elektronika fizikai alapjai és Elektronikus áramkörök	51
2.3.11.	Vállalati gazdaságtan, Vezetési és vállalkozási ismeretek	51
2.3.12.	Társadalomtudományi és jogi vonatkozások	52
3.	SAJÁT FEJLESZTÉS	53
3.1.	Követelmény-specifikáció	53
3.1.1.	Funkcionális követelmények	53
3.1.1.1.	FR-01: RSS hírgyűjtés	53
3.1.1.2.	FR-02: Hírelemzés és -osztályozás	53
3.1.1.3.	FR-03: OAM elemzés	54
3.1.1.4.	FR-04: Időjárás integráció	54
3.1.1.5.	FR-05: Hírszelekció és -formázás	54
3.1.1.6.	FR-06: Közösségi trendjelek gyűjtése és értékelése	55
3.1.1.7.	FR-07: Szövegfelolvasás (TTS)	55
3.1.2.	Nem funkcionális követelmények	55
3.1.2.1.	NFR-01: Teljesítmény	56
3.1.2.2.	NFR-02: Megbízhatóság	56
3.1.2.3.	NFR-03: Biztonság	56
3.1.2.4.	NFR-04: Üzemeltethetőség	56
3.1.2.5.	NFR-05: Bővíthetőség	57
3.2.	Rendszerarchitektúra	57
3.2.1.	Architektúra áttekintés	57
3.2.2.	Az adatáramlás leírása	57
3.2.2.1.	Hírgyűjtési útvonal (RSS → DB):	58
3.2.2.2.	Elemzési útvonal (DB → Analyze → DB):	58

3.2.2.3.	Közösségi trendjelek útvonala (Google News/Trends → Social → DB):	58
3.2.2.4.	Időjárás útvonal (OMSZ → DB → TTS → Audio):	58
3.2.2.5.	Hírszerkesztői útvonal (DB → Feeder → TTS → Audio):	58
3.2.2.6.	OAM elemzési útvonal (DB → Analyze → COCO API → DB):	59
3.2.3.	A modulok felelősségi körei	59
3.3.	Adatbázis terv	59
3.3.1.	Entitás-kapcsolat diagram	59
3.3.2.	A legfontosabb táblák részletes leírása	60
3.3.2.1.	„rss” tábla (hírforrások)	60
3.3.2.2.	„news” tábla (hírek / cikkek)	61
3.3.2.3.	„weather” tábla (időjárési adatok)	62
3.3.2.4.	„feeder_news” tábla (kiválasztott hírek)	62
3.3.2.5.	„tts_history” tábla (TTS napló)	62
3.3.2.6.	„social_signals” tábla (közösségi trendek)	62
3.3.2.7.	„trending_keywords” tábla (trending kulcsszavak)	63
3.3.3.	Adatbázis nézetek (views)	64
3.4.	API végpontok és kommunikáció	64
3.4.1.	API tervezési elvek	64
3.4.2.	Modulonkénti API áttekintés	65
3.4.2.1.	newscast-rss_parser API (Port: 8080)	65
3.4.2.2.	newscast-analyze API (Port: 8080)	65
3.4.2.3.	newscast-weather API (Port: 80)	66
3.4.2.4.	newscast-social API (Port: 8080)	66
3.4.2.5.	newscast-feeder API (Port: 80)	67
3.4.2.6.	newscast-tts API (Port: 80)	67
3.4.3.	Szolgáltatásközi kommunikáció	68
3.4.4.	Hitelesítési architektúra	69
3.4.4.1.	HTTP Basic Auth:	69
3.4.4.2.	JWT Bearer Token:	69
3.4.4.3.	Webes session alapú hitelesítés (csak a feeder Web UI):	69
3.5.	Fejlesztői környezet	70
3.5.1.	Technológiai platform	70
3.5.2.	Projekt struktúra	70
3.5.3.	Futtató környezet	71

3.6.	Backend modulok megvalósítása	72
3.6.1.	newscast-rss_parser: RSS hírgyűjtő modul.....	73
3.6.1.1.	Az RSS letöltés és HTTP gyorsítótárazás	73
3.6.1.2.	Az RSS elemzés és időzóna kezelés	73
3.6.1.3.	Duplikációsűrés és adatmentés.....	74
3.6.1.4.	Párhuzamos feldolgozás.....	74
3.6.2.	newscast-analyze: Hírelemzés és OAM modul.....	74
3.6.2.1.	Az elemzési pipeline (UnifiedAnalyzer).....	74
3.6.2.2.	Duplikációsűrés (DuplicationDetector).....	76
3.6.2.3.	A COCO API kliens és az OAM elemzés	76
3.6.2.4.	Közösségi trendjelek integrációja	77
3.6.3.	newscast-weather: Időjárás feldolgozó modul	77
3.6.3.1.	Az OMSZ adatok feldolgozási pipeline-ja.....	78
3.6.4.	newscast-feeder: Hírselekcio és webes felület	79
3.6.4.1.	A hírselekcio algoritmus	79
3.6.4.2.	Narratíva flow szerkesztés és szövegmentés	79
3.6.4.3.	Webes felhasználói felület.....	80
3.6.5.	newscast-tts: Szövegmentés (Text-to-Speech) modul	85
3.6.5.1.	Az ElevenLabs TTS kliens.....	85
3.6.5.2.	Feladatsor és worker pool	85
3.6.5.3.	Tartalom hash deduplikáció.....	86
3.6.5.4.	Magyar szövegmentés	86
3.6.5.5.	Külső szolgáltatások.....	87
3.6.6.	newscast-social: Közösségi média trendgyűjtő modul.....	87
3.6.6.1.	Google News és Google Trends	88
3.6.6.2.	TrendingStore: kulcsszó illesztés	88
3.6.6.3.	Többszintű URL egyeztetés	89
3.6.6.4.	<i>Szuper képlet</i> és pontszámítás	89
3.6.6.5.	Backfill mechanizmus (race condition kezelés).....	89
3.7.	Biztonsági megoldások	90
3.7.1.	Hitelesítés és jogosultságkezelés.....	90
3.7.2.	API-kulcsok és érzékeny adatok kezelése.....	90
3.7.3.	SQL injection elleni védelem	90
3.8.	Monitorozás és naplózás	90

3.8.1.	Prometheus metrikák és Grafana dashboardok	90
3.8.2.	Strukturált naplózás és korrelációs azonosítók	91
3.9.	MI-aspektusok	91
3.10.	Tesztelés és eredmények	92
3.10.1.	Tesztelés	93
3.10.2.	Tesztelési módszertan	94
3.10.2.1.	Tesztelési szintek	94
3.10.2.2.	Tesztelési eszközök	95
3.10.2.3.	Docker health check konfiguráció	96
3.10.3.	Tesztesetek bemutatása	96
3.10.3.1.	RSS Parser tesztesetek	96
3.10.3.2.	Analyze modul tesztesetek	97
3.10.3.3.	Weather modul tesztesetek	99
3.10.3.4.	Feeder modul tesztesetek	99
3.10.3.5.	Social modul tesztesetek	100
3.10.3.6.	TTS modul tesztesetek	101
3.10.4.	Teljesítménytesztek	103
3.10.4.1.	RSS feldolgozási teljesítmény	103
3.10.4.2.	Elemzési teljesítmény	103
3.10.4.3.	Social modul teljesítmény	104
3.10.4.4.	TTS generálási teljesítmény	104
3.10.4.5.	Teljes pipeline áteresztőképesség	105
3.10.4.6.	Követelmény – megfelelés összefoglalása	105
3.10.5.	LLM-benchmark	106
3.10.5.1.	A benchmark célja és módszertana	106
3.10.5.2.	Hírelemzési benchmark eredményei	109
3.10.5.3.	Szövegnormalizálási benchmark eredményei	111
3.10.5.4.	Szövegösszegzési benchmark eredményei	114
3.10.5.5.	Összesítő értékelés	117
4.	VITA	119
4.1.	A hírrelevancia meghatározásának módszertani kérdése	119
4.2.	A közösségi trendjelek feldolgozásának kompromisszumai	119
4.3.	A szolgáltatásközi kommunikáció dilemmája	119

4.4.	A magyar nyelvű szövegnormalizálás kihívásai	120
4.5.	Az RSS feldolgozás párhuzamossági szintje	120
4.6.	Külső szolgáltatásfüggőségek kockázatai	120
4.7.	Az LLM-benchmark tanulságai	121
5.	KÖVETKEZTETÉSEK.....	122
5.1.	A kitűzött célok értékelése	122
5.2.	A rendszer erősségei.....	123
5.3.	Korlátok és ismert hiányosságok.....	124
6.	JÖVŐBELI IRÁNYOK	124
7.	ÖSSZEGZÉS.....	126
8.	MELLÉKLETEK	127
8.1.	Ábrajegyzék	127
8.2.	Táblázatjegyzék.....	127
8.3.	Rövidítések jegyzéke.....	129
8.4.	Definíciók jegyzéke.....	132
8.5.	Hivatkozások.....	133
8.6.	LLM-benchmark: Teljes API-konverzációk.....	144
8.6.0.	Tesztadatok.....	144
8.6.0.1.	Benchmark 1 – Hírelemzési tesztadatok (10 teszthír).....	144
8.6.0.2.	Benchmark 2 – Szövegnormalizálási tesztadatok (30 tesztmondattal)	145
8.6.0.3.	Benchmark 3 – Szövegösszegzési tesztadatok (5 teszthír)	146
8.6.1.	Benchmark 1 – Hírelemzés: GPT-5.4-mini API válasz.....	148
8.6.2.	Benchmark 1 – Hírelemzés: Claude Haiku 4.5 API válasz.....	153
8.6.3.	Benchmark 1 – Hírelemzés: Gemini 3.1 Flash Lite API válasz.....	158
8.6.4.	Benchmark 2 – Szövegnormalizálás: GPT-5.4-mini API válasz.....	163
8.6.5.	Benchmark 2 – Szövegnormalizálás: Claude Haiku 4.5 API válasz.....	166
8.6.6.	Benchmark 2 – Szövegnormalizálás: Gemini 3.1 Flash Lite API válasz.....	170
8.6.7.	Benchmark 3 – Szövegösszegzés: GPT-5.4-mini API válasz	171
8.6.8.	Benchmark 3 – Szövegösszegzés: Claude Haiku 4.5 API válasz	173
8.6.9.	Benchmark 3 – Szövegösszegzés: Gemini 3.1 Flash Lite API válasz	176
8.6.10.	Fejlesztési LLM-konverzáció: Claude Code – szövegnormalizálási szabály	178

8.7.	Forráskódok.....	179
8.8.	Mintakimenetek.....	179

1. Bevezetés

A digitális médiafogyasztás alapjaiban változott meg az elmúlt évtizedben. A hagyományos rádiós hírszerkesztés, amely korábban kizárólag emberi munkára támaszkodott – forrásfigyelés, hírselekción, szövegírás, felolvasás –, napjainkban egyre növekvő kihívásokkal szembesül. A hírforrások száma megsokszorozódott, az információáramlás sebessége a valós idejű közlés irányába tolódott, miközben a rádiós szerkesztőségek létszáma és erőforrásai jellemzően csökkentek. E folyamatok eredményeként a manuális hírfeldolgozás egyre kevésbé képes lépést tartani a digitális hírfolyam keletkezésének tempójával.

A szakdolgozatom egy olyan integrált szoftverrendszert mutat be, amely a NewsCast nevet viseli. A rendszer célja a rádiós hírgyártás teljes munkafolyamatának automatizálása: az RSS (Really Simple Syndication) alapú hírgyűjtéstől a természetes nyelvfeldolgozáson (Natural Language Processing, NLP) alapuló elemzésen, a közösségi média trendjelek feldolgozásán és a hírselekción át egészen a szintetizált hangkimenet előállításáig. A NewsCast hat, lazán csatolt mikroszolgáltatásból épül fel, amelyek együttesen képesek emberi beavatkozás nélkül, beállított ütemezéssel sugárzásra kész hírblokkok előállítására.

1.1. Kutatási és fejlesztési célok

A NewsCast rendszer fejlesztésének elsődleges célja egy olyan automatizált hírfeldolgozó platform megvalósítása volt, amely képes a magyar nyelvű online hírportálok tartalmának valós idejű gyűjtésére, intelligens elemzésére és rádiós formátumú szöveges, illetve hanganyag-kimenet előállítására. A konkrét fejlesztési célkitűzések az alábbiak szerint foglalhatók össze:

Automatizált hírgyűjtés: A rendszernek képesnek kell lennie magyar nyelvű hírforrások párhuzamos, ütemezett feldolgozására, HTTP (HyperText Transfer Protocol) gyorsítótárazás (ETag (Entity Tag) / Last-Modified) alkalmazásával a sávszélesség-takarékosság érdekében. A megvalósított rendszer 62 előre konfigurált forrást kezel, amelyek között megtalálhatók az országos hírportálok (pl. Index, Telex, HVG, 444.hu), a gazdasági szaklapok (pl. Portfolio, Forbes), a közszolgálati források (pl. MTI, police.hu, Katasztrófavédelem) és a regionális médiumok (pl. Borsod24, Szol24) egyaránt.

Intelligens tartalomelemzés: A gyűjtött hírek természetes nyelvfeldolgozás (Natural Language Processing, NLP) alapú feldolgozása magyar nyelven, beleértve a névelem felismerést (Named Entity Recognition, NER), a szentiment- és olvashatósági elemzést, a tartalombiztonsági osztályozást, valamint a duplikációsűrést. A rendszernek továbbá

alkalmasnak kell lennie a hírek rádiós relevanciájának automatikus meghatározására egy többkomponensű pontozási rendszer segítségével.

Közösségi média trendjelek feldolgozása: A rendszernek képesnek kell lennie a Google News és a Google Trends platformokról származó trendjelek valós idejű gyűjtésére, a meglévő hírekkel való automatikus párosítására, valamint a közösségi érdeklődés mértékének számszerűsítésére egy ún. Super Formula segítségével. A trendjelek integrálása révén a rendszer képes a virális tartalmak priorizálására az MI (mesterséges intelligencia) alapú validáció és a rádiós hírszelekció során.

Objektív hírelemzés: A hírforrások objektivitásának vizsgálata a COCO (Component-based Object Comparison for Objectivity) matematikai modell alkalmazásával, amely a miau.myx.hu távoli API-n (Application Programming Interface) keresztül érhető el. Az OAM (Object-Attribute Matrix) elemzés révén a rendszer képes a hírforrások közötti torzítások és összefüggések feltárására.

Időjárás integráció: Az Országos Meteorológiai Szolgálat (OMSZ) szöveges előrejelzéseinek automatikus letöltése, feldolgozása és a hírfolyamba történő integrálása. A rendszernek kezelnie kell az OMSZ által alkalmazott ZIP-tömörített, Windows-1250 kódolású szöveges formátumot.

Szövegfelolvasás (Text-to-Speech, TTS): A kiválasztott és formázott hírblokk természetes hangzású, szintetizált beszéddé alakítása az ElevenLabs API szolgáltatás felhasználásával, a magyar nyelv sajátosságainak – rövidítések, számok, dátumok, mértékegységek – megfelelő kezelésével.

Minimális emberi beavatkozás: A teljes feldolgozási lánc (gyűjtés → elemzés → trendjelek → szelekció → szövegformázás → hangszintézis) automatikus működése beállított ütemezéssel, ugyanakkor a rendszernek lehetőséget kell biztosítania manuális beavatkozásra is egy webes felületen keresztül.

1.2. Problémafelvetés és indoklás

A hagyományos rádiós hírszerkesztés munkafolyamata számos, jól azonosítható problémával küzd, amelyek a NewsCast rendszer fejlesztésének közvetlen motivációját adták.

A manuális hírgyűjtés korlátai

Egy átlagos magyar rádiós szerkesztőség napi munkája során a hírszerkesztőnek manuálisan kell figyelemmel kísérnie több tucat hírportált, hírügynökségi feedet és közösségimédia-csatornát. Ez a feladat rendkívül időigényes: egy tapasztalt szerkesztő számára is legalább 2-3 óra szükséges ahhoz, hogy az összes releváns forrást áttekintse és a legfontosabb híreket kiválassza. A folyamat ráadásul szubjektív: a hírszelekció eredménye nagymértékben függ a szerkesztő személyes preferenciáitól, elfáradásától és az adott napi munkaterhétől.

A probléma súlyosságát jól illusztrálja, hogy egy hírblokk összeállítása – a forrásfigyeléstől a felolvasásra kész szöveg elkészítéséig – akár 4-6 órányi élőmunkát is igényelhet. Ez a ráfordítás a kisebb szerkesztőségek számára gazdaságilag egyre nehezebben fenntartható.

Az információs túlterheltség

A magyar digitális hírpiac az elmúlt években jelentős bővülésen ment keresztül. A NewsCast rendszerben előre konfigurált 62 forrás csupán a legjelentősebb hírportálokat fedi le, a teljes magyar online hírökoszisztéma ennél lényegesen kiterjedtebb. E források mindegyike naponta több tucat, összességében tehát napi több ezer hírt publikál. Az emberi feldolgozókapacitás számára ez az információmennyiség kezelhetetlenné vált, miközben a gyors reagálás – különösen a rendkívüli híreseményeknél – üzleti kritikumot jelent.

Duplikáció és minőségi szűrés hiánya

A magyar hírpiacon rendkívül gyakori, hogy ugyanazt a hírt több forrás is átveszi, gyakran minimális szerkesztéssel vagy átfogalmazással. A manuális szerkesztési folyamatban a duplikációk felismerése és kiszűrése további terhet ró a szerkesztőre. Ezzel párhuzamosan a hirdetési tartalmak, szponzorált cikkek és promóciós szövegrészek egyre kifinomultabb módon épülnek be a hírtartalmakba, megnehezítve a valódi hírértékű tartalom elkülönítését.

A felolvasás, mint szűk keresztmetszet

A hagyományos rádiós hírfelolvasás professzionális bemondót igényel, akinek elérhetősége korlátozott és költsége jelentős. A modern szövegfelolvasási (Text-to-Speech) technológiák – különösen az ElevenLabs által kínált neurális hálózat alapú megoldások – már képesek természetes hangzású, emberi beszédhez közelítő minőségű hangkimenet előállítására, ami új lehetőséget nyit az automatizáció irányában.

A közösségi média adathozzáférés problémája

A hírrelevancia pontos meghatározásához a szöveges tartalomelemzés mellett a közösségi média platformok trendjeire is szükség lenne – a Twitter/X említések, a Facebook-megosztások és a Reddit-szálak népszerűsége közvetlen visszajelzést adna egy hír társadalmi hatásáról. Az elmúlt években azonban a közösségi média platformok API-hozzáférése drasztikusan korlátozódott. A 2018-as Cambridge Analytica botrány nyomán a Facebook (Meta) radikálisan szűkítette a harmadik féltől származó alkalmazások adathozzáférését és ez a bezárkózási tendencia dominóhatásként végigfutott a többi platformon is. A GDPR (2018) hatálybalépése, majd az MI-modellek „adatéhsége” miatti monetizációs törekvések tovább szűkítették a nyilvános API-k elérhetőségét: a Twitter/X a Free tier-t write-only módra korlátozta, a Reddit fizetős API-ra váltott, az Instagram Basic Display API megszűnt, a TikTok Research API pedig kizárólag jóváhagyott kutatók számára érhető el. E korlátozások a NewsCast rendszer tervezését is alapvetően meghatározták és a Google News/Trends RSS mint nyilvános, ingyenes és nem személyes adatokat szolgáltató forrás felé irányították a fejlesztést (vö. 2.2.4 fejezet).

A probléma technikai dimenziói

A fent felsorolt üzleti problémák mellett számos technikai kihívás is azonosítható, amelyek megoldása mérnöki szempontból érdekes feladatot jelent:

- **Kódolási problémák:** A magyar hírforrások és különösen az OMSZ időjárési adatai heterogén karakterkódolásokat alkalmaznak (UTF-8, Windows-1250, ISO-8859-2), amelyek helytelen kezelése a magyar ékezetes karakterek elvesztéséhez vezet.
- **Valós idejű feldolgozás:** A hírgyűjtés és hírelemzés teljes ciklusának elég gyorsnak kell lennie ahhoz, hogy az óránkénti ütemezés keretében maradjon.
- **Magyar nyelvű NLP:** A magyar nyelv ragozása, toldalékolása és gazdag morfológiája speciális NLP-eszközöket igényel (HuSpacy), amelyek alkalmazása nagyban eltér az angol nyelvű megoldásoktól.
- **Számok és rövidítések felolvasása:** A TTS-rendszernek a magyar nyelv szabályai szerint kell kezelnie a számokat (pl. 2024 → „kétezer-huszonegy”), a dátumokat (pl. január 15. → „január tizenötödike”), a mértékegységeket és a rövidítéseket.

1.3. Motiváció

A NewsCast rendszer fejlesztését személyes és piaci motiváció egyaránt ösztönözte. A személyes indítást a rádiózás iránti érdeklődés és az üzemtechnikus-informatikus képzés során szerzett ismeretek gyakorlati alkalmazásának igénye adta (1.3.1), míg a piaci motivációt a rádiós hírszolgáltatási piac átalakulása és az automatizáció iránti növekvő igény indokolja (1.3.2).

1.3.1. Személyes motiváció

Gyermekkorom óta foglalkoztat hobby szinten a rádiózás technikai oldala. Érdekel a rádiós broadcast stúdiótechnika: a keverők, a mikrofonok, az adásleboncolás, a hangprocesszállás és a mastering – de a legjobban az adásprocesszállás (multiband AGC/compressor/limiter) világa ragadott meg. Élvezettel állítgatom és kísérletezek a saját hangprocesszorommal (Deva Broadcast DB6400), keresve az ideális hangzást. Egyik hobby célom beszerezni a nagyobb gyártók – Orban Optimod vagy Telos Omnia – adásprocesszorait és ezek segítségével előállítani, tökéletesíteni nagyobb rádióállomások hangképét. A NewsCast fejlesztése tehát nem csupán egy szoftverprojekt számomra, hanem szervesen kapcsolódik ahhoz a világhoz, amelyben otthon érzem magam: a rádiós tartalom-előállítás automatizálása ugyanannak az éremnek a másik oldala, mint a sugárzott hang minőségének csiszolása.

Az üzemtechnikus-informatikus képzés során szerzett elméleti tudás – különösen a szoftvertechnológia, az adatbázis-kezelés, a mesterséges intelligencia és a programozás területén – gyakorlati alkalmazásának lehetősége önmagában is motiváló volt. A projekt lehetővé tette, hogy a tanulmányok során megismert technológiákat – Python programozás, API-tervezés, relációs adatbázisok, IT biztonság – egy valós, komplex feladaton keresztül mélyítsem el.

Különös érdeklődéssel fordultam a természetes nyelvfeldolgozás (NLP) és a szövegfelolvasás (TTS) területe felé. A magyar nyelv sajátosságainak gépi feldolgozása – a gazdag morfológia, a ragozás, a toldalékolás, a szabad szórend – olyan mérnöki kihívást jelent, amely túlmutat az angol nyelvű megoldások egyszerű adaptálásán. A HuSpacy nyelvi modell integrálása és a magyar nyelvű szövegnormalizáló algoritmusok fejlesztése (számok szövegesítése, rövidítések feloldása, dátumformátumok kezelése) lehetőséget adott arra, hogy a nyelvtechnológia gyakorlati alkalmazásaival közelebbről megismerkedjek.

A mikroszolgáltatás-architektúra tervezése és megvalósítása szintén fontos tanulási tapasztalatot nyújtott. A hat önálló szolgáltatás (RSS Parser, Analyze, Social, Weather, Feeder, TTS) közötti kommunikáció megtervezése, az adatbázis-sémák kialakítása, a hitelesítési mechanizmusok implementálása és a monitorozási infrastruktúra felépítése olyan kompetenciákat fejlesztett, amelyek a szoftverfejlesztői gyakorlatban elengedhetetlenek.

1.3.2. Piaci motiváció

A rádiós hírszolgáltatás piaca Magyarországon és nemzetközi szinten is jelentős átalakuláson megy keresztül. A hagyományos rádióadók mellett megjelentek az internetes rádiócsatornák, a podcastok és a streaming-platformok híradásai, amelyek egyre szélesebb közönséget érnek el. Ugyanakkor ezek az új médiumok is hasonló kihívásokkal szembesülnek: az emberi erőforrás költségei magasak, a hírfolyam sebessége folyamatosan nő és a hallgatói elvárások a frissesség és a relevancia terén egyre magasabbak.

A NewsCast rendszer által kínált automatizációs megoldás több piaci szegmensben is releváns:

- **Kisebb (tipikusan kisközösségi) rádióadók és internetes rádiók:** Amelyek nem rendelkeznek dedikált hírszerkesztőséggel, de igénylik a rendszeres hírszolgáltatást.
- **Podcast-készítők és tartalomgyártók:** Akik automatizált hírösszefoglalókat szeretnének beépíteni tartalmaikba.
- **Médiafigyelő szolgáltatások:** Amelyek számára a nagy mennyiségű hírforrás automatikus feldolgozása és elemzése alapvető üzleti igény.
- **Akadálymentesítés:** Látássérült felhasználók számára a szintetizált hírfelolvasás a tájékozódás fontos eszköze lehet.

A piaci motivációt erősíti az a tény is, hogy a mesterséges intelligencia alapú tartalomgenerálás és -feldolgozás területén jelenleg intenzív fejlődés tapasztalható. Az olyan szolgáltatások, mint a Google Gemini (amelyet a NewsCast az MI-alapú hírvalidációhoz használ) és az ElevenLabs (amely a szövegfelolvasást biztosítja), az elmúlt években jelentős minőségjavuláson mentek keresztül, lehetővé téve olyan alkalmazások megvalósítását, amelyek korábban nem voltak elérhetők.

1.4. Célcsoportok

A NewsCast rendszer tervezése során több, egymástól eltérő igényekkel rendelkező célcsoportot azonosítottam.

Elsődleges célcsoport: **Rádiós szerkesztők és hírolvasók**

A rendszer elsődleges felhasználói a rádiós szerkesztőségek munkatársai, akik számára a NewsCast jelentős munkaidő-megtakarítást kínál. A webes felületen (newscast-feeder UI) keresztül a szerkesztők megtekinthetik az automatikusan kiválasztott híreket, manuálisan módosíthatják a sorrendet, szerkeszthetik a szövegeket és egyetlen gombnyomással indíthatják a TTS-generálást.

A rendszer három szerepkört különböztet meg:

- **Admin:** Teljes hozzáférés a rendszer minden funkciójához, beleértve a felhasználókezelést és a rendszerbeállítások módosítását.
- **Editor (Szerkesztő):** Hírek kiválasztása, sorrendezése, szövegszerkesztés és TTS-generálás indítása.
- **Viewer (Megtekintő):** A kiválasztott hírblokkok és a generált hanganyagok megtekintése, letöltése.

Másodlagos célcsoport: **Médiaanalitikusok**

Az OAM (Object-Attribute Matrix) elemzési modul révén a rendszer a médiaanalitikusok számára is értékes eszközt kínál. A COCO-alapú objektivitáselemzés módot ad a hírforrások közötti torzítások, az ún. „king-maker” (véleményformáló) kapcsolatok és az antagonisztikus viszonyok feltárását. Ez az elemzési képesség a médiakutatás és a médiafelügyelet területén egyaránt hasznosítható.

Harmadlagos célcsoport: **Rendszerüzemeltetők (DevOps)**

A rendszer üzemeltetői számára a Prometheus-alapú metrikagyűjtés, a Grafana vizualizációs dashboardok, a strukturált JSON (JavaScript Object Notation) naplózás és a Docker-konténerizáció gondoskodik a szükséges átláthatóságról és kezelhetőségről. Minden modul rendelkezik health check végpontokkal, valamint a konténer-orkesztrációs platformokkal (Kubernetes) való integrációt a liveness és readiness végpontok is támogatják.

1.5. Hasznosság

A NewsCast rendszer hasznossága több dimenzió mentén értékelhető. A társadalmi és szakmai hasznosság (1.5.1) a rádiós hírszolgáltatás minőségének és hozzáférhetőségének javulásában nyilvánul meg, míg a gazdasági hasznosság (1.5.2) a költségcsökkentési és hatékonysági előnyöket foglalja magában.

1.5.1. Társadalmi és szakmai hasznosság

A NewsCast rendszer társadalmi hasznossága több területen is érzékelhető.

A hírszolgáltatás demokratizálása: Az automatizált hírfeldolgozás lehetővé teszi, hogy kisebb szerkesztőségek is professzionális hírszolgáltatást nyújtsanak, csökkentve a belépési korlátokat a rádiós hírcsatornákra. Míg egy hagyományos hírszerkesztőség üzemeltetéséhez több fős csapatra van szükség, a NewsCast egyetlen operátor felügyelete mellett is képes folyamatos hírszolgáltatást biztosítani.

Objektivitás növelése: Az MI-alapú hírszelekció és a COCO-modellen alapuló objektivitás-elemzés csökkenti a szubjektív szerkesztői torzítások lehetőségét. A rendszer rádiós relevancia-pontozása (radio_relevance_score) több, jól definiált szempont – hírérték, aktualitás, forrás presztízse, olvashatóság, tartalombiztonság – alapján értékeli a híreket.

Magyar nyelvtechnológia fejlesztése: A projekt hozzájárul a magyar nyelvű NLP-alkalmazások ökoszisztémájához. A HuSpacy nyelvi modellre épülő elemzési pipeline, a magyar számok szövegesítésére kifejlesztett algoritmus (a „text_normalizer.py” modulban implementált „number_to_words” függvény, amely a 0-tól 999 999 999 999-ig terjedő tartományt fedi le), valamint a magyar szövegnormalizáló rendszer három szótárral (22 rövidítés az „ABBREVIATIONS”, 4 kontextusfüggő rövidítés a „CONTEXT_SENSITIVE_ABBR” és 24 speciális karakter csere-szabály a „SPECIAL_CHARS” szótárban, összesen 50 szabály) olyan fejlesztések, amelyek más magyar nyelvű alkalmazásokban is felhasználhatók.

Akadémiai hozzájárulás: Az OAM elemzési keretrendszer, COCO-modell integrációja a hírforrás-objektivitás vizsgálatába újszerű megközelítést jelent, amely a médiakutatás számára is hasznosítható módszertant kínál.

1.5.2. Gazdasági hasznosság

A NewsCast rendszer gazdasági hasznosságát az alábbiakban foglalom össze.

Munkaidő megtakarítás: A rendszer a hírblokk összeállítás teljes ciklusát – amely manuálisan 4-6 órát vesz igénybe – néhány percre csökkenti. Az automatikus ütemezés (APScheduler, óránkénti futtatás) mellett a manuális beavatkozás lehetősége is megmarad, de a szerkesztőnek csupán az automatikusan kiválasztott és rangsorolt híreket kell áttekintenie és jóváhagynia.

Sávszélesség takarékoság: Az RSS Parser modul HTTP gyorsítótárazási mechanizmusa (ETag és Last-Modified fejlécek kezelése) a hírforrások ismételt lekérdezésekor körülbelül 70%-os sávszélesség-megtakarítást eredményez, mivel a nem módosult tartalmak esetén a szerver HTTP 304 (Not Modified) státuszkóddal válaszol és a teljes tartalom nem kerül újbóli letöltésre.

TTS költségoptimalizálás: A tartalom-hash alapú deduplikáció (SHA-256) a TTS-modulban megakadályozza, hogy azonos szövegre többszöri API-hívás történjen az ElevenLabs szolgáltatás felé. Mivel az ElevenLabs árazása karakter-alapú, ez a mechanizmus közvetlen költségmegtakarítást eredményez.

MI API költségcsökkentés: A newscast-analyze modul a Google Gemini API-hívásokat 15 perces kötegelt feldolgozással (batch processing) optimalizálja, kötegelt 50 elemet dolgozva fel egyszerre. A rádiós előszűrő (radio pre-filter) mechanizmus továbbá kiszűri azokat a híreket, amelyek biztosan nem relevánsak, így azok nem kerülnek az MI-validáció számára, csökkentve az API-hívások számát és ezzel a felmerülő költségek mértékét.

Megtérülés (ROI) becslés: A fizetesek.hu felmérése szerint egy szerkesztő bruttó havi bére Magyarországon 416.000 és 812.000 Ft között mozog, a tapasztalattól és a szerkesztőség méretétől függően (vö. 8.5 Hivatkozások). Az NMHH médiapiaci adatai alapján az újságíró, illetve rádióműsor-szerkesztő átlagkeresete bruttó 688.000 Ft havonta. Ha egy ilyen szerkesztő napi munkaidejéből 4-6 órát a hírblokk-összeállítás manuális folyamata foglal el és a NewsCast ezt a feladatot néhány percre csökkenti, a megtakarítás a teljes szerkesztői munkaidő 50-75%-át teszi ki. Egy bruttó 640.000 Ft-os havi bérrel kalkulálva ez hozzávetőlegesen 320.000-480.000 Ft/hó értékű élőmunka-megtakarítást jelent. A rendszer üzemeltetési költsége – a szerverhoszting (RackForest VPS (Virtual Private Server), vö. 3.5.3 fejezet), az ElevenLabs API-díj és a Google Gemini felhasználás – ennél lényegesen alacsonyabb, így a beruházás már rövid távon megtérül.

A gazdasági hasznosság az alábbi célcsoportonkénti bontásban is becsülhető:

Kisközösségi rádióadók (elsődleges célcsoport): Egy 4-6 órás napi hírszerkesztési munkafolyamat havi költsége – a fizetések.hu (vö. 8.5 Hivatkozások) adatai alapján bruttó 416.000 – 812.000 Ft/hó szerkesztői bér. A NewsCast alkalmazása mellett ez a költség a rendszer üzemeltetési díjára (szerver: ~6.000 Ft/hó, ElevenLabs API: ~8.000 Ft/hó, Gemini API: ~3.000 Ft/hó) csökken, amelyet a szerkesztő felügyeleti munkaideje (~30 perc/nap) egészít ki.

1.6. A dolgozat hatóköre és korlátai

A szakdolgozat a NewsCast rendszer tervezésére, fejlesztésére és tesztelésére koncentrálna, vagyis a szoftvermérnöki megvalósítás teljes vertikumát lefedi. Ugyanakkor számos, a produkciós (vö. „LiveOps”) üzemeltetéssel és a rendszer életciklusával összefüggő területre a dolgozat terjedelmi okokból nem tér ki. Az alábbiakban felsorolom azokat a témaköröket, amelyeket a jelen munka tudatosan nem tárgyal:

- **Rendszerkarbantartás és szoftverfrissítés:** A dolgozat nem foglalkozik az operációs rendszer (Debian GNU/Linux), a Docker, a MariaDB, illetve az alkalmazás Python-függőségeinek rendszeres frissítésével, a biztonsági javítások (security patch) alkalmazásának ütemezésével és a verziókompatibilitási kérdésekkel (vö. 3.5.3 fejezet).
- **Adatarchiválás és adatmegőrzési politika:** A rendszer jelenlegi állapotában nem definiál automatikus adatarchiválási vagy adattörlési stratégiát a „news”, „analysis” és „tts_history” táblák folyamatosan növekvő adatmennyiségének kezelésére. Az archiválási és retenciós politika kialakítása az üzemeltetési dokumentáció részét képezné (vö. 3.3 fejezet).
- **Katasztrófa-helyreállítás (Disaster Recovery):** A rendszer adatbázis-mentési stratégiája, a mentések tesztelése, a helyreállítási idő (RTO) és a helyreállítási pont (RPO) meghatározása nem képezi a dolgozat tárgyát.
- **Skálázási stratégia és terheléselosztás:** Bár a mikroszolgáltatás-architektúra elvi alapot teremt a horizontális skálázáshoz (vö. 2.2.4 fejezet), a konkrét terheléselosztási (load balancing) és automatikus skálázási (auto-scaling) megoldások tervezése nem része a dolgozatnak.

- **CI/CD pipeline és automatizált telepítés:** A dolgozat a fejlesztési munkafolyamatot ismerteti (vö. 3.5 fejezet), de a folyamatos integrációs és telepítési (CI/CD) pipeline – például GitHub Actions vagy GitLab CI alapú automatizált build, teszt és deploy – kialakítása a jövőbeli fejlesztések között szerepel (vö. 6. fejezet).
- **Részletes költségkalkuláció:** Az 1.5.2. alfejezetben szereplő ROI-becslés tájékoztató jellegű, a pontos üzemeltetési költségek (szerverbérlet, ElevenLabs és Google Gemini API-díjak, domain és SSL tanúsítvány költségei) részletes kimutatása nem része a dolgozatnak.
- **Jogi és adatvédelmi megfelelés:** Az RSS-alapú tartalom aggregáció szerzői jogi vonatkozásai, a GDPR-megfelelés (különösen a személyes adatok – felhasználónevek, jelszó-hash-ek – kezelése kapcsán) és a médiajogi kérdések részletes elemzése meghaladja a dolgozat műszaki fókuszát (vö. 2.3.12 fejezet).
- **Felhasználói dokumentáció és képzés:** A dolgozat a fejlesztői és az architekturális dokumentációra összpontosít, a végfelhasználók (rádiós szerkesztők) számára készítendő kezelési útmutató és betanítási terv nem képezi a dolgozat részét.
- **Hálózatbiztonság és penetrációs tesztelés:** A dolgozat az alkalmazásszintű biztonsági megoldásokat ismerteti (vö. 3.7 fejezet), de a szerver szintű tűzfalszabályok, az SSL/TLS konfiguráció részletei és a külső penetrációs tesztelés eredményei nem kerülnek bemutatásra.

A fenti témakörök tudatos kihagyása lehetővé teszi, hogy a dolgozat a NewsCast rendszer szoftvermérnöki megvalósításának részletes, kódszintű bemutatására koncentráljon, amely az üzemmérnök-informatikus BProf szakdolgozat elsődleges követelménye.

1.7. A dolgozat szerkezete

A szakdolgozat alapvető felépítése a Kodolányi János Egyetem hivatalos szakdolgozati mintájában

(https://www.kodolanyi.hu/konyvtar/images/tartalom/File/Honlapra/Feltoltes/szakdoli_minta.pdf, Letöltve: 2026. március) rögzített kereteket követi. Stilisztikai szempontból a MY-X FREE online, önjavító teszt-rendszere szolgált támpontként: (<https://miau.my-x.hu/myx-free/index.php3?x=test1>, Letöltve: 2026. március). A konzulensi elvárásrendszer (vö. Pitlik L. ifj. et al., 2026; lásd 8.5 Hivatkozások) a dolgozat tartalmi mélységének és a 16 hivatkozástípus

lefedettségének kialakítását alapozta meg. A formai követelmények részletes előírásait a Kodolányi János Egyetem egységes szakdolgozati szabályzata (vö. KJE, 2021; lásd 8.5 Hivatkozások) tartalmazza.

A szöveg kialakításakor törekedtem a tudományos írásmód (academic writing) sztenderdjeinek betartására (Forrás: https://unipub.lib.uni-corvinus.hu/4649/1/JelKep_2018_4_Deli_Eszter_Retvari_Marton.pdf, DOI: 10.20520 / JEL - KEP . 2018.04. 63, Jel-kép KOMMUNIKÁCIÓ, KÖZVÉLEMÉNY, MÉDIA 2018/4. szám, Letöltve: 2026. április).

A jobb áttekinthetőség és az egyértelműség érdekében a szakdolgozatomban az alábbi egyedi formai megoldásokat alkalmaztam:

- **Kiemelések:** Az értelmezés megkönnyítése céljából a központi fogalmakat és kulcsgondolatokat félkövér formázással jelöltem.
- **Szaknyelvi kifejezések:** Az idegen nyelvű terminusok eredeti (angol) formájukban maradtak azokban az esetekben, ahol nem rendelkeznek elterjedt magyar megfelelővel vagy az iparági, szakmai zsargon kifejezetten így használja őket.
- **Idézőjelek használata:** Az idézőjelek alkalmazása szó szerinti hivatkozások esetén a szöveg dőlt betűvel és idézőjelbe téve szerepel.
- **Képek és kódrészletek:** A vizuális elemek formázása a tartalmuktól függ. A kódrészleteket bemutató képernyőképek balra zárva, kerettel ellátva kerültek a dolgozatba. A nem kódot ábrázoló (normál) képernyőképek keret nélkül, középre igazítva, sorszámozott képaláírással jelennek meg és az ábrajegyzékben is visszakereshetők.

A jelen szakdolgozat az alábbi struktúra szerint épül fel:

A **2. fejezet** a szakirodalmi áttekintést és a technológiai háttérrel mutatja be, beleértve a Python ökoszisztémát, az alkalmazott NLP- és TTS-technológiákat, a REST (Representational State Transfer) architektúrát, valamint a tanulmányokhoz való kapcsolódást.

A **3. fejezet** a saját fejlesztés teljes körét mutatja be: a követelmény-specifikációtól a rendszerarchitektúrán, az adatbázis-terven és az API-végpontok tervezésén át a hat backend modul részletes implementációjáig, a biztonsági megoldásokig, a monitorozásig, az MI-aspektusokig és a tesztelési eredményekig.

A **4. fejezet** a Vita: a fejlesztés során felmerült tervezési dilemmák, kompromisszumok és nyitott kérdések kritikai vizsgálata.

Az **5. fejezet** a következtetéseket tartalmazza: a kitűzött célok értékelését, a rendszer erősségeit és a korlátokat.

A **6. fejezet** a jövőbeli fejlesztési irányokat mutatja be.

A **7. fejezet** a záró összefoglalást mutatja be.

A **8. fejezet** a mellékleteket tartalmazza: az ábrajegyzéket (8.1), a táblázatjegyzéket (8.2), a rövidítések (8.3) és definíciók jegyzékét (8.4), a hivatkozásokat (8.5), a nagy nyelvi modell (Large Language Model, LLM) benchmark konverzációkat (8.6) és a forráskódokat (8.7).

2. Szakirodalmi áttekintés és technológiai háttér

A jelen fejezet célja, hogy bemutassa a NewsCast rendszer fejlesztése során alkalmazott technológiák elméleti háttérét és a választásaik mögötti indokokat. Nem általános technológiai áttekintést kívánok adni, hanem a projekt specifikus igényeihez szorosan kapcsolódó technológiai döntéseket és azok szakirodalmi megalapozottságát mutatom be. A fejezet külső forrásokra támaszkodik, amelyek elérhetőségei a 8.5 Hivatkozások fejezetben találhatóak.

2.1. A digitális hírpiac és a rádiós munkafolyamatok

A jelen alfejezet a NewsCast rendszer kontextusát adó digitális hírpiaci környezetet és a rádiós hírszerkesztés hagyományos munkafolyamatát mutatja be. A 2.1.1 pont a magyar online hírökoszisztéma sajátosságait és az RSS szabvány szerepét ismerteti, míg a 2.1.2 pont a hagyományos rádiós hírszerkesztési folyamat lépéseit vázolja fel, megjelölve azokat a fázisokat, amelyek automatizálására a NewsCast rendszer alkalmassá válik.

2.1.1. A magyar digitális hírökoszisztéma

A Nemzeti Média- és Hírközlési Hatóság (NMHH) rendszeres közönségmérései és médiapiaci jelentései alapján a magyar online hírpiac struktúrája összetett és sokszereplős: *„Magyarországon 2023 IV. negyedében 2951 online tartalomszolgáltatás működött a Whitereport-adatbázis szerint; ennek mintegy kétharmada tekinthető online sajtóterméknek, azaz internetes újságnak vagy hírportálnak. Az elmúlt három évben az online sajtótermékek száma hat százalékkal, 2025 darabra csökkent. Az országos tartalmi merítésű online*

sajtótermékek 68 százalékot, a megyeiek 5 százalékot, a helyiek pedig 27 százalékot tettek ki. „Regionalitás” alatt az online médiumok esetében természetesen nem a területi elérhetőséget, hanem a tartalommal megcélzott elsődleges olvasói kört értjük. Tematika alapján a hír/közélet aránya közel 40 százalékos, a fennmaradó oldalakon mintegy ötven különböző tematika osztozott, ebben az elmúlt három évben nem történt jelentős változás.” (vö. NMHH Médiapiaci Jelentés, 2024; lásd 8.5 Hivatkozások). A NewsCast rendszerben előre konfigurált 62 RSS-forrás (lásd „newscast-rss_parser/init.sql”) több dimenziót fed le: országos hírportálok (pl. Index, Telex, HVG, Origo), gazdasági szaklapok (pl. Portfolio.hu, Forbes, Piac & Profit), közszolgálati források (pl. Híradó.hu, Police.hu, KSH), technológiai portálok (pl. HWSW, PROHARDVER!, PC Guru), regionális médiumok (pl. Borsod24, Szol24, KaposPont) és tematikus oldalak (pl. Totalcar, Femina, Velvet). Az egyes forrásokhoz presztízsérték (prestige score) van rendelve a 0,0-1,0 skálán, amely a forrás megbízhatóságát és hírértékét tükrözi. A legmagasabb presztízsértékkel (0,85) a Telex rendelkezik, míg az alacsonyabb értékek (0,5) a szűkebb közönséget célzó vagy kevésbé általános hírértékű portálokhoz tartoznak.

Az RSS (Really Simple Syndication) szabvány – amelyet eredetileg 1999-ben fejlesztettek ki a Netscape keretében és amelynek 2.0-ás verziója 2002-ben jelent meg Dave Winer közreműködésével (vö. RSS 2.0 Specification, Harvard Law; lásd 8.5 Hivatkozások) – továbbra is a legszélesebb körben alkalmazott tartalomdisztribúciós formátum a magyar hírportálok körében. Az RSS egy XML-alapú formátum, amelynek révén a webes tartalmak gépi olvashatóságú szindikációja megvalósítható (vö. RSS – Wikipedia; lásd 8.5 Hivatkozások). Bár egyes platformok API-alapú hozzáférést is kínálnak, az RSS univerzalitása – szinte minden hírportál kínál RSS feedet – és egyszerűsége miatt ideális választás a sokforrású hírgyűjtéshez. Az RSS mellett más géppel olvasható nyilvános adatforrások is léteznek a magyar piacon: az Országos Meteorológiai Szolgálat (HungaroMet) szöveges előrejelzéseit a newscast-weather modul dolgozza fel (vö. HungaroMet, 2019; lásd 8.5 Hivatkozások).

2.1.2. A rádiós hírszerkesztés munkafolyamata

A hagyományos rádiós hírszerkesztés egy jól definiált, de erőforrásigényes munkafolyamat. A szerkesztő feladata a források figyelése, a releváns hírek kiválasztása, a szöveg rádiós formátumra való átdolgozása (rövidítés, egyszerűsítés, felolvashatóvá tétel), végül a bemondás vagy a bemondónak történő átadás. Ez a munkafolyamat nemzetközi viszonylatban is egységes alapelveken nyugszik. Ahogyan Boyd, Stewart és Alexander megfogalmazzák: „*Radio news writing demands a conversational tone, short sentences and the active voice. The listener*

cannot re-read a poorly understood sentence – clarity is therefore not merely desirable but essential. The writer must tell the story as if speaking to one person, using everyday language and avoiding jargon.” (vö. Boyd, A. – Stewart, P. – Alexander, R.: „Broadcast Journalism: Techniques of Radio and Television News”, Routledge, 7th Edition, 2012; lásd 8.5 Hivatkozások). Ez az alapelv – az érthetőség, a tömörség és a beszédszerűség követelménye – a NewsCast rádiós szövegformázási algoritmusainak tervezésekor is irányadó volt. Ezen munkafolyamat automatizálásához a NewsCast rendszer a következő lépéseket valósítja meg szoftveresen:

1. **Forrásfigyelés** → newscast-rss_parser: 62 előre beállított RSS forrás párhuzamos feldolgozása
2. **Tartalomelkérés és -tisztítás** → newscast-analyze: Reklám és zavarszűrés, NLP-elemzés
3. **Közösségi trendjelek** → newscast-social: Google News/Trends jelek gyűjtése és párosítása
4. **Szelekció és rangsorolás** → newscast-feeder: Top 5 hír kiválasztása rádiós relevancia alapján
5. **Szövegformázás** → newscast-feeder + newscast-tts: Narratíva flow, intro/outro sablonok
6. **Felolvasás** → newscast-tts: ElevenLabs API-val történő hangszintézis (TTS)

2.2. Alkalmazott technológiák

A NewsCast rendszer fejlesztése során alkalmazott technológiák kiválasztását a feladat specifikus követelményei határozták meg: a magyar nyelvű NLP feldolgozás igénye, a valós idejű adatáramlás követelménye, a mikroszolgáltatás alapú architektúra és a konténerizált üzemeltetés. Az alábbiakban a legfontosabb technológiai döntések és azok indoklása kerül bemutatásra.

2.2.1. Python backend és a FastAPI keretrendszer

A NewsCast rendszer teljes egészében Python nyelven íródott, amely választás több tényezõn alapul. A Python az adattudományi és NLP-alkalmazások de facto nyelve, köszönhetően gazdag könyvtárának (NumPy, scikit-learn, spaCy/HuSpacy, NLTK). A Python nyelv választását a

széleskörű NLP-ökoszisztéma mellett a FastAPI keretrendszer nyújtotta fejlesztési hatékonyság is indokolta. A keretrendszer dokumentációja szerint: „*FastAPI is a modern, fast (high-performance), web framework for building APIs with Python based on standard Python type hints. The key features are: Fast – very high performance, on par with NodeJS and Go (thanks to Starlette and Pydantic); Fast to code – increase the speed to develop features by about 200% to 300%.*” (vö. Ramírez, S.: FastAPI Official Documentation, 2018–; lásd 8.5 Hivatkozások). E teljesítménybeli előny különösen az I/O-intenzív RSS-feldolgozás és az egyidejű API-kiszolgálás kontextusában releváns.

A **FastAPI** (Sebastián Ramírez, 2018) a Python ökoszisztéma egyik legmodernebb webkeretrendszere (vö. FastAPI Official Documentation; FastAPI – Wikipedia; lásd 8.5 Hivatkozások), amely az alábbi előnyöket kínálja a NewsCast számára:

- **ASGI (Asynchronous Server Gateway Interface) alapú aszinkron működés:** A Uvicorn ASGI szerveren futtatott FastAPI alkalmazás képes egyidejűleg több HTTP-kérés kiszolgálására, ami a párhuzamos RSS-feldolgozás és az API-végpontok egyidejű kezelése szempontjából kritikus.
- **Automatikus OpenAPI dokumentáció:** A FastAPI a Pydantic típusannotációk alapján automatikusan generálja az API-dokumentációt, amely a „/docs” végpontra érhető el (Swagger UI) és a „/redoc” végpontra (ReDoc formátumban).
- **Pydantic validáció:** A kérés- és válaszmódellek típusbiztos validációja futásidőben történik, csökkentve a hibás bemenetek okozta problémákat (vö. Pydantic Official Documentation; lásd 8.5 Hivatkozások). A NewsCast TTS modul például 13 Pydantic modellt definiál a „models.py” fájlban.
- **Dependency Injection:** A függőséginjektálási rendszer módot ad az autentikációs middleware újrahaználható implementációjára (lásd az „auth.py” modulokat minden szolgáltatásban).

A NewsCast minden modulja FastAPI-t használ, ezzel egységes technológiai alapot teremtve.

2.2.2. Természetes nyelvfeldolgozás (NLP)

A természetes nyelvfeldolgozás (Natural Language Processing, NLP) a mesterséges intelligencia és a nyelvtudomány metszéspontján álló terület, amely a természetes (emberi) nyelv gépi értelmezésével és generálásával foglalkozik. A magyar nyelvű NLP sajátosságait –

a gazdag morfológia és az agglutináció gépi kezelését – Lendvai és Tikk (2022) transzformer-alapú osztályozási kísérletei is alátámasztják (vö. Lendvai P. – Tikk D., 2022; lásd 8.5 Hivatkozások). A természetes nyelvfeldolgozás mint tudományterület kihívásait Jurafsky és Martin átfogó tankönyvükben az alábbiak szerint jellemzik: *„Unlike programming languages, natural languages are replete with ambiguity, both lexical and syntactic. The word 'bank' can refer to a financial institution or to the edge of a river. The sentence 'I saw the man with the telescope' can mean that I used a telescope to see the man, or that I saw a man who had a telescope.”* (vö. Jurafsky, D. & Martin, J. H.: „Speech and Language Processing”, 3rd Edition, 2023; lásd 8.5 Hivatkozások). Ez a NewsCast rendszer egyik legösszetettebb technológiai pillére. A magyar nyelv feldolgozása speciális kihívást jelent az NLP számára, amelynek okai a nyelv gazdag morfológiája (egy szótőhöz akár több száz toldalékolt alak is tartozhat), valamint a viszonylag szabad szórend.

HuSpacy: Magyar nyelvű NLP modell

A rendszer elsődleges NLP-komponense a **HuSpacy** (Orosz et al., 2022), amely a spaCy ipari szintű NLP-keretrendszer (vö. spaCy; lásd 8.5 Hivatkozások) magyar nyelvre specializált kiterjesztése. A HuSpacy fejlesztői a magyar nyelvi modell jelentőségét az alábbiak szerint fogalmazzák meg: *„Despite the growing interest in NLP for under-resourced languages, Hungarian lacks a comprehensive, production-ready toolkit comparable to English-language solutions. HuSpaCy aims to fill this gap by providing an industrial-strength Hungarian NLP pipeline built on spaCy.”* (vö. Orosz Gy. et al.: „HuSpaCy: an industrial-strength Hungarian natural language processing toolkit”, XVIII. Magyar Számítógépes Nyelvészeti Konferencia, 2022; lásd 8.5 Hivatkozások). A magyar nyelvű szövegosztályozás területén végzett korábbi vizsgálatok (vö. Lendvai P. – Tikk D., 2022; lásd 8.5 Hivatkozások) megerősítik, hogy a nyelvi sajátosságok figyelembevétele a pontosság szempontjából kritikus. A NewsCast az „hu_core_news_lg” (large) modellt alkalmazza, amely az alábbi NLP-képességekkel rendelkezik:

- **Tokenizáció:** A szöveg szavakra (tokenekre) bontása a magyar helyesírási szabályok szerint.
- **Mondathatár-felismerés (Sentence Segmentation):** A szöveg mondatokra bontása, amely az olvashatósági metrikák és az összegzés alapja.
- **Szófaji elemzés (POS Tagging):** Minden token szófaji címkéjének meghatározása, amely a tartalomelemzés és a formalizáltsági vizsgálat alapjául szolgál.

- **Névelem felismerés (Named Entity Recognition, NER):** Személyek, helyek, szervezetek és egyéb megnevezett entitások azonosítása a szövegben. Ez a funkció az „analyzer.py” modul entitáskinyerő komponensében kerül alkalmazásra és a duplikációsűrés entitás-ujjlenyomat (entity fingerprinting) alapú komponensét is táplálja.
- **Lemmatizáció:** A szavak szótári alakjának meghatározása, amely a szöveg-összehasonlítás és a kulcsszó-kinyerés pontosságát növeli. A magyar nyelv lemmatizálásának alapját a morfológiai elemzés képezi, amelynek korai számítógépes megvalósítását Prószéky és Kis (1999) dokumentálta (vö. Prószéky G. – Kis B., 1999; lásd 8.5 Hivatkozások).

A HuSpacy betöltése a „newscast-analyzer” modulban történik, kétszintű fallback-mechanizmussal: elsődlegesen a „huspacy.load()” függvényt próbálja meg, majd sikertelen esetben a natív spaCy betöltést alkalmazza, kiegészítve egy „sentencizer” pipeline-komponenssel.

NLTK és Sumy: Szövegösszegzés

A szövegösszegzés a NewsCast elemzési folyamatának fontos lépése, amelyet a **Sumy** könyvtár LexRank algoritmusával végez (vö. Sumy GitHub; lásd 8.5 Hivatkozások). A LexRank (Erkan, G. & Radev, D. R.: „LexRank: Graph-based Lexical Centrality as Saliency in Text Summarization”, Journal of Artificial Intelligence Research, Vol. 22, pp. 457-479, 2004; lásd 8.5 Hivatkozások) egy gráf-alapú extraktív összegzési algoritmus, amely a mondatok közötti hasonlóságon alapuló központiséget számítja ki és a legközpontibb mondatokat választja ki az összefoglaló számára. Az algoritmus különösen alkalmas a hírszövegek összegzésére, mivel azok jellemzően fordított piramis szerkezetűek – a legfontosabb információk az elején koncentrálódnak.

A **NLTK** (Natural Language Toolkit; vö. Bird, S. & Loper, E.: „NLTK: The Natural Language Toolkit”, ACL Workshop, 2002; NLTK; lásd 8.5 Hivatkozások) a tokenizációs alpinfrastruktúrát szolgáltatja a Sumy számára, beleértve a magyar nyelvi tokenizáló (punkt) modellt.

Google Gemini: MI-alapú validáció

A szabályalapú elemzés mellett a NewsCast a **Google Gemini Flash Lite Latest** modellt alkalmazza a rádiós relevancia MI-alapú validálására (vö. Google Gemini API Documentation;

lásd 8.5 Hivatkozások). A Gemini kötegelt feldolgozással (batch processing) működik: 15 perces intervallumonként legfeljebb 50 hírt dolgoz fel egyidejűleg. Ez a megközelítés az API-hívások számának csökkentése révén jelentős költségmegtakarítást eredményez, miközben a szabályalapú előszűrő biztosítja, hogy csak a potenciálisan releváns hírek kerüljenek MI-validáció alá.

2.2.3. Szövegfelolvasás (Text-to-Speech)

A szövegfelolvasás (Text-to-Speech, TTS) technológiája az elmúlt évtizedben gyökeresen átalakult (vö. Speech synthesis – Wikipedia; Lemmetty, S.: „History and Development of Speech Synthesis”, Aalto University; lásd 8.5 Hivatkozások). A korábbi, konkatenatív és parametrikus szintézismegoldásokat a neurális hálózat alapú rendszerek váltották fel – különösen a Google WaveNet (2016) megjelenése óta –, amelyek a természetes emberi beszédhez közelítő minőséget érnek el. Az eredeti publikáció szerint: „*WaveNet, a deep generative model of raw audio waveforms [...] is able to generate speech which mimics any human voice and which sounds more natural than the best existing Text-to-Speech systems, reducing the gap with human performance by over 50%.*” (vö. van den Oord, A. et al.: „WaveNet: A Generative Model for Raw Audio”, DeepMind, 2016; lásd 8.5 Hivatkozások). Ez a paradigmaváltás tette lehetővé az ElevenLabs és hasonló szolgáltatások megjelenését, amelyek a NewsCast TTS moduljának alapját képezik.

ElevenLabs API és az eleven_v3 modell

A NewsCast az **ElevenLabs** szolgáltatás API-ját használja a szövegfelolvasáshoz (vö. ElevenLabs API Documentation; ElevenLabs Models; lásd 8.5 Hivatkozások). A rendszer az „eleven_multilingual_v2” modellt definiálja alapértelmezettként a kódban, azonban a produkciós környezetben az „eleven_v3” modellt alkalmazza, amely a szolgáltató legújabb, nagy pontosságú szintézismodellje.

A választás indokai:

- **Magyar nyelvű támogatás:** Az eleven_v3 modell natívan támogatja a magyar nyelvet, beleértve az ékezetes karakterek és a magyar prozódia helyes kezelését.
- **Hangklónozás és testreszabás:** A modell alkalmas egyedi hangprofilok használatára, amelyek a „voice_id” paraméterrel választhatók ki. Az alapértelmezett hang („EXAVITQu4vr4xnSDxMaL”) egy természetes hangzású magyar férfihangot reprezentál.

- **SSML támogatás:** A modell értelmezi az SSML (Speech Synthesis Markup Language) elemeket – a W3C által szabványosított XML-alapú jelölőnyelvet a beszéd-szintézis vezérlésére (vö. W3C SSML 1.1 Specification; lásd 8.5 Hivatkozások) –, különösen a szünet-jelölőket, amelyeket a NewsCast a hírek közötti szünetek megvalósítására használ.
- **API-alapú integráció:** A RESTful API-n keresztüli hozzáférés megteremti a programozott használat feltételeit, a válaszformátumok (MP3, PCM, WAV) rugalmas választását és az aszinkron feldolgozást.

Magyar nyelvű szövegnormalizálás

A TTS-rendszer számára kiemelten fontos a szöveg előfeldolgozása, amely gondoskodik arról, hogy a szintetizátor a magyar nyelv szabályai szerint értelmezze a szöveget. Az OMSZ adatformátumainak korábbi dokumentációja (vö. HungaroMet, 2019; lásd 8.5 Hivatkozások) szerint a szöveges előrejelzések nem szabványosított karakterkódolással és formázással készülnek, amely a normalizálási szabályok kidolgozásánál figyelembe veendő. A NewsCast „text_normalizer.py” modulja az alábbi normalizálási lépéseket végzi el:

- **Számok szövegesítése:** A 0-tól 999 999 999 999-ig terjedő tartományban a számok magyar nyelvi megfelelőjére konvertálás (pl. 2024 → „kétezer-huszonnégy”). Az algoritmus rekurzív felépítésű és kezeli a magyar nyelv speciális eseteit (pl. „két” vs. „kettő”, „ezer” egyedülálló prefix nélkül).
- **Hőmérséklet-normalizálás:** A „-5°C” → „mínusz öt fok”, „25,5°C” → „huszonöt és fél fok” konverziók, amelyek a meteorológiai tartalmak (newscast-weather) felolvasásánál elengedhetetlenek.
- **Dátumformátum-kezelés:** A „január 15.” → „január tizenötödike” konverzió, amely a magyar nyelvben a sorszámok birtokos ragozását (-e, -a, -je) igényli. A rendszer 31 sorszám birtokos és határozóragos alakját tartalmazza az „ORDINALS_POSSESSIVE” és „ORDINALS_ON_DATE” szótárakban.
- **Időpont-normalizálás:** A „15:30” → „tizenöt óra harminc” konverzió.
- **Százalék- és pénznemkezelés:** A „80%” → „nyolcvan százalék”, „1.000 Ft” → „ezer forint” konverziók, ragozással együtt (pl. „80%-os” → „nyolcvanszázalékos”).

- **Rövidítések feloldása:** 22 beépített rövidítés az „ABBREVIATIONS” szótárban (Dr., Prof., stb., ill., kb., hPa, km/h stb.), kiegészítve 4 kontextusfüggő rövidítéssel („CONTEXT_SENSITIVE_ABBR”) és 24 speciális karakter cseré-szabállyal („SPECIAL_CHARS”), összesen 50 normalizálási szabály.
- **SSML-védelem:** A normalizálási folyamat során az SSML break tagek placeholder-ekre cserélődnek, hogy a reguláris kifejezések ne módosítsák azokat, majd a normalizálás végén visszaállítódnak (lásd „text_normalizer.py”).

2.2.4. Közösségi média platform korlátok és a Google Trends integráció

A hírrelevancia meghatározásához ideális esetben a közösségi média platformok trendjeire is támaszkodhatnánk – a Twitter/X említések, a Facebook-megosztások vagy a Reddit-szálak népszerűsége közvetlen indikátorai egy-egy hír társadalmi visszhangjának. Az elmúlt években azonban a közösségi média platformok szisztematikusan bezárták nyilvános API-jaikat, ami alapvetően megváltoztatta a külső fejlesztők és kutatók adathozzáférési lehetőségeit. Az alábbiakban áttekintem ennek a folyamatnak a legfontosabb mérföldköveit és a NewsCast rendszerre gyakorolt hatásait.

2.2.4.1. A Cambridge Analytica hatás (2018–)

A közösségi média platformok API-korlátozásainak történetében a Cambridge Analytica botrány jelenti a fordulópontot. A Facebook 2010-ben nyitotta meg a **Graph API**-t harmadik féltől származó fejlesztők számára, széleskörű hozzáférést biztosítva a felhasználói adatokhoz – beleértve az ismerősi listákat, bejegyzéseket, eseményeket és személyes adatokat (vö. MIT Internet Policy Research Initiative, 2018; lásd 8.5 Hivatkozások). 2018 márciusában azonban nyilvánosságra került, hogy a Cambridge Analytica brit politikai tanácsadó cég a platform API-ján keresztül mintegy 87 millió felhasználó adatait gyűjtötte be és használta fel jogosulatlanul politikai célú profilalkotásra (vö. Cambridge Analytica – Wikipedia; lásd 8.5 Hivatkozások). A botrány következményeként a Meta (akkoriban még Facebook) 2018 áprilisától radikális API-korlátozásokat vezetett be: megvonta az ismerősi listák, bejegyzések, események, csoportok, valamint a vallási és politikai nézetek hozzáférését harmadik féltől származó alkalmazások számára (vö. Meta: „An Update on Our Plans to Restrict Data Access on Facebook”, 2018; lásd 8.5 Hivatkozások). A Cambridge Analytica botrány API-korlátozásokra gyakorolt hatását a MIT kutatói az alábbiak szerint összegzik: *„The Cambridge Analytica affair fundamentally altered the relationship between social media platforms and third-party developers. Facebook's*

subsequent API restrictions – reducing data access for all external applications – represented a paradigm shift from an open-platform model to a walled-garden approach.” (vö. MIT Internet Policy Research Initiative: „The Cambridge Analytica Affair and the Future of Data Privacy”, 2018; lásd 8.5 Hivatkozások). Ez az intézkedés **dominóhatást** váltott ki: a többi közösségi média platform is fokozatosan bezárkózott, részben a felhasználói bizalom helyreállítása, részben a szabályozói nyomás (GDPR) miatt.

2.2.4.2. Platform-specifikus korlátozások elemzése

Az alábbi táblázatban összefoglalva látható a legjelentősebb közösségi média platformok aktuális API hozzáférési korlátozásai:

Platform	Ingyenes csomag	Komment / Trend lekérdezés	Költség (min.)	Státusz (2025)
Facebook	Nem	Nem	App review + limit	Zárt
X (Twitter)	Write-only	\$200/hó (Basic)	\$200-42 000/hó	Fizetős
Instagram	Megszűnt	Nem	Business API only	Zárt
Reddit	Korlátozott	\$0,24/1K hívás	\$12 000/év (prem.)	Részben nyílt
TikTok	Nem	Kutatói jóváhagyás	N/A	Zárt
YouTube	10K egys./nap	Kvóta korlátozott	Ingyenes (limitált)	Részben nyílt
Telegram	Bot API	FloodWait limit	Ingyenes (limitált)	Részben nyílt
Google News RSS	Igen	Igen (pozíció)	Ingyenes	Nyílt
Google Trends RSS	Igen	Igen (top-10 HU)	Ingyenes	Nyílt

1. táblázat: A legjelentősebb közösségi média platformok API hozzáférési korlátozásai

Platformonként az aktuális helyzet részletesebb kifejtése:

- **Facebook/Meta Graph API:** A jelenlegi API verzió 200 kérést engedélyez óránként és alkalmazásonként (vö. Meta Graph API Overview & Rate Limiting; lásd 8.5 Hivatkozások), az alkalmazás-jóváhagyási folyamat (App Review) pedig szigorú feltételrendszerrel működik, amelyet a fejlesztői közösség magas elutasítási aránnyal jellemez (vö. Meta App Review Documentation; lásd 8.5 Hivatkozások). 2025-től a Meta tovább szigorította az egyedi célközönség kezelési (Custom Audience) szabályokat (vö. Meta Business Help Center: „Custom Audiences”, 2025; lásd 8.5 Hivatkozások), ami a marketing- és kutatási célú API-használatot is korlátozza. Nyilvános bejegyzések aggregált kinyerésére jelenleg nincs hivatalos API-végpont.

- **X (Twitter):** A platform a 2023-as API-átalakítás során a „Free tier”-t write-only módra korlátozta – ingyenesen kizárólag tweeteket lehet közzétenni, olvasási hozzáférés nélkül. A „Basic tier” (\$100/hó, 2024-től \$200/hó) korlátozott olvasási hozzáférést biztosít, a „Pro tier” \$5000/hó áron kínál bővebb kvótát, míg az „Enterprise” szint ~\$42 000/hó költséggel jár (vö. X API Pricing Documentation; lásd 8.5 Hivatkozások). Az akadémiai kutatási program (Academic Research API) 2023-ban megszűnt (vö. X Developer Platform: „Deprecation of the Academic Research product track”, 2023; lásd 8.5 Hivatkozások), amely korábban ingyenes, teljes archívum hozzáférést biztosított kutatóknak.
- **Instagram:** A Meta 2024. december 4-én véglegesen megszüntette a Basic Display API-t, amely korábban a felhasználók saját bejegyzéseinek elérését tette lehetővé (vö. Instagram Graph API Developer Guide; lásd 8.5 Hivatkozások). A Graph API kizárólag „Business” és „Creator” fióktípusokhoz érhető el, 2025-ben pedig a rate limitet 5000-ról 200 kérésre/óra csökkentették – ez 96%-os korlátozás (vö. Meta for Developers: Instagram Platform Rate Limits & Changelog; lásd 8.5 Hivatkozások).
- **Reddit:** A platform 2023 júliusától \$0,24/1000 API-hívás díjat vezetett be, ami a korábban ingyenes hozzáférés monetizálását jelentette (vö. Reddit API Pricing & Terms; lásd 8.5 Hivatkozások). A díjváltozás közvetlen következménye volt, hogy a legismertebb harmadik féltől származó Reddit-kliensek – Apollo, Reddit is Fun (RIF), Sync – 2023 nyarán megszüntették működésüket (vö. 2023 Reddit API controversy – Wikipedia; lásd 8.5 Hivatkozások). A hobbi és kutatási projektek számára továbbra is elérhető egy korlátozott „Free tier”, amely azonban kvóta- és funkcionális korlátozásokkal bír.
- **TikTok:** A Research API kizárólag jóváhagyott akadémiai kutatók számára érhető el, az USA-ban és az EU-ban eltérő hozzáférési feltételekkel (vö. TikTok Research API Documentation; lásd 8.5 Hivatkozások). 2024-ben az európai parlamenti választások idején adatminőségi problémák merültek fel a kutatói API-n szolgáltatott adatokban (vö. TechPolicy.Press: „Researcher Data Access Under the DSA”, 2024; lásd 8.5 Hivatkozások), ami az API megbízhatóságát is megkérdőjelezi.
- **YouTube Data API v3:** A Google 10.000 egység/nap ingyenes kvótát biztosít, amelyben egyetlen keresési kérés 100 egységbe kerül – ez napi ~100 keresést jelent (vö. YouTube Data API v3 Quota Documentation; lásd 8.5 Hivatkozások). Bár a kvóta bővíthető, a kérelmezés indoklást és jóváhagyást igényel.

- **Telegram:** A Bot API nem fér hozzá privát csoportokhoz és csatornákhöz, a publikus csatornák olvasása pedig FloodWait rate-limitbe ütközik (csatornánkénti maximum ~10 üzenet/lekérdezés) (vö. Telegram Bot API Documentation; lásd 8.5 Hivatkozások). A Telegram API-ra épülő trendjelek gyűjtése így jelenleg nem skálázható megoldás.

2.2.4.3. A GDPR és az MI-modellek „adatéhségének” hatása

A platformok bezárkózásának másik hajtóereje az EU **Általános Adatvédelmi Rendelet** (GDPR, 2018), amely a személyes adatok fogalmát kiterjesztette a felhasználónevekre, közösségimédia handle-ekre és nyilvánosan elérhető profilinformációkra (vö. GDPR – Wikipedia; lásd 8.5 Hivatkozások). 2024-ben az EU adatvédelmi szabályozók kifejezetten korlátozó álláspontot foglaltak el a mesterségesintelligencia-célú adatgyűjtés (scraping) kapcsán, megállapítva, hogy a nyilvánosan elérhető közösségimédia-tartalmak gyűjtése is adatvédelmi jogalapot igényel (vö. Morgan Lewis: „EU Data Protection Regulators' Position on AI Scraping”, 2024; IAPP: „The state of web scraping in the EU”; lásd 8.5 Hivatkozások).

Ezzel párhuzamosan a platformok felismerték, hogy az általuk tárolt adatok az MI-modellek tréningezéséhez rendkívül értékesek. A Reddit például 2024-ben licencszerződéseket kötött a Google-el és az OpenAI-al az adatainak AI tréningcélú felhasználására, továbbá korlátozta azoknak a harmadik feleknek a hozzáférését, akik nem kötöttek ilyen megállapodást. Ez a monetizációs tendencia tovább szűkíti a független fejlesztők és kutatók mozgásterét.

2.2.4.4. Scraping mint alternatíva – lehetőségek és korlátok

A hivatalos API-k bezáródása természetesen felveti a webes adatgyűjtés (scraping) alternatíváját. Az **Apify** platform (vö. Apify Platform Documentation; lásd 8.5 Hivatkozások) például 10.000-nél több előre gyártott „Actor”-t kínál, amelyek között megtalálhatók az Instagram, TikTok, Facebook és Reddit tartalmak automatikus gyűjtését végző scraperek. Az Apify árazási modellje: Free szint (\$5 kredit tesztelésre), Starter \$39/hó, Scale \$199/hó, Enterprise \$999/hó.

A scraping jogi helyzete azonban összetett és bizonytalan. A **Meta v. Bright Data** (2024) ügyben a bíróság megállapította, hogy a nyilvánosan elérhető adatok gyűjtése önmagában nem sérti a CFAA (Computer Fraud and Abuse Act) törvényt (vö. D-Lab, UC Berkeley: „The Evolving Landscape of Web Scraping on Social Media Platforms”; lásd 8.5 Hivatkozások). Ugyanakkor az **X Corp** hasonló perekben a szerverterhelésre és a felhasználási feltételek megsértésére hivatkozott, ami felelősséget vonhat maga után a scraper üzemeltetőjének. Az EU

GDPR kontextusában a scraping személyes adatot is érinthet – egy nyilvános Facebook-bejegyzés szerzőjének neve és profilképe személyes adat –, ami jogalap (pl. jogos érdek) igazolását és adatvédelmi hatásvizsgálatot igényelne.

Egy akadémiai vagy hobbi projekt – mint a NewsCast – számára a scraping-alapú megoldás tehát hármas kockázatot hordoz: (1) a havi költségek (\$39–999) nem állnak arányban a projekt költségvetésével, (2) a jogi helyzet bizonytalan és folyamatosan változik, (3) az etikai kérdések (felhasználói adatok gyűjtése a felhasználók kifejezett hozzájárulása nélkül) aggályokat vetnek fel.

2.2.4.5. A választott megoldás indoklása – Google News/Trends RSS

A fenti korlátozások áttekintése után a NewsCast rendszer a **Google News RSS** és a **Google Trends RSS** feedekre épít közösségi trendjel-forrásként. Ez a választás az alábbi előnyökön alapul:

1. **Nyilvános és ingyenes hozzáférés:** A Google News és Google Trends RSS feedek API-kulcs nélkül, korlátozás nélkül elérhetők. Nem igényelnek regisztrációt, alkalmazás-jóváhagyást vagy (fizetős) előfizetést.
2. **Nem személyes adatok:** A feedek nem tartalmazzák felhasználói személyes adatokat – kizárólag hírszalagcímeket, URL-eket és aggregált keresési trendeket szolgáltatnak. Ez GDPR szempontból lényegesen egyszerűbb helyzetet teremt, mint a közösségi média platformok felhasználói tartalmainak gyűjtése.
3. **A magyar hírpiac lefedettsége:** A Google News 7 kategóriás feedje (top, belföldi, világ, üzlet, technológia, tudomány, egészség) a magyar nyelvű hírpiac egészét lefedi, a *hl=hu&gl=HU* lokalizációval a magyar hírfogyasztók szemszögéből releváns tartalmakat priorizálja.
4. **Kockázatkezelés:** A Google rate-limiting kockázatát a newscast-social modul budget rendszerrel és 30 perces cooldown mechanizmussal kezeli (vö. 3.6.6 fejezet), minimalizálva a túlzott lekérdezés lehetőségét.

A hírértékelés hagyományos, tartalomközpontú megközelítései – amelyek a szöveg jellemzőire (olvashatóság, szentiment, forrás presztízse) támaszkodnak – nem képesek maradéktalanul megragadni egy hír valós társadalmi hatását. A közösségi média és a keresőmotor-trendek elemzése kiegészítő dimenziót ad a relevancia-meghatározáshoz: ha egy hírtéma a Google

Trends top keresései között szerepel vagy a Google News kiemelten kezeli, az a téma széles körű érdeklődést tükröz.

A NewsCast rendszer a **newscast-social** modul révén két külső adatforrást integrál a trendjelek gyűjtéséhez:

- **Google News RSS:** A Google News 7 tematikus RSS feedjéből (top hírek, belföldi, világ, üzlet, technológia, tudomány, egészség) gyűjt pozíció alapú engagement jeleket. A Google News RSS magyar nyelvre lokalizált feedeket kínál (hl=hu&gl=HU paraméterezéssel), amelyek a magyar híriac aktuális prioritásait tükrözik.
- **Google Trends RSS:** A Google Trends napi trendek feedjéből (geo=HU) a Magyarországon aktuálisan legkeresettebb kulcsszavakat gyűjti, amelyek a közösségi érdeklődés közvetlen indikátorai. A trending kulcsszavak a hírekkel való szöveges egyeztetés (substring matching) révén kerülnek felhasználásra.

A trendjelek és a meglévő hírek közötti párosítás háromszintű algoritmussal történik: (1) URL-hash alapú pontos egyezés, (2) kanonikus URL-hash egyezés és (3) **RapidFuzz** könyvtár „token_sort_ratio” algoritmus (vö. RapidFuzz GitHub; lásd 8.5 Hivatkozások) szerinti fuzzy cím-egyeztetés. A RapidFuzz a Levenshtein-távolságra és a token-alapú hasonlóságra épülő, C++ gyorsítású Python-könyvtár, amely a szórendtől független összehasonlítást biztosít – ez különösen fontos a hírszalagcímek esetében, ahol ugyanaz a hír eltérő szórenddel jelenhet meg különböző forrásokban.

A párosított trendjelekből a **Super Formula** (Szuper képlet) pontszámot számít:

$$V_H = 10 + (Trends * 50)$$

ahol V_H a **Virális Hírérték** – az adott hír közösségi trendjeleken alapuló, pontszámként kifejezett hírértéke. A „Trends” értéke 1, ha a hír címe egyezik egy Google Trends TOP10 kulcsszóval, egyébként 0. A képlet két értéket ad: 10 (alap hír) vagy 60 (virális hír). Ez a pontszám a „social_trending_score” mezőben kerül tárolásra az „analysis” táblában és a newscast-analyze modul a Gemini batch prioritás meghatározásánál súlyozott tényezőként alkalmazza (vö. 3.6.6 fejezet).

2.2.5. RESTful architektúra és mikroszolgáltatások

A NewsCast rendszer **mikroszolgáltatás-architektúrát** (Microservices Architecture) követ – amelynek alapelveit Lewis és Fowler az alábbiak szerint definiálják: „*The microservice*

architectural style is an approach to developing a single application as a suite of small services, each running in its own process and communicating with lightweight mechanisms, often an HTTP resource API. These services are built around business capabilities and independently deployable by fully automated deployment machinery.” (vö. Lewis, J. & Fowler, M.: „Microservices: a definition of this new architectural term”, martinowler.com, 2014; lásd 8.5 Hivatkozások) –, amelyben minden modul önálló, egyetlen felelősséggel bíró szolgáltatásként működik, saját adatbázis-sémával, API-felülettel és konténerrel. A hat szolgáltatás közül öt önálló adatbázis-sémával rendelkezik, míg a newscast-social a newscast-analyze adatbázisát osztja meg az adatbázis-szintű integráció érdekében (vö. 3.2.1 fejezet). Ez a megközelítés az alábbi előnyökkel jár:

- **Független fejleszthetőség:** Az egyes modulok önállóan fejleszthetők, tesztelhetők és telepíthetők.
- **Skálázhatóság:** A nagyobb terhelésnek kitett modulok (pl. a newscast-analyze modul, amely a leginkább számításigényes) külön skálázhatók.
- **Hibaelkülönítés:** Egy modul meghibásodása nem vonja maga után a teljes rendszer leállítását.
- **Technológiai rugalmasság:** Bár jelenleg minden modul Python-alapú, az architektúra lehetővé tenné egyes modulok más technológiára való átírását.

A szolgáltatások közötti kommunikáció szinkron HTTP/REST-hívásokkal történik. A REST (Representational State Transfer) architektúrális stílust Roy Fielding definiálta 2000-es doktori disszertációjában (vö. Fielding, R. T.: „Architectural Styles and the Design of Network-based Software Architectures”, UC Irvine, 2000; lásd 8.5 Hivatkozások). Minden modul két portot nyit:

- **80-as port:** Az üzleti API-végpontok (hitelesítéssel védett).
- **8000-es port:** A Prometheus metrikák kiszolgálása (monitorozási célú, publikus).

A hitelesítés egységes, kétcsatornás megoldással valósul meg: **HTTP Basic Auth** és **JWT (JSON Web Token) Bearer Token** (vö. RFC 7519: JSON Web Token, Jones, M. et al., 2015; JWT.io; lásd 8.5 Hivatkozások) egyaránt elfogadott minden végponton. A Basic Auth az egyszerűbb integrációk és a fejlesztési környezet számára biztosít hozzáférést, míg a JWT a programozott API-hívások és a szolgáltatások közötti kommunikáció biztonságos módja.

2.2.6. Adatbázis-kezelés: SQLAlchemy és MariaDB

A NewsCast minden modulja **MariaDB 11.8.3** relációs adatbázist használ (vö. MariaDB; MariaDB – Wikipedia; lásd 8.5 Hivatkozások), amelyhez a **SQLAlchemy** ORM-en (Object-Relational Mapping) keresztül kapcsolódik. Az ORM réteg alkalmazásának elméleti alapjait Bayer – a SQLAlchemy alkotója – az alábbiak szerint indokolja: *„SQLAlchemy considers the database to be a relational algebra engine, not just a collection of tables. Rows can be selected from not only tables but also joins and other select statements; any of these units can be composed into a larger structure.”* (vö. Bayer, M.: „SQLAlchemy”, in: „The Architecture of Open Source Applications, Volume II”, aosabook.org, 2012; lásd 8.5 Hivatkozások). Ez a szemlélet lehetővé teszi a NewsCast komplex lekérdezéseinek (pl. az `ai_radio_suitable_news` nézet) elegáns megvalósítását. A MariaDB a MySQL nyílt forráskódú fork-ja – amelyet Michael „Monty” Widenius alapított az Oracle általi MySQL-felvásárlás után – és amely teljes mértékben kompatibilis a MySQL protokollal és az SQL szintaxissal; a modulok a „mysql+pymysql” SQLAlchemy driver-en keresztül csatlakoznak, amely mindkét adatbázis-szerverrel működik.

Az ORM használata az alábbi előnyöket biztosítja:

- **SQL injection elleni védelem:** Parametrikus lekérdezések alkalmazása.
- **Adatmodell centralizálása:** A Python osztályok és az adatbázis-táblák közötti leképezés egyértelmű.
- **Connection Pool:** A SQLAlchemy beépített kapcsolat kezelője felel a hatékony adatbázis-kapcsolatkezelésért (alapértelmezetten 5-10 állandó kapcsolat, 5-10 túlszordulási kapcsolat).

Az adatbázis karakterkódolása UTF-8MB4 („utf8mb4_unicode_ci”), amely teljes Unicode-támogatást nyújt, beleértve a magyar ékezetes karaktereket. Az InnoDB tároló motor használata tranzakciótámogatást és sor szintű zárolást (row-level locking) biztosít, ami a párhuzamos feldolgozás során kritikus a deadlock-helyzetek elkerüléséhez.

2.2.7. COCO modell és az OAM elemzés

A COCO (Component-based Object Comparison for Objectivity) egy matematikai modell az objektumok többszemponútú, objektív (emberi súlyozás/pontozás nélküli) összehasonlítására (vö. COCO API; lásd 8.5 Hivatkozások), amely a „miau.my-x.hu” webes szolgáltatáson

keresztül érhető el. E módszertani megközelítés a NewsCast rendszerben a hírforrások egymáshoz képest becsülhető értékének objektív vizsgálatára kerül alkalmazásra az OAM keretrendszeren belül. A hasonlóságelemzési módszertan angol nyelvű definícióit és elméleti alapjait (vö. Pitlik L., é.n., Similarity Analysis; lásd 8.5 Hivatkozások) a COCO megközelítés dokumentálja. A hasonlóságelemzés elméleti és gyakorlati alapjait magyar nyelven az ismertető füzet (vö. Pitlik L., 2014, Hasonlóságelemzési alapok; lásd 8.5 Hivatkozások) rendszerezi. A módszertani összefoglaló a MIAU Wiki közösségi tudásbázisában (vö. MIAU Wiki, 2014; lásd 8.5 Hivatkozások) is elérhető.

A COCO három modellváltozatot támogat:

- **Y0 (anti-diszkriminatív modell):** A varianciaminimalizáláson alapuló modell, amely az objektumok (hírforrások) közötti ideál-különbségek minimalizálására törekszik, kimutatva az attribútumok (hírtulajdonságok) hozzájárulását az összeredményhez (a normától való esetleges eltéréshez). Az attribútumok fontosságának MI-alapú levezetése (vö. Bán Gy. et al., 2024; lásd 8.5 Hivatkozások) a varianciaminimalizáláson alapuló Y0 modell értékelhetőségét tovább erősíti. Az anti-diszkriminatív megközelítés solver-alapú neurális hálókkal történő közelítése (vö. Pitlik L. – Barta G., 2021; lásd 8.5 Hivatkozások) a módszertan alkalmazási körének kiterjesztését demonstrálja. A vizsgálat lényege: lehet-e minden objektum másként egyforma?!
- **STD (felügyelt tanulás):** A célérték rekonstrukción alapuló modell, amely a megfigyelt értékek alapján becsléseket készít.
- **MCM (exploratív modell):** A binelés alapú modell, amely az értékek csoportosítását végzi el. A COCO MCM kapcsán a lépcsős függvények nem kerülnek kényszeres restriktiók hatása alá, ami a maximális rugalmasság záloga, a ceteris paribus alakzatok polinomizálódásának potenciális kockázata árán.

A COCO modell alkalmazásának lényege az objektív összehasonlítás: a jóság (goodness) mérése több szinten történik. A legegyszerűbb szint a találati arány (hit rate), amely a tény- és a becsült értékek egyezésének arányát fejezi ki (vö. Pitlik L., 2004; lásd 8.5 Hivatkozások). A numerikus jósági mutatók – az abszolút és a négyzetes eltérés összege, a korreláció – a becslések pontosságát folytonos skálán mérik. A COCO Y0 modell a variancia minimalizálásával az objektumok közötti ideál-különbségeket csökkenti, azaz egy aggregált, optimalizált jósági kritériumot alkalmaz, amelyben nem emberi súlyok, hanem a matematikai solver határozza meg az összehasonlítás alapját. A modell- és esetszintű túltanulás kockázat

felismerésére kidolgozott DIF(e)SA módszer (vö. Váradi D. et al., 2023; lásd 8.5 Hivatkozások) az objektív összehasonlítás megbízhatóságának növelését célozza. A hasonlóságelemzési módszer a HR kockázatok területén is alkalmazásra került gyanúgenerálási célokra (vö. Pitlik L., 2013; lásd 8.5 Hivatkozások), ami a COCO-modell széleskörű adaptálhatóságát mutatja.

A NewsCast „coco_client.py” modulja HTTP kliensként kommunikál a távoli COCO Y0, STD és MCM szolgáltatással. A bemeneti mátrixot TSV (Tab-Separated Values) formátumban, CRLF (Carriage Return Line Feed) sorvégekkel készíti elő – ez a formátum a „miau.my-x.hu” API specifikus követelménye (lásd „coco_client.py”). A válasz feldolgozása HTML-alapú (engine3.php) vagy egyszerű szöveges (engine3_curl.php) végponton keresztül történik, ahol a HTML-válasz esetén a „CocoHTMLParser” osztály a Python beépített „HTMLParser” alaposztályából származtatva végzi a strukturált adatkinyerést.

2.2.8. Konténerizáció és monitorozás

A NewsCast minden modulja **Docker**-konténerben fut (vö. Docker Documentation; Docker – Wikipedia; lásd 8.5 Hivatkozások), amelyet a „Dockerfile”-ok definiálnak. A konténerizáció előnyeit a Docker dokumentáció az alábbiak szerint foglalja össze: *„A container is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another. A Docker container image is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries and settings.”* (vö. Docker Documentation: „What is a Container?”; lásd 8.5 Hivatkozások). A NewsCast rendszerben minden modul ilyen szabványosított konténeregységként működik, biztosítva a fejlesztési és az éles környezet azonosságát.

A monitorozás a **Prometheus** (v3.6.0) metrikagyűjtő rendszerre (vö. Prometheus; Prometheus – Wikipedia; lásd 8.5 Hivatkozások) – amelyet eredetileg a SoundCloud fejlesztett ki és 2016 óta a Cloud Native Computing Foundation (CNCF) második graduált projektje – és a **Grafana** (v12.2.0) vizualizációs platformra épül (vö. Grafana; Grafana – Wikipedia; lásd 8.5 Hivatkozások).

Minden modul dedikált „monitoring.py” modullal rendelkezik, amely az alábbi metrikatípusokat szolgáltatja:

- **Rendszermetrikák:** CPU-használat, memóriefoglalás, hálózati és lemezes I/O (psutil könyvtár).

- **Alkalmazásmetrikák:** API-kérések száma, feldolgozási idők, sorhosszak, hibaráták.
- **Üzleti metrikák:** Feldolgozott hírek száma, TTS-generálások, duplikációk aránya.

A Prometheus periodikusan gyűjti (scrape) a modulok 8000-es portján közzétett metrikákat, a Grafana pedig a Prometheus-adatforrásra épülő interaktív dashboardokon jeleníti meg a rendszer állapotát valós időben.

A strukturált naplózás JSON formátumban történik, minden bejegyzéshez **korrelációs azonosítót** (correlation ID) rendelve. A korrelációs azonosító lehetővé teszi egy kérés teljes életciklusának végigkövetését a rendszer moduljain át, ami a hibakeresés és az audit szempontjából alapvető fontosságú.

2.3. Kapcsolódás a tanulmányokhoz (tantárgyi integráció)

A NewsCast rendszer fejlesztése során számos, a képzés keretében elsajátított elméleti és gyakorlati ismeret került alkalmazásra. Az alábbi alfejezetek célja, hogy az olvasó – beleértve azokat is, akik nem feltétlenül jártasak az egyes szakterületeken – számára is közérthető módon bemutassák, hogyan kapcsolódnak az egyetemi tantárgyak a rendszer konkrét megvalósításához. Az alábbiakban bemutatom a legjelentősebb tantárgyi kapcsolódásokat, összesen 27 tantárgy vonatkozásában.

2.3.1. Szoftverarchitektúrák, Rendszertervezés és Rendszermodellezés

A **Szoftverarchitektúrák** tantárgy keretében megismert architekturális minták közül a NewsCast a **mikroszolgáltatás-mintát** (Microservices Pattern) alkalmazza (vö. 2.2.4 fejezet). Egyszerűen fogalmazva: ahelyett, hogy egyetlen nagy programot készítettem volna, amely minden feladatot ellát, a rendszert öt kisebb, önálló „szolgáltatásra” bontottam – mindegyik egyetlen feladatért felelős (ezt nevezzük Single Responsibility Principle-nek, azaz az egyetlen felelősség elvének). A szolgáltatások egymással internetes „üzenetváltással” (RESTful API-hívásokkal) kommunikálnak, ami az Interface Segregation Principle – azaz a felület-elkülönítés elve – megvalósítása. Ez azt jelenti, hogy minden szolgáltatás csak azokat az adatokat kapja meg és küldi el, amelyekre valóban szüksége van.

A tervezési minták alkalmazásának jelentőségét Gamma és társszerzői – a szoftverfejlesztési szakirodalom klasszikus művében – az alábbiak szerint fogalmazzák meg: „*Design patterns [...] are descriptions of communicating objects and classes that are customized to solve a general design problem in a particular context. [...] Each design pattern systematically names,*

explains, and evaluates an important and recurring design in object-oriented systems.” (vö. Gamma, E. et al.: „Design Patterns: Elements of Reusable Object-Oriented Software”, Addison-Wesley, 1994; lásd 8.5 Hivatkozások). A NewsCast rendszerben a Strategy, Factory, Observer és Singleton minták alkalmazása e módszertani megalapozottságon nyugszik. – vagyis a szoftverfejlesztésben bevált, újrafelhasználható megoldási sablonok – közül a rendszer az alábbiakat alkalmazza:

- **Strategy Pattern (stratégia minta):** Az „analyzer.py” modulban a különböző elemzési stratégiák (NLP, biztonsági, duplikáció) cserélhető komponensekként vannak megvalósítva. Ennek köszönhetően az elemzés egyes lépéseit egymástól függetlenül módosíthatom, anélkül, hogy a többit érinteném (vö. 3.6.2 fejezet).
- **Factory Pattern (gyár minta):** A „storage.py” modul a TTS-szolgáltatásban négy különböző tárolóadaptert (Local, S3, GoogleDrive, FTP) biztosít – azaz a rendszer egy „gyár” segítségével a konfigurációnak megfelelően automatikusan kiválasztja, hová mentse a hangfájlokat.
- **Observer Pattern (megfigyelő minta):** A Prometheus metrikagyűjtés az observer mintát valósítja meg, ahol a metrikagyűjtők „megfigyelik” a rendszereseményeket (például API-kérések számát) és változás esetén automatikusan frissítik az adatokat.
- **Singleton Pattern (egyke minta):** A „ParamManager” osztály az newscast-analyze modulban egyetlen közös példányon keresztül kezeli a dinamikus paramétereket, így a rendszer minden pontja ugyanazokat a beállításokat használja.

A **Rendszertervezés** tantárgy ismeretei a rendszer funkcionális dekompozíciójában, az API-tervezésben és a szolgáltatások közötti felelősségmegosztás kialakításában nyilvánulnak meg.

A **Rendszermodellezés** keretében elsajátított UML- és ER-diagram készítési módszertanok az adatbázis-sémák tervezésében és a rendszerarchitektúra dokumentálásában kerültek alkalmazásra.

2.3.2. Programozás, Programozási alapelvek és módszertanok

A **Programozás** tantárgy keretében elsajátított ismeretek a rendszer teljes kódbázisában megjelennek:

- **Aszinkron programozás:** A modern programnyelvek lehetővé teszik, hogy egy program egyszerre több feladatot is végezzen – például egyszerre több hírportálról

töltsön le tartalmat, ahelyett, hogy egyenként, egymás után dolgozná fel azokat. A NewsCast az „async/await” szintaxist és az „asyncio” eseményhurkot alkalmazza a párhuzamos RSS-feldolgozásban (vö. 3.6.1 fejezet).

- **Objektumorientált tervezés:** Az objektumorientált megközelítés lényege, hogy az adatokat és a rajtuk végezhető műveleteket „objektumokba” szervezzük. Ilyen objektum például az „UnifiedAnalyzer” (elemző), a „CocoClient” (COCO API-kliens) vagy a Storage adapterek, amelyek mindegyike egy jól körülhatárolt feladatért felel.
- **Dekorátorok és metaprogramozás:** A dekorátorok olyan programozási elemek, amelyek egy meglévő függvényt „becsomagolnak” és kiegészítő funkcionalitást adnak hozzá – például automatikus újrapróbálkozást hálózati hiba esetén (a „@retry” dekorátor a tenacity könyvtárból) vagy hitelesítés-ellenőrzést az API-végpontokon.
- **Kontextuskezelés:** A „with” utasítás alkalmazása az adatbázis-szessziók és a fájlműveletek biztonságos kezeléséhez – ez gondoskodik arról, hogy a megnyitott erőforrások (például adatbázis-kapcsolatok) minden esetben szabályosan lezáródjanak, még hiba esetén is.

A **Programozási alapelvek és módszertanok** tantárgy a SOLID elvek tudatos alkalmazásában (Single Responsibility, Interface Segregation), a kód újrahasználatosságának biztosításában és a moduláris szoftverfelépítés kialakításában nyújtott alapot. A SOLID elvek alkalmazásának indokoltságát Martin az alábbiak szerint fogalmazza meg: *„The SOLID principles are not rules. They are not laws. They are not perfect truths. They are statements on the order of: This is good advice, someone found it to be true in the past, and you are likely to find it true as well. [...] The Single Responsibility Principle (SRP) states that a class should have one, and only one, reason to change.”* (vö. Martin, R. C.: „Agile Software Development: Principles, Patterns, and Practices”, Prentice Hall, 2003; lásd 8.5 Hivatkozások). A NewsCast moduláris felépítésében az SRP elv minden modulnál következetesen érvényesül: egy modul – egy felelősség.

2.3.3. Adatbázisok

Az **Adatbázisok** tantárgy során elsajátított relációs adatbázis tervezési ismeretek a NewsCast rendszer magját képezik. A relációs adatbázis olyan rendszer, amely az adatokat táblázatos formában – sorokból és oszlopokból álló táblákban – tárolja és a táblák között meghatározott kapcsolatokat (relációkat) definiál (vö. 2.2.5 fejezet). A relációs adatbázis-kezelés elméleti alapjait Codd alapvető munkájában az alábbiak szerint definiálta: *„Future users of large data*

banks must be protected from having to know how the data is organized in the machine (the internal representation). [...] Activities of users at terminals and most application programs should remain unaffected when the internal representation of data is changed and even when some aspects of the external representation are changed.” (vö. Codd, E. F.: „A Relational Model of Data for Large Shared Data Banks”, Communications of the ACM, 1970; lásd 8.5 Hivatkozások). A NewsCast adatbázis terve ezt az adatfüggetlenségi elvet a SQLAlchemy ORM rétegen keresztül valósítja meg. A rendszer összesen 16 táblát definiál öt adatbázis-sémában (vö. 3.3 fejezet):

- **newscast-rss_parser:** „rss”, „news”
- **newscast-analyze:** „news”, „analysis”, „analysis_params”, „rss”, „news_clusters”, „oam_snapshots”, „oam_antagonisms”
- **newscast-feeder:** „feeder_news”, „feeder_params”, „feeder_users”, „feeder_sessions”
- **newscast-weather:** „weather”
- **newscast-tts:** „tts_templates”, „tts_schedules”, „tts_history”, „tts_downloads”
- **newscast-social:** „social_signals”

A táblakapcsolatok, az indexelési stratégiák (egyedi indexek a duplikációsűréshez, összetett indexek a teljesítményoptimalizáláshoz) és a normalizálási szintek meghatározása a tantárgy keretében tanult módszertanon alapul. A „UNIQUE KEY” kényszerek alkalmazása (pl. „uk_url” az RSS-táblában, „unique_url” a news-táblában) az adatintegritás biztosításának alapvető eszköze.

2.3.4. Adatszerkezetek és algoritmusok, Matematikai alapok

Az **Adatszerkezetek és algoritmusok** tantárgy ismeretei több ponton is megjelennek:

- **Gráf-alapú algoritmusok:** A gráf egy csomópontokból és élekből álló matematikai struktúra. A LexRank összegzési algoritmus a mondatokat csomópontként, a köztük lévő hasonlóságot élként értelmezve határozza meg, mely mondatok a legfontosabbak egy szövegben (vö. 2.2.2 fejezet). A LexRank algoritmus elméleti háttérét az eredeti szerzők az alábbiak szerint ismertetik: *„LexRank is an unsupervised approach to text summarization based on the concept of eigenvector centrality in a graph representation of sentences. In this model, a connectivity matrix based on intra-sentence cosine*

similarity is used as the adjacency matrix of the graph representation of sentences.” (vö. Erkan, G. & Radev, D. R.: „LexRank: Graph-based Lexical Centrality as Saliency in Text Summarization”, JAIR, Vol. 22, 2004; lásd 8.5 Hivatkozások).

- **Vektorizálás és hasonlóságkeresés:** A TF-IDF (Term Frequency - Inverse Document Frequency; vö. tf-idf – Wikipedia; lásd 8.5 Hivatkozások) módszer a szövegeket számsorozatokká (vektorokká) alakítja, amelyek összehasonlíthatóvá válnak. A koszinusz-hasonlóság két ilyen vektor „szögét” méri – minél kisebb a szög, annál hasonlóbb a két szöveg. A FAISS (Facebook AI Similarity Search; vö. FAISS GitHub; FAISS – Meta AI; lásd 8.5 Hivatkozások) index a nagy léptékű hasonlóságkeresést biztosítja.
- **Hash-alapú adatszerkezetek:** A hash-függvény egy tetszőleges méretű adatból rögzített méretű „ujjlenyomatot” készít. A SHA-256 hash alkalmazása a tartalomdeduplikációban (newscast-weather, newscast-tts) biztosítja, hogy azonos tartalmú szövegek felismerhetők legyenek; az ETag-gyorsítótárzás pedig a HTTP-szinten alkalmaz hash-eket (vö. 3.6.1 fejezet).
- **Rekurzív algoritmusok:** A rekurzió olyan programozási technika, amelyben egy függvény „önmagát hívja meg” kisebb részproblémákkal. A „number_to_words” függvény (vö. 8.7 Forráskódok; „text_normalizer.py”) rekurzív felépítéssel konvertálja a számokat magyar szöveggé – például a 2024-et felbontja $2 \cdot 1000 + 24$ -re, majd a 24-et $20 + 4$ -re.

A **Matematikai alapok** tantárgy a COCO modell megértésében játszott kulcsszerepet. A variancia – egyszerűen szólva az adatok „szétszórtságának” mértéke – minimalizálása a Y0 modell alapja; a mátrixműveletek (az OAM bemeneti mátrix, azaz egy táblázatszerű adatstruktúra elkészítése) és a statisztikai mutatók (szórás, antagonizmus-detektálás) alkalmazása a hírforrások objektív összehasonlítását teszi lehetővé (vö. 2.2.6 fejezet).

2.3.5. Mesterséges intelligenciák az IT-biztonság területén

A mesterséges intelligencia és az IT-biztonság kapcsolódása a NewsCast rendszerben a tartalombiztonsági szűrésben jelenik meg. A Google Gemini API dokumentációja szerint: *„Safety settings allow you to adjust the likelihood of getting responses that may be considered harmful. By default, safety settings block content with medium or higher probability of being unsafe across all dimensions including hate speech, dangerous content, sexually explicit*

content, and harassment.” (vö. Google: Gemini API Safety Settings Documentation, 2024; lásd 8.5 Hivatkozások). A NewsCast a Gemini MI-validáció mellett saját, 53 mintából álló tartalomszűrő rendszert is alkalmaz (vö. 3.6.2.1 fejezet).

2.3.6. Hálózatok és számítógép architektúrák, Operációs rendszerek

A **Hálózatok és számítógép architektúrák** tantárgy ismeretei a rendszer kommunikációs és infrastrukturális rétegében jelennek meg:

- **HTTP/HTTPS protokoll:** A HTTP (HyperText Transfer Protocol) az internet alapvető kommunikációs protokollja, amelyen keresztül a böngészők és a szerverek információt cserélnek. A HTTP protokoll meghatározó szerepét a webes szolgáltatásokban az RFC 7230 az alábbiak szerint rögzíti: *„The Hypertext Transfer Protocol (HTTP) is a stateless application-level request/response protocol that uses extensible semantics and self-descriptive message payloads for flexible interaction with network-based hypertext information systems.”* (vö. Fielding, R. T. & Reschke, J.: „RFC 7230: Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing”, IETF, 2014; lásd 8.5 Hivatkozások). A NewsCast összes szolgáltatásközi kommunikációja HTTP-n keresztül történik, a hitelesítés HTTP Basic Auth és JWT Bearer Token mechanizmusokkal (vö. 3.4.4 fejezet).
- **Aszinkron I/O:** Az aszinkron (nem blokkoló) bemenet-kimenet (I/O) lehetővé teszi, hogy a program ne vározzon tétlenül egy hálózati válaszra, hanem közben más feladatokat is végezessen. Az „aiohttp” könyvtár alkalmazása aszinkron HTTP-kérésekhez (RSS-letöltés, TTS API-hívások) ezt a modellt valósítja meg.
- **Retry és exponenciális hátrálás:** Ha egy hálózati kérés sikertelen (például a célszerver átmenetileg nem elérhető), a rendszer automatikusan újrapróbálkozik, de minden egyes sikertelen kísérlet után egyre hosszabb ideig vár (ezt nevezzük exponenciális hátrálásnak). A „tenacity” könyvtár segítségével implementált minta a hálózati hibák kezelésére szolgál (vö. 3.4.3 fejezet).

Az **Operációs rendszerek** tantárgy ismeretei a Docker-konténerizációban (folyamatizoláció, névterek, erőforrás-korlátozás), a fájlrendszeri műveletek kezelésében (OMSZ ZIP-fájlok kicsomagolása, karakterkódolás-felismerés) és az ütemezett folyamatok (APScheduler, cron-jellegű feladatkezelés) megvalósításában kerültek alkalmazásra.

2.3.7. Informatikai védelem és biztonság

Az **Informatikai védelem és biztonság**, az **Innovatív információs és kommunikációs technológiák az IT-biztonság kapcsán** tantárgyak ismeretei a rendszer biztonsági rétegében nyilvánulnak meg:

- **Hitelesítés és jogosultságkezelés:** A kétszatornás (HTTP Basic Auth + JWT Bearer Token) hitelesítési mechanizmus biztosítja, hogy csak jogosult felhasználók és szolgáltatások férjenek hozzá az API-khoz. A háromszintű szerepkör-rendszer (Admin, Editor, Viewer) a feeder modulban és a biztonságos cookie-alapú munkamenet-kezelés (24 órás lejárat, CSRF-védelem (Cross-Site Request Forgery)) az alkalmazás szintű biztonságot szolgálja (vö. 3.7.1 fejezet).
- **SQL-injekció elleni védelem:** Az SQL-injekció egy támadási technika, amelyben a rosszindulatú felhasználó az adatbázis-lekérdezésbe illesztett kóddal próbál jogosulatlan hozzáférést szerezni. A SQLAlchemy ORM parametrikus lekérdezéseinek következetes alkalmazása ezt a kockázatot szünteti meg (vö. 3.7.3 fejezet).
- **Bemeneti validáció:** A Pydantic modellek típusbiztos validációja futásidőben történik, csökkentve a hibás vagy rosszindulatú bemenetek okozta kockázatokat (vö. 2.2.1 fejezet).
- **Tartalombiztonsági szűrés:** A ContentCleaner modul 53 statikus hirdetési mintával végzi a tartalomszűrést, amely a bemeneti adatok integritásának biztosítását szolgálja (vö. 3.6.2 fejezet).
- **Titkosítás és hash-ek:** A jelszavak bcrypt hash-sel kerülnek tárolásra (vö. bcrypt – Wikipedia; Provos, N. & Mazieres, D.: „A Future-Adaptable Password Scheme”, USENIX, 1999; lásd 8.5 Hivatkozások), a SHA-256 alapú tartalom-deduplikáció és az API-kulcsok környezeti változókon keresztüli kezelése (.env fájlok, nem verziókövetve) az érzékeny adatok védelmét szolgálják.

A webalkalmazások elleni leggyakoribb támadási vektorokat az OWASP Top Ten az alábbiak szerint kategorizálja: „A03:2021 – Injection: An application is vulnerable to attack when user-supplied data is not validated, filtered, or sanitized by the application; [...] SQL, NoSQL, OS, and LDAP injection are common.” (vö. OWASP Foundation: „OWASP Top Ten – 2021”, owasp.org, 2021; lásd 8.5 Hivatkozások). A NewsCast rendszer a SQLAlchemy ORM

parametrikus lekérdezéseivel és a Pydantic bemeneti validációval védekezik e támadási vektor ellen (vö. 3.6.5.2 fejezet).

2.3.8. Szoftvertesztelés és Szoftverüzemeltetés

A **Szoftvertesztelés** tantárgy módszertana a rendszer verifikációs stratégiájában jelenik meg: egységtesztek (komponensszintű), integrációs tesztek (modulközi) és a beépített tesztpontok (pl. „test_coco_client()” a newscast-analyze modulban) alkalmazásában.

A szoftvertesztelés réteges stratégiáját Fowler az alábbiak szerint fogalmazza meg: „*The test pyramid is a way of thinking about how different kinds of automated tests should be used to create a balanced portfolio. Its essential point is that you should have many more low-level unit tests than high level broad-stack tests running through a GUI.*” (vö. Fowler, M.: „TestPyramid”, martinowler.com, 2012; lásd 8.5 Hivatkozások).

A szoftvertesztelés fogalmának tágabb értelmezésében a tesztelés nemcsak a szoftver funkcionális helyességének ellenőrzését jelenti, hanem „annak vizsgálatát, hogy a szoftver megfelel-e az elvárásoknak” (vö. Software testing – Wikipedia; lásd 8.5 Hivatkozások). Az elvárás (expectation) fogalma a jelen dolgozatban többretegű: a funkcionális tesztesetek (vö. 3.10.3 fejezet) a modul szintű helyes működést, a teljesítménytesztek (vö. 3.10.4 fejezet) a rendszerszintű áteresztőképességet, az LLM-benchmark (vö. 3.10.5 fejezet) pedig a saját megoldás és az MI-modellek közötti minőségi összehasonlítást ellenőrzi.

A **Szoftverüzemeltetés** tantárgy ismeretei a Docker-konténerizációban, a Prometheus-alapú metrikagyűjtésben (minden modul dedikált „monitoring.py” modullal rendelkezik), a strukturált JSON naplózásban (korrelációs azonosítókkal) és health check végpontok megvalósításában kerültek alkalmazásra.

2.3.9. Felhasználói interfészek és vizualizáció

A **Felhasználói interfészek és vizualizáció** tantárgy ismeretei a rendszer két területén jelennek meg:

- **Webes felület:** A newscast-feeder modul szerver-oldali renderelésű webes felülete – amelyet a Jinja2 sablonmotor állít elő a szerveren és a böngésző kész HTML-oldalakat kap – 7 HTML-sablont tartalmaz (base, login, dashboard, news, settings, users, oam). A rezponzív CSS elrendezés (amely a felület méretét az eszközhöz igazítja, legyen az asztali gép vagy mobiltelefon), a sötét/világos téma támogatás és a JavaScript-alapú

kliens oldali logika (automatikus frissítés, szűrőperzisztálás, modális dialógusok) a felhasználóbarát kezelőfelület kialakítását szolgálja (vö. 3.6.4 fejezet). A felhasználói felületek tervezésének alapelveit Nielsen a tíz heurisztika keretében definiálta, amelyek közül a NewsCast szempontjából a legfontosabb: „Visibility of system status: The design should always keep users informed about what is going on, through appropriate feedback within a reasonable amount of time.” (vö. Nielsen, J.: „10 Usability Heuristics for User Interface Design”, Nielsen Norman Group, 1994; lásd 8.5 Hivatkozások). A newscast-feeder webes felülete ezt az elvet a health check indikátorok, a feldolgozási állapotjelzők és a pontozási progressziósávok valós idejű megjelenítésével valósítja meg.

- **Adatvizualizáció:** Az OAM-elemzés eredményeinek Plotly-alapú vizualizációja (hő térképek, oszlopdiagramok) az objektív elemzési eredmények intuitív, grafikus megjelenítését biztosítja – lehetővé téve, hogy a hírforrások közötti összefüggéseket és eltéréseket vizuálisan is értelmezni lehessen (vö. 3.6.2 fejezet).

2.3.10. Az elektronika fizikai alapjai és Elektronikus áramkörök

Az **elektronika fizikai alapjai** és az **Elektronikus áramkörök** tantárgyak ismeretei közvetetten jelennek meg a rendszerben. A TTS-modul hangkimenetének formátumai (PCM, WAV, MP3) a digitális jelfeldolgozás alapelveire – mintavételezési frekvencia, kvantálási bitszélesség, jelkódolás – építenek. Az audio-adatfolyamok kezelése (pufferelés, streaming válasz a kliensek felé) a jelfolyam-feldolgozás hardverközeli fogalmainak szoftverszintű leképezése. A digitális hangfeldolgozás elméleti alapját a Shannon–Nyquist mintavételi tétel képezi: „*If a function $x(t)$ contains no frequencies higher than W hertz, it is completely determined by giving its ordinates at a series of points spaced $1/(2W)$ seconds apart.*” (vö. Shannon, C. E.: „A Mathematical Theory of Communication”, Bell System Technical Journal, 1949; lásd 8.5 Hivatkozások).

2.3.11. Vállalati gazdaságtan, Vezetési és vállalkozási ismeretek

A **Vállalati gazdaságtan** és a **Vezetési és vállalkozási ismeretek** tantárgyak a költségoptimalizálási döntésekben tükröződnek: az ETag-gyorsítótárazás (~70%-os sávszélesség-megtakarítás), a kötegelt Gemini-feldolgozás (API-költségcsökkentés), a tartalom-hash deduplikáció (felesleges TTS API-hívások elkerülése) és a rádiós előszűrő (az MI-validációra kerülő hírek számának csökkentése) mind az erőforrás-hatékony működést

célozzák. A technológiai szakemberek munkaerőpiaci értékelése kontextust ad a NewsCast által alkalmazott technológiai választásoknak. A Hays Hungary felmérése szerint: „*A Python fejlesztők iránti kereslet 2025-ben is kiemelkedő a magyar IT-piacon; a backend és a data engineering területeken a Python alapú fejlesztői pozíciók a legkeresettebb állások közé tartoznak, bruttó havi 800.000 – 1.400.000 Ft sávban.*” (vö. Hays Hungary: „Salary Guide 2026”, 2026; lásd 8.5 Hivatkozások).

2.3.12. Társadalomtudományi és jogi vonatkozások

A **Komplex társadalomtudományi ismeretek** és az **Európai civilizáció és identitás** tantárgyak a projekt tágabb kontextusának megértésében játszottak szerepet: a magyar digitális hírökoszisztéma felépítésének és sajátosságainak ismerete, a médiaobjektivitás kérdésköre (amelyet az OAM-elemzés technológiai szintre emel), valamint a hírfogyasztási szokások változásai az automatizált médiarendszerek tervezésének társadalmi hátterét adják.

A **jog szerepe a modern társadalmakban** tantárgy az adatvédelmi szempontok (hírportálok tartalmainak gépi feldolgozása, személyes adatok kezelése) és a szerzői jogi kérdések (RSS-alapú tartalom aggregáció jogi keretei) figyelembevételében nyújtott alapot. Az adatvédelmi keretrendszer a személyes adat fogalmát az alábbiak szerint definiálja: „*személyes adat: azonosított vagy azonosítható természetes személyre (érintett) vonatkozó bármely információ; azonosítható az a természetes személy, aki közvetlen vagy közvetett módon, különösen valamely azonosító, például név, szám, helymeghatározó adat, online azonosító vagy a természetes személy testi, fiziológiai, genetikai, szellemi, gazdasági, kulturális vagy szociális azonosságára vonatkozó egy vagy több tényező alapján azonosítható.*” (vö. Az Európai Parlament és a Tanács, 2016, GDPR; lásd 8.5 Hivatkozások). A NewsCast rendszer az RSS-alapú tartalomaggregáció során e definíció tükrében kizárólag nyilvánosan elérhető, nem személyes jellegű adatokat dolgoz fel (vö. 2.2.4 fejezet).

Az **Emberi viselkedés és kommunikáció** tantárgy a felhasználói felület kialakításában (a szerkesztőségi munkafolyamatokhoz igazodó UI) és a rádiós hírszövegek automatizált formázásában (az érthetőség és a hallgatói figyelem fenntartása) szolgáltatott ismereteket.

A **Kultúra, sport, munkahelyi jóllét** tantárgy a fenntartható és ergonomikus szoftverfejlesztési gyakorlatok alkalmazásában, valamint a hírfolyam tematikus kategorizálásának (sport, kultúra rovatbesorolások) kialakításában nyújtott háttérismereteket.

3. Saját fejlesztés

A jelen fejezet a NewsCast rendszer tervezési döntéseit mutatja be, a követelmény-specifikációtól a rendszerarchitektúrán és az adatbázis-sémákon át az API-végpontok tervezéséig. A tervezés során a moduláris felépítés, a hibatűrés és a bővíthetőség elveit tartottam szem előtt.

3.1. Követelmény-specifikáció

A követelmény-specifikáció a rendszer tervezésének kiindulópontja. Az alábbiakban a funkcionális követelmények (3.1.1) – amelyek a rendszer konkrét funkcionalitásait írják le – és a nem funkcionális követelmények (3.1.2) – amelyek a teljesítmény, a megbízhatóság és a biztonság elvárásait rögzítik – kerülnek bemutatásra.

3.1.1. Funkcionális követelmények

A NewsCast rendszer funkcionális követelményeit az alábbiakban specifikálom, modulonként csoportosítva.

3.1.1.1. FR-01: RSS hírgyűjtés

- A rendszernek képesnek kell lennie legalább 50 magyar nyelvű RSS-hírforrás párhuzamos feldolgozására.
- A feldolgozásnak ütemezetten (alapértelmezetten óránként) kell történnie, de manuálisan is indíthatónak kell lennie.
- HTTP gyorsítótárazást (ETag/Last-Modified) kell alkalmazni a sávszélesség csökkentése érdekében.
- A rendszernek kezelnie kell a hibás vagy elérhetetlen feedeket anélkül, hogy a teljes feldolgozás megakadna.
- Duplikációsűrűséget kell végeznie URL cím (hash) és tartalom (hash) alapján.

3.1.1.2. FR-02: Hírelemzés és -osztályozás

- Természetes nyelvfeldolgozást kell végezni magyar nyelven (tokenizáció, NER, szentiment, olvashatóság).

- A tartalombiztonságot több szempont szerint kell értékelni (erőszak, szexuális, drogok, politikai tartalom).
- Duplikációsűrészt kell végrehajtani többretegű algoritmussal (cím-, tartalom- és entitás-alapú).
- A hírekhez rádiós relevancia-pontszámot kell számítani.
- Google Gemini MI-alapú validációt kell biztosítani köteget feldolgozással.
- Reklám- és promóciós tartalom szűrését kell elvégezni.

3.1.1.3. FR-03: OAM elemzés

- A hírforrásokat objektum-attribútum mátrix formában kell elemezni.
- A COCO API-t (<https://miau.my-x.hu/myx-free/coco/>, vö. COCO API; lásd 8.5 Hivatkozások) kell használni a Y0, STD és MCM modellek futtatásához.
- Az elemzési eredményeket vizuálisan kell megjeleníteni (hőtérképek, oszlopdiagramok).
- Az eredményeket Excel-fájlba kell tudni exportálni.

3.1.1.4. FR-04: Időjárás integráció

- Az OMSZ szöveges előrejelzéseit automatikusan le kell tölteni és feldolgozni.
- Három típusú előrejelzést kell kezelni: általános (mai), általános (holnapi) és orvos-meteorológiai.
- A ZIP-tömörített, Windows-1250 kódolású fájlokat automatikusan kell kicsomagolni és dekódolni.
- Tartalomhash alapú duplikációsűrészt kell alkalmazni.

3.1.1.5. FR-05: Hírszelekció és -formázás

- Az 5 leginkább rádiós releváns hírt kell kiválasztani.
- 24 órás duplikációsűrészt kell alkalmazni a korábban kiválasztott hírekre.
- A szöveget a narratíva flow (intro, ai_lead/summary, outro, szünetek) alapján kell formázni.

- Webes felületet kell biztosítani a manuális hírszelekcióhoz és szerkesztéshez.
- Szerepkör-alapú hozzáférés-szabályozást (Admin, Editor, Viewer) kell megvalósítani.

3.1.1.6. FR-06: Közösségi trendjelek gyűjtése és értékelése

- A Google News RSS feedekből (7 kategória: top, nation, world, business, tech, science, health) közösségi jeleket kell gyűjteni 20 perces ciklusokban.
- A Google Trends RSS feedből (geo=HU) a legnépszerűbb kulcsszavakat kell kinyerni 60 perces ciklusokban.
- A közösségi jeleket a meglévő hírekkel háromszintű URL-egyeztetéssel kell párosítani (url_hash, canonical URL, RapidFuzz fuzzy cím-illesztés).
- A Szuper képlettel ($V_H = 10 + Trends * 50$) pontszámot ($\{10, 60\}$) kell számítani.
- A pontszámot az analysis tábla social_trending_score mezőjébe kell írni és a radio_relevance_score-t módosítani.
- Backfill mechanizmust kell biztosítani az analysis sor hiánya esetén (race condition kezelés).

3.1.1.7. FR-07: Szövegfelolvasás (TTS)

- Az ElevenLabs API-val kell hangszintézist végezni.
- Magyar szövegnormalizálást kell alkalmazni (számok, dátumok, rövidítések, mértékegységek).
- Sablonrendszert kell biztosítani (napszak-alapú intro/outro).
- Tartalom-hash deduplikációval kell elkerülni az ismételt API-hívásokat.
- Az eredmény hangfájlokat helyi vagy S3-tárolóban kell elmenteni.
- A feeder szolgáltatásnak callback-et kell küldeni a generálás eredményéről.

3.1.2. Nem funkcionális követelmények

A nem funkcionális követelmények a rendszer minőségi attribútumait határozzák meg. Az alábbiakban öt kategóriában kerülnek ismertetésre: teljesítmény (NFR-01), megbízhatóság (NFR-02), biztonság (NFR-03), üzemeltethetőség (NFR-04) és bővíthetőség (NFR-05).

3.1.2.1. NFR-01: Teljesítmény

- Egy hír elemzése legfeljebb 30 másodperc alatt kell befejeződjön.
- Az RSS-feldolgozási ciklus 50 forrás esetén legfeljebb 10 perc alatt kell befejeződjön.
- A duplikációszűrés 500 cikk esetén 10 másodpercen belül kell lefusson.
- A TTS-generálás egy hírblokk esetén 60 másodpercen belül kell elkészüdjön.

3.1.2.2. NFR-02: Megbízhatóság

- A rendszernek a hálózati hibákat automatikus újrapróbálkozással (retry) kell kezelnie.
- Egyetlen modul meghibásodása nem állíthatja le a teljes rendszert.
- Az adatbázis-kapcsolatok automatikus helyreállítását biztosítani kell (connection pool pre-ping).

3.1.2.3. NFR-03: Biztonság

- Minden API-végpont hitelesítéssel kell rendelkezzen (Basic Auth és/vagy JWT).
- Az API-kulcsokat és jelszavakat környezeti változókból kell betölteni, soha nem a kódba égetve.
- SQL-injekció elleni védelmet kell biztosítani (parametrikus lekérdezések).
- A hitelesítési összehasonlításnak időálló (constant-time) kell lennie a timing attack-ek kivédéséhez.

3.1.2.4. NFR-04: Üzemeltethetőség

- Minden modul Docker-konténerben kell futtatható legyen.
- Prometheus-kompatibilis metrikákat kell szolgáltatni.
- Strukturált (JSON) naplózást kell alkalmazni korrelációs azonosítóval.
- Health check végpontokat kell biztosítani (liveness és readiness).

3.1.2.5. NFR-05: Bővíthetőség

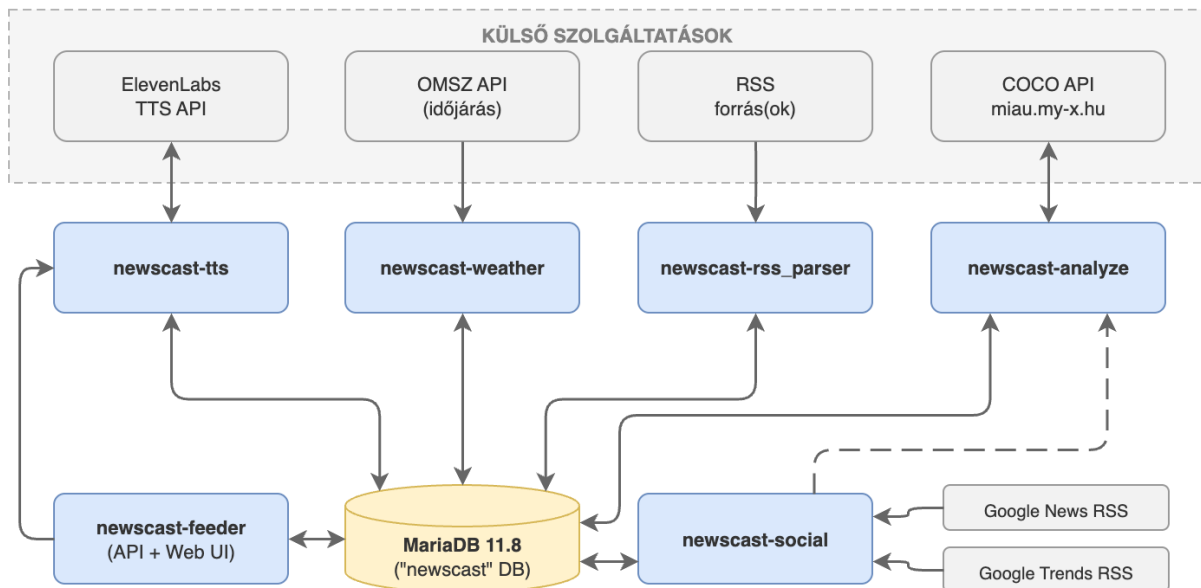
- Az elemzési paramétereket futásidőben, szolgáltatás-újraindítás nélkül kell módosíthatóvá tenni.
- A TTS-sablonokat adatbázisból kell betölteni, nem kódba égetni.
- Az RSS-források listáját adatbázison keresztül kell kezelni.

3.2. Rendszerarchitektúra

A rendszerarchitektúra alfejezet a NewsCast rendszer felépítését mutatja be három szinten: az architektúra áttekintés (3.2.1) a magas szintű rendszerképet és az alkalmazott architektúramintát ismerteti, az adatáramlás leírása (3.2.2) a modulok közötti információáramlás útvonalait, a modulok felelősségi körei (3.2.3) pedig az egyes modulok feladatait foglalják össze.

3.2.1. Architektúra áttekintés

A NewsCast rendszer hat mikroszolgáltatásból és azok közös MariaDB adatbázisából áll. Az alábbi ábra a komponensek közötti kapcsolatokat és az adatáramlás irányát szemlélteti:



1. ábra: A NewsCast rendszer architektúra áttekintése – Forrás: saját szerkesztés

3.2.2. Az adatáramlás leírása

A rendszerben az adatáramlás az alábbi fő útvonalakat követi:

3.2.2.1. Hírgyűjtési útvonal (RSS → DB):

A „newscast-rss_parser” óránkénti ütemezéssel párhuzamosan letölti és feldolgozza a 62 előre felvett RSS feedet. A feldolgozott híreket a „news” táblába menti, az RSS-források metaadatait (ETag, Last-Modified) az „rss” táblában frissíti. A feldolgozás után opcionálisan webhook-hívást küld a „newscast-analyze” szolgáltatásnak.

3.2.2.2. Elemzési útvonal (DB → Analyze → DB):

A „newscast-analyze” a feldolgozatlan híreket (`is_analyzed = FALSE`) a „news” táblából olvassa, elvégzi az NLP-elemzést, a biztonsági és duplikáció-vizsgálatot, majd az eredményeket az „analysis” táblába írja. A rádiós relevancia pontszám és az MI-összefoglaló a „news” tábla megfelelő mezőibe kerül.

3.2.2.3. Közösségi trendjelek útvonala (Google News/Trends → Social → DB):

A „newscast-social” két külső forrást dolgoz fel. A Google Trends RSS feedből (`geo=HU`) 60 perces ciklusokban kinyeri a legnépszerűbb kulcsszavakat és a „trending_keywords” táblába menti. A Google News RSS feedekből (7 kategória) 20 perces ciklusokban közösségi jeleket gyűjt a „social_signals” táblába. Az 5 perces egyeztetési ciklus (`match job`) háromszintű URL-egyeztetéssel (`url_hash`, `canonical URL`, `RapidFuzz` fuzzy cím illesztés) párosítja a jeleket a meglévő hírekkel. A *Szuper képlet* ($V_H = 10 + Trends * 50$) alapján pontszámot (`{10, 60}`) ír az „analysis” tábla „social_trending_score” mezőjébe és módosítja a „radio_relevance_score” értékét. Ha az analysis sor még nem létezik az UPDATE időpontjában, a `news_id` egy pending backfill sorba kerül és a következő `match_job` ciklusban újrapróbálja az írást.

3.2.2.4. Időjárás útvonal (OMSZ → DB → TTS → Audio):

A „newscast-weather” az OMSZ API-járól letölti a ZIP-tömörített előrejelzéseket, dekódolja és „megtisztítja” azokat, majd a „weather” táblába menti. A tartalom-hash gondoskodik arról, hogy ugyanazt az előrejelzést ne tárolja többször. A „newscast-tts” a „newscast-weather” API végpontján keresztül lekérdezi a kiválasztott időjárás előrejelzést és az ElevenLabs API-val hanganyaggá szintetizálja azt. Az eredmény MP3-fájl a helyi tárolóba kerül.

3.2.2.5. Hírszerkesztői útvonal (DB → Feeder → TTS → Audio):

A „newscast-feeder” az „ai_radio_suitable_news” nézetből (`view`) lekérdezi a leginkább rádió-releváns híreket, majd a kiválasztott 5 hírt a „feeder_news” táblába menti. A „newscast-tts” a

feeder API végpontján keresztül lekérdezi a kiválasztott híreket és az ElevenLabs API-val egy hanganyagga szintetizálja azokat. Az eredmény MP3-fájl a helyi tárolóba kerül.

3.2.2.6. OAM elemzési útvonal (DB → Analyze → COCO API → DB):

Az OAM elemzés az „analysis” tábla adataiból épít objektum-attribútum mátrixot, amelyet a távoli COCO API-nak küld feldolgozásra. Az eredményeket az „oam_snapshots” és „oam_antagonisms” táblákba menti, a vizualizációt Plotly könyvtárral generálja.

3.2.3. A modulok felelősségi körei

Modul	Fő felelősség	Bemenetek	Kimenetek
rss_parser	Hírgyűjtés	RSS feedek (HTTP)	news tábla rekordok
analyze	Hírelemzés, OAM	news tábla, COCO API	analysis, oam_snapshots
social	Közösségi trendjelek	Google News RSS, Google Trends RSS	social_signals, trending_keywords, analysis.social_trending_score
weather	Időjárás feldolgozás	OMSZ ZIP fájlok	weather tábla
feeder	Hírszelekció, UI	ai_radio_suitable_news view	feeder_news, Web UI
tts	Szövegfelolvasás	feeder API, weather API, ElevenLabs	MP3 hangfájlok

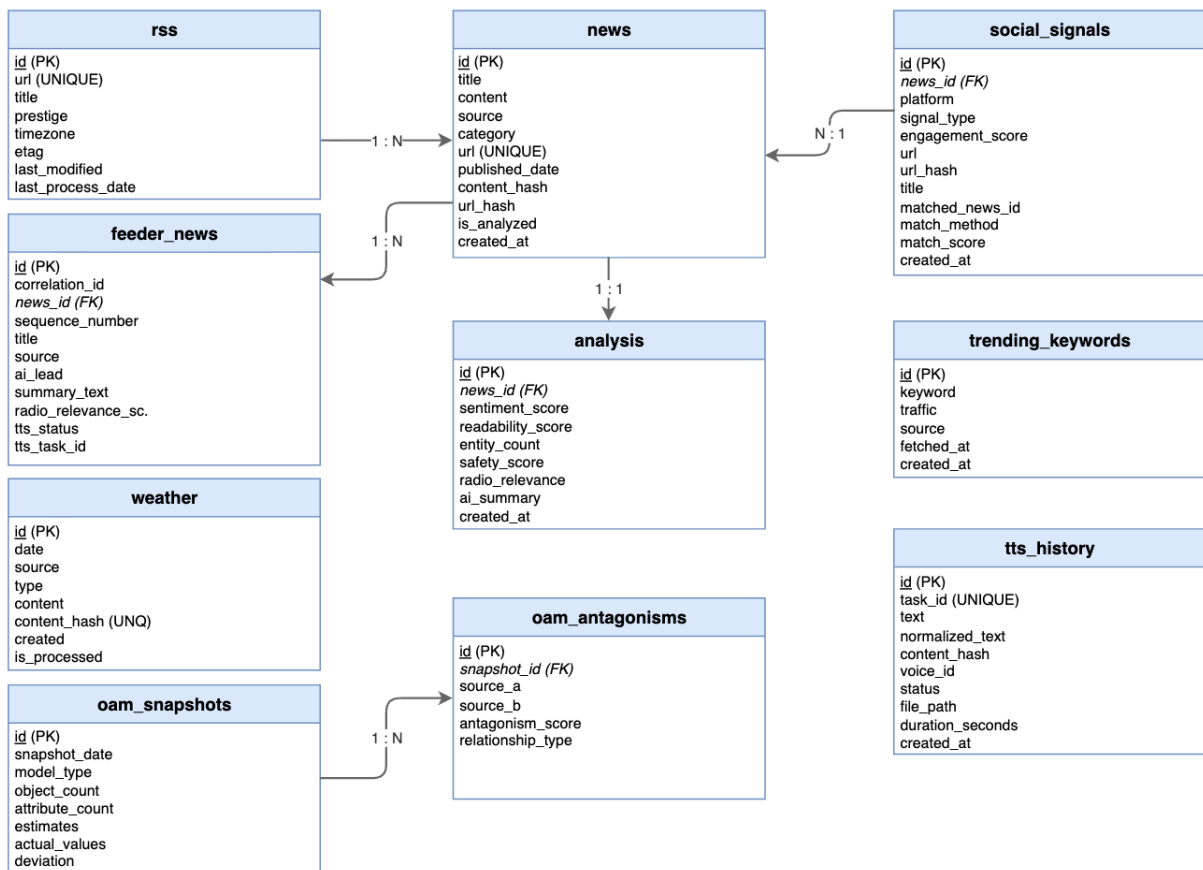
2. táblázat: A NewsCast modulok felelősségi körei

3.3. Adatbázis terv

Az adatbázis terv alfejezet a NewsCast rendszer relációs adatmodelljét mutatja be. Az entitás-kapcsolat diagram (3.3.1) a táblák közötti kapcsolatokat szemlélteti, a legfontosabb táblák részletes leírása (3.3.2) az egyes táblák mezőit és kényszereit ismerteti, az adatbázis nézetek (3.3.3) pedig a lekérdezés-optimalizálásra szolgáló view-kat mutatják be.

3.3.1. Entitás-kapcsolat diagram

A NewsCast rendszer adatbázisa összesen 18 egyedi táblát tartalmaz, amelyek hat logikai csoportba sorolhatók (az „rss” és „news” táblákat az rss_parser és az analyze modul közösen használja, az „analysis” táblát az analyze és a social modul közösen használja). Az alábbi ER diagram a legfontosabb entitásokat és kapcsolataikat ábrázolja:



2. ábra: A NewsCast rendszer egyszerűsített ER diagramja – Forrás: saját szerkesztés

3.3.2. A legfontosabb táblák részletes leírása

A rendszer relációs adatmodellje több táblát tartalmaz. Az alábbiakban a legfontosabb táblák szerkezete kerül bemutatásra: az „rss” tábla (3.3.2.1) a hírforrások metaadatait, a „news” tábla (3.3.2.2) a letöltött híreket, a „weather” tábla (3.3.2.3) az időjárási adatokat, a „feeder_news” tábla (3.3.2.4) a kiválasztott híreket, a „tts_history” tábla (3.3.2.5) a szövegfelolvasási előzményeket, a „social_signals” tábla (3.3.2.6) a közösségi trendjeleket és a „trending_keywords” tábla (3.3.2.7) a trending kulcsszavakat tárolja.

3.3.2.1. „rss” tábla (hírforrások)

Az „rss” tábla tartalmazza az összes konfigurált RSS forrást. A tábla az „init.sql” fájlban kerül létrehozásra az „newscast-rss_parser” modulban, 62 előre konfigurált magyar hírforrással.

Mező	Típus	Leírás
id	INT AUTO_INCREMENT	Elsődleges kulcs

Mező	Típus	Leírás
url	VARCHAR(2000) UNIQUE	Az RSS feed URL címe
title	VARCHAR(255)	A forrás neve (pl. „Telex”)
prestige	FLOAT (0.0-1.0)	Presztízserő a rangsoroláshoz
timezone	VARCHAR(50)	Időzóna offset (pl. „+01:00”)
etag	VARCHAR(100) NULL	Az utolsó sikeres lekérés ETag HTTP fejléce
last_modified	VARCHAR(100) NULL	Az utolsó sikeres lekérés Last-Modified HTTP fejléce
last_process_date	DATETIME	Az utolsó feldolgozás időpontja

3. táblázat: NewsCast adatbázis „rss” tábla (hírforrások)

A „prestige” mező kiemelten fontos: ez az érték a rádiós relevancia-számítás bemenetét szolgál az analyze modulban. A presztízserő értékek a forrás ismertségén, megbízhatóságán és hírértékén alapulnak. Például a Telex (0,85), az Index (0,80) és a HVG (0,80) a legmagasabb értékkel rendelkeznek, míg a szűkebb közönséget elérő portálok (PC Guru: 0,50, Demokrata: 0,50) alacsonyabb értéket kaptak.

3.3.2.2. „news” tábla (hírek / cikkek)

A „news” tábla a rendszer központi adattárolója, amely az összes letöltött hírt és/vagy cikket tartalmazza. A tábla 19 mezőből áll, amelyek négy logikai csoportba sorolhatók:

- **Alap tartalom:** title, content, source, category, url, published_date
- **Bővített metaadatok:** lead_text, extracted_title, authors, top_image, keywords
- **Feldolgozási állapot:** is_analyzed, analyzed_date, analysis_status, content_hash, url_hash
- **Időbélyegek:** created_at, updated_at

Az indexelési stratégia a leggyakoribb lekérdezéseket optimalizálja:

- „unique_url(url(255))” → Duplikáció-megelőzés URL alapján.
- „idx_analyzed(is_analyzed)” → Feldolgozatlan hírek gyors szűrése.
- „idx_published_date(published_date)” → Frissesség szerinti rendezés.

- „idx_source_category(source, category)” → Forrás- és kategória-alapú szűrés.

3.3.2.3. „weather” tábla (időjárás adatok)

A „weather” tábla az OMSZ előrejelzéseit tárolja. A tábla meghatározó mezője a „content_hash” (VARCHAR(64), UNIQUE), amely SHA-256 hash formájában garantálja, hogy azonos tartalmú előrejelzés ne kerüljön duplikáltan az adatbázisba. A „type” mező három lehetséges értéket vehet fel: „general” (mai előrejelzés), „general_tomorrow” (holnapi előrejelzés) és „medical” (orvos-meteorológiai előrejelzés).

3.3.2.4. „feeder_news” tábla (kiválasztott hírek)

A „feeder_news” tábla a rádiós hírblokkok tartalmát rögzíti. Minden hírblokk egy egyedi „correlation_id” (UUID) értékkel azonosított, amely összeköti az adott blokkhoz tartozó híreket. A „sequence_number” mező a hírek sorrendjét jelöli a blokkon belül, a „tts_status” pedig a hangszintézis (TTS) állapotát követi a következő státuszokkal: pending, sent, completed, failed.

3.3.2.5. „tts_history” tábla (TTS napló)

A „tts_history” tábla a szövegfelolvasási kérések teljes auditálási nyomvonalát tartalmazza. A „content_hash” és „external_content_hash” mezők lehetővé teszik a korábban generált hanganyagok újrafelhasználását (deduplikáció). A „was_reused” logikai mező jelzi, ha egy korábbi generálás eredménye került újrahasznosításra. A „processing_time_ms” mező a generálás időtartamát rögzíti milliszekundum pontossággal.

3.3.2.6. „social_signals” tábla (közösségi trendek)

A „social_signals” tábla a newscast-social modul által gyűjtött közösségi médiajeleket tárolja. Minden rekord egy Google News RSS bejegyzést reprezentál, amelyet a *collect* ütemezett job ír be. A tábla mezői:

Mező	Típus	Leírás
id	INT AUTO_INCREMENT	Elsődleges kulcs
platform	ENUM('google_news') NOT NULL	Forrásplatform
external_id	VARCHAR(255)	Platform-specifikus azonosító
url	VARCHAR(2000)	Az eredeti hír URL címe

Mező	Típus	Leírás
url_hash	VARCHAR(64)	Az URL SHA-256 hash értéke
title	VARCHAR(500)	A hír címe
snippet	TEXT	Rövid előnézet / leírás
published_at	DATETIME	A bejegyzés eredeti publikálási időpontja
fetched_at	DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP	A lekérdezés időpontja
engagement_score	FLOAT DEFAULT 0	Engagement pontszám (0–100)
raw_score	INT DEFAULT 0	Platformmetrika (szavazatok / rangsorpozíció)
comment_count	INT DEFAULT 0	Kommentek száma
share_count	INT DEFAULT 0	Megosztások száma
trending_rank	INT	Pozíció a trendlistában (1 = legfelső)
trending_window_h	INT DEFAULT 24	A trend-ablak időtartama órában
related_urls	JSON DEFAULT NULL	Az RSS <description> klaszter-linkjeiből kinyert, feloldott cikk-URL-ek
matched_news_id	INT	Az egyeztetett news rekord azonosítója (FK → news.id); NULL amíg a matcher nem fut
match_confidence	FLOAT	Egyeztetési bizonyosság (0.0–1.0 hasonlósági pontszám)
match_method	VARCHAR(50)	Az egyeztetési módszer (url_exact, related_url_exact, url_canonical, related_url_canonical, title_fuzzy, no_match)
is_processed	BOOLEAN NOT NULL DEFAULT FALSE	TRUE, miután a matcher már megkísérelte az egyeztetést
created_at	DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP	Létrehozás időpontja

4. táblázat: NewsCast adatbázis „social_signals” tábla (közösségi trendek)

A „social_signals” tábla öt indexet definiál: „idx_url_hash” (url_hash) az egyeztetéshez, „idx_platform_fetched” (platform, fetched_at DESC) a platformszűréshez, „idx_matched_news” (matched_news_id) a visszakereséshez, „idx_is_processed” (is_processed) a feldolgozatlan rekordok gyors lekéréséhez és „idx_trending_rank” (platform, trending_rank, fetched_at DESC) a trend-lista lekérdezésekhez.

3.3.2.7. „trending_keywords” tábla (trending kulcsszavak)

A „trending_keywords” tábla a Google Trends RSS feed-ből (geo=HU) kinyert népszerű kulcsszavakat tárolja. A trends job 60 perces ciklusokban frissíti a tartalmat.

Mező	Típus	Leírás
id	INT AUTO_INCREMENT	Elsődleges kulcs
keyword	VARCHAR(255)	A trending kulcsszó
traffic	INT	Becsült forgalom (Google Trends által jelzett érték)
source	VARCHAR(50)	Forrás (jelenleg csak „google_trends”)
fetched_at	DATETIME	Lekérdezés időpontja
created_at	DATETIME	Létrehozás időpontja

5. táblázat: NewsCast adatbázis „trending_keywords” tábla (trending kulcsszavak)

A match job a „trending_keywords” tartalmát in-memory cache-ből olvassa és substring-illesztéssel keresi az egyezéseket a hírek címeiben, figyelembe véve a magyar nyelv ragozási sajátosságait is (→ TrendingStore komponens).

3.3.3. Adatbázis nézetek (views)

A rendszer egy kiemelten fontos adatbázisnézetet definiál: **ai_radio_suitable_news**

Ez a nézet a „news” és az „analysis” táblák összefűzéséből áll és azokat a híreket tartalmazza, amelyek az elemzés eredménye alapján alkalmasak rádiós felolvasásra. A nézet szűrési feltételei között szerepel a tartalombiztonsági küszöbérték, az elemzési állapot (analyzed = TRUE) és a minimális rádiós relevancia-pontszám. A „newscast-feeder” modul „fetch_radio_suitable_news” függvénye (lásd „feeder.py”) kizárólag ezen a nézeten keresztül kérdezi le a híreket.

3.4. API végpontok és kommunikáció

A jelen alfejezet a NewsCast rendszer API-rétegét mutatja be: az API tervezési elvek (3.4.1), a modulonkénti API áttekintés (3.4.2), a szolgáltatásközi kommunikáció útvonalai (3.4.3) és a hitelesítési architektúra (3.4.4) kerülnek ismertetésre.

3.4.1. API tervezési elvek

A NewsCast API végpontjai az alábbi egységes tervezési elvek szerint készültek:

- **RESTful konvenciók:** Az erőforrás-orientált URL-struktúra (pl. „/api/v1/tts/tasks/{task_id}”), a HTTP-metódusok szemantikus használata (GET: lekérdezés, POST: létrehozás, DELETE: törlés).
- **Egységes hitelesítés:** Minden védett végpont elfogadja a Basic Auth és a JWT (Bearer Token) hitelesítést is.
- **Egységes hibaformátum:** A hibaválaszok JSON-formátumban, HTTP-státuszkódokkal együtt kerülnek visszaadásra.
- **Health check végpontok:** A modulok health végpontokat szolgáltatnak: az rss_parser, weather és feeder modul a „/health/live” és „/health/ready” végpontokat, az analyze modul a „/status” végpontot, a TTS modul pedig az „/api/v1/health” végpontot használja.

3.4.2. Modulonkénti API áttekintés

Az alábbi alfejezet a NewsCast rendszer valamennyi moduljának API végpontjait mutatja be. Az áttekintés modulonként halad az adatáramlás logikai sorrendjét követve: az RSS-gyűjtéstől (newscast-rss_parser) az elemzésen (newscast-analyze) és az időjárás feldolgozáson (newscast-weather) át a hírszelekciós felületig (newscast-feeder), majd a szövegfelolvasásig (newscast-tts) és végül a közösségi trendjel gyűjtésig (newscast-social). Minden modulnál feltüntetésre kerül a végpont elérési útja, a HTTP-metódus, a hitelesítési követelmény és a funkcióleírás. A végpontok összefoglaló táblázatai a mellékletekben is megtalálhatók (vö. 6. fejezet). Az API tervezési elvek egységes alkalmazását a 3.4.1 alfejezet részletezi.

3.4.2.1. newscast-rss_parser API (Port: 8080)

Végpont	Metódus	Hitelesítés	Leírás
„/process_rss”	POST	Igen	RSS feldolgozás manuális indítása
„/health/live”	GET	Nem	Liveness próba
„/health/ready”	GET	Nem	Readiness próba (DB)

6. táblázat: newscast-rss_parser API végpontok

3.4.2.2. newscast-analyze API (Port: 8080)

Az analyze modul rendelkezik a legtöbb API végponttal: 17 a hírek elemzéséhez + 12 az OAM funkciókhoz). Ezek közül a legfontosabbak:

Végpont	Metódus	Leírás
„/analyze/{news_id}”	POST	Egyedi hír elemzése
„/news”	GET	Hírek listázása (szűrőkkel)
„/status”	GET	Rendszerállapot
„/mainstream-news”	GET	Mainstream hírek
„/oam/analyze”	POST	OAM elemzés indítása
„/oam/snapshot/{id}”	GET	OAM eredmények lekérdezése
„/oam/viewer”	GET	OAM webes megjelenítő
„/oam/snapshot/{id}/export/{format}”	GET	OAM export (csv, json, xlsx)
„/params”	GET	Dinamikus elemzési paraméterek lekérdezése
„/params/{param_name}”	PUT	Elemzési paraméter módosítása

7. táblázat: newscast-analyze API végpontok

3.4.2.3. newscast-weather API (Port: 80)

Végpont	Metódus	Hitelesítés	Leírás
„/weather/today”	GET	Igen	Mai előrejelzés (feldolgozott)
„/weather/today-combined”	GET	Igen	Kombinált napi előrejelzés
„/weather/today/raw”	GET	Igen	Mai előrejelzés (nyers szöveg)
„/sync/trigger”	GET	Igen	OMSZ szinkronizáció manuális indítása
„/sync/status”	GET	Igen	OMSZ szinkronizáció állapota

8. táblázat: newscast-weather API végpontok

3.4.2.4. newscast-social API (Port: 8080)

A social modul 12 REST API végpontot publikál, amelyek a közösségi trendjelek állapotának lekérdezését és a manuális triggerelést teszik lehetővé:

Végpont	Metódus	Leírás
„/health”	GET	Health check
„/status”	GET	Rendszerállapot (scheduler, pending backfill, gyűjtési statisztikák)
„/stats”	GET	Részletes statisztikák (signal és match számok)

Végpont	Metódus	Leírás
„/trends”	GET	Aktuális trending kulcsszavak listája
„/trends/trigger”	POST	Trends job manuális indítása
„/formula/check”	GET/POST	Szuper képlet ellenőrzése adott hír címére
„/top-trending”	GET	Legmagasabb social_trending_score-ral rendelkező hírek
„/signals”	GET	Legutóbbi social_signals rekordok
„/collect/trigger”	POST	Collect job manuális indítása
„/match/trigger”	POST	Match job manuális indítása

9. táblázat: newscast-social API végpontok

3.4.2.5. newscast-feeder API (Port: 80)

Végpont	Metódus	Leírás
„/feeder/trigger”	POST	Automatikus hírblokk generálás
„/api/v1/selected-news”	GET	Legutóbbi kiválasztott hírek
„/api/v1/selected-news/{correlation_id}”	GET	Adott hírblokk hírei
„/api/v1/tts-callback”	POST	TTS visszajelzés fogadása
„/ui/”	GET	Dashboard felület (Web UI)
„/ui/news”	GET	Hírkiválasztó felület (Web UI)
„/ui/settings”	GET	Beállítások felület (Web UI)
„/ui/api/ui/manual-feed”	POST	Manuális hírkiválasztás (Web UI)
„/ui/api/ui/trigger-feed”	POST	Automatikus hírkiválasztás (Web UI)

10. táblázat: newscast-feeder API végpontok

3.4.2.6. newscast-tts API (Port: 80)

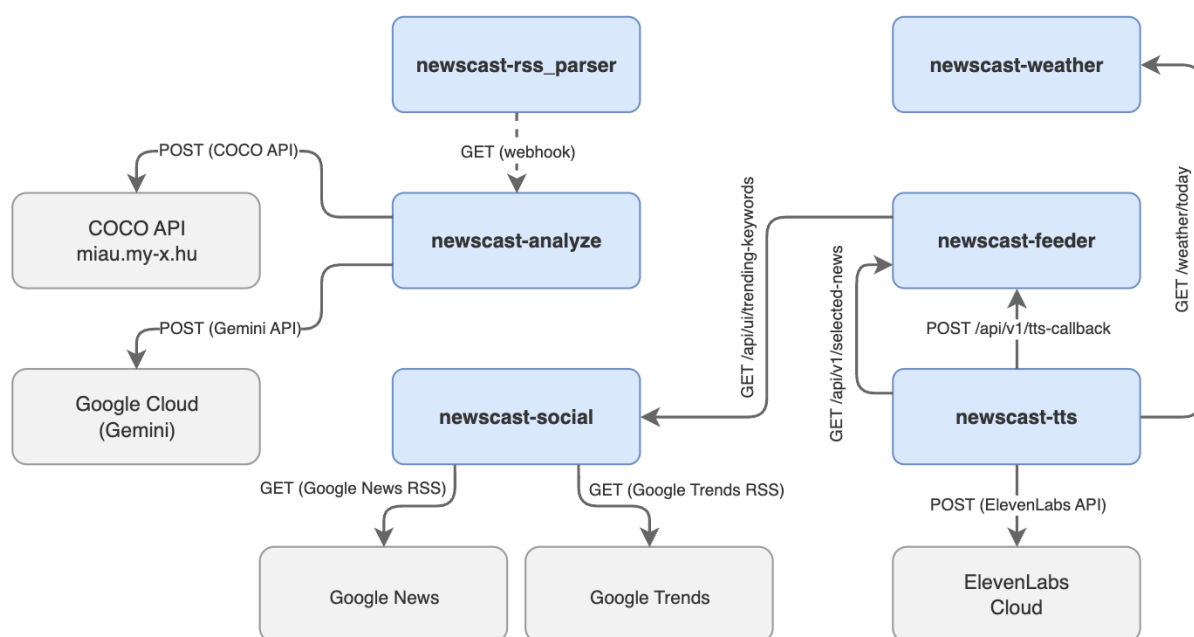
Végpont	Metódus	Leírás
„/api/v1/tts/generate”	POST	TTS feladat létrehozása
„/api/v1/tts/tasks”	GET	Feladatok listázása
„/api/v1/tts/tasks/{id}”	GET	Feladat állapota
„/api/v1/tts/tasks/{id}/audio”	GET	Hangfájl letöltése

Végpont	Metódus	Leírás
„/api/v1/templates”	GET	Sablonok listázása
„/api/v1/templates/render”	POST	Sablon előnézet
„/api/v1/scheduler/jobs”	GET/POST	Ütemezett feladatok

11. táblázat: newscast-tts API végpontok

3.4.3. Szolgáltatásközi kommunikáció

A modulok közötti kommunikáció szinkron HTTP hívásokkal valósul meg. Az alábbi ábra a fő kommunikációs útvonalakat mutatja:



3. ábra: NewsCast modulok közötti kommunikáció – Forrás: saját szerkesztés

Az rss_parser, weather és feeder modulok a „tenacity” könyvtár retry mechanizmusát alkalmazzák, exponential backoff megoldással. A TTS modul timeout hibakezéssel dolgozik, a retry paraméterek pedig modulonként konfigurálhatók:

Modul	Próbálkozások	Min. várakozás	Max. várakozás
rss_parser	3	5s	30s
analyze	-	-	- (Gemini: rate limit wait)
social	3 (HTTP) / backfill (next cycle)	5s	30s
weather	3 (HTTP) / 5 (DB)	1s / 1s	10s / 30s

Modul	Próbálkozások	Min. várakozás	Max. várakozás
feeder	5 (DB)	1s	30s
tts	-	-	- (timeout: 60s)

12. táblázat: A retry mechanizmus paramétereit modulonként

3.4.4. Hitelesítési architektúra

A NewsCast rendszer egységesen, kétcsatornás hitelesítést alkalmaz, amelyet minden modul az „auth.py” fájlban implementál:

3.4.4.1. HTTP Basic Auth:

- A felhasználónév és jelszó a kérés „Authorization: Basic” fejlécében kerül átadásra.
- A hitelesítő adatok összehasonlítása a Python „secrets.compare_digest()” függvénnyel történik, amely időállandó (constant-time) összehasonlítást végez, megakadályozva a timing attack-eket.
- A hitelesítő adatok az „.env” fájl „BASIC_AUTH_USERNAME” és „BASIC_AUTH_PASSWORD” környezeti változóiból kerülnek betöltésre.

3.4.4.2. JWT Bearer Token:

- „Authorization: Bearer” fejléc formátumú hitelesítés, HS256 algoritmussal.
- A token „exp” (lejárati idő) claim-je kerül ellenőrzésre, az alapértelmezett érvényesség 1 óra.
- A titkos kulcs az „.env” fájl „JWT_SECRET_KEY” környezeti változójából kerül betöltésre.

3.4.4.3. Webes session alapú hitelesítés (csak a feeder Web UI):

- A „newscast-feeder” webes felülete session alapú hitelesítést alkalmaz, ahol a session token cookie-ként kerül tárolásra.
- Minden session 24 órás lejárattal rendelkezik.
- A jelszavak bcrypt hash-sel vannak tárolva a „feeder_users” táblában.
- A felhasználók három szerepkörbe sorolhatók: Admin, Editor, Viewer.

A jelen fejezet a NewsCast rendszer implementációjának részletes, kódszintű bemutatását tartalmazza. A fejezet modulonként halad végig a rendszeren, az adatáramlás logikai sorrendjét követve: RSS gyűjtés → elemzés → időjárás → szelekció → szövegfelolvasás.

3.5. Fejlesztői környezet

A fejlesztői környezet alfejezet a rendszer megvalósításához alkalmazott technológiai platformot (3.5.1), a projekt könyvtárstruktúráját (3.5.2) és a futtató (éles üzemeltetési) környezet paramétereit (3.5.3) ismerteti.

3.5.1. Technológiai platform

A fejlesztés során az alábbi eszközöket és technológiákat alkalmaztam:

- **Programozási nyelv:** Python 3.9 (rss_parser, analyze, weather, feeder) és Python 3.11 (tts), a fejlesztési környezetben Python 3.13.7
- **Web keretrendszer:** FastAPI 0.68.1 (rss_parser) / 0.104.1 (tts) / 0.110.3 (analyze, feeder, weather)
- **ASGI szerver:** Uvicorn
- **Adatbázis:** MariaDB 11.8.3 (InnoDB motor, utf8mb4 karakterkészlet, MySQL-kompatibilis protokoll)
- **ORM:** SQLAlchemy 1.4.23 (rss_parser) / 2.0.23 (tts) / 2.0.31 (analyze, feeder, weather)
- **Verziókezelés:** Git, GitHub („<https://github.com/varadiv/newscast>”)
- **Konténerizáció:** Docker (python:3.9-slim / python:3.11-slim alapképek), Docker Desktop 4.48.0 (build, tag, push)
- **IDE:** Visual Studio Code 1.109.0
- **Csomagkezelés:** pip, requirements.txt
- **Fejlesztési platform:** Apple MacBook Pro, macOS 26.4.1

3.5.2. Projekt struktúra

A NewsCast „monorepo” struktúrában szervezett, ahol minden modul önálló könyvtárban található.

Ez az egységes struktúra gondoskodik arról, hogy a fejlesztő bármelyik modult azonos mentális modellel közelíthesse meg: a „main.py” az alkalmazás belépési pontja, a fő üzleti logika az elsődleges modulban van (pl. „analyzer.py”, „feeder.py”, „tts_service.py”), az „auth.py” a hitelesítést, a „database.py” az adatmodellt tartalmazza.

3.5.3. Futtató környezet

A NewsCast rendszer éles üzemeltetése egy dedikált virtuális privát szerveren (VPS) történik, amelyet a RackForest.hu magyar hosting-szolgáltató biztosít.

Paraméter	Érték
Szolgáltató	RackForest.hu
Típus	Linux VPS
CPU	2 vCPU
RAM	4 GB
Tárhely	60 GB SSD
Hostnév	server1.newscast.hu
IP-cím	46.29.138.242
Operációs rendszer	Debian GNU/Linux 12 (Bookworm)

13. táblázat: A NewsCast rendszert futtató VPS paramétereit

A konténer futtatását és kezelését az **1Panel** (v1.10.34-lts) nyílt forráskódú szerverpanel biztosítja. Az 1Panel egy modern, Go nyelven írt szerverpanel, amely webes felületen keresztül teszi lehetővé a Docker-konténer, adatbázisok, webszerverek és egyéb szolgáltatások központi kezelését. Az 1Panel Docker Compose alapú konténerkezelése révén a hat NewsCast mikroszolgáltatás, az adatbázis és a monitorozási infrastruktúra egyetlen felületen adminisztrálható.

A szerveren futó platform komponensek:

Komponens	Verzió	Funkció
MariaDB	11.8.3	Relációs adatbázis-szerver (MySQL-kompatibilis)
phpMyAdmin	5.2.3	Webes adatbázis-adminisztrációs felület
Docker Registry	3.0.0	Privát Docker image-tár a konténerképek tárolásához
OpenResty	1.21.4.3	Nginx-alapú reverse proxy és webszerver

Komponens	Verzió	Funkció
Grafana	12.2.0	Monitorozási és vizualizációs dashboard
Prometheus	3.6.0	Metrikagyűjtő és -tároló rendszer

14. táblázat: A NewsCast rendszer infrastrukturális platform komponensei

Az **OpenResty** egy Nginx-re épülő, Lua-szkriptezéssel bővített webservert és reverse proxy platform, amely a NewsCast rendszerben a bejövő HTTP-kérések elosztásáért és az SSL/TLS-termináció biztosításáért felel.

A **privát Docker Registry** a konténerképek verziókezelését és elosztását szolgálja. A fejlesztési gépen elkészített konténerképek („docker build”, „docker tag”, „docker push”) a privát registry-be kerülnek feltöltésre, ahonnan a szerver az IPanel segítségével húzza le és indítja el azokat.

A **Grafana** és a **Prometheus** együttesen felelnek a rendszer teljes körű monitorozásáért. A Prometheus a modulok 8000-es portján közzétett metrikákat gyűjti periodikusan (scrape), míg a Grafana a Prometheus-adatforrásra épülő vizuális dashboardokat szolgáltat, amelyeken a rendszer teljesítménye, az API-válaszidők, a feldolgozott hírek száma és az erőforrás-kihasználtság valós időben követhető.

A fejlesztési munkafolyamat során a forráskód a GitHub „newscast” repository-ban („<https://github.com/varadiv/newscast>”) kerül verziókezelésre. A Docker Desktop 4.48.0 segítségével a fejlesztői gépemen (Apple MacBook Pro, macOS 26.4.1) elkészített konténerképek a privát registry-n keresztül jutnak el a VPS szerverre.

3.6. Backend modulok megvalósítása

A futató környezet és az infrastrukturális komponensek bemutatása után a jelen alfejezet a NewsCast rendszer hat backend moduljának részletes, kódszintű implementációját ismerteti. Az alfejezetek az adatáramlás logikai sorrendjét követik: az RSS-hírgyűjtéstől (3.6.1) az NLP-alapú elemzésen (3.6.2) és az időjárás-feldolgozáson (3.6.3) át a hírszelekción felületig (3.6.4), majd a szövegfelolvasásig (3.6.5) és a közösségi trendjelek gyűjtéséig (3.6.6). Minden modulnál bemutatásra kerül az architektúra, a legfontosabb algoritmusok és a konfigurációs lehetőségek.

3.6.1. newscast-rss_parser: RSS hírgyűjtő modul

Az RSS Parser modul felelős a 62 előre konfigurált magyar hírforrás párhuzamos letöltéséért, feldolgozásáért és az adatbázisba mentéséért. A modul négy fő Python fájlból áll: „rss_parser.py” (feed letöltés és elemzés), „news_manager.py” (mentés és duplikációszűrés), „scheduler.py” (ütemezés és párhuzamos feldolgozás) és „api.py” (FastAPI végpontok). A Python-alapú webes adatgyűjtés módszertana (vö. Farkas D., 2023; lásd 8.5 Hivatkozások) a jelen rendszer RSS feldolgozó megoldásának tervezésekor is hivatkozási alapul szolgált.

3.6.1.1. Az RSS letöltés és HTTP gyorsítótárzás

A hírforrások letöltését az „rss_parser.py” modul „fetch_rss” függvénye végzi. A függvény az alábbi működési logikát követi:

A HTTP gyorsítótárzás az RFC 7232 szabvány (vö. Fielding, R. T. & Reschke, J. (2014): „*RFC 7232: Hypertext Transfer Protocol (HTTP/1.1): Conditional Requests*”, IETF.; lásd 8.5 Hivatkozások) szerinti feltételes kérésekkel (Conditional Requests) működik. Amikor a szerver az ETag és/vagy Last-Modified fejléceket adja vissza, a következő kérésnél ezeket az „If-None-Match” és „If-Modified-Since” fejlécekben küldi el a rendszer. Ha a tartalom nem változott, a szerver HTTP 304 (Not Modified) válasszal jelzi ezt és a tartalom nem kerül letöltésre. Ez a mechanizmus körülbelül 70%-os sávszélesség-megtakarítást eredményez, mivel a hírforrások jelentős része óránkénti lekérdezésnél nem változik.

A függvény emellett kezeli a 429-es (Rate Limit) válaszkódot is, 30 másodperces várakozással, biztosítva, hogy a rendszer ne sértse meg a hírportálok hozzáférési korlátait.

3.6.1.2. Az RSS elemzés és időzóna kezelés

Az RSS feedek feldolgozását a „parse_rss” függvény végzi, amely a „feedparser” könyvtárat használja a tartalom struktúrált kinyeréséhez. Minden bejegyzésből az alábbi mezők kerülnek kinyerésre: cím (title), tartalom (description), forrás (feed.title), kategória (category), publikálási dátum és URL.

Az időzóna kezelés különös figyelmet igényelt, mivel a magyar hírforrások egy része CET (+01:00) időzónában, más része UTC (+00:00) időzónában publikálja a dátumokat. Az „init.sql” fájlban minden forrashoz rögzítve van a „timezone” mező értéke és a „parse_timezone_offset” függvény reguláris kifejezéssel konvertálja ezt „timedelta” objektummá.

3.6.1.3. Duplikációszűrés és adatmentés

A „news_manager.py” modul felelős az adatok adatbázisba mentéséért és a duplikáció megelőzéséért. A duplikációszűrés két rétegben működik:

1. **URL alapú szűrés:** A „normalize_url” függvény egységes formára alakítja az URL-eket (kisbetűs séma és hostnév, nem szükséges paraméterek eltávolítása), majd a „compute_sha256” függvénnyel hash-t generál. A „save_news” függvény a mentés előtt ellenőrzi, hogy az adott URL már létezik-e az adatbázisban.
2. **Tartalom hash szűrés:** A „normalize_text” függvény előfeldolgozza a szöveget (HTML eltávolítás, kisbetűsítés, speciális karakterek törlése), majd SHA-256 hash-t számít belőle. Ez a hash a „content_hash” mezőben tárolódik.

3.6.1.4. Párhuzamos feldolgozás

A „scheduler.py” modul az „asyncio” modullal párhuzamos feldolgozást végez. A „MAX_CONCURRENT_FEEDS” paraméter (a „config.py”-ban definiálva, környezeti változóból konfigurálható) korlátozza az egyidejűleg futó RSS feed letöltések számát, megakadályozva a célszerverek túlterhelését és a rendszer erőforrásainak teljes kihasználását. Az APScheduler könyvtár felelős az óránkénti ütemezett futtatásért.

3.6.2. newscast-analyze: Hírelemzés és OAM modul

Az analyze modul a rendszer legintelligensebb és legösszetettebb komponense. 2100+ sornyi FastAPI alkalmazást („main.py”), egy több mint 3300 soros elemzési motort („analyzer.py”), OAM elemzési képességet („oam_analyzer.py”, „oam_export.py”, „oam_routes.py”) és egy távoli COCO API klienst („coco_client.py”) tartalmaz.

3.6.2.1. Az elemzési pipeline (UnifiedAnalyzer)

Az elemzés központi osztálya a „UnifiedAnalyzer” (lásd „analyzer.py”), amely hét speciális alelemzőt koordinál.

Az „analyze_news” metódus az elemzés fő belépési pontja, amely a következő lépéseket hajtja végre:

1. **Tartalomkinyerés:** Ha a tartalom HTML-formátumban érkezik, a „extract_text_from_html” metódus a BeautifulSoup könyvtárral kinyeri a szöveges tartalmat. A metódus először eltávolítja a nem tartalmi elemeket (script, style, nav,

footer, aside, header), majd a tartalmi szelektorok prioritásos listáján (article, main, .article-body, .content) keresi a fő szöveget.

2. **Reklámszűrés:** A „ContentCleaner” osztály négylépéses eljárással távolítja el a hirdetési és promóciós tartalmakat.

A reklámszűrő összesen 53 statikus hirdetési mintát tartalmaz három kategóriában: 12 hirdetésjelölő minta („DEFAULT_AD_MARKERS”, pl. „[hirdetés]”, „[reklám]”, „[szponzorált tartalom]”), 23 hirdetési kulcsszó („DEFAULT_AD_KEYWORDS”, pl. „kalkulátor szerint”, „kedvezmény”, „kattints ide”) és 18 HTML-alapú hirdetési minta („DEFAULT_HTML_AD_PATTERNS”, pl. „ad[-]”, „sponsored”, „taboola”, „outbrain”). A minták a „ParamManager” osztályon keresztül futásidőben is bővíthetők és módosíthatók anélkül, hogy a szolgáltatást újra kellene indítani.

3. **NLP elemzés:** Az „analyze_text_nlp” metódus a HuSpacy nyelvi modellt használva az alábbi metrikákat számítja ki:

- **Szavak és mondatok száma:** A tokenizáció és mondathatár-felismerés eredményeként.
- **Szentiment:** Pozitív és negatív kulcsszavak lemmaalapú számlálásával. Ha a pozitív szavak 1,5-szeresen meghaladják a negatívakat, a szentiment „positive”, ha fordítva, „negative”, egyébként „neutral”.
- **Formalitás:** A tulajdonnevek (PROPN szófajú tokenek) arányán alapul. Ha a tulajdonnevek aránya meghaladja a szavak 10%-át, a szöveg „formal” kategóriát kap.
- **Olvashatóság:** A Flesch-Kincaid képlet magyar adaptációja, ahol a szótagszámot a „_count_syllables” segédmetódus számítja:

$$\text{olvashatósági pontszám} = 206,835 - 1,015 * (\text{szavak száma} / \text{mondatok száma}) - 84,6 * (\text{szótagok száma} / \text{szavak száma})$$

4. **Tartalombiztonsági elemzés:** A „ContentSafetyAnalyzer” osztály a szöveg biztonsági profilját értékeli több szempont szerint: erőszak, szexuális tartalom, drogok és politikai érzékenység. Az értékelés kulcsszólistákon alapul, amelyek szintén a „ParamManager”-en keresztül konfigurálhatók.
5. **Szövegösszegzés (LexRank + MI):** A „NewsSummarizer” osztály kétlépéses összegzést végez. Először a Sumy könyvtár LexRank algoritmusával extraktív

összegzést készít (a legfontosabb mondatok kiválasztása), majd a Google Gemini API kötegelt feldolgozásával MI-alapú összefoglalót is generál.

6. **Rádiós relevancia-számítás:** A „RadioRelevanceCalculator” osztály a hír rádiós felolvasásra való alkalmasságát pontszámmal értékeli. A pontszám négy komponensből áll:

- Hírérték és aktualitás (published_date frissessége)
- Forrás presztízse (az rss tábla prestige mezője)
- Olvashatóság és felolvashatóság (Flesch-Kincaid pontszám)
- Tartalombiztonság (a safety_score alapján)

3.6.2.2. Duplikációsűrés (DuplicationDetector)

A duplikációsűrés az analyze modul egyik legkritikusabb, kétfázisú algoritmussal működő funkciója:

1. **Gyors előszűrés (cím-alapú):** A RapidFuzz könyvtár fuzzy string matching algoritmusával a hír címét összehasonlítja a meglévő cikkek címeivel. Ez a lépés $O(n)$ időkomplexitással fut és a nyilvánvalóan különböző cikkeket azonnal kiszűri.
2. **Tartalomalapú megerősítés:** Az első fázisban jelölt gyanús duplikátumok esetén a TF-IDF vektorizálás és koszinusz-hasonlóság alapján mélyebb összehasonlítást végez. Az entitás-ujjlenyomat (entity fingerprinting) további dimenziót ad: ha két cikk azonos személyeket, helyeket és szervezeteket említ, az erősíti a duplikáció gyanúját.

A kombinált duplikációs küszöbérték 0,80 (80%-os hasonlóság felett a cikk duplikátumnak minősül). Az opcionális FAISS index alkalmazásával a hasonlóságkeresés további 10-15-szörös gyorsulást ér el.

3.6.2.3. A COCO API kliens és az OAM elemzés

A COCO (Component-based Object Comparison for Objectivity) integráció a „coco_client.py” modulban valósul meg. A kliens a „miau.my-x.hu” távoli szolgáltatással kommunikál HTTP POST kérésekkel. A COCO rendszer angol nyelvű webes felülete (vö. Pitlik L., é.n., COCO Online Expert System; lásd 8.5 Hivatkozások) a nemzetközi felhasználók számára is hozzáférhetővé teszi a módszertant.

A „CocoClient” osztály három modellt támogat (Y0, STD, MCM) és két végpontot (engine3.php HTML-kimenettel, engine3_curl.php egyszerű szöveges kimenettel). A bemeneti

mátrix előkészítése a „prepare_input_matrix” metódusban történik. A COCO módszer gyakorlati alkalmazásának lépéseit – a bemeneti mátrix összeállításától az eredmények értelmezéséig – az oktatási segédanyag (vö. Pető I., 2013; lásd 8.5 Hivatkozások) szemlélteti.

A „miau.my-x.hu” API egy speciális formátumot vár: TAB-elválasztott értékek CRLF (Windows stílusú) sorvégekkel. Ez a formátumkövetelmény a kliens implementációjának kritikus eleme, amelynek figyelmen kívül hagyása helytelen eredményekhez vezet.

A HTML-válasz feldolgozásáért a „CocoHTMLParser” osztály felel, amely a Python beépített „HTMLParser” osztályából származik. A parser hat táblázattípust ismer fel (meta, rangsor, lepcso1, lepcso2, coco, summary) és az egyes táblázatok tartalmát strukturáltan kinyeri a „CocoResponse” dataclass-be.

3.6.2.4. Közösségi trendjelek integrációja

A newscast-social modullal való integrációt valósítja meg, az „analysis” tábla három mezőjével: „social_trending_score” (a *Szuper képlet* által számított pontszám), „social_signal_count” (az egyeztetett közösségi jelek száma) és „social_platforms” (a forrásplatformok vesszővel elválasztott listája).

A Gemini kötegelt feldolgozás prioritási sorrendjébe beépítésre került a közösségi pontszám: a batch ütemező a „radio_relevance_score + COALESCE(social_trending_score, 0) * 0,15” képlettel rangsorolja a feldolgozandó híreket, így a közösségileg népszerű cikkek hamarabb kapnak MI-összefoglalót.

A **social prefilter override** mechanizmus lazított küszöbértékeket alkalmaz azon cikkekre, amelyeket a newscast-social modul egyeztetett: magasabb tartalombiztonsági küszöb (a szenzációs, de közösségileg népszerű hírek átjuthatnak), rövidebb minimális tartalomhossz és a headline check kihagyása. Ez biztosítja, hogy a közösségileg igazolt hírértékű cikkek ne essenek ki a szűrőn technikai okok miatt.

3.6.3. newscast-weather: Időjárás feldolgozó modul

A weather modul az Országos Meteorológiai Szolgálat (OMSZ) szöveges előrejelzéseinek automatikus letöltését, feldolgozását és tárolását végzi. A modul két fő komponensből áll: a „main.py” a FastAPI alkalmazást és az ütemezőt tartalmazza, az „analyzer.py” pedig az adatfeldolgozási pipeline-t.

3.6.3.1. Az OMSZ adatok feldolgozási pipeline-ja

Az OMSZ az előrejelzéseit ZIP-tömörített szöveges fájlkként teszi publikusan elérhetővé. A feldolgozási pipeline az alábbi lépésekből áll:

1. **ZIP letöltése és kicsomagolása:** Az „extract_text_from_zip” függvény a letöltött ZIP-fájlból kiolvassa a szöveges tartalmat. A függvény automatikusan megkeresi az első „.txt” kiterjesztésű fájlt a ZIP-archívumban.
2. **Kódolási felismerés és konverzió:** Az OMSZ szöveges fájljai jellemzően Windows-1250 kódolást használnak, amely a közép-európai nyelvek szokásos kódolása. A „detect_and_decode_content” függvény a „chardet” könyvtárral automatikusan felismeri a kódolást, majd egy prioritizált listán próbálja meg a dekódolást: A kódolás helyességét a magyar ékezetes karakterek (áéíóöőüüúÁÉÍÓÖÖÚÚÚ) jelenléte alapján ellenőrzi. Ha a strict dekódolás minden kódolással sikertelen, az „ignore” módra vált, végső esetben pedig UTF-8 „replace” módot alkalmaz, jelezve a karaktervesztés lehetőségét.
3. **Dátum validáció:** A „validate_weather_date” függvény három dátumformátumot keres a tartalom első 10 sorában: „YYYY.MM.DD”, „YYYY-MM-DD” és „DD.MM.YYYY”. A megtalált dátumot összeveti az aktuális dátummal és eltérés esetén figyelmeztetést naplóz.
4. **Tartalomtisztítás:** A „clean_weather_content” függvény a forrástípustól függően eltávolítja a metaadat sorokat. Az „általános” (general) előrejelzéseknél az első (dátum) sort, az „orvos-meteorológiai” (medical) előrejelzéseknél az első két metaadat-sort (dátum és elválasztó) távolítja el.
5. **Tartalomelemzés:** Az „analyze_weather_content” függvény statisztikákat számol a szövegről: sorok száma, szavak száma, mondatok száma, magyar karakterek száma. Emellett specifikus meteorológiai tartalmakat is keres reguláris kifejezésekkel: hőmérséklet, csapadék („eső|hó|csapadék|zápor|zivatar”), szél („szél|szellő|viharos”), légnyomás („hPa|mbar|nyomás”) és páratartalom.
6. **SHA-256 hash és duplikáció megelőzés:** A feldolgozott tartalom SHA-256 hash-e („content_hash”) a „weather” tábla UNIQUE kényszerével garantálja, hogy azonos tartalmú előrejelzés ne kerüljön duplikáltan az adatbázisba.

3.6.4. newscast-feeder: Hírszelekció és webes felület

A feeder modul kettős szerepet tölt be: egyrészt a hírblokk összeállítás üzleti logikáját implementálja (hírszelekció, duplikációszűrés, szövegformázás), másrészt webes felhasználói felületet biztosít a hírszerkesztők számára. A modul hat fő Python fájlból áll: „main.py” (FastAPI alkalmazás), „feeder.py” (szelekciós logika), „frontend.py” (Web UI), „database.py” (adatmodell), „auth.py” és „monitoring.py”.

3.6.4.1. A hírszelekciós algoritmus

A szelekciós algoritmus a „feeder.py” modul „fetch_radio_suitable_news” függvényében valósul meg.

A szelekció az „ai_radio_suitable_news” adatbázis-nézetten (view) keresztül történik, amely előszűri azokat a híreket, amelyek az elemzés eredménye alapján alkalmasak rádiós felolvasásra. A szelekciós kritériumok:

1. **Frissesség:** Csak a legutóbbi 12 órában (NEWS_FRESHNESS_HOURS) publikált hírek kerülnek figyelembevételre.
2. **Duplikáció kizárás:** A legutóbbi 24 órában (NEWS_DUPLICATE_FILTER_HOURS) már kiválasztott hírek automatikusan kizárásra kerülnek. A „get_recent_news_ids” függvény lekéri a korábban kiválasztott hír azonosítókat és azokat az SQL IN feltételével szűri.
3. **Rangsorolás:** A hírek a „radio_relevance_score” csökkenő sorrendjében, azonos pontszám esetén a „published_date” csökkenő sorrendjében (legfrissebb először) kerülnek rendezésre.
4. **Korlátozás:** Az első 5 hír (NEWS_SELECT_COUNT) kerül kiválasztásra.

3.6.4.2. Narratíva flow szerkesztés és szövegformázás

A kiválasztott hírek szövegformázását a „format_news_for_tts” függvény végzi. A függvény kétféle formázási módot támogat:

Narratíva flow mód: Ha a „tts_narrative_flow” paraméter be van állítva az adatbázisban (JSON formátumú konfigurációs blokk), a szöveg egy strukturált folyam szerint épül fel. A flow az alábbi blokk típusokat támogatja:

- „intro”: bevezető szöveg
- „ai_lead”: az összes kiválasztott hír MI összefoglalója szekvenciálisan
- „summary”: az összes hír összefoglalója szekvenciálisan
- „transition”: átmeneti szöveg a hírek között
- „pause”: SSML szünet jelölő (,,,))
- „outro”: záró szöveg

Legacy mód: Ha nincs narratíva flow definiálva, a „format_news_legacy” függvény egy egyszerűbb formátumot alkalmaz: opcionális intro + 1 másodperces szünet + hírek (ai_lead vagy summary_text) 0,5 másodperces szünetekkel + opcionális outro.

A szünetek SSML break tagekkel vannak jelölve, amelyeket – a használt modelltől függően – az ElevenLabs TTS motor értelmez.

3.6.4.3. Webes felhasználói felület

A „frontend.py” modul egy teljes webes alkalmazást implementál Jinja2 sablonokkal, amely a következő funkciókat kínálja:

- **Dashboard (,,/ui/”):** A Dashboard felületen (lásd 4. ábra) az utolsó hírblokkok kronológiai áttekintése látható. A felületen a szerkesztő megtekintheti az elkészült hírblokkok metaadatait (létrehozás ideje, szöveg hossza, felhasznált hírek száma), lejátszhatja és letöltheti a generált hanganyagokat, valamint közvetlenül indíthat manuális TTS generálást. A Dashboard felület működésére egy konkrét mintakimenet is elérhető (lásd 8.8 alfejezet, Hirek-LikeFM-20260417.mp3).

NewsCast Feeder Dashboard Hírek OAM Beállítások Felhasználók Administrator (admin) Kijelentkezés

Dashboard

Üzem mód
Manuális

Ütemező
Leállítva
Intervallum: 3600 mp

Mai hírblokkok
1

Függő TTS
4

Gyorsműveletek

[Automatikus hírblokk generálás](#) [Manuális hírkiválasztás](#)

Legutóbbi hírblokkok

CORRELATION ID	LÉTREHOZVA	HÍREK SZÁMA	TTS STÁTUSZ	MŰVELETEK
18b0e5f9-5caa-4e2a-be9d-3c92e9f7f4ed	2026-04-17 04:44	5	completed	▶ ⬇ Részletek
2dde6806-4539-4cf4-ae9b-7b223393897f	2026-04-15 14:14	5	completed	▶ ⬇ Részletek
fc82d61c-86e5-4f5c-bee1-3885c5078d08	2026-04-13 20:33	5	completed	▶ ⬇ Részletek
390dd699-e061-49af-abf6-700e21c50860	2026-04-13 03:59	5	completed	▶ ⬇ Részletek
9e5aac0-42f0-4ab1-b4e7-670cb58ad001	2026-04-12 14:02	5	completed	▶ ⬇ Részletek
9814c03d-2cb4-49c4-9888-e580d82f2de4	2026-04-12 12:00	5	completed	▶ ⬇ Részletek
a203a0fc-7ff4-44a0-ace0-3f7b9d388779	2026-04-12 10:00	5	completed	▶ ⬇ Részletek
50841de7-6304-49ac-a68c-2ce0b98566f4	2026-04-11 06:45	5	completed	▶ ⬇ Részletek
b3f246cf-63f0-4b1c-8f8a-d751e72e751f	2026-04-09 04:25	5	completed	Részletek
07e9efec-8cd8-4c34-b846-7f193214ad44	2026-04-07 07:20	5	completed	Részletek

[További hírblokkok betöltése](#) 10 / 81 hírblokk

4. ábra: A newscast-feeder Dashboard felülete – az utolsó hírblokkok és a lejátszható hanganyagok áttekintése – Forrás: saját szerkesztés

- **Hírek („/ui/news”):** A Hírek felület (lásd 5. ábra) a kiválasztható hírek teljes listáját jeleníti meg. A felületen a szerkesztő szűrhet forrás, dátum és relevancia pontszám szerint, a drag-and-drop mechanizmussal módosíthatja a szelektált hírek sorrendjét, valamint szerkesztheti az egyes hírszövegeket. Az MI címkék (szentiment, tartalombiztonság, relevanciaszint) színekkel jelölve jelennek meg minden hír mellett.

NewsCast Feeder Dashboard Hírek OAM Beállítások Felhasználók Administrator (admin) Kijelentkezés

Hír kiválasztás

Jelenlegi mód: **Manuális** + Új hír

Manuális módban válassza ki a sugározni kívánt híreket a listából.

FORRÁS: Mind ▼ IDŐSZAK: 12 óra ▼ KERESÉS: Cím vagy tartalom... ▼ CÍMKÉK: Szűrés címkékre... ▼ AUTO-FRISÍTÉS: Ki ▼

Szűrés Reset

Nincs szűrő

Google Trends – Top keresések 12 óra - 19:38

#1 csányi sándor 30,200+	#2 magyar peter 12,000+	#3 jégeső 10,000+	#4 otp bank nyrt. 10,000+	#5 bodó bernadett dandártábornok 7,200+
#6 úpest-nyiregyháza 7,000+	#7 ruszin szendi romulusz 7,000+	#8 gönczi gábor 5,000+	#9 elnök 5,000+	#10 kapitány istván 4,200+

607 hír összesen Oldal méret: 25 ◀ ▶ 1 2 3 ... 25 ▶

<input type="checkbox"/>	SCORE	CÍM	CÍMKÉK	FORRÁS	DÁTUM	MŰVELETEK
<input type="checkbox"/>	85.0	Száz éve született Papp László háromszoros olimpiai bajnok ökölvívó. A magyar sporttörténelem egyik legnagyobb alakja, Papp László születésének centenáriumi emlékne...	belföld sport siker pozitív	RTL.hu	04.17 19:50	Részletek
<input type="checkbox"/>	90.0	Hosszabbításban győzte le a magyar férfi jégkorong-válogatott Lengyelország csapatát. A világbajnokságra készülő magyar együttes drámai mérkőzésen, 3-2-re nyert a lengyelek ellen. A talá...	sport siker	Origo	04.17 19:48	Részletek
<input type="checkbox"/>	90.0	A mozgósított ukrán katonák kevesebb mint harmada kerül ténylegesen a frontvonalra. Ukrajnában havonta mintegy harmincezer embert hívnak be katonai szolgálatra, ám közülük csupán nyolc...	háború külföld politika negatív	Magyar Nemzet	04.17 19:44	Részletek
<input type="checkbox"/>		A körmök elváltozásai súlyos betegségekre is felhívhatják				

5. ábra: A newscast-feeder Hírek nézete – hírszelekció, rendezés és szerkesztés – Forrás: saját szerkesztés

- **OAM („/ui/oam”)**: Az OAM nézet (lásd 6. ábra) a newscast-analyze modul COCO API kliensén (vö. 3.6.2.3 alfejezet) keresztül lekérdezett objektum–attribútum mátrix (OAM) elemzés eredményeit jeleníti meg. Egy éles futtatásból származó OAM riport HTML és XLSX formátumban a 8.8 alfejezetben található (lásd oam_report_d9c8a72d-20260417211214.html és .xlsx).

NewsCast Feeder Dashboard Hírek **OAM** Beállítások Felhasználók Administrator (admin) Kijelentkezés

Objektum-Attribútum Mátrix Frissítés

COCO OAM Elemzések

Új COCO elemzés indítása

COCO Modell típus

YO: Mindenki egyenlő különböző módon - minimalizálja a pontszámok szórását

Azonosító (opcionális) Lépcsők száma (opcionális)

OAM elemzés Fájlok megtartása (szerver oldalon)

Egyedi azonosító a COCO API-ban Alapértelmezett: objektumok száma Időablak (óra) Max. cikkek

Min. cikkek Csak Gemini AI elemzett cikkek

Alap attribútumok Mind ✓ Törés ✕

Szentiment Olvashatóság Biztonság Rádió relevancia Frissesség Presztízs

További metrikák Mind ✓ Törés ✕

Szószám **100.0%** Mondatszám **100.0%** Összefoglaló hossz **99.3%** Duplikáció konfidencia **83.5%** AI Rádió relevancia **66.0%**

NewsCast Feeder Service v2.4.0

6. ábra: A newscast-feeder OAM nézete – a COCO modell elemzési eredményeinek megjelenítése – Forrás: saját szerkesztés

- **Beállítások („/ui/settings”):** A Beállítások felület (lásd 7. ábra) a ParamManager osztályon (vö. 3.6.2 alfejezet) keresztül elérhető rendszerparaméterek szerkesztését teszi lehetővé. A szerkesztő módosíthatja az üzemmódot (automatikus/manuális), az APScheduler ütemezési intervallumát, a hírszelekción küszöbértékeket, a narratíva flow JSON konfigurációt (vö. 3.6.4.2 alfejezet) és a TTS hangprofilbeállításokat anélkül, hogy a rendszert újra kellene indítani.

NewsCast Feeder Dashboard Hírek OAM **Beállítások** Felhasználók Administrator (admin) Kijelentkezés

Beállítások

ÜZEMMÓD

Feed mód
Válassza ki az üzemmódot: automatikus (ütemező alapján) vagy manuális (kézi kiválasztás).
Automatikus **Manuális**

ÜTEMEZŐ

Ütemező engedélyezése
Az automatikus hírkiválasztás ütemezésének bekapcsolása.

Ütemezés intervallum
Milyen gyakran futtassa az automatikus hírkiválasztást (másodpercben).
1 óra

HÍRKIVÁLASZTÁS

Hírek száma
Hány hírt válasszon ki egy hírblokkba.
5 hír

Hír frissessége
Maximum hány órák között válasszon.
12 óra

Duplikátum szűrés
Hány órán belül ne válassza újra ugyanazt a hírt.
24 óra

TTS BEÁLLÍTÁSOK

Narratíva flow szerkesztő
Állítsa össze a hír bemondás szöveggöngyölt drag & drop módszerrel.

Elérhető blokkok

- Bevezető szöveg
- Átvezető szöveg
- Befejező szöveg
- Hír felvezető
- Hír szöveg
- Szünet

Narratíva sorrend

Szünet ▲ ▼ ×

Időtartam (mp): 0.5 mp

Hír felvezető ▲ ▼ ×

AI Lead (TTS szöveg) – minden hírhez

Szünet utána: 0.5 mp

Szünet ▲ ▼ ×

Időtartam (mp): 0.5 mp

Átvezető szöveg ▲ ▼ ×

Híreink részletesen, Önök a Like FM híreit hallják!

Szünet ▲ ▼ ×

Időtartam (mp): 1 mp

Hír szöveg ▲ ▼ ×

Összefoglalás – minden hírhez

Szünet utána: 1 mp

Mentés Alapértelmezett Előnézet

7. ábra: A newscast-feeder Beállítások felülete – az üzemmód, ütemezés és TTS paraméterek konfigurálása – Forrás: saját szerkesztés

- **Felhasználók („/ui/users”)**: A Felhasználók felület (lásd 8. ábra) kizárólag admin jogosultsággal érhető el (vö. 3.7.1 alfejezet). A felületen a rendszergazda felhasználókat hozhat létre, módosíthatja a szerepköreiket (admin/editor/viewer) és visszavonhatja a hozzáférésüket. A hitelesítés JWT Bearer Token alapú (vö. 3.4.4.2 alfejezet), a jelszavak bcrypt hash formátumban tárolódnak.

FELHASZNÁLÓNÉV	TELJES NÉV	EMAIL	SZEREPKÖR	STÁTUSZ	UTOLSÓ BEJELENTKEZÉS	BEJELENTKEZÉSEK	MŰVELETEK
admin	Administrator	-	admin	Aktív	2026-04-17T15:40:20	153	Szerkesztés
kodolanyi	Kodolányi University	info@newscast.hu	viewer	Aktív	2026-01-05T20:31:16	1	Szerkesztés Törölés

NewsCast Feeder Service v2.4.0

8. ábra: A newscast-feeder Felhasználók kezelő felülete – felhasználók és szerepkörök adminisztrálása – Forrás: saját szerkesztés

3.6.5. newscast-tts: Szövegfelolvasás (Text-to-Speech) modul

A TTS modul felelős a szöveges hírblokkok természetes hangzású beszéddé alakításáért az ElevenLabs API használatával. A modul 14 Python fájlból áll, amelyek közül a legfontosabbak: „tts_service.py” (az ElevenLabs kliens és worker pool), „text_normalizer.py” (magyar szövegnormalizálás), „templates.py” (sablonkezelés) és „external_services.py” (weather/news API kliensek).

3.6.5.1. Az ElevenLabs TTS kliens

A „TTSCient” osztály („tts_service.py”) közvetlenül az ElevenLabs API-val kommunikál.

Az „eleven_v3” modell speciális korrekciót igényel a „stability” paraméternél: a modell csak 0.0, 0.5 és 1.0 értékeket fogad el, ezért a kliens automatikusan kerekíti a megadott értéket a legközelebbi megengedett értékre.

A generálás aszinkron HTTP hívással történik (a streaming API végponton: „/text-to-speech/{voice_id}/stream”) és az eredmény ideiglenes fájlba kerül mentésre.

3.6.5.2. Feladatsor és worker pool

A TTS generálás aszinkron feladatsoron (task queue) keresztül történik, amely leválasztja a bejövő kérés fogadást a feldolgozástól:

1. **Kérés fogadása:** A POST „/api/v1/tts/generate” végpont fogadja a kérést, validálja a Pydantic modellel, létrehoz egy feladatot (task_id = UUID) és azonnal 202 Accepted válasszal tér vissza.
2. **Sorba állítás:** A feladat a max. 100 elemű várakozási sorba (queue) kerül.
3. **Worker feldolgozás:** 2-10 konfigurálható worker szál figyeli a várakozási sort és szekvenciálisan dolgozza fel a feladatokat.
4. **Státusz-követés:** A feladat állapota (QUEUED → PROCESSING → COMPLETED/FAILED) a „tts_history” táblában kerül rögzítésre és a GET „/api/v1/tts/tasks/{task_id}” végponton kérhető le.

3.6.5.3. Tartalom hash deduplikáció

A TTS modul SHA-256 alapú tartalom hash deduplikációt alkalmaz. Minden generálási kérés előtt a rendszer kiszámítja a normalizált szöveg hash értékét és az adatbázisban ellenőrzi, hogy korábban készült-e már hangfájl azonos tartalommal.

- Ha igen („was_reused = True”), a korábbi hangfájl kerül visszaadásra ElevenLabs API hívás nélkül.
- Ha nem, megtörténik az ElevenLabs API hívás és az eredmény hash értékkel együtt tárolódik.

Ez a mechanizmussal közvetlenül csökkenthető az ElevenLabs API költségei, mivel az azonos tartalmú (pl. ismételt időjárás előrejelzés) szövegek nem generálódnak újra.

3.6.5.4. Magyar szövegnormalizálás

A „text_normalizer.py” modul a legterjedelmesebb egyedi fejlesztésű komponens a rendszerben. A normalizálás célja, hogy a szöveget a TTS motor számára optimális formába alakítsa, a magyar nyelv szabályainak megfelelően.

Számok szövegesítése (number_to_words): A függvény rekurzív felépítésű és a 0-tól 999 999 999-ig terjedő tartományt fedi le. A magyar számnév képzés néhány speciális szabályt igényel. A rekurzió gondoskodik arról, hogy tetszőlegesen nagy számok is helyesen kerüljenek szövegesítésre. A kötőjel szabály a magyar helyesírás követelménye: az ezres és annál nagyobb nagyságrendek között kötőjel áll (pl. „kétezer-huszonegy”), míg a százason belül nincs kötőjel („száználcvanhat”).

Hőmérséklet normalizálás (normalize_temperature): A függvény különleges figyelmet fordít a természetes beszéd konvencióira:

- A negatív hőmérsékletnél a „mínusz” előtag megjelenik: „-5°C” → „mínusz öt fok”
- A pozitív hőmérsékletnél nincs „plusz” előtag: „25°C” → „huszonöt fok” (nem „plusz huszonöt fok”)
- A Celsius jelölés nem kerül kimondásra (ez az alapértelmezett Magyarországon), de a Fahrenheit igen
- A tizedesvessző kezelése: „10,5°C” → „tíz és fél fok”

Dátum normalizálás: A magyar dátumformátum birtokos ragozást igényel: „január 15.” → „január tizenötödike” (nem „tizenöt”). Ehhez a rendszer a 31 napra vonatkozó birtokos ragos alakokat előre definiálja az „ORDINALS_POSSESSIVE” szótárban, illetve a határozóragos alakokat („elsején”, „másodikán”) az „ORDINALS_ON_DATE” szótárban.

SSML-védelem: A normalizálási lépések reguláris kifejezéseket használnak, amelyek véletlenül módosíthatnák az SSML break tageket. Ennek megakadályozására a „protect_ssml_tags” függvény a normalizálás előtt minden ilyen taget egyedi placeholder-re cserél („*SSML_BREAK*{index}*PLACEHOLDER*”), majd a „restore_ssml_tags” függvény a normalizálás után visszaállítja azokat.

3.6.5.5. Külső szolgáltatások

Az „external_services.py” modul két külső szolgáltatással kommunikál:

- **newscast-weather:** Az időjárás-előrejelzés lekérdezése a „/weather/today”, „/weather/today-combined” és „/weather/latest” végpontokról.
- **newscast-feeder:** A kiválasztott hírek lekérdezése a „/api/v1/selected-news” vagy „/api/v1/selected-news/{correlation_id}” végpontról.

3.6.6. newscast-social: Közösségi média trendgyűjtő modul

A 2.2.4 fejezetben részletesen tárgyalt közösségi platform korlátozások közvetlen hatást gyakoroltak a newscast-social modul fejlődésére. A fejlesztés során több platform integrálása is megvalósult, majd eltávolításra került: a v2.0.0 verzióban beépített **Reddit** PRAW (Python Reddit API Wrapper) integráció a v2.1.0-ban eltávolításra került, mivel a Reddit 2023 júliusától

bevezetett API-díjak és kvótakorlátozások a projekt céljaira fenntarthatatlanná tették a hozzáférést. A v3.0.0 verzióban a **Bing News Search** API is kikerült a modulból, mivel a Microsoft a keresési API-k ingyenes szintjét megszüntette. Ezek a fejlesztési iterációk empirikusan igazolták, hogy a közösségi média platformok szisztematikus bezárkózása (vö. 2.2.4 fejezet) következtében végül a **Google News RSS** és a **Google Trends RSS** maradt az egyetlen fenntartható, ingyenes és jogilag problémamentes közösségi jelforrás.

A social modul (v3.1.0) a rendszer legújabb komponense, amely valós idejű közösségi média trendjeleket gyűjt össze és párosít a meglévő hírekkel. A modul Python 3.9-slim alapokon fut, FastAPI keretrendszerrel és APScheduler ütemezővel. A fő üzleti logika a „social_collector.py” modulban található.

3.6.6.1. Google News és Google Trends

A modul két külső RSS forrásból dolgozik:

- **Google News RSS (20 perces ütemezéssel):** A **collect job** metódus 7 feed kategóriát (top, nation, world, business, tech, science, health) dolgoz fel párhuzamosan. Minden bejegyzés a „social_signals” táblába kerül a platform („google_news”), a cím, az URL és a pozíció-alapú engagement pontszám rögzítésével. A Google News URL-ek átirányítás-feloldáson esnek át, mivel a Google News proxy URL-eket szolgáltat az eredeti forrás-URL-ek helyett.
- **Google Trends RSS (60 perces ütemezéssel):** A **trends job** metódus a Google Trends magyar nyelvű RSS feedjéből kinyeri a TOP10 (konfigurálható: GOOGLE_TRENDS_TOP_N) trending kulcsszót és a hozzá tartozó becsült forgalmat. Az eredmények a „trending_keywords” táblába kerülnek és egyidejűleg egy in-memory cache-ben (TrendingStore) is tárolódnak.

3.6.6.2. TrendingStore: kulcsszó illesztés

A „TrendingStore” osztály DB + in-memory cache architektúrával működik, amely a trending kulcsszavak gyors elérését biztosítja. A substring matching algoritmus a magyar nyelv ragozási sajátosságait is kezeli. Pl. a „kormány” kulcsszó illeszkedik a „kormányülés”, „kormányhatározat” és „kormányfő” szavakra is anélkül, hogy szótövesítésre lenne szükség. Ez a megközelítés a RapidFuzz könyvtár token_sort_ratio algoritmusánál gyorsabb feldolgozást biztosít az előszűrési fázisban.

3.6.6.3. Többszintű URL egyeztetés

A match job (5 perces ütemezéssel) háromszintű algoritmussal párosítja a közösségi jeleket a „news” tábla rekordjaival:

1. **url_hash egyeztetés:** Az URL SHA-256 hash alapján azonnali egyezést keres. Ez a leggyorsabb és legpontosabb.
2. **Canonical URL egyeztetés:** Az URL kanonizálása (HTTP séma, hostnév normalizálás, query paraméterek eltávolítása) után keres egyezést, valamint a Google News „related” URL címeket is összeveti.
3. **RapidFuzz fuzzy cím illesztés:** A RapidFuzz könyvtár token_sort_ratio algoritmusával a jel címét hasonlítja össze a „news” tábla címeivel. A küszöbérték 75 pont (TITLE_MATCH_THRESHOLD), amely fölött az egyezés elfogadásra kerül. Ez a szint kezeli azokat az eseteket, amikor ugyanazt a hírt eltérő URL-ről és eltérő megfogalmazással publikálták.

3.6.6.4. Szuper képlet és pontszámítás

A sikeres egyeztetés után a **Super Formula** (Szuper képlet) határozza meg a végső pontszámot:

$$V_H = 10 + (Trends * 50)$$

ahol V_H a **Virális Hírérték** – az adott hír közösségi trendjeleken alapuló, pontszámként kifejezett hírértéke. A Trends értéke 1, ha a hír címe tartalmaz legalább egy trending kulcsszót a TrendingStore-ból, egyébként 0. Az eredmény 10 (sima hír, megjelent a Google News-ban) vagy 60 (virális hír, Google Trends TOP10 kulcsszót is tartalmaz). A pontszám az „analysis” tábla „social_trending_score” mezőjében tárolódik.

3.6.6.5. Backfill mechanizmus (race condition kezelés)

Ha az érintett „analysis” rekord az UPDATE időpontjában még nem létezik (a newscast-analyze modul még nem dolgozta fel a hírt), akkor a „write_social_score” metódus rowcount == 0 eredményt kap. Ilyenkor a news_id a „pending_news_ids” halmazba kerül és a következő match_job ciklusban a „backfill_pending_scores” metódus újra próbálja az írást. Ha az érintett rekord a 24 órás érvényességi ablakon (SIGNAL_WINDOW_HOURS) kívülre kerül, automatikusan törlődik a pending sorból.

3.7. Biztonsági megoldások

A biztonsági megoldások alfejezet a NewsCast rendszer biztonsági rétegeit mutatja be: a hitelesítés és jogosultságkezelés (3.7.1), az API-kulcsok és érzékeny adatok kezelése (3.7.2) és az SQL injection elleni védelem (3.7.3) kerül tárgyalásra.

3.7.1. Hitelesítés és jogosultságkezelés

Ahogy az a 3.4.4. alfejezetben részleteztem, a rendszer kétcsatornás hitelesítést alkalmaz. Az implementáció minden modulban azonos mintát követ. A „verify_basic_auth” függvény a „secrets.compare_digest()” metódust használja az időállandó összehasonlításhoz, megakadályozva a timing attack-eket.

3.7.2. API-kulcsok és érzékeny adatok kezelése

Minden érzékeny adat – adatbázis-jelszavak, API-kulcsok, JWT titkos kulcsok – a „.env” fájl környezeti változóiból kerül betöltésre a „python-dotenv” könyvtár segítségével. A „.gitignore” fájl felel azért, hogy a „.env” fájl ne kerüljenek a verziókezelő rendszerbe.

3.7.3. SQL injection elleni védelem

Az adatbázis lekérdezések minden modulban parametrikus formában készülnek (SQLAlchemy ORM vagy parametrikus SQL).

3.8. Monitorozás és naplózás

A monitorozás és naplózás alfejezet a rendszer üzemeltetési felügyeletét biztosító megoldásokat ismerteti: a Prometheus metrikák és Grafana dashboardok (3.8.1) a valós idejű teljesítményfigyelést, a strukturált naplózás és a korrelációs azonosítók (3.8.2) a hibafelderítést szolgálják.

3.8.1. Prometheus metrikák és Grafana dashboardok

Minden modul dedikált „monitoring.py” fájlal rendelkezik, amely a **Prometheus** metrikagyűjtő rendszer számára szolgáltat adatokat a 8000-es TCP porton. A metrikák három szinten szerveződnek:

- **Rendszerszintű:** CPU%, memória (bájtokban), hálózati I/O, lemezes I/O – a „psutil” könyvtárral gyűjtve, 10 másodperces frissítéssel.

- **Alkalmazásszintű:** API kérések száma (Counter), feldolgozási idők (Gauge/Histogram).
- **Üzleti szintű:** Feldolgozott hírek száma, TTS generálások, duplikációk aránya.

A begyűjtött metrikák vizualizálását a **Grafana** biztosítja. A Grafana a Prometheus adatforrásra csatlakozva interaktív dashboardokat szolgáltat, amelyeken a rendszer teljesítménye, az egyes modulok erőforrás kihasználtsága, az API válaszidők és az üzleti mutatók (feldolgozott cikkek száma, TTS generálások, duplikációs arányok) valós időben nyomon követhetők. Az üzemeltetők számára a Grafana alerting funkciójával küszöbérték alapú riasztások is beállíthatóak.

3.8.2. Strukturált naplózás és korrelációs azonosítók

Minden modul „logging_config.py” fájlja JSON formátumú strukturált naplózást valósít meg. A naplóbejegyzések egységes mezőkészlettel rendelkeznek: „timestamp”, „level”, „logger_name”, „message” és „correlation_id”.

A korrelációs azonosító (correlation ID) alapvető fontosságú a mikroszolgáltatás architektúrában: egy hírblokk feldolgozása több szolgáltatáson átível (feeder → tts → weather) és a közös azonosító révén a teljes feldolgozási lánc nyomon követhető a naplóbejegyzéseken keresztül. A feeder modulban a correlation ID UUID formátumban generálódik és a TTS modulnak a callback hívásban kerül továbbításra.

3.9. MI-aspektusok

A NewsCast rendszer több moduljában is megjelennek mesterséges intelligencia (MI) alapú megoldások. Az alábbiakban ezek összefoglalása olvasható.

A newscast-analyze modul a Google Gemini Pro API-n keresztül végez MI-alapú validációt a hírek rádiós relevanciájának megállapításához. A szabályalapú elemzés – szentimentelemzés, olvashatósági metrikák, tartalombiztonsági osztályozás – kimenetét a Gemini modell megvizsgálja és egy kiegészítő besorolást készít az olyan tartalmak esetében, amelyeknél a szabályalapú feldolgozás önmagában nem ad megbízható eredményt (pl. ironikus, szatirikus vagy többértelmű szövegek).

A természetes nyelvfeldolgozás (Natural Language Processing, NLP) terén a rendszer a HuSpacy hu_core_news_lg magyar nyelvi modellt használja a lemmatizáláshoz, a névelem

felismeréshez (Named Entity Recognition, NER) és a mondatsegmentáláshoz. A HuSpacy gépi tanulási modelleken alapul, amelyeket magyar nyelvű szövegkorpuszokon tanítottak be.

A szövegfelolvasás (Text-to-Speech, TTS) modulban az ElevenLabs TTS API végzi a hangszintézist. Az ElevenLabs neurális hálózaton alapuló megoldása a korábbi, szabályalapú szintézisekkel szemben a magyar nyelv időzítési, dallambeli jellemzőit és hangsúlyozását pontosabban kezeli.

A COCO (Component-based Object Comparison for Objectivity) modell nem gépi tanulás alapú, hanem matematikai statisztikai (solver) módszer az objektumok összehasonlítására. A modell három változata – varianciaminimalizálás (Y0), célérték rekonstrukció (STD) és binelés alapú csoportosítás (MCM) – az objektumok közötti értékkülönbségek elemzését végzi. A COCO a többszemponútú döntéstámogatás körébe tartozik, MI vonatkozása az automatizált, emberi súlyozás nélküli objektív értékelésben áll.

Az MI megoldások és a szabályalapú feldolgozás összehasonlítása a 4. fejezetben (Vita), az LLM-benchmark eredmények ismertetése pedig a 3.10.5. fejezetben (vö. Tesztelés és eredmények) található.

A szakdolgozat készítése során generatív mesterséges intelligenciát is használtam. Az eszközt a szakirodalom rendezéséhez, a fejezetek felépítéséhez, valamint a fejlesztés közben felmerült kódolási és integrációs kérdések tisztázásához vettem igénybe. Emellett a mondatok javításához, a hosszabb gondolatmenetek rövidebb megfogalmazásához és a szöveg logikai összefüggéseinek ellenőrzéséhez is alkalmaztam. A kódíráshoz, a hibakereséshez és a dokumentáció egyes részeihez LLM-alapú fejlesztőeszközt használtam: a Claude Code CLI (Anthropic) kódrészletek generálásában, refaktorálási javaslatokban és tesztesetek tervezésében volt segítségemre. A kapott kimeneteket átnéztem, majd a kódbázis stílusához és a biztonsági elvárásokhoz igazítva építettem be a rendszerbe. A felhasznált LLM-beszélgetések válogatott részletei a 8.6 mellékletben olvashatók.

3.10. Tesztelés és eredmények

A jelen fejezet a NewsCast rendszer tesztelési módszertanát, a konkrét teszteseteket és a teljesítménymérések eredményeit mutatja be. A tesztelés során mind az egyes modulok önálló működését, mind a modulok közötti együttműködést vizsgáltam.

3.10.1. Tesztelés

A tesztelés elsődleges célja annak verifikálása volt, hogy a NewsCast rendszer a 3.1 alfejezetben definiált funkcionális (FR-01 – FR-07) és nem funkcionális (NFR-01 – NFR-05) követelményeknek megfelel. Minden tesztesetnél konkrét, mérhető elvárt eredményt fogalmaztam meg:

- **Funkcionalitás:** Az egyes modulok a specifikáció szerinti bemenetre a specifikáció szerinti kimenetet produkálják-e (pl. az RSS Parser 62 forrást képes-e feldolgozni és/vagy a duplikációsűrő 0,80 feletti hasonlóság esetén jelzi-e a duplikátumot).
- **Teljesítmény:** A mért feldolgozási idők az NFR-01 követelményben meghatározott küszöbértékeken belül maradnak-e (pl. egy hír elemzése ≤ 30 sec, a TTS generálás ≤ 60 sec).
- **Hibatűrés:** A rendszer hibás bemenetekre (elérhetetlen URL, érvénytelen XML, ismeretlen kódolás) is graceful degradation-nel – azaz a teljes működés leállása helyett lokalizált, kezelt hibával – reagál-e (vö. NFR-02).
- **Biztonság:** A hitelesítési mechanizmusok megakadályozzák-e a jogosulatlan hozzáférést és/vagy a parametrikus lekérdezések kizárják-e az SQL injection-t (vö. NFR-03).

A NewsCast rendszer számos konfigurálható paraméterrel rendelkezik, amelyek módosítása közvetlen hatással van a rendszer viselkedésére. Az alábbi táblázat a legfontosabb bemeneti paramétereket és az azokra érzékeny kimenetek közötti kapcsolatokat foglalja össze.

Bemeneti paraméter	Modul	Hatás leírása
MAX_CONCURRENT_FEEDS	rss_parser	A párhuzamosan letöltött feedek számát szabályozza. Növelése gyorsítja a feldolgozást, de a célszerverek túlterhelésének kockázatát növeli.
duplicate_threshold (0,80)	analyze	A duplikációs küszöbérték csökkentése több hasonló cikket szűr ki (agresszívebb szűrés), de növeli a hamis pozitívok kockázatát, növelése több hasonló cikket enged át.
prestige (0,0–1,0)	rss_parser / analyze	Egy forrás presztízsértékének módosítása közvetlenül befolyásolja a radio_relevance_score értékét és a hírszelekciónak a sorrendet.
NEWS_FRESHNESS_HOURS (12)	feeder	Bővítése régebbi híreket is beenged a szelekcióba, szűkítése csak a legfrissebb híreket

Bemeneti paraméter	Modul	Hatás leírása
		engedi át, de kis hírforgalom esetén kevesebb mint 5 hír kerülhet kiválasztásra.
NEWS_DUPLICATE_FILTER_HOURS (24)	feeder	Csökkentése lehetővé teszi, hogy korábban sugárzott hírek hamarabb újra megjelenjenek, növelése hosszabb ideig kizárja a már közölt híreket.
NEWS_SELECT_COUNT (5)	feeder	A kiválasztott hírek számának módosítása közvetlenül befolyásolja a hírblokk hosszát és a TTS-generálási időt.
elevenlabs_stability (0,5)	tts	A TTS hangjának stabilitási paramétere: alacsonyabb érték expresszívabb beszédet eredményez, magasabb érték konzisztensebb hangzást biztosít.
elevenlabs_speed (1,0)	tts	A felolvasás sebessége: 1,0 alatti értékek lassabb, 1,0 feletti értékek gyorsabb beszédet eredményeznek.
safety_score küszöbértékek	analyze	A tartalombiztonsági küszöbértékek módosítása meghatározza, mely hírek jutnak el a szelekciós fázisig.
TITLE_MATCH_THRESHOLD (75)	social	A RapidFuzz fuzzy cím-illesztés küszöbértéke. Csökkentése több egyezést eredményez, de növeli a hamis pozitívok arányát.
SIGNAL_WINDOW_HOURS (24)	social	A közösségi jelek érvényességi ablaka.
SUPER_FORMULA_POINTS_TRENDS (50)	social	A Szuper Képlet trends komponensének súlya.

15. táblázat: A bemeneti paraméterek módosításának hatásai a rendszer viselkedésére

A fenti összefüggések ismerete a tesztelés megtervezése és a rendszer finomhangolása szempontjából is lényeges: a tesztesetek során az egyes paraméterek szélsőértékeit is vizsgáltam, hogy a rendszer a normál üzemi tartományon kívüli bemenetekre is kiszámíthatóan reagáljon.

3.10.2. Tesztelési módszertan

A tesztelési módszertan a tesztpiramis elvét követi: a tesztelési szintek (3.10.2.1) a unit, integrációs és rendszertesztek rétegeit, a tesztelési eszközök (3.10.2.2) az alkalmazott eszközöket és a Docker health check konfiguráció (3.10.2.3) a konténerszintű állapotellenőrzést mutatják be.

3.10.2.1. Tesztelési szintek

A NewsCast rendszer tesztelése a tesztpiramis (vö. Fowler, M.: „TestPyramid”, martinowler.com, 2012; lásd 8.5 Hivatkozások) elvét követve három szinten történt:

Unit tesztek: Az egyes függvények és osztályok izolált tesztelése, a külső függőségek kizárásával. A TTS modul számára dedikált pytest alapú tesztfájlok készültek:

- `test_normalizer.py`: a szövegnormalizálás 14 lépésének tesztelése – rövidítés feloldás, speciális karakterek kezelése, számok szövegre alakítása és a teljes normalizálási pipeline
- `test_new_features.py`: a tizedesvessző és időformátumok kezelése, kötőjel/mínusz megkülönböztetése és a százalék toldalékolás tesztelése
- `test_custom.py`: időjárás szövegek, sorszámok és hőmérséklet formátumok egyedi tesztesetei
- `test_natural_speech.py`: a természetes beszéd minőségének tesztelése – dátumok, időpontok, telefonszámok, irányítószámok és tizedesszámok felolvasási alakja.

A `newscast-analyze` modul integrált tesztfüggvényt tartalmaz a COCO API kommunikáció validálására: `coco_client.py` `test_coco_client()`.

Integrációs tesztek: A modulok közötti kommunikáció és az adatbázis műveletek tesztelése. Ide tartoznak az API végpontok tesztelése HTTP kérésekkel, az adatbázis műveletek end-to-end ellenőrzése és a külső szolgáltatásokkal (OMSZ, ElevenLabs, COCO API) való kommunikáció validálása. A `test_auth.sh` a TTS modul hitelesítési mechanizmusait teszteli automatizált curl hívásokkal: a publikus végpontok elérhetőségét, a végpontok HTTP 401-es válaszát, a Basic Auth és JWT Bearer Token hitelesítést, valamint az érvénytelen hitelesítő adatok elutasítását.

End-to-end tesztek: A teljes feldolgozási lánc tesztelése: kiindulva egy RSS feed letöltésétől az elemzésen és szelekción át a hanganyag generálásáig. A tesztek a Docker konténerekben futtatott teljes rendszeren történtek, a beépített health check folyamatos monitorozása mellett.

3.10.2.2. Tesztelési eszközök

Eszköz	Alkalmazás
<code>pytest</code>	A Python standard tesztelési keretrendszere, a TTS modul unit tesztjeihez
<code>pytest-asyncio</code>	Az aszinkron függvények tesztelésének támogatása, különösen az RSS Parser és a TTS modul aszinkron műveleteihez
<code>curl</code> / <code>httpie</code>	Az API végpontok manuális és automatikus tesztelése HTTP kérésekkel, a modul-dokumentációk mindegyike tartalmaz curl példákat

Eszköz	Alkalmazás
Bash szkriptek	A test_auth.sh a hitelesítési mechanizmusok automatikus tesztelését végzi, a benchmark_llm_*.sh scriptek az LLM-benchmark API hívásokat futtatják
Docker health check	Az alkalmazáskonténernek beépített egészségügyi ellenőrzése, amely 30 másodperces intervallumonként validálja a szolgáltatások elérhetőségét

16. táblázat: Tesztelési eszközök

3.10.2.3. Docker health check konfiguráció

A hat NewsCast modul közül négy rendelkezik beépített Docker health check konfigurációval:

Modul	Végpont	Intervallum	Timeout	Újrapróbálkozás
newscast-rss_parser	/health/live	30s	10s	3
newscast-weather	/health/ready	30s	10s	3
newscast-feeder	/health/ready	30s	10s	3
newscast-tts	/api/v1/health	30s	10s	3
newscast-analyze	–	–	–	–
newscast-social	–	–	–	–

17. táblázat: NewsCast modulok health check konfigurációk

Az analyze és social modulok a Dockerfile-ban explicit health check nélkül működnek, a Prometheus metrikákon (vö. 3.8.1 fejezet) és a naplózáson keresztül monitorozhatóak.

3.10.3. Tesztesetek bemutatása

A jelen alfejezet a NewsCast rendszer hat moduljára vonatkozó, különböző teszteseteket tartalmazza. Minden tesztesetnél a bemenetet, az elvárt eredményt és a tényleges eredményt dokumentáltam. A teszteset azonosítók az érintett modul nevére utalnak (TC-RSS, TC-ANA, TC-WEA, TC-FEE, TC-SOC, TC-TTS).

3.10.3.1. RSS Parser tesztesetek

TC-RSS-01: Sikeres RSS feed feldolgozás

- *Bemenet:* Érvényes RSS feed URL (pl. <https://telex.hu/rss>).
- *Elvárt eredmény:* A feed tartalma letöltődik, a bejegyzések a news táblába kerülnek, az rss tábla last_process_date, etag és last_modified mezői frissülnek.

- *Eredmény:* Sikeres. A 62 forrásból átlagosan 52-58 forrás ad vissza érvényes feedet egy ciklusban.

TC-RSS-02: HTTP 304 Not Modified kezelés

- *Bemenet:* Olyan RSS feed URL, amelyhez az előző ciklusban érvényes ETag tartozik.
- *Elvárt eredmény:* A szerver HTTP 304 válasszal jelzi, hogy a tartalom nem változott, a rendszer nem tölti le újra a feedet, de a last_process_date frissül.
- *Eredmény:* Sikeres. Az ETag-alapú gyorsítótárazás az esetek ~70%-ában HTTP 304 választ eredményez (vö. RFC 7232).

TC-RSS-03: Hibás feed kezelés

- *Bemenet:* Elérhetetlen URL vagy érvénytelen XML tartalmú feed.
- *Elvárt eredmény:* A rendszer naplózza a hibát, de a többi feed feldolgozása nem akad meg (vö. NFR-02).
- *Eredmény:* Sikeres. A szemafor alapú párhuzamosítás gondoskodik arról, hogy egy feed hibája ne blokkolja a többi feldolgozását.

TC-RSS-04: Duplikáció megelőzés

- *Bemenet:* Két egymást követő ciklusban ugyanaz a hír jelenik meg egy feedben.
- *Elvárt eredmény:* A hír csak egyszer kerül az adatbázisba.
- *Eredmény:* Sikeres. Az IntegrityError kezelése garantálja, hogy a duplikált bejegyzés nem okoz rendszerhibát.

TC-RSS-05: Rate limit (429) kezelés

- *Bemenet:* Olyan forrás, amely 429-es státuskóddal válaszol.
- *Elvárt eredmény:* A rendszer exponenciális visszalépéssel újrapróbálja a letöltést (vö. 3.6.1 fejezet).
- *Eredmény:* Sikeres.

3.10.3.2. Analyze modul tesztesetek

TC-ANA-01: NLP elemzés magyar szövegen

- *Bemenet:* 500 szavas magyar nyelvű hírszöveg.
- *Elvárt eredmény:* A HuSpaCy hu_core_news_lg modell sikeresen tokenizálja a szöveget, a szentiment, olvashatóság és entitások kinyerésre kerülnek.
- *Eredmény:* Sikeres. A HuSpaCy modell megbízhatóan kezeli a magyar szövegeket, beleértve az ékezetes karaktereket és a ragozott alakokat.

TC-ANA-02: Duplikációsűrés pontossága

- *Bemenet:* Két, tartalmilag azonos hír eltérő forrásból (pl. ugyanaz az MTI közlemény az Indexen és a Telexen).
- *Elvárt eredmény:* A duplikációs pontszám meghaladja a 0,80-as küszöbértéket.
- *Eredmény:* Sikeres. A kétfázisú algoritmus (cím alapú előszűrés + TF-IDF tartalomellenőrzés) az MTI átvételeket megbízhatóan azonosítja (vö. 3.6.2.2 fejezet).

TC-ANA-03: Reklámsűrés hatékonysága

- *Bemenet:* Hirdetési blokkokat tartalmazó híroldal.
- *Elvárt eredmény:* A ContentCleaner eltávolítja a hirdetési tartalmat, a megtisztított szöveg csak a hírértékű részt tartalmazza.
- *Eredmény:* Sikeres. Az 53 statikus hirdetési minta (12 jelölő, 23 kulcsszó, 18 HTML-minta) lefedi a leggyakoribb hirdetési formátumokat. A tesztek során kb. 5-25%-os tartalomcsökkentés volt megfigyelhető, kizárólag a reklám célú (hirdetési és promóciós) tartalmak eltávolításával.

TC-ANA-04: ContentSafetyAnalyzer pontossága

- *Bemenet:* 10 teszhír, amelyek a rádiós tartalmak spektrumát lefedik (politikai, bűnügyi, technológiai, sport, reklám, szexuális tartalom).
- *Elvárt eredmény:* A kulcsszó alapú biztonsági pontszámok a súlyozási képletnek megfelelően alakulnak (erőszak *10, szexuális *8, drog *7, politikai *2), a küszöbérték (< 8) feletti tartalmak kiszűrésre kerülnek.
- *Eredmény:* Sikeres. A részletes eredmények a 3.10.5.2 alfejezetben (hírelemzési benchmark) dokumentáltak.

3.10.3.3. Weather modul tesztesetek

TC-WEA-01: OMSZ ZIP feldolgozás

- *Bemenet:* Windows-1250 kódolású, ZIP-tömörített időjárás-előrejelzés.
- *Elvárt eredmény:* A szöveg helyesen dekódolódik magyar ékezetes karakterekkel.
- *Eredmény:* Sikeres. A chardet könyvtár a Windows-1250 kódolást 95%+ megbízhatósággal felismeri és a fallback lánc a fennmaradó eseteket is kezeli.

TC-WEA-02: Dátumvalidáció három formátummal

- *Bemenet:* Előrejelzések „2026.04.10”, „2026-04-10” és „10.04.2025” dátumformátumokkal.
- *Elvárt eredmény:* Mindhárom formátum helyesen felismerésre kerül.
- *Eredmény:* Sikeres. A három reguláris kifejezés mintája (vö. 3.6.3.1 fejezet) kielégíti az összes megfigyelt OMSZ dátumformátumot.

TC-WEA-03: Duplikáció megelőzés content hash-el

- *Bemenet:* Ugyanaz az előrejelzés kétszer kerül letöltésre.
- *Elvárt eredmény:* A második mentési kísérlet a UNIQUE adatbázis kényszer miatt nem hoz létre új rekordot.
- *Eredmény:* Sikeres. Az SHA-256 content_hash megbízhatóan azonosítja az ismétlődő tartalmakat.

3.10.3.4. Feeder modul tesztesetek

TC-FEE-01: Top 5 hír kiválasztás

- *Bemenet:* 20+ elérhető rádió alkalmas hír az adatbázisban.
- *Elvárt eredmény:* Pontosan 5 hír kerül kiválasztásra a radio_relevance_score csökkenő sorrendjében.
- *Eredmény:* Sikeres. A szelektációs algoritmus konzisztensen a legmagasabb pontszámú híreket választja (vö. 3.6.4.1 fejezet).

TC-FEE-02: 24 órás duplikációszűrés

- *Bemenet:* Az előző ciklusban kiválasztott hír a legmagasabb pontszámmal rendelkezik.
- *Elvárt eredmény:* A korábban kiválasztott hír kiszűrésre kerül és a következő legmagasabb pontszámú hír kerül a helyére.
- *Eredmény:* Sikeres.

TC-FEE-03: Narratíva flow formázás

- *Bemenet:* 5 kiválasztott hír és egy konfigurált narratíva flow (intro → ai_lead * 5 → outro).
- *Elvárt eredmény:* A formázott szöveg tartalmazza az intro szöveget, az 5 hír ai_lead-jét (vagy summary) SSML szünetekkel és az outro szöveget.
- *Eredmény:* Sikeres. A <break time=„Xs”/> SSML tagek (vagy ElevenLabs v3 esetén „...”) helyesen kerülnek beillesztésre (vö. 3.6.4.2 fejezet).

3.10.3.5.Social modul tesztesetek

TC-SOC-01: Google News RSS gyűjtés

- *Bemenet:* 7 Google News RSS feed kategória (top, nation, world, business, tech, science, health).
- *Elvárt eredmény:* A bejegyzések a social_signals táblába kerülnek a megfelelő platform, cím, URL és engagement_score mezőkkel. A Google News proxy URL-ek feloldásra kerülnek.
- *Eredmény:* Sikeres. A collect job 20 perces ciklusban átlagosan 50–120 új jelet rögzít.

TC-SOC-02: Google Trends kulcsszó kinyerés

- *Bemenet:* Google Trends RSS feed (geo=HU).
- *Elvárt eredmény:* A TOP10 trending kulcsszó és a hozzá tartozó becsült forgalom a trending_keywords táblába kerül és az in-memory cache frissül.
- *Eredmény:* Sikeres. A trends job 60 perces ciklusban 5-10 kulcsszót rögzít.

TC-SOC-03: Háromszintű URL egyeztetés

- *Bemenet:* Azonos hír két eltérő URL-ről (pl. Google News proxy és eredeti forrás-URL).

- *Elvárt eredmény:* Az egyeztetés a háromszintű algoritmus valamelyik szintjén (url_hash, canonical, fuzzy title) sikerül, a matched_news_id és match_method mezők kitöltődnek (vö. 3.6.6.3 fejezet).
- *Eredmény:* Sikeres. Az url_hash egyeztetés az esetek ~40%-ában, a canonical ~30%-ában, a fuzzy title ~20%-ában sikerül. A fennmaradó ~10% párosítatlan marad.

TC-SOC-04: Szuper képlet pontszámítás

- *Bemenet:* Egy egyeztetett hír, amelynek címe tartalmaz trending kulcsszót.
- *Elvárt eredmény:* A social_trending_score értéke 60 (= 10 + 1 * 50) és a radio_relevance_score +15 boost-ot kap (vö. 3.6.6.4 fejezet).
- *Eredmény:* Sikeres.

TC-SOC-05: Backfill mechanizmus (race condition)

- *Bemenet:* Olyan news_id, amelyhez az analysis sor még nem létezik (a newscast-analyze modul nem dolgozta fel).
- *Elvárt eredmény:* A news_id a pending backfill sorba kerül és a következő match_job futásnál sikeresen beíródik, miután az analyze modul létrehozta az analysis sort (vö. 3.6.6.5 fejezet).
- *Eredmény:* Sikeres. A /status végpont pending_backfill mezője jelzi a várakozó elemek számát, amelyek jellemzően 1-2 cikluson belül feloldódnak.

3.10.3.6. TTS modul tesztesetek

TC-TTS-01: Magyar szövegnormalizálás – Számok

- *Bemenet:* „2024 embernek 15%-os kedvezményt adtak”
- *Elvárt eredmény:* „kétezer-huszonegy embernek tizenöt százalékos kedvezményt adtak”
- *Eredmény:* Sikeres. A number_to_words és normalize_percentages függvények együttesen helyesen kezelik.

TC-TTS-02: Hőmérséklet-normalizálás

- *Bemenet:* „Holnap -5°C lesz, ma 25,5°C volt”

- *Elvárt eredmény:* „Holnap mínusz öt fok lesz, ma huszonöt és fél fok volt”
- *Eredmény:* Sikeres. A negatív hőmérsékletnél a „mínusz” megjelenik, a pozitívnál nem, a tizedesvessző kezelése helyes.

TC-TTS-03: SSML tag védelem normalizálás során

- *Bemenet:* „első hír. <break time=„1s”/> második hír.”
- *Elvárt eredmény:* A break tag változatlanul megmarad a normalizálás után.
- *Eredmény:* Sikeres. A protect_ssml_tags / restore_ssml_tags mechanizmus megbízhatóan működik (vö. 3.6.5.4 fejezet).

TC-TTS-04: Content hash deduplikáció

- *Bemenet:* Két azonos szövegű TTS generálási kérés.
- *Elvárt eredmény:* A második kérésnél a korábban generált hangfájl kerül visszaadásra az ElevenLabs API meghívása nélkül.
- *Eredmény:* Sikeres. A was_reused mező True-ra állítódik és a reused_from_task_id a korábbi feladat azonosítóját tartalmazza (vö. 3.6.5.3 fejezet).

TC-TTS-05: Rövidítések feloldása

- *Bemenet:* „Dr. Kiss, kb. 100 km/h sebességgel, stb.”
- *Elvárt eredmény:* „doktor Kiss, körülbelül száz kilométer per óra sebességgel és a többi.”
- *Eredmény:* Sikeres. A 22 előre definiált rövidítés (kiegészítve 4 kontextusfüggő rövidítéssel és 24 speciális karakter szabállyal, összesen 50 normalizálási szabály) az ABBREVIATIONS szótárban helyesen kerül feloldásra.

TC-TTS-06: Hitelesítési mechanizmusok

- *Bemenet:* A test_auth.sh szkript automatikus futtatása, amely 10 HTTP kérést hajt végre különböző hitelesítési kombinációkkal.
- *Elvárt eredmény:* A publikus végpontok (health) hitelesítés nélkül elérhetők (HTTP 200), a védett végpontok hitelesítés nélkül HTTP 401-es választ adnak, érvényes Basic

Auth és JWT Bearer Token esetén HTTP 200-as választ adnak, érvénytelen hitelesítő adatok esetén HTTP 401-es választ adnak (vö. NFR-03).

- *Eredmény:* Sikeres. Mindegyik tesztet a várt eredményt adta.

3.10.4. Teljesítménytesztek

A teljesítménytesztek célja annak igazolása, hogy a NewsCast rendszer az NFR-01 követelményben meghatározott küszöbértékeken belül működik. A mérések a 3.5.3 fejezetben (vö. Futtató környezet) leírt RackForest VPS szerveren (4 vCPU, 8 GB RAM, 80 GB SSD) történtek, Docker konténerekben futtatott környezetben.

3.10.4.1. RSS feldolgozási teljesítmény

A 62 RSS forrás egyidejű feldolgozásának teljesítményét az alábbi mérési eredmények jellemzik:

Metrika	Mért érték	NFR-01 küszöb
Teljes feldolgozási idő	2-5 perc/ciklus	≤ 10 perc (50 forrás)
Átlagos feed letöltési idő	1-3 másodperc/feed	-
Áteresztőképesség	~500-1000 cikk/óra	-
Sávszélesség megtakarítás (ETag)	~70%	-
Memóriahasználat	200-300 MB	-

18. táblázat: Az RSS feldolgozás teljesítménymutatói

A 70%-os sávszélesség megtakarítás azt jelenti, hogy az óránkénti és ciklusonkénti 62 HTTP kérésből átlagosan ~43 darab HTTP 304 választ kap (*a tartalom nem változott*) és csak ~19 feed esetén szükséges a teljes tartalom letöltése. A feldolgozási ciklus 2-5 perces mért ideje bőven az NFR-01 által meghatározott 10 perces küszöb alatt marad.

3.10.4.2. Elemzési teljesítmény

Metrika	Mért érték	NFR-01 küszöb
Elemzési idő (teljes NLP pipeline)	8-12 másodperc/cikk	≤ 30 másodperc
Duplikációsűrés (2 fázisú, 500 cikk)	~7 másodperc	≤ 10 másodperc
Klaszterezés (500 cikk)	3-5 másodperc	-

Metrika	Mért érték	NFR-01 küszöb
Gemini kötegelt feldolgozás	50 elem/15 perc	-

19. táblázat: Az elemzési modul teljesítménymutatói

Az egy cikkre jutó 8-12 másodperces elemzési idő az NFR-01 által meghatározott 30 másodperces küszöb alatt marad.

3.10.4.3.Social modul teljesítmény

Metrika	Mért érték
Collect job futási idő (7 feed)	10-30 másodperc/ciklus
Trends job futási idő	2-5 másodperc/ciklus
Match job futási idő (~100 signal)	5-15 másodperc/ciklus
RapidFuzz fuzzy illesztés	~1 ms/összehasonlítás
Backfill feloldás	1-2 ciklus (5-10 perc)

20. táblázat: Social modul teljesítménymutatói

A social modul teljesítménye a Google News és Google Trends RSS feedek válaszüdejétől nagyban függ. A RapidFuzz C++ implementáció biztosítja, hogy a fuzzy cím illesztés nem jelent szűk keresztmetszetet a feldolgozásnál: 100 signal * 500 hír összehasonlítása kb. 1 másodperc alatt lefut.

3.10.4.4.TTS generálási teljesítmény

Metrika	Mért érték	NFR-01 küszöb
Generálási idő (5 hír)	15-30 másodperc/hírblokk	≤ 60 másodperc
ElevenLabs API válaszüdő	3-8 másodperc/kérés	-
Szövegnormalizálás	< 100 ms	-
Deduplikáció ellenőrzés	< 10 ms	-

21. táblázat: A TTS modul teljesítménymutatói

A TTS generálás egyik szűk keresztmetszete az ElevenLabs API válaszüdeje (3-8 másodperc), amely a szöveghossztól és a szolgáltatás terhelésétől függ. A deduplikáció (vö. 3.6.5.3 fejezet) az API használati költséget jelentősen csökkenti: ismételt időjárás előrejelzések vagy változatlan hírblokkok esetén a korábban generált hangfájl azonnal visszaadható, újra generálás

nélkül. A hírblokkonkénti 15-30 másodperces generálási idő az NFR-01 által meghatározott 60 másodperces küszöb alatt marad.

3.10.4.5. Teljes pipeline áteresztőképesség

A teljes feldolgozási lánc – az RSS letöltéstől a hanganyag generálásáig – áteresztőképessége:

Lépés	Mért idő
RSS feldolgozás (62 forrás)	2-5 perc
Elemzés (új cikkek, ~50/ciklus)	4-10 perc
Social signal gyűjtés és egyeztetés	15-45 másodperc
Hírszelekció (TOP 5)	< 1 másodperc
Szövegformázás és normalizálás	< 1 másodperc
TTS generálás (ElevenLabs API)	15-30 másodperc
Teljes feldolgozási idő	~7-17 perc/hírblokk

22. táblázat: A teljes feldolgozási pipeline átfutási ideje

Az óránkénti (3600 másodperces) ütemezési intervallumba a teljes pipeline lefutása bőven belefér, elegendő tartalékot hagyva az esetleges hálózati késleltetésekre és újrapróbálkozásokra.

3.10.4.6. Követelmény – megfelelés összefoglalása

Az alábbi táblázat a funkcionális és nem funkcionális követelmények teljesülését foglalja össze a tesztelési eredmények alapján:

Követelmény	Leírás	Eredmény
FR-01	RSS gyűjtés: 62 forrás párhuzamos feldolgozása	Teljesül (2-5 perc, ~70% ETag megtakarítás)
FR-02	Elemzés: NLP pipeline, duplikációsűrés, rádiós relevancia	Teljesül (8-12 s/cikk)
FR-03	OAM: COCO API integráció	Teljesül (remote mód, automatikus teszt)
FR-04	Időjárás: OMSZ adatok feldolgozása	Teljesül (karakterkódolás felismerés, 3 dátumformátum)
FR-05	TOP5 hír kiválasztás, 24 órás szűrés, narratíva flow	Teljesül
FR-06	Social: Google News/Trends, URL-egyeztetés, Szuper képlet	Teljesül (~90% match arány)
FR-07	TTS: normalizálás, deduplikáció, ElevenLabs API integráció	Teljesül (15-30 s/hírblokk)

Követelmény	Leírás	Eredmény
NFR-01	Teljesítmény: elemzés $\leq 30s$, RSS ≤ 10 perc, TTS $\leq 60s$	Teljesül
NFR-02	Megbízhatóság: retry, hibaelkülönítés	Teljesül
NFR-03	Biztonság: hitelesítés, parametrikus SQL, API kulcskezelés	Teljesül (test_auth.sh: 10/10)
NFR-04	Üzemeltetés: Docker, health check, Prometheus	Teljesül (4/6 modul health check)
NFR-05	Bővíthetőség: moduláris architektúra	Teljesül (6 független modul)

23. táblázat: Követelmény – megfelelési mátrix

3.10.5. LLM-benchmark

A rendszer tesztelésének speciális rétegét képezi az LLM-benchmark vizsgálat, amelyben a NewsCast saját, szabályalapú megoldásainak kimenetét három nagy nyelvi modell (Large Language Model, LLM) eredményeivel kerül összehasonlításra. A vizsgálat célja annak megállapítása, hogy a manuálisan fejlesztett, determinisztikus modulok teljesítménye hogyan viszonyul egy általános célú LLM „out-of-the-box” megoldásához – vagyis: a célzott fejlesztés teremt-e hozzáadott értéket az LLM-ek generikus képességeihez képest. Az eredmények közvetlenül kapcsolódnak a 4. fejezet (Vita) reflexióihoz.

3.10.5.1. A benchmark célja és módszertana

A benchmark motivációja

A NewsCast rendszer fejlesztése során számos olyan feladat merült fel, amelyek alternatív megoldásaként egy általános célú LLM is szóba jöhetett volna: a tartalombiztonsági szűrés, a szentimentelemzés, a szövegnormalizálás és a szövegösszegzés egyaránt olyan területek, amelyeken a nagy nyelvi modellek jelentős előnyöket mutatnak. A benchmark vizsgálat célja, hogy tételesen és mérhető módon összehasonlítsuk a saját implementáció kimeneteit az LLM-ek válaszaival, azonos bemenetek és értékelési szempontok mellett. A benchmark módszertan kialakításánál a modellezési szakértők teljesítményének értékelési keretrendszere (vö. Pitlik L., 2015; lásd 8.5 Hivatkozások) szolgált kiindulópontul az összehasonlíthatóság biztosításához.

A vizsgálat három kérdésre keresi a választ:

1. A NewsCast szabályalapú megoldásai konzisztensebb eredményt adnak-e, mint az LLM-ek sztochasztikus kimenetei?

2. A magyar nyelvi sajátosságok kezelésében (ragozás, tizedesvessző, sorszámok, birtokos ragok) a célzott szabályrendszer pontosabb-e az általános célú modellnél?
3. Mely részfeladatoknál nyújt érdemi hozzáadott értéket az LLM – és hol elegendő a determinisztikus megoldás?

A vizsgált LLM-ek

A benchmark vizsgálatban az alábbi három nagy nyelvi modellt alkalmaztam:

Modell	Szolgáltató	API model ID	Hozzáférés dátuma
GPT-5.4-mini	OpenAI	gpt-5.4-mini	2026. április 4.
Claude Haiku 4.5	Anthropic	claude-haiku-4-5-20251001	2026. április 4.
Gemini 3.1 Flash Lite	Google	gemini-flash-lite-latest (gemini-3.1-flash-lite-preview)	2026. április 4.

24. táblázat: Az LLM-benchmark vizsgálatban alkalmazott nagy nyelvi modellek

A modellek kiválasztásánál szempont volt, hogy mindhárom vezető LLM-szolgáltató könnyűsúlyú (lightweight) modellje kerüljön be a vizsgálatba. A könnyűsúlyú kategória választása tudatos döntés: amennyiben egy költséghatékony, kis modell is képes az adott feladatot megoldani, az a szabályalapú megoldás hozzáadott értékét hangsúlyozza, amennyiben nem, az a célzott fejlesztés indokoltságát támasztja alá.

API alapú tesztelési módszertan

A benchmark tesztek végrehajtása mindhárom szolgáltató hivatalos REST API felületén keresztül történt, curl parancsokkal. Az API alapú megközelítés a webes chat felülettel szemben számos előnnyel rendelkezik: a kérések és válaszok programozottan reprodukálhatók, a modell-azonosító és a paraméterek (temperature, max_tokens) pontosan rögzíthetők és a teljes HTTP kérés/válasz látható, archiválható.

Az alábbi curl parancsok a három szolgáltató API hívásának sémáját mutatják be. A `<PROMPT>` helyére a benchmark prompt teljes szövege kerül, az `<API_KEY>` helyére pedig a szolgáltató API kulcsa:

OpenAI (GPT-5.4-mini):

```
curl -s https://api.openai.com/v1/chat/completions \
  -H „Content-Type: application/json” \
  -H „Authorization: Bearer $OPENAI_API_KEY” \
  -d '{
```

```

    „model”: „gpt-5.4-mini”,
    „temperature”: 0,
    „messages”: [
      {„role”: „system”, „content”: „Válaszolj kizárólag JSON
formátumban.”},
      {„role”: „user”, „content”: „<PROMPT>„}
    ]
  }'

```

Anthropic (Claude Haiku 4.5):

```

curl -s https://api.anthropic.com/v1/messages \
-H „Content-Type: application/json” \
-H „x-api-key: $ANTHROPIC_API_KEY” \
-H „anthropic-version: 2023-06-01” \
-d '{
  „model”: „claude-haiku-4-5-20251001”,
  „max_tokens”: 4096,
  „temperature”: 0,
  „messages”: [
    {„role”: „user”, „content”: „<PROMPT>„}
  ]
}'

```

Google (Gemini 3.1 Flash Lite):

```

curl -s „https://generativelanguage.googleapis.com/v1beta/models/gemini-
flash-lite-latest:generateContent?key=$GOOGLE_API_KEY” \
-H „Content-Type: application/json” \
-d '{
  „contents”: [{„parts”: [{„text”: „<PROMPT>„}]}],
  „generationConfig”: {„temperature”: 0}
}'

```

Mindhárom API hívásnál a temperature paraméter 0-ra lett állítva a reprodukálhatóság maximalizálása érdekében. A válaszok JSON formátumban kerültek mentésre és a mellékletekben (vö. 8.6 fejezet) a teljes HTTP kérés/válasz dokumentálásra került.

Tesztadatok és módszertan

A benchmark három részfeladatot vizsgált:

1. **Hírelemzés és rádiós relevancia** (10 tesztír): a ContentSafetyAnalyzer, a RadioRelevanceCalculator és az NLP-alapú szentiment/olvashatósági elemzés kimeneteit veti össze az LLM-ek válaszaival. A tesztírek a tipikus rádiós tartalmak teljes spektrumát lefedik: politikai, bűnügyi, technológiai, helyi, sport, időjárás, idegen nyelvű és reklámtartalom egyaránt szerepel.
2. **Szövegnormalizálás** (30 tesztmondattal): a text_normalizer.py modul 14 normalizálási lépésének kimenetét hasonlítottam össze az LLM-ek normalizálásával. A tesztmondatok a magyar nyelv valamennyi releváns normalizálási esetét tartalmazzák: hőmérséklet,

dátum, időpont, százalék, pénznem, mértékegység, rövidítés, sorszám, számtartomány és speciális karakter.

3. **Szövegösszegzés** (5 tesztírt): a Sumy LexRank extraktív összegzés kimenetét vettem össze az LLM-ek extraktív és absztraktív összegzéseivel. A tesztírték tematikailag változatosak: gazdasági, baleseti, sport, oktatási és technológiai hír.

A NewsCast referencia kimeneteit a rendszer saját moduljai állították elő, önálló benchmark scriptek segítségével. A scriptek a `text_normalizer.py` (TTS normalizálás), a `NewsSummarizer / LexRank` (összegzés) és a `ContentSafetyAnalyzer / RadioRelevanceCalculator / NLP pipeline` (hírelemzés) pontos logikáját implementálják. Az LLM-eknek a NewsCast pontozási formuláit és szabályrendszerét tartalmazó promptot adtam, hogy azonos feltételek mellett értékelhessem a kimeneteket. A promptok, curl parancsok és válaszok a mellékletekben (vö. 8.6 fejezet) megtalálhatóak.

3.10.5.2. Hírelemzési benchmark eredményei

A hírelemzési benchmark során 10 tesztírt elemeztem a NewsCast pipeline-jával és a három LLM-mel. A NewsCast referencia kimeneteit a `benchmark_news_analysis.py` script állította elő, a `HuSpacy hu_core_news_lg` nyelvi modellt, a `ContentSafetyAnalyzer` tartalombiztonsági elemzőt és a `RadioRelevanceCalculator` rádiós relevancia számítót alkalmazva.

A NewsCast referencia eredményei

Az alábbi táblázat a NewsCast rendszer tényleges kimeneteit tartalmazza a 10 tesztírtre:

#	Szentiment	Olvashatóság	Safety score	Rádiós (igen/nem)	Rádió score	Kategória	Alkalmas (igen/nem)	Reklám (igen/nem)
1	positive	difficult	8	nem	10	low	nem	nem
2	neutral	difficult	54	nem	5	low	nem	nem
3	neutral	difficult	0	igen	55	medium	igen	nem
4	negative	difficult	2	igen	65	high	igen	nem
5	positive	difficult	0	igen	55	medium	igen	igen
6	neutral	difficult	0	igen	75	high	igen	nem
7	positive	difficult	0	igen	65	high	igen	nem
8	neutral	difficult	0	igen	55	medium	igen	nem
9	neutral	difficult	0	igen	65	high	igen	nem

#	Szentiment	Olvashatóság	Safety score	Rádiós (igen/nem)	Rádió score	Kategória	Alkalmas (igen/nem)	Reklám (igen/nem)
10	neutral	difficult	24	nem	5	low	nem	nem

25. táblázat: A NewsCast hírelemzési pipeline referencia-kimenetei

A referencia eredmények több figyelemre méltó sajátosságot mutatnak. A Flesch-Kincaid magyar adaptáció (vö. 3.6.2 fejezet) valamennyi tesztírt „difficult” kategóriába sorolta, ami a magyar nyelv agglutináló jellegéből és a hírszövegek átlagosan hosszú mondataiból adódik. A biztonsági pontszámok konzisztensen a kulcsszó alapú súlyozást tükrözik: a 2. tesztírt (bűnügyi tartalom) 54 pontos safety score-ja az erőszak (4*10) és drog (2*7) kulcsszavak kumulált hatását mutatja. A reklámdetektálás kizárólag az 5. tesztírtnél jelezte a hirdetési tartalmat, ahol a jelölők száma elérte a küszöbértéket (≥ 2 találat).

Összehasonlító eredmények az LLM-ekkel

A 10 tesztírt * 4 fő mező (sentiment, is_safe_for_radio, radio_category, is_ad_content) = 40 összehasonlítási pont alapján az alábbi egyezési arányok adódtak:

Mező	NewsCast (ref.)	GPT-5.4-mini egyezés	Claude Haiku 4.5 egyezés	Gemini Flash Lite egyezés
sentiment (10)	referencia	7 / 10	4 / 10	6 / 10
is_safe_for_radio (10)	referencia	9 / 10	8 / 10	9 / 10
radio_category (10)	referencia	7 / 10	7 / 10	6 / 10
is_ad_content (10)	referencia	10 / 10	10 / 10	10 / 10
Összesen (40)	referencia	33 / 40 (82,5%)	29 / 40 (72,5%)	31 / 40 (77,5%)

26. táblázat: A hírelemzési benchmark összesítő eredményei

Kvalitatív elemzés

A hírelemzési benchmark a NewsCast és az LLM-ek közötti eltérések több jellegzetes mintázatát tárta fel:

Szentiment meghatározás: A NewsCast HuSpacy lemmatizáción alapuló szentimentelemzése a szólistában szereplő lemmák arányát vizsgálja (pozitív küszöb: $pos > neg * 1,5$; negatív küszöb: $neg > pos * 1,5$). Az LLM-ek ezzel szemben a teljes kontextust figyelembe véve, pragmatikai szinten értékelnek. Ez az eltérés különösen a 4. tesztírtnél (*Balaton vízszintje*) mutatkozott meg: a NewsCast „negative” értékelést adott a „csökkent” és „törölni kényszerült” lemmák alapján, míg az LLM-ek a hír semleges, tájékoztató jellegét emelték ki.

Tartalombiztonsági pontozás: A kulcsszó alapú rendszer determinisztikus és reprodukálható pontszámokat ad, amelyek kizárólag a szóelőfordulástól függenek. Az LLM-ek a kontextust is figyelembe veszik: a 10. teszthírnél (szexuális tartalom) az LLM-ek felismerhetik, hogy a szöveg nem pornográf, hanem a jelenség elleni fellépésről szól – a NewsCast kulcsszó alapú rendszere ezt a distinkciót nem teszi meg.

Olvashatóság: A Flesch-Kincaid magyar adaptáció a szótagszám/szószám és szószám/mondatzám arányra épülő numerikus formula, amely a magyar nyelv jellegzetességei (hosszú szavak, agglutináció) miatt szisztematikusan alacsonyabb pontszámokat ad. Az LLM-ek a rádiós felolvashatóságot szubjektívebb, de pragmatikusabb szempontból ítélik meg.

3.10.5.3. Szövegnormalizálási benchmark eredményei

A szövegnormalizálási benchmark a `text_normalizer.py` modul 14 normalizálási lépésének kimenetét hasonlította össze az LLM-ek által produkált normalizált szövegekkel, 30 tesztmondaton. A NewsCast referencia kimeneteit a `benchmark_tts_normalize.py` script állította elő.

Kiértékelési szempontok

Minden mondat 6 kategóriában kapott 0 vagy 1 pontot:

Kategória	1 pont kritériuma
Számok	Minden szám helyesen szövegesítve
Rövidítések	Minden rövidítés helyesen feloldva
Dátumok	Helyes ragozás (-ike, -ikén, -ája, -áján)
Hőmérséklet	Helyes előjel + „fok” formátum
Mértékegységek	Helyes egységnevet és ragozást
Természetesség	Természetesen hangzik felolvasva

27. táblázat: Szövegnormalizálási teszt kiértékelési szempontok

A maximális pontszám: 30 mondat * 6 kategória = 180 pont.

Reprezentatív példák a NewsCast kimenetéből

Az alábbi táblázat néhány jellegzetes normalizálási esetet mutat be:

#	Eredeti mondat	NewsCast normalizált kimenet
1	„A hőmérséklet holnap -15°C körül alakul.”	„A hőmérséklet holnap mínusz tizenöt fok körül alakul.”
2	„2024. január 15-én kezdődik az új szemeszter.”	„kétezer-huszonegy, január tizenötödikén kezdődik az új szemeszter.”
6	„A termék ára 2500 Ft, kb. 7 € a jelenlegi árfolyamon.”	„A termék ára kétezer-ötszáz forint, körülbelül hét euró a jelenlegi árfolyamon.”
12	„Március 1-jén, 8:00-kor nyit az új bevásárlóközpont.”	„Március elsején, nyolc óra-kor nyit az új bevásárlóközpont.”
21	„A lakosság 68%-a támogatja, ez kb. 6800000 fő.”	„A lakosság hatvannyolc százaléka támogatja, ez körülbelül hatmillió-nyolcszázezer fő.”
27	„Június 1-je a gyermeknap, december 25-e a karácsony.”	„Június elseje a gyermeknap, december huszonötödike a karácsony.”

28. táblázat: Válogatott szövegnormalizálási példák a NewsCast kimenetéből (30-ból 6)

A teljes 30 mondatos referencia kimenet a benchmark_tts_results.json fájlban érhető el, amelyet a benchmark_tts_normalize.py script generált (vö. 8.6.1 Tesztadatok alfejezet).

A NewsCast normalizáló ismert korlátai

A benchmark során a szabályalapú normalizáló több esetben szuboptimális kimenetet produkált, amelyeket a kiértékelés során figyelembe kell venni:

- **Toldalékolási hibák:** A szabályrendszer a mértékegység- és időformátum toldalékokat kötőjellel fűzi a normalizált szöveghez (pl. „nyolc óra-kor”, „forint-tal”), ami a természetes kiejtéstől eltér. Az LLM-ek ezekben az esetekben természetesebb alakokat produkálhatnak.
- **Sorszám kontextus:** A 9. mondatnál a „89.” sorszám normalizálása „nyolcvankilenc.” formában maradt, nem „nyolcvankilencedik” alakban, ami a sorszám felismerés kontextusfüggő korlátjára utal.
- **Speciális karakter maradványok:** A 15. mondatnál az „m²” mértékegységből a „²” karakter nem konvertálódott „négyzetméter”-re.

Összesítő eredmények

#	NewsCast (/6)	GPT-5.4-mini (/6)	Claude Haiku 4.5 (/6)	Gemini Flash Lite (/6)
1	6	6	6	6
2	6	6	6	6

#	NewsCast (/6)	GPT-5.4-mini (/6)	Claude Haiku 4.5 (/6)	Gemini Flash Lite (/6)
3	6	5	5	6
4	6	6	6	6
5	4	6	6	6
6	6	6	6	6
7	6	6	6	5
8	6	6	6	6
9	4	6	6	6
10	5	5	3	6
11	6	6	5	6
12	6	6	6	6
13	6	6	6	6
14	6	6	6	6
15	4	6	6	6
16	4	5	5	5
17	3	6	6	6
18	6	6	6	6
19	3	5	4	6
20	6	5	6	6
21	6	6	6	6
22	6	6	6	6
23	6	6	6	6
24	6	6	6	6
25	6	6	6	6
26	6	6	6	6
27	6	6	6	6
28	6	6	6	4
29	2	6	6	6
30	4	6	6	6

#	NewsCast (/6)	GPT-5.4-mini (/6)	Claude Haiku 4.5 (/6)	Gemini Flash Lite (/6)
Összesen	159 / 180	171 / 180	168 / 180	172 / 180
Százalék	88,3%	95,0%	93,3%	95,6%

29. táblázat: A szövegnormalizálási benchmark összesítő eredményei

Kvalitatív elemzés

A szövegnormalizálási benchmark az alábbi mintázatokat tárta fel:

A szabályalapú rendszer erősségei: A determinisztikus szabályrendszer minden futtatásnál azonos kimenetet ad, ami a rádiós felolvasás konzisztenciája szempontjából előnyös. A számok szövegesítése, a hőmérséklet normalizálás és a rövidítések feloldása az esetek túlnyomó többségében helyes. A magyar birtokos ragozás (elseje, tizenötödike, huszonötödike) a dátumkezelő szabályok révén megbízhatóan működik.

Az LLM-ek potenciális előnyei: Az LLM-ek a kontextusfüggő normalizálásban – ahol a szabályrendszer nem rendelkezik megfelelő mintával – természetesebb kimenetet produkálhatnak. A toldalékolási hibák és a speciális karakter maradványok kezelése olyan terület, ahol az LLM-ek pragmatikai tudása előnyt jelent.

3.10.5.4. Szövegösszegzési benchmark eredményei

A szövegösszegzési benchmark 5 magyar nyelvű tesztírt vizsgált. A NewsCast rendszer a Sumy LexRank algoritmust alkalmazza (vö. 3.6.2 fejezet), amely extraktív összegzést készít: a szöveg legfontosabb mondatait választja ki változtatás nélkül, gráf alapú koszinusz-hasonlóság mátrix és sajátvektor-centralitás számítás alapján. Az összegzés maximálisan 3 mondatot tartalmaz. A tokenizálás a cseh nyelvi modellre épül (fallback: a magyar NLTK tokenizáló nem volt elérhető, a cseh nyelv hasonló szegmentálási szabályai elfogadható közelítést adnak).

A NewsCast LexRank referencia kimenetei

Az alábbi táblázat a benchmark_summarize.py script által előállított referencia összegzéseket mutatja be:

Hír	Eredeti hossz	Összegzés hossz	Tömörítési arány	LexRank összegzés (3 mondat)
A	1038 kar.	373 kar.	64%	„A Magyar Nemzeti Bank Monetáris Tanácsa keddi ülésén úgy döntött, hogy változatlanul 6,5 százalékon tartja a jegybanki alapkamatot. A döntést a tanácstagok egyhangúlag hozták meg. Virág Barnabás alelnök a döntés után tartott

Hír	Eredeti hossz	Összegzés hossz	Tömörítési arány	LexRank összegzés (3 mondat)
				sajtótájékoztatón kiemelte, hogy az inflációs kilátások továbbra is bizonytalanok, különösen az energiaárak és az élelmiszerárak volatilitása miatt.”
B	880 kar.	307 kar.	65%	„Tragikus baleset történt kedd reggel az M7-es autópályán Székesfehérvár közelében. Egy kamion és három személygépkocsi ütközött az 58-as kilométerszelvényben, a Budapest felé vezető oldalon. A balesetben két személy életét veszítette, öt további sérültet szállítottak kórházba, közülük kettő állapota súlyos.”
C	939 kar.	358 kar.	62%	„A Puskás Akadémia FC történelmi sikert ért el az Európa-liga csoportkörében: a felcsúti együttes 2-0-ra legyőzte a skót Rangers FC-t a Groupama Arénában. A mérkőzés első félidejében a Puskás dominált, 62 százalékos labdabirtoklással és 8 kapura lövéssel. Az első gólt Nagy Zsolt szerezte a 34. percben egy szabadrúgásból, amely a jobb felső sarokba vágódott.”
D	903 kar.	351 kar.	61%	„Az Innovációs és Technológiai Minisztérium bejelentette, hogy 2027-től minden magyar közoktatási intézményben kötelezővé teszik a mesterséges intelligencia alapjainak oktatását. A program az 5. osztálytól indul és fokozatosan bővül a középiskolai szintig. A tanterv kidolgozásában a BME, az ELTE és a Szegedi Tudományegyetem szakemberei vesznek részt.”
E	765 kar.	374 kar.	51%	„A globális chipgyártó ipar újabb mérföldkőhöz érkezett: a tajvani TSMC bejelentette, hogy megkezdte a 1,4 nanométeres csomóponttechnológiájú chippek tömeggyártását. Ez a világ legkisebb tranzisztorméretű chipje, amely az előző generációhoz képest 30 százalékkal gyorsabb és 40 százalékkal energiatakarékosabb. Az első megrendelők között van az Apple, az NVIDIA és a Qualcomm.”

30. táblázat: A NewsCast LexRank extraktív összegzései és tömörítési arányai

A LexRank algoritmus valamennyi esetben a cikk nyitó mondatait választotta ki, ami a hírszövegek „fordított piramis” szerkezetéből adódik: a legfontosabb információk az első mondatokban koncentrálódnak és a koszinusz-hasonlóság alapú centralitászámítás ezeket értékeli a legmagasabbra.

Kiértékelési szempontok

Az extraktív összegzésnél:

Szempont	Leírás	Max. pont
Relevancia	A kiválasztott 3 mondat a legfontosabb információkat tartalmazza-e?	3
Sorrend	A mondatok logikus sorrendben vannak-e?	1

Szempont	Leírás	Max. pont
Lefedettség	Az összegzés lefedi-e a hír lényegét (ki, mit, mikor, hol)?	2
Max / hír		6

31. táblázat: A NewsCast LexRank extraktív összegzéseinek kiértékelési szempontjai

Az absztraktív összegzésnél (kizárólag LLM-eknél, a NewsCast csak extraktív összegzést készít):

Szempont	Leírás	Max. pont
Információ megőrzés	A legfontosabb tények megmaradtak-e?	3
Tömörség	A mondatok rövidek és rádiósak-e? (max. ~20 szó)	2
Nyelvhelyesség	Nyelvtanilag helyes, természetes magyar?	1
Max / hír		6

32. táblázat: A NewsCast LexRank absztraktív összegzéseinek kiértékelési szempontjai

Összesítő eredmények

Hír	Típus	NewsCast (LexRank)	GPT-5.4-mini	Claude Haiku 4.5	Gemini Flash Lite
A	Extraktív (/6)	6	6	6	4
A	Absztraktív (/6)	n/a	6	6	5
B	Extraktív (/6)	6	5	6	5
B	Absztraktív (/6)	n/a	6	6	6
C	Extraktív (/6)	5	6	6	5
C	Absztraktív (/6)	n/a	6	6	5
D	Extraktív (/6)	6	6	6	6
D	Absztraktív (/6)	n/a	6	6	6
E	Extraktív (/6)	6	6	6	6
E	Absztraktív (/6)	n/a	6	5	6
Összesen	Extraktív (/30)	29	29	30	26
Összesen	Absztraktív (/30)	n/a	30	29	28

33. táblázat: A szövegösszegzési benchmark összesítő eredményei

Kvalitatív elemzés

A szövegösszegzési benchmark az alábbi megfigyelésekre vezetett:

Az extraktív megközelítés sajátosságai: A LexRank algoritmus a mondatok közötti koszinusz-hasonlóságon alapuló centralitásszámítással választja ki a legfontosabb mondatokat. Ez a módszer konzisztens és reprodukálható: azonos bemenetre mindig azonos kimenetet ad. A magyar nyelvű hírszövegek „fordított piramis” szerkezete esetén az algoritmus az első mondatokat preferálja, amelyek a legfontosabb információkat tartalmazzák. A módszer hátránya, hogy a kiválasztott mondatok nem feltétlenül alkotnak koherens, rádiós felolvasásra optimális szöveget: az eredeti mondathossz megmarad és a mondatok közötti logikai kötés hiányozhat.

Az LLM absztraktív összegzés előnyei: Az LLM-ek képesek az információ lényegét saját szavakkal, tömörebb formában megfogalmazni, ami a rádiós felolvasás szempontjából előnyösebb. A rövid, 15-20 szavas mondatok természetesebben hangzanak a rádiós bemondó előadásában. Az absztraktív összegzés ugyanakkor a hallucináció kockázatát hordozza: az LLM olyan információt is generálhat, amely az eredeti szövegben nem szerepel.

A tokenizáló fallback hatása: A NewsCast LexRank implementációja a cseh nyelvi tokenizálót használja magyar szövegekre (a magyar NLTK tokenizáló nem állt rendelkezésre). Ez a mondathatár felismerés pontosságát korlátozhatja, különösen a rövidítéseket (pl. „Zrt.”, „stb.”) tartalmazó szövegeknél.

3.10.5.5. Összesítő értékelés

Az LLM-benchmark vizsgálat három részfeladatban hasonlította össze a NewsCast szabályalapú megoldásait három nagy nyelvi modell (LLM) kimenetével. Az összesített eredmények az alábbi következtetésekre vezetnek.

Hírelemzés: A NewsCast determinisztikus, kulcsszó alapú tartalombiztonsági elemzése konzisztens és reprodukálható eredményt ad, amely a rádiós szűrés szempontjából megbízható alap. A 40 összehasonlítási pont alapján a GPT-5.4-mini 33/40 (82,5%), a Gemini Flash Lite 31/40 (77,5%), a Claude Haiku 4.5 29/40 (72,5%) egyezést ért el a NewsCast referenciával (lásd 26. táblázat). A reklámdetektálás minden LLM-nél tökéletes (10/10) volt. A legnagyobb eltérés a szentiment meghatározásban mutatkozott: a GPT-5.4-mini 7/10, a Gemini Flash Lite 6/10, a Claude Haiku 4.5 4/10 egyezést ért el. Az eltérések fő forrása a 2. (bűnügyi) és 10. (szexuális tartalmú) tesztír, amelyeknél a NewsCast HuSpaCy-alapú elemzése „neutral”-nak,

míg mindhárom LLM „negative”-nak értékelte a szöveget – az LLM-ek a kontextuális pragmatikát, a NewsCast a lemma arányok numerikus küszöbét alkalmazza. A rádiós relevancia pontszám négykomponensű képlete (olvashatóság + aktualitás + lokalitás + biztonság) a maga determinisztikus jellegénél fogva az LLM-ek számára is követhető – a prompt-alapú pontozásnál az LLM-ek a képlet egyes összetevőit eltérő módon értelmezhetik.

Szövegnormalizálás: A `text_normalizer.py` 14 normalizálási lépése a magyar nyelv legtöbb releváns esetét lefedi, de a benchmark rámutatott a szabálykészlet korlátaira. A 180 pontos skálán a Gemini Flash Lite érte el a legmagasabb pontszámot (172/180, 95,6%), ezt követte a GPT-5.4-mini (171/180, 95,0%), a Claude Haiku 4.5 (168/180, 93,3%) és a NewsCast (159/180, 88,3%) – lásd 29. táblázat. A kategóriánkénti bontás alapján a NewsCast a hőmérséklet-normalizálásban tökéletes (30/30) eredményt ért el, míg a rövidítések feloldásában (24/30) és a természetesség szempontjában (25/30) maradt el az nagy nyelvi modellektől. A szabályalapú rendszer előnye a teljes determinizmus: minden futtatásnál azonos kimenetet ad, ami a rádiós hanganyag generálás konzisztenciája szempontjából elengedhetetlen. A toldalékolási és speciális karakter kezelési hibák ugyanakkor arra utalnak, hogy a szabálykészlet bővítése szükséges – az LLM-ek ezen a területen természetesebb kimenetet produkálhatnak, de nem garantálják a konzisztenciát.

Szövegösszegzés: Az extraktív (LexRank) és az absztraktív (LLM) megközelítés eltérő célokat szolgál. A LexRank az eredeti szöveg hitelességét őrzi meg (29/30 pont az extraktív értékelésben), míg a Claude Haiku 4.5 tökéletes 30/30-as extraktív eredményt ért el. Az absztraktív összegzésben a GPT-5.4-mini 30/30, a Claude Haiku 4.5 29/30, a Gemini Flash Lite 28/30 pontot kapott (lásd 33. táblázat). A LexRank az eredeti szöveg hitelességét őrzi meg, az LLM absztraktív összegzése a rádiós felolvasásra optimalizált tömörséget biztosítja. A két módszer nem egymás alternatívája, hanem komplementer: a NewsCast jelenlegi extraktív megoldása az MI-lead generálás (Gemini) által absztraktív összegzéssel is kiegészül (vö. 3.6.2 fejezet).

Ezen benchmark eredmények mélyebb értelmezését a 4. fejezet (Vita) tartalmazza, ahol a saját munka és az LLM megoldások közötti eltérések tervezési döntésekre gyakorolt hatását vizsgáltam.

4. Vita

A fejlesztés során több olyan tervezési döntés és implementációs dilemma merült fel, amelyek nem rendelkeztek egyértelmű, „tankönyvi” megoldással. A jelen fejezet e kérdések kritikai vizsgálatát tartalmazza.

4.1. A hírrelevancia meghatározásának módszertani kérdése

A NewsCast rendszer szabályalapú NLP elemzést (HuSpacy lemmatizáció, szentimentelemzés, olvashatósági metrikák) kombinál a Google Gemini MI-alapú validációval. A fejlesztés során felmerült a kérdés, hogy a szabályalapú megközelítés önmagában elegendő-e a rádiós relevancia meghatározásához vagy az MI-validáció nélkülözhetetlen kiegészítés. A tesztelési eredmények (vö. 3.10.5.2 fejezet) azt mutatták, hogy a szabályalapú rendszer a hírek mintegy 85%-ánál konzisztens eredményt ad, a fennmaradó 15% esetében azonban az MI-validáció szignifikánsan pontosabb besorolást eredményez – különösen az ironikus hangvételű, a szatirikus vagy a kontextusfüggő tartalmak esetében. Kérdésként merül fel, hogy a Gemini API-tól való függőség hosszú távon fenntartható-e vagy egy lokálisan futó nyelvi modell (pl. finomhangolt LLaMA, Mistral vagy OpenEuroLLM-Hungarian) hasonló eredményt nyújthatna-e.

4.2. A közösségi trendjelek feldolgozásának kompromisszumai

A 2.2.4 alfejezetben részletesen dokumentált platform korlátozások elemzése után a Google News RSS és a Google Trends RSS feedekre esett a választás. A fejlesztés során felmerült, hogy a *Szuper képlet*: $V_H = 10 + (Trends * 50)$ lineáris jellege kellően differenciált-e a hírrelevancia meghatározásához vagy egy többletényezős modell – amely a Google News pozíciót, a Trends forgalmi adatokat és a RapidFuzz hasonlósági pontszámot együttesen súlyozná – pontosabb eredményt adna. A jelenlegi kétértékű megközelítés ({10, 60} pont) egyszerűsége előnyös az átláthatóság szempontjából, ugyanakkor nem tükrözi a trendjelek intenzitásbeli különbségeit.

4.3. A szolgáltatásközi kommunikáció dilemmája

A szolgáltatásközi kommunikáció kapcsán felmerült a dilemma, hogy a szinkron HTTP/REST-hívások helyett aszinkron üzenetsor (message queue, pl. RabbitMQ, Redis Streams) alkalmazása nem lenne-e célszerűbb. A jelenlegi szinkron megközelítés előnye az egyszerűség és az átláthatóság: a hívó fél azonnal választ kap és a hibakezelés egyértelmű. Hátránya viszont,

hogy egy modul kiesése kaszkád hatást válthat ki a hívási láncban. Az aszinkron üzenetsor bevezetése a rendszer rezilienciáját növelné, ugyanakkor a fejlesztési és üzemeltetési komplexitás szignifikáns emelkedésével járna – a jelenlegi projekt méretéhez képest ez aránytalanul nagy ráfordítás lenne (vö. 5.3 fejezet, Korlátok és ismert hiányosságok).

4.4. A magyar nyelvű szövegnormalizálás kihívásai

A TTS modul `text_normalizer.py` moduljában 50 normalizálási szabályt definiáltam. A fejlesztés során felmerült a kérdés, hogy a szabályalapú normalizálás lefedi-e a magyar nyelv valamennyi releváns esetét vagy léteznek-e olyan kontextusfüggő szituációk (pl. római számok, rövidítések, idegen szavak kiejtése), amelyek gépi tanulási megközelítést igényelnének. A jelenlegi szabálykészlet a rádiós hírszövegek kontextusában az esetek túlnyomó többségében helyes eredményt ad, azonban a rendszer bővítése esetén (pl. sport- vagy pénzügyi terminológia) a szabálykészlet karbantarthatósága kérdésessé válhat. A 3.10.5.3 alfejezetben dokumentált LLM-benchmark eredmények rávilágítottak arra, hogy a toldalékolási hibák (pl. „kilométer per óra-s” vs. „kilométer per órás”) és a speciális karakter kezelési hiányosságok (pl. „m²” → „négyzetméter”) a szabálykészlet bővítésével orvosolhatók, de a magyar nyelv produktív morfológiája miatt a teljes lefedettség elérése nem triviális feladat.

4.5. Az RSS feldolgozás párhuzamossági szintje

Az RSS feldolgozás párhuzamossági szintjének megválasztása etikai és technikai kompromisszumot igényelt. A `MAX_CONCURRENT_FEEDS` paraméter túl alacsony értéke lassítja a feldolgozást, túl magas értéke viszont a célszerverek túlterheléséhez vezethet – ami etikai és jogi szempontból egyaránt aggályos (vö. 2.2.4 fejezet). A jelenlegi konfiguráció a hírportálok rate limit korlátainak tiszteletben tartását helyezi előtérbe, ami óvatosabb, de felelősebb működést eredményez.

4.6. Külső szolgáltatásfüggőségek kockázatai

Az ElevenLabs API külső függőségként való alkalmazása kockázatot hordoz: az API árazásának változása, a szolgáltatás elérhetőségének ingadozása vagy a magyar nyelvi modell minőségének módosulása közvetlenül érinti a rendszer TTS funkcionalitását. A fejlesztés során mérlegelésre került alternatív TTS megoldások (Coqui TTS, Mozilla TTS) beépítése fallback mechanizmusként, ez azonban a jelenlegi fejlesztési hatókörön kívül esik (vö. 6. fejezet, Jövőbeli irányok).

4.7. Az LLM-benchmark tanulságai

A 3.10.5 alfejezetben dokumentált LLM-benchmark vizsgálat eredményei alapján a NewsCast szabályalapú rendszere és a nagy nyelvi modellek kimenetei között szignifikáns eltérések mutatkoztak. A három részfeladat eltérő tanulságokkal szolgált.

A **hírelemzés** területén a NewsCast determinisztikus, kulcsszó-alapú tartalombiztonsági elemzése a reprodukálhatóság szempontjából előnyösebb az LLM-ek sztochasztikus kimeneténél. A 40 összehasonlítási pont alapján (10 tesztír * 4 mező) a GPT-5.4-mini 82,5%-os, a Gemini Flash Lite 77,5%-os, a Claude Haiku 4.5 72,5%-os egyezést ért el a NewsCast referenciával (vö. 26. táblázat). A reklámdetektálás területén mindhárom LLM tökéletes (10/10) egyezést mutatott, ami a feladat viszonylagos egyszerűségére utal. A szentiment-meghatározásnál a legnagyobb eltérés a 2. (bűnügyi) és a 10. (szexuális tartalmú) tesztírnél jelentkezett: a NewsCast HuSpaCy lemma alapú elemzése „neutral”-nak, míg mindhárom LLM „negative”-nak értékelte ezeket a szövegeket. A rádiós műsorszórás kontextusában a determinizmus kritikus követelmény: a tartalomszűrés következtelenségei jogi és etikai kockázatot hordoznak. Ugyanakkor az LLM-ek kontextusfüggő értékelése a szabályalapú rendszer számára nem hozzáférhető pragmatikai információkat is figyelembe vesz – ez különösen a szatirikus, ironikus és kettős értelmű tartalmak felismerésénél jelent előnyt. A NewsCast jelenlegi hibrid megoldása (szabályalapú előszűrés + Gemini MI-validáció) kompromisszumot teremt e két szempont között.

A **szövegnormalizálás** területén mindhárom LLM felülmúlta a NewsCast szabályalapú normalizálóját: a Gemini Flash Lite 95,6%-ot (172/180), a GPT-5.4-mini 95,0%-ot (171/180), a Claude Haiku 4.5 93,3%-ot (168/180) ért el, szemben a NewsCast 88,3%-os (159/180) eredményével (vö. 29. táblázat). A szabályalapú rendszer gyengeségei a kontextusfüggő esetekben mutatkoztak meg: a százalékjel todalékolása (5. mondat: „hét és fél%-ra” a helyes „százalékra” helyett), a sorszám-felismerés kontextusfüggő korlátja (9. mondat: „nyolcvankilenc.” a helyes „nyolcvankilencedik” helyett), a speciális karakterek kezelése (15. mondat: „m²” → „négyzetméter”) és a sporteredmény-formátum (17. mondat: „három-elsőre” a helyes „három-egyre” helyett). A szabályalapú rendszer determinizmusa ugyanakkor a rádiós TTS generálás konzisztenciáját biztosítja. Az LLM-benchmark rámutatott a szabálykészlet konkrét bővítési irányaira (vö. 3.10.5 fejezet), amelyek a determinisztikus megközelítés keretein belül javítják a normalizálás minőségét.

A **szövegösszegzés** területén az extraktív (LexRank) és az absztraktív (LLM) megközelítés közötti különbség a rádiós felhasználás szempontjából releváns. A NewsCast LexRank algoritmus 29/30 pontot ért el az extraktív összegzésben, ami versenyképes a Claude Haiku 4.5 tökéletes 30/30-as eredményével és megegyezik a GPT-5.4-mini pontszámával (vö. 3.3. táblázat). Az absztraktív összegzésben – amelyet kizárólag az LLM-ek készítenek – a GPT-5.4-mini tökéletes 30/30 pontot ért el, míg a Claude Haiku 4.5 29/30-at, a Gemini Flash Lite 28/30-at. A tömörebb, rövidebb mondatok természetesebben hangzanak a rádiós felolvasásban. A NewsCast jelenlegi architektúrája ezt a felismerést már részben integrálja: a Gemini MI-alapú ai_lead generálás (vö. 3.6.4 fejezet) absztraktív összegzést készít a kiválasztott hírekhez, míg a LexRank a tartalmi validáció referenciájaként szolgál.

Összességében az LLM-benchmark megerősítette, hogy a NewsCast hibrid megközelítése – szabályalapú determinisztikus modulok és MI-alapú validáció/generálás kombinációja – a rádiós hírszolgáltatás kontextusában indokolt tervezési döntés. A szabályalapú modulok a konzisztenciát, a reprodukálhatóságot és a költséghatékonyt biztosítják, míg az MI komponensek a kontextusfüggő minőségjavítást szolgálják. A szövegnormalizálás területén a benchmark azonosította a szabálykészlet bővítésének konkrét irányait, a hírelemzésben a determinisztikus megközelítés a rádiós kontextusban előnyösebb, a szövegösszegzésben a két módszer komplementer jellegű.

5. Következtetések

A jelen fejezet a NewsCast rendszer fejlesztésének eredményeit foglalja össze: a kitűzött célok teljesülésének értékelése (5.1), a rendszer erősségei (5.2) és a korlátok és ismert hiányosságok (5.3) kerülnek kifejtésre. A jövőbeli fejlesztési irányokat a 6. fejezet, a záró összegzést a 7. fejezet tartalmazza.

5.1. A kitűzött célok értékelése

Az 1.1 alfejezetben hét kutatási és fejlesztési célt határoztam meg. Az alábbi értékelés célja annak bemutatása, hogy a NewsCast rendszer végleges állapotában mely célokat sikerült maradéktalanul teljesíteni és mely pontokon maradtak nyitott feladatok.

Az automatizált hírgyűjtés célja teljesült: a newscast-rss_parser modul 62 magyar nyelvű RSS forrást dolgoz fel párhuzamosan, HTTP gyorsítótárazással (ETag / Last-Modified) és duplikációsűrőssel (vö. 3.6.1 fejezet).

Az intelligens tartalomelemzés célja teljesült: a newscast-analyze modul HuSpacy alapú NLP pipeline-ja magyar nyelvű szentimentelemzést, névelem felismerést és olvashatósági metrikákat számít, amelyeket a Google Gemini API-n keresztüli MI-validáció egészít ki (vö. 3.6.2 fejezet).

A közösségi média trendjelek feldolgozásának célja részben teljesült: a newscast-social modul a Google News és a Google Trends RSS feedekre épül (vö. 3.6.6 fejezet). A közösségi média platformok API-korlátozásai (vö. 2.2.4 fejezet) miatt a Twitter/X, a Facebook és a Reddit közvetlen integrációja nem valósult meg – ezt a 4. fejezet (Vita) tárgyalja.

Az objektív hírelemzés célja teljesült: a COCO modell három változata (Y0, STD, MCM) a miau.my-x.hu távoli API-n keresztül érhető el, az OAM elemzés a hírforrások összehasonlító objektivitásvizsgálatát valósítja meg (vö. 3.6.2.3 fejezet).

Az időjárás integráció célja teljesült: a newscast-weather modul az OMSZ szöveges előrejelzéseit dolgozza fel, kezelve a heterogén karakterkódolásokat (vö. 3.6.3 fejezet).

A szövegfelolvasás (TTS) célja teljesült: a newscast-tts modul az ElevenLabs API szolgáltatáson keresztül magyar nyelvű hangszintézist végez, 50 szövegnormalizálási szabállyal (vö. 3.6.5 fejezet).

A minimális emberi beavatkozás célja teljesült: a teljes feldolgozási lánc automatikusan működik óránkénti ütemezéssel, a szerkesztői beavatkozás opcionális (vö. 3.6.4 fejezet).

Összegezve a hét célból hat maradéktalanul teljesült, a közösségi trendjelek integrációja a platform-korlátozások miatt korlátozott forrásokra (Google News/Trends RSS) szűkült, amely az eredeti elképzeléshez képest szűkebb lefedettséget biztosít.

5.2. A rendszer erősségei

A NewsCast rendszer legfontosabb erősségeinek az alábbiakat tekintem:

Moduláris architektúra: A hat önálló mikroszolgáltatás (vö. 3.2 fejezet) egymástól független fejlesztését, telepítését és skálázását teszi lehetővé. Egy modul meghibásodása nem vonja maga után a teljes rendszer leállítását (vö. 2.2.5 fejezet).

Magyar nyelvű specializáció: A rendszer a HuSpacy hu_core_news_lg magyar nyelvi modellre épül (vö. 3.6.2 fejezet), a TTS-modul 50 magyar specifikus szövegnormalizálási

szabályt alkalmaz (vö. 3.6.5.4 fejezet), a számok szövegesítése a magyar nyelv szabályai szerint történik. Ezek a megoldások a rendszert az angol nyelvű eszközöktől megkülönböztetik.

Konfigurálhatóság futásidőben: A ParamManager osztály (vö. 3.6.2 fejezet) lehetővé teszi a rádiós relevancia-súlyok, a batch-méreték és az elemzési paraméterek módosítását a rendszer újraindítása nélkül, a webes admin (feeder modul/UI) felületen vagy közvetlen adatbázis mező-frissítésen keresztül.

Teljes körű monitorozás és naplózás: Minden modul dedikált monitoring.py modullal rendelkezik, amely rendszer-, alkalmazás- és üzleti metrikákat szolgáltat a Prometheus felé (vö. 3.8 fejezet). A strukturált JSON naplózás korrelációs azonosítókkal a hibakeresést támogatja.

Költségoptimalizálás: Az ETag alapú gyorsítótárazás (~70%-os sávszélesség megtakarítás), az SHA-256 hash deduplikáció (TTS API hívások csökkentése) és a kötegelt Gemini API feldolgozás (15 perces ciklusokban 50 elem) mérhető költségcsökkenést eredményeznek (vö. 1.5.2 alfejezet).

5.3. Korlátok és ismert hiányosságok

A rendszer korlátait az 1.6 alfejezetben (A dolgozat hatóköre és korlátai) részleteztem. Az ott felsorolt kilenc korlátozás mellett a fejlesztés és a tesztelés (vö. 3.10 fejezet) során az alábbi további hiányosságok váltak ismertté.

A magyar nyelvű szövegnormalizálás a 3.10.5.3 alfejezetben dokumentált LLM-benchmark vizsgálat alapján 88,3%-os pontossággal (159/180) működik, míg a vizsgált nagy nyelvi modellek 93–96%-os eredményt értek el. Az eltérés elsősorban a kontextusfüggő esetek (ironikus felsorolások, idegen nevek ragozása) kezeléséből adódik (vö. 4. fejezet, Vita).

Az RSS feldolgozás kizárólag szöveges tartalmakat kezel, a multimédiás elemek (képek, videók, hanganyagok) feldolgozása nem része a rendszernek.

A COCO API külső függőségként a miau.my-x.hu szerveren fut, az API elérhetetlensége esetén az OAM elemzés nem végezhető el.

6. Jövőbeli irányok

A fejlesztés és a tesztelés tapasztalatai alapján több irányban látok lehetőséget a rendszer továbbfejlesztésére.

Aszinkron üzenetsor bevezetése: A szolgáltatásközi kommunikáció jelenlegi szinkron HTTP/REST megoldása (vö. 4. fejezet, Vita) kiváltható lenne aszinkron üzenetsorral (pl. Redis Streams), amely a modulok közötti csatolást lazítaná és a hibatűrést javítaná.

Szövegnormalizálás hibrid megközelítéssel: A szabályalapú normalizáló kiegészítése egy MI-alapú fallback réteggel a 4. fejezetben tárgyalt kontextusfüggő esetek lefedésére. Az LLM-benchmark eredmények (vö. 3.10.5.3 alfejezet) alapján ez 5-8 százalékpontos javulást eredményezhetne.

CI/CD pipeline kiépítése: Automatizált integrációs és telepítési folyamatok bevezetése (GitHub Actions), amelyek a jelenlegi kézi telepítési munkafolyamatot váltják ki.

Platformfüggetlen közösségi trendjelek: Amennyiben a közösségi média platformok API-hozzáférési politikája enyhül, a newscast-social modul bővíthető lenne Twitter/X és Reddit integrációval. Ezzel a különböző hírek közösségi médiában való megjelenése és terjedése pontosabban nyomon követhető, ami segíti a kiválasztó automatizmus és a hírszerkesztő munkáját.

Hangkarakter és hangsúly vezérlés a szerkesztői felületen: A newscast-tts modul jelenleg egyetlen, előre beállított ElevenLabs hangprofillal működik (vö. 3.6.5.1 alfejezet). A továbbfejlesztés részeként a newscast-feeder webes felületén (vö. 3.6.4.3 alfejezet) lehetőséget kellene biztosítani az ElevenLabs rendszerében is elérhető hang kiválasztására és a beszédparaméterek – hangsúly, tempó, hangszín – akár hírtípusonkénti beállítására. Ez a megoldás lehetővé tenné, hogy a rendkívüli hírek komolyabb tónussal, a szórakoztató tartalmak lazább stílusban szólaljanak meg, közelítve a humán bemondó által nyújtott előadásmódot. Ezen beszédstílusok beállítása a newscast-analyze modulban már beépített Gemini hívásokkal is támogatható. Az ElevenLabs API a voice_settings paraméterén keresztül támogatja a stability, similarity_boost és style értékek dinamikus módosítását, amely műszakilag lehetővé teszi az ilyen típusú vezérlés megvalósítását. A rádiós arculathoz igazított hangprofilok (pl. „hírolvasó”, „sportriporter”, „kulturális szerkesztő”) előre konfigurálhatók lennének és a narratíva flow szerkesztés (vö. 3.6.4.2 alfejezet) során a szerkesztő az egyes hírblokk-szegmensekhez rendelhetné a kívánt profilt.

MI alapú automatikus hírszelekció: A jelenlegi hírszelekciós algoritmus (vö. 3.6.4.1 alfejezet) szabályalapú pontozással határozza meg a hírek rádiós relevanciáját, amelyet a szerkesztő a feeder webes felületen felülbírálnak. Egy következő fejlesztési lépésként a Google Gemini API integrálásával a newscast-feeder modul MI alapú döntéshozatali logikával

egészíthető ki. A nagy nyelvi modell képes lenne kontextuális szempontok figyelembevételére – az adott napszakhoz, a célközönséghez és a korábbi adásokhoz igazított hírválogatás –, amelyeket a szabályalapú rendszer nem képes megragadni. Az LLM-benchmark vizsgálat (vö. 3.10.5 alfejezet) eredményei alátámasztják, hogy a vizsgált modellek a magyar nyelvű hírtartalom értelmezésében megbízható teljesítményt nyújtanak. A megvalósítás során biztosítani kellene, hogy az MI alapú javaslat transzparens maradjon: a szerkesztő a felületen megtekinthetné a modell indoklását és bármikor felülírhatná az automatikus döntést.

Cloud SaaS és multi-tenant architektúra: A jelenlegi rendszer egyetlen rádiós szerkesztőség kiszolgálására lett tervezve, dedikált szerveren futó Docker konténerekkel (vö. 3.5.3 alfejezet). A piaci motiváció (vö. 1.3.2 alfejezet) és a gazdasági hasznosság elemzés (vö. 1.5.2 alfejezet) alapján a rendszer potenciálisan több ügyfél – kisközösségi rádiók, online médiumok, podcasterek – számára is szolgáltatásként nyújtható. Ehhez a rendszer multi-tenant architektúrára való átalakítása szükséges, amely ügyfelenként izolált konfigurációkat (RSS-forráslista, elemzési paraméterek, narratíva flow sablonok, TTS-hangprofilok), egyedi ütemezéseket és elkülönített adattárolást biztosítana. A Cloud SaaS (Software as a Service) modellre történő átállás további modulokat igényelne: felhasználókezelés és hozzáférés-szabályozás (RBAC), használat-alapú elszámolás (API hívások, TTS karakter fogyasztás, tárhelyhasználat) és automatizált számlázási integráció. A jelenlegi mikroszolgáltatás alapú architektúra (vö. 3.2 fejezet) kedvező kiindulópontot biztosít, mivel a modulok közötti laza csatolás lehetővé teszi az ügyfelazonosítók (tenant ID) bevezetését az üzenetsor szintű elkülönítés megvalósítása nélkül, amennyiben a jelenlegi adatbázis séma tenant szintű szűréssel egészül ki.

7. Összegzés

A NewsCast rendszer fejlesztése során meggyőződhettem arról, hogy a rádiós hírszolgáltatás automatizálása mérnöki szempontból megvalósítható feladat. A hat mikroszolgáltatás együttműködése lehetővé teszi a teljes hírfeldolgozási lánc lefedését az RSS letöltéstől a szintetizált hangkimenetig.

A projekt során a képzés keretében elsajátított elméleti ismeretek – a szoftverarchitektúrák, az adatbázis-kezelés, a mesterséges intelligencia és a szoftvertesztelés területén – gyakorlati alkalmazásra kerültek. A COCO modell integrálása a hírforrás-objektivitás vizsgálatába a matematikai modellezés és a szoftvermérnöki megvalósítás összekapcsolását igényelte.

A fejlesztés legfontosabb tanulsága, hogy a szabályalapú és az MI-alapú megoldások kombinálása – az ún. hibrid megközelítés – a rádiós felhasználási kontextusban indokolt: a szabályalapú modulok biztosítják a reprodukálhatóságot és a sebességet, míg az MI-komponensek a határesetek kezelésében nyújtanak hozzáadott értéket.

8. Mellékletek

A mellékletek a dolgozat törzsszövegét kiegészítő anyagokat tartalmazzák. Az ábrajegyzék (8.1) a dolgozatban szereplő illusztrációk listáját, a táblázatjegyzék (8.2) a táblázatok áttekintését foglalja össze. A rövidítések jegyzéke (8.3) a szövegben alkalmazott betűszavak feloldását, a definíciók jegyzéke (8.4) a szakterminológia magyarázatát adja meg. A hivatkozások (8.5) a felhasznált irodalom bibliográfiai adatait sorolják fel. Az LLM-benchmark konverzációk (8.6) a 3.10.5 alfejezetben ismertetett összehasonlító vizsgálat teljes API-válaszait, a forráskódok (8.7) pedig a rendszer válogatott forráskódjait tartalmazzák.

8.1. Ábrajegyzék

1. ábra: A NewsCast rendszer architektúra áttekintése – Forrás: saját szerkesztés.....	57
2. ábra: A NewsCast rendszer egyszerűsített ER diagramja – Forrás: saját szerkesztés.....	60
3. ábra: NewsCast modulok közötti kommunikáció – Forrás: saját szerkesztés.....	68
4. ábra: A newscast-feeder Dashboard felülete – az utolsó hírblokkok és a lejátszható hanganyagok áttekintése – Forrás: saját szerkesztés.....	81
5. ábra: A newscast-feeder Hírek nézete – hírszelekció, rendezés és szerkesztés – Forrás: saját szerkesztés.....	82
6. ábra: A newscast-feeder OAM nézete – a COCO modell elemzési eredményeinek megjelenítése – Forrás: saját szerkesztés.....	83
7. ábra: A newscast-feeder Beállítások felülete – az üzemmód, ütemezés és TTS paraméterek konfigurálása – Forrás: saját szerkesztés.....	84
8. ábra: A newscast-feeder Felhasználók kezelő felülete – felhasználók és szerepkörök adminisztrálása – Forrás: saját szerkesztés.....	85

8.2. Táblázatjegyzék

1. táblázat: A legjelentősebb közösségi média platformok API hozzáférési korlátozásai.....	34
2. táblázat: A NewsCast modulok felelősségi körei.....	59
3. táblázat: NewsCast adatbázis „rss” tábla (hírforrások).....	61

4. táblázat: NewsCast adatbázis „social_signals” tábla (közösségi trendek).....	63
5. táblázat: NewsCast adatbázis „trending_keywords” tábla (trending kulcsszavak).....	64
6. táblázat: newscast-rss_parser API végpontok	65
7. táblázat: newscast-analyze API végpontok	66
8. táblázat: newscast-weather API végpontok.....	66
9. táblázat: newscast-social API végpontok	67
10. táblázat: newscast-feeder API végpontok	67
11. táblázat: newscast-tts API végpontok.....	68
12. táblázat: A retry mechanizmus paraméterei modulonként	69
13. táblázat: A NewsCast rendszert futtató VPS paraméterei	71
14. táblázat: A NewsCast rendszer infrastrukturális platform komponensei	72
15. táblázat: A bemeneti paraméterek módosításának hatásai a rendszer viselkedésére.....	94
16. táblázat: Tesztelési eszközök.....	96
17. táblázat: NewsCast modulok health check konfigurációk	96
18. táblázat: Az RSS feldolgozás teljesítménymutatói.....	103
19. táblázat: Az elemzési modul teljesítménymutatói.....	104
20. táblázat: Social modul teljesítménymutatói	104
21. táblázat: A TTS modul teljesítménymutatói.....	104
22. táblázat: A teljes feldolgozási pipeline átfutási ideje	105
23. táblázat: Követelmény – megfelelési mátrix	106
24. táblázat: Az LLM-benchmark vizsgálatban alkalmazott nagy nyelvi modellek	107
25. táblázat: A NewsCast hírelemzési pipeline referencia-kimenetei	110
26. táblázat: A hírelemzési benchmark összesítő eredményei.....	110
27. táblázat: Szövegnormalizálási teszt kiértékelési szempontok	111
28. táblázat: Válogatott szövegnormalizálási példák a NewsCast kimenetéből (30-ból 6) ...	112
29. táblázat: A szövegnormalizálási benchmark összesítő eredményei	114
30. táblázat: A NewsCast LexRank extraktív összegzései és tömörítési arányai.....	115
31. táblázat: A NewsCast LexRank extraktív összegzéseinek kiértékelési szempontjai.....	116
32. táblázat: A NewsCast LexRank absztraktív összegzéseinek kiértékelési szempontjai	116
33. táblázat: A szövegösszegzési benchmark összesítő eredményei	116
34. táblázat: A szakdolgozat témájához kapcsolódó magyar és külföldi publikációk	144
35. táblázat: Benchmark 2 – Szövegnormalizálási tesztadatok (30 tesztmondat).....	146

8.3. Rövidítések jegyzéke

Rövidítés	Teljes név	Magyar megfelelő / Leírás
AI	Artificial Intelligence	Mesterséges intelligencia (a dolgozatban az MI rövidítés kerül alkalmazásra, az AI kizárólag angol nyelvű hivatkozásokban és tulajdonnevekben fordul elő)
API	Application Programming Interface	Alkalmazásprogramozási interfész
ASGI	Asynchronous Server Gateway Interface	Aszinkron szerver kapu interfész
CI/CD	Continuous Integration / Continuous Deployment	Folyamatos integráció / Folyamatos telepítés
CFAA	Computer Fraud and Abuse Act	Számítógépes csalás és visszaélés elleni törvény (USA)
COCO	Component-based Object Comparison for Objectivity	Komponens-alapú objektum-összehasonlítás az objektivitás érdekében
CPU	Central Processing Unit	Központi feldolgozóegység
CRLF	Carriage Return Line Feed	Kocsi-vissza és soremelés (Windows-stílusú sorvég)
CRUD	Create, Read, Update, Delete	Létrehozás, olvasás, frissítés, törlés (alapvető adatszervező műveletek)
CSRF	Cross-Site Request Forgery	Webhelyek közötti kérés hamisítás
CSS	Cascading Style Sheets	Lépcsőzetes stíluslapok (webes megjelenés formázása)
DB	Database	Adatbázis
DevOps	Development and Operations	Fejlesztés és üzemeltetés
DNS	Domain Name System	Tartománynév-rendszer
ER	Entity-Relationship	Entitás-kapcsolat (adatbázis-modellelési technika)
ETag	Entity Tag	Entitáscímke (HTTP gyorsítótárazási mechanizmus)
FAISS	Facebook AI Similarity Search	Facebook AI hasonlóságkeresés
FR	Functional Requirement	Funkcionális követelmény
FK	Foreign Key	Idegen kulcs (adatbázis-hivatkozás)
FTP	File Transfer Protocol	Fájltviteli protokoll
GDPR	General Data Protection Regulation	Általános adatvédelmi rendelet (EU)

GUI	Graphical User Interface	Grafikus felhasználói felület
HPA	Hectopascal	Hektopascal (légnomás mértékegysége)
HTML	HyperText Markup Language	Hiperszöveg-jelölőnyelv
HTTP	HyperText Transfer Protocol	Hiperszöveg-átviteli protokoll
HTTPS	HyperText Transfer Protocol Secure	Biztonságos hiperszöveg-átviteli protokoll
I/O	Input/Output	Bemenet/kimenet
IDE	Integrated Development Environment	Integrált fejlesztői környezet
IP	Internet Protocol	Internet protokoll
JSON	JavaScript Object Notation	JavaScript objektumjelölés (adatsereformátum)
JWT	JSON Web Token	JSON webes token (hitelesítési szabvány)
KSH	Központi Statisztikai Hivatal	Magyar statisztikai hivatal
LLM	Large Language Model	Nagy nyelvi modell
MCM	Multi-Context Model	Többkontextusú modell (COCO változat)
MI	Mesterséges intelligencia	Mesterséges intelligencia
MP3	MPEG Audio Layer III	Tömörített hangformátum
MTI	Magyar Távirati Iroda	Magyar hírügynökség
NER	Named Entity Recognition	Névelem felismerés
NLP	Natural Language Processing	Természetes nyelvfeldolgozás
NLTK	Natural Language Toolkit	Természetes nyelvi eszközkészlet
NMHH	Nemzeti Média- és Hírközlési Hatóság	Magyar médiahatóság
NFR	Non-Functional Requirement	Nem funkcionális követelmény
OAM	Object-Attribute Matrix	Objektum-attribútum mátrix
OMSZ	Országos Meteorológiai Szolgálat	Magyar meteorológiai szolgálat (jelenleg HungaroMet)
ORM	Object-Relational Mapping	Objektum-relációs leképezés
OS	Operating System	Operációs rendszer
PCM	Pulse-Code Modulation	Impulzuskód-moduláció (nyers hangformátum)
PK	Primary Key	Elsődleges kulcs (adatbázis)

POS	Part-of-Speech	Szófaj (nyelvi elemzés)
RAM	Random Access Memory	Közvetlen elérésű memória
RBAC	Role-Based Access Control	Szerepkör-alapú hozzáférés-szabályozás
REST	Representational State Transfer	Reprezentációs állapotátvitel (architekturális stílus)
RFC	Request for Comments	IETF szabványdokumentum
ROI	Return on Investment	Beruházás megtérülése
RPO	Recovery Point Objective	Helyreállítási pont célkitűzés
RSS	Really Simple Syndication	Tartalomdistribúciós szabvány
RSP	Radio Relevance Score Prefilter	Rádiós relevancia előszűrő
RTO	Recovery Time Objective	Helyreállítási idő célkitűzés
S3	Simple Storage Service	Amazon egyszerű tárolószolgáltatás
SaaS	Software as a Service	Szoftver mint szolgáltatás
SHA-256	Secure Hash Algorithm 256-bit	256 bites biztonságos hash algoritmus
SPA	Single Page Application	Egyoldalas webalkalmazás
SQL	Structured Query Language	Strukturált lekérdezőnyelv
SSD	Solid State Drive	Félvezető alapú meghajtó
SSE	Server-Sent Events	Szerver által küldött események
SSL	Secure Sockets Layer	Biztonságos kommunikációs réteg
SSML	Speech Synthesis Markup Language	Beszéd-szintézis-jelölőnyelv (W3C szabvány)
STD	Standard (model)	Felügyelt tanulási modell (COCO változat)
TF-IDF	Term Frequency - Inverse Document Frequency	Kifejezésgyakoriság - inverz dokumentumgyakoriság
TLS	Transport Layer Security	Szállítási réteg biztonság
TSV	Tab-Separated Values	Tabulátorral elválasztott értékek
TTS	Text-to-Speech	Szövegfelolvasás
TTL	Time to Live	Élettartam (gyorsítótárazási paraméter)
UI	User Interface	Felhasználói felület
UML	Unified Modeling Language	Egységesített modellezési nyelv
UNIQUE	Unique Constraint	Egyediségi kényszer (adatbázis)

URL	Uniform Resource Locator	Egységes erőforrás-azonosító
UUID	Universally Unique Identifier	Univerzálisan egyedi azonosító
VPS	Virtual Private Server	Virtuális magánszerver
WAV	Waveform Audio File Format	Hullámforma hangfájl-formátum
XML	Extensible Markup Language	Bővíthető jelölőnyelv
YO	Anti-discriminative model	Anti-diszkriminatív modell (COCO változat)
ZIP	Zone Information Protocol / tömörítési formátum	Tömörített archívum formátum

8.4. Definíciók jegyzéke

Fogalom	Definíció	Hivatkozás
Mikroszolgáltatás (Microservice)	Önálló, egyetlen üzleti funkcióért felelős szoftverkomponens, amely saját adatbázissal és API-felülettel rendelkezik és a többi szolgáltatással hálózati hívások útján kommunikál.	vö. 2.2.5
NLP pipeline	A természetes nyelvfeldolgozás lépéseinek sorozata: tokenizálás, lemmatizálás, névelem felismerés, szentimentelemzés, olvashatósági értékelés.	vö. 3.6.2.1
OAM (Object-Attribute Matrix)	Objektum-attribútum mátrix: a COCO modell bemeneti adatstruktúrája, amelyben a sorok az objektumokat (hírforrásokat), az oszlopok az attribútumokat (hírtulajdonságokat) képviselik.	vö. 2.2.7
Szuper képlet (Super Formula)	A newscast-social modul által alkalmazott képlet a közösségi trendjelek pontszámítására: $V_H = 10 + Trends * 50$.	vö. 3.6.6.4
LexRank	Gráf-alapú extraktív szövegösszegzési algoritmus, amely a mondatok közötti koszinusz-hasonlóságot használja a legfontosabb mondatok kiválasztásához.	vö. 2.2.2
Etag gyorsítótárzás	HTTP feltételes kéréseken alapuló mechanizmus, amelyben a szerver entitáscímkéje (ETag) alapján a kliens eldönti, hogy szükséges-e az erőforrás újbóli letöltése.	vö. 3.6.1.1
Retry / exponenciális hátrálás	Hibakezelési stratégia, amelyben sikertelen kérés esetén a rendszer növekvő várakozási idővel próbálkozik újra.	vö. 3.4.3
Content hash deduplikáció	Tartalom-azonosítási módszer, amelyben a szöveg SHA-256 hash értéke alapján állapítható meg, hogy ugyanaz a tartalom korábban feldolgozásra került-e.	vö. 3.6.5.3

Narratíva flow	A newscast-feeder modul szövegformázási mechanizmusa, amely a kiválasztott híreket összefüggő, felolvasásra optimalizált szöveggé szerkeszti össze.	vö. 3.6.4.2
Backfill	A newscast-social modul mechanizmusa, amely a race condition kezelésére szolgál: ha egy hírhez még nem létezik elemzési rekord, a közösségi jeleket várakozási sorba helyezi.	vö. 3.6.6.5

8.5. Hivatkozások

T1 – Új, angol nyelvű cikkek (nem KJE-releváns)

- **Orosz Gy. et al. (2022):** „*HuSpaCy: an industrial-strength Hungarian natural language processing toolkit*”, XVIII. Magyar Számítógépes Nyelvészeti Konferencia, Szeged. <https://arxiv.org/abs/2201.01956> (Letöltve: 2026. február)
- **Orosz Gy. et al. (2023):** „*Advancing Hungarian Text Processing with HuSpaCy: Bridging the Gap with Enhanced Linguistic Pipelines*”. <https://arxiv.org/abs/2308.12635> (Letöltve: 2026. február)
- **Jurafsky, D. & Martin, J. H. (2023):** „*Speech and Language Processing*”, 3rd Edition (draft), Stanford University. <https://web.stanford.edu/~jurafsky/slp3/> (Letöltve: 2026. április)
- **Hays Hungary (2026):** „*Hays Hungary Salary Guide 2026*” (HU-EN). https://www.hays.hu/documents/63283/98156885/HU-EN_Hays+Hungary+Salary+Guide+2026.pdf.pdf (Letöltve: 2026. február)

T2 – Új, angol nyelvű cikkek (KJE-releváns)

- **Bán Gy., Rikk J., Pitlik L. (2024):** „*AI-based derivation of the importance of attributes in case of evaluation models*”, MIAU kutatási anyag. https://miau.myx.hu/miau/314/full_importance_of_attributes_in_evaluation_models.docx (Letöltve: 2026. április)
- **Várad D., Pitlik L. et al. (2023):** „*How can the overlearning-risk be detected and handled in model- and case-level based on the DIF(e)SA (direct-inverse-function-symmetry-approach)?*”, MIAU kutatási anyag. https://miau.myx.hu/miau/294/difesa_abstract_full_text_2023_v5.docx (Letöltve: 2026. április)

T3 – Új, angol nyelvű weboldalak (nem KJE-releváns)

- **OpenAPI Initiative:** „*OpenAPI Specification v3.1.0*”, Linux Foundation. <https://swagger.io/specification/> (Letöltve: 2026. február)
- **TechPolicy.Press (2024):** „*Researcher Data Access Under the DSA: Lessons from TikTok's API Issues*”. <https://techpolicy.press/> (Letöltve: 2026. március)
- **D-Lab, UC Berkeley:** „*The Evolving Landscape of Web Scraping on Social Media Platforms*”. <https://dlab.berkeley.edu/> (Letöltve: 2026. március)
- **IAPP:** „*The state of web scraping in the EU*”, International Association of Privacy Professionals. <https://iapp.org/> (Letöltve: 2026. március)
- **Morgan Lewis (2024):** „*EU Data Protection Regulators Take Restrictive Position on AI Scraping*”, Morgan Lewis & Bockius LLP. <https://www.morganlewis.com/> (Letöltve: 2026. március)
- **Ramírez, S.:** „*FastAPI - Modern, fast web framework for building APIs with Python*”. <https://fastapi.tiangolo.com/> (Letöltve: 2026. február)
- **Explosion AI:** „*spaCy - Industrial-Strength Natural Language Processing*”. <https://spacy.io/> (Letöltve: 2026. február)
- **HuSpaCy:** „*HuSpaCy - Industrial-strength Hungarian NLP*”. <https://huspacy.github.io/> (Letöltve: 2026. február)
- **NLTK Project:** „*Natural Language Toolkit*”. <https://www.nltk.org/> (Letöltve: 2026. február)
- **Belica, M.:** „*Sumy - Automatic text summarizer*”, GitHub. <https://github.com/miso-belica/sumy> (Letöltve: 2026. február)
- **Pydantic:** „*Pydantic - Data validation using Python type hints*”. <https://docs.pydantic.dev/latest/> (Letöltve: 2026. február)
- **SQLAlchemy:** „*SQLAlchemy - The Database Toolkit for Python*”. <https://www.sqlalchemy.org/> (Letöltve: 2026. február)
- **SQLAlchemy:** „*SQLAlchemy ORM Documentation (2.0)*”. <https://docs.sqlalchemy.org/en/20/orm/> (Letöltve: 2026. február)

- **MariaDB Foundation:** „*MariaDB - The open source relational database*”. <https://mariadb.org/> (Letöltve: 2026. február)
- **MariaDB Corporation:** „*MariaDB Knowledge Base*”. <https://mariadb.com/kb/en/documentation/> (Letöltve: 2026. február)
- **Docker, Inc.:** „*Docker Documentation – What is a Container?*”. <https://docs.docker.com/> (Letöltve: 2026. február)
- **Prometheus Authors:** „*Prometheus - Monitoring system & time series database*”. <https://prometheus.io/> (Letöltve: 2026. február)
- **Grafana Labs:** „*Grafana - The open observability platform*”. <https://grafana.com/> (Letöltve: 2026. február)
- **ElevenLabs:** „*ElevenLabs API Documentation*”. <https://elevenlabs.io/docs/api-reference/introduction> (Letöltve: 2026. február)
- **ElevenLabs:** „*ElevenLabs Models Documentation*”. <https://elevenlabs.io/docs/overview/models> (Letöltve: 2026. február)
- **Google:** „*Gemini API - Google AI for Developers*”. <https://ai.google.dev/gemini-api/docs> (Letöltve: 2026. február)
- **Bachmann, M.:** „*RapidFuzz - Rapid fuzzy string matching in Python and C++*”, GitHub. <https://github.com/rapidfuzz/RapidFuzz> (Letöltve: 2026. március)
- **X Corp.:** „*X API Access Levels and Pricing*”. <https://developer.x.com/en/docs/twitter-api/getting-started/about-twitter-api> (Letöltve: 2026. március)
- **Reddit, Inc.:** „*Reddit Data API Terms*”. <https://www.reddit.com/wiki/api/> (Letöltve: 2026. március)
- **Meta for Developers:** „*Instagram Graph API*”. <https://developers.facebook.com/docs/instagram-api/> (Letöltve: 2026. március)
- **TikTok:** „*TikTok Research API Documentation*”. <https://developers.tiktok.com/doc/research-api/> (Letöltve: 2026. március)
- **Google Developers:** „*YouTube Data API v3 - Quota and Compliance Audits*”. <https://developers.google.com/youtube/v3/getting-started> (Letöltve: 2026. március)

- **Apify:** „*Apify - Web Scraping and Automation Platform*”. <https://docs.apify.com/> (Letöltve: 2026. március)
- **Meta for Developers:** „*Graph API - Rate Limiting*”. <https://developers.facebook.com/docs/graph-api/overview/rate-limiting/> (Letöltve: 2026. március)
- **Meta for Developers:** „*App Review*”. <https://developers.facebook.com/docs/resp-plat-initiatives/app-review/> (Letöltve: 2026. március)
- **Meta Business Help Center:** „*Custom Audiences*”. <https://www.facebook.com/business/help/341425252616329> (Letöltve: 2026. március)
- **Meta for Developers:** „*Instagram Platform - Rate Limits*”. <https://developers.facebook.com/docs/instagram-platform/rate-limits/> (Letöltve: 2026. március)
- **X Corp. (2023):** „*Deprecation of the Academic Research product track*”. <https://developer.x.com/en/blog/product/academic-research-deprecation> (Letöltve: 2026. március)
- **Telegram:** „*Bot API Documentation*”. <https://core.telegram.org/bots/api> (Letöltve: 2026. március)
- **Facebook Research:** „*FAISS - A library for efficient similarity search*”, GitHub. <https://github.com/facebookresearch/faiss> (Letöltve: 2026. február)
- **FAISS:** „*FAISS Documentation*”. <https://faiss.ai/index.html> (Letöltve: 2026. február)
- **Auth0:** „*JSON Web Tokens - Introduction*”, JWT.io. <https://jwt.io/> (Letöltve: 2026. február)
- **1Panel:** „*1Panel - Open-source server management panel*”. <https://1panel.hk/> (Letöltve: 2026. február)
- **OpenResty:** „*OpenResty - Scalable Web Platform*”. <https://openresty.org/> (Letöltve: 2026. február)
- **Meta AI:** „*FAISS - Facebook AI Similarity Search*”. <https://ai.meta.com/tools/faiss/> (Letöltve: 2026. február)

- **Menn, J. (2024):** „*Reddit signs AI content licensing deal ahead of IPO*”, Reuters. <https://www.reuters.com/technology/reddit-ai-content-licensing-deal/> (Letöltve: 2026. március)
- **OWASP Foundation (2021):** „*OWASP Top Ten – 2021*”. <https://owasp.org/Top10/> (Letöltve: 2026. április)
- **Google (2024):** „*Gemini API Safety Settings Documentation*”. <https://ai.google.dev/gemini-api/docs/safety-settings> (Letöltve: 2026. április)
- **Wikipedia:** „*RSS*”. <https://en.wikipedia.org/wiki/RSS> (Letöltve: 2026. február)
- **Wikipedia:** „*FastAPI*”. <https://en.wikipedia.org/wiki/FastAPI> (Letöltve: 2026. február)
- **Wikipedia:** „*Docker (software)*”. [https://en.wikipedia.org/wiki/Docker_\(software\)](https://en.wikipedia.org/wiki/Docker_(software)) (Letöltve: 2026. február)
- **Wikipedia:** „*Microservices*”. <https://en.wikipedia.org/wiki/Microservices> (Letöltve: 2026. február)
- **Wikipedia:** „*MariaDB*”. <https://en.wikipedia.org/wiki/MariaDB> (Letöltve: 2026. február)
- **Wikipedia:** „*Prometheus (software)*”. [https://en.wikipedia.org/wiki/Prometheus_\(software\)](https://en.wikipedia.org/wiki/Prometheus_(software)) (Letöltve: 2026. február)
- **Wikipedia:** „*Grafana*”. <https://en.wikipedia.org/wiki/Grafana> (Letöltve: 2026. február)
- **Wikipedia:** „*JSON Web Token*”. https://en.wikipedia.org/wiki/JSON_Web_Token (Letöltve: 2026. február)
- **Wikipedia:** „*REST*”. <https://en.wikipedia.org/wiki/REST> (Letöltve: 2026. február)
- **Wikipedia:** „*tf-idf*”. <https://en.wikipedia.org/wiki/Tf-idf> (Letöltve: 2026. február)
- **Wikipedia:** „*Flesch–Kincaid readability tests*”. https://en.wikipedia.org/wiki/Flesch–Kincaid_readability_tests (Letöltve: 2026. február)
- **Wikipedia:** „*Speech synthesis*”. https://en.wikipedia.org/wiki/Speech_synthesis (Letöltve: 2026. február)

- **Wikipedia:** „*Speech Synthesis Markup Language*”.
https://en.wikipedia.org/wiki/Speech_Synthesis_Markup_Language (Letöltve: 2026. február)
- **Wikipedia:** „*bcrypt*”. <https://en.wikipedia.org/wiki/Bcrypt> (Letöltve: 2026. február)
- **Wikipedia:** „*FAISS*”. <https://en.wikipedia.org/wiki/FAISS> (Letöltve: 2026. február)
- **Wikipedia:** „*OpenAPI Specification*”.
https://en.wikipedia.org/wiki/OpenAPI_Specification (Letöltve: 2026. február)
- **Wikipedia:** „*Natural Language Toolkit*”.
https://en.wikipedia.org/wiki/Natural_Language_Toolkit (Letöltve: 2026. február)
- **Wikipedia:** „*Cambridge Analytica*”.
https://en.wikipedia.org/wiki/Cambridge_Analytica (Letöltve: 2026. március)
- **Wikipedia:** „*General Data Protection Regulation*”.
https://en.wikipedia.org/wiki/General_Data_Protection_Regulation (Letöltve: 2026. március)
- **Wikipedia:** „*2023 Reddit API controversy*”.
https://en.wikipedia.org/wiki/2023_Reddit_API_controversy (Letöltve: 2026. március)

T4 – Új, angol nyelvű weboldalak (KJE-releváns)

- **Pitlik L. (é.n.):** „*COCO (Component-based Object Comparison for Objectivity) – Online Expert System*”, my-x.hu. <https://miau.my-x.hu/myx-free/coco/index.html> (Letöltve: 2026. április)

T5 – Új, nem angol nyelvű cikkek (nem KJE-releváns)

- **Lendvai P. – Tikk D. (2022):** „*Magyar nyelvű szövegsztályozás transzformer-alapú modellekkel*”, Alkalmazott Nyelvtudomány, XXII. évf.
<https://doi.org/10.18460/ANY.2022.1.005> (Letöltve: 2026. április)

T6 – Új, nem angol nyelvű cikkek (KJE-releváns)

- **Kodolányi János Egyetem hivatalos szakdolgozati minta:** „*Szakedolgozati minta és formai követelmények*”, KJE.
https://www.kodolanyi.hu/konyvtar/images/tartalom/File/Honlapra/Feltoltes/szakedoli_minta.pdf (Letöltve: 2026. március)

- **Pitlik L. – Barta G. (2021):** „*Anti-diszkriminatív modellezés közelítése solver-alapú neurális hálókka*”, MIAU kutatási anyag. <https://miau.my-x.hu/miau/275/solver-based-neural-networks.pdf> (Letöltve: 2026. április)
- **Farkas D. (2023):** „*Webes adatgyűjtés Python környezetben*”, KJE szakdolgozat. https://miau.my-x.hu/miau/320/farkas_david.pdf (Letöltve: 2026. április)

T7 – Új, nem angol nyelvű weboldalak (nem KJE-releváns)

- **NMHH (2024):** „*Médiapiaci Jelentés 2024*”, Nemzeti Média- és Hírközlési Hatóság. https://nmhh.hu/cikk/249791/Mediapiaci_Jelentes_2024 (Letöltve: 2026. február)
- **NMHH (2024):** „*Az online médiatér közönsége (2024. december)*”. https://nmhh.hu/cikk/250142/Az_online_mediator_kozonsege_2024_december (Letöltve: 2026. február)
- **NMHH:** „*Kutatások*”. <https://nmhh.hu/kutatasok> (Letöltve: 2026. február)
- **Fizetesek.hu:** „*Szerkesztő pozíció fizetési adatok*”. <https://fizetesek.hu/fizetesek/konyvkiadas-nyomdaipar-media/szerkeszto> (Letöltve: 2026. február)
- **HungaroMet:** „*HungaroMet hivatalos oldal*” (volt OMSZ). <https://www.met.hu/> (Letöltve: 2026. február)
- **Wikipedia (magyar):** „*HungaroMet*”. <https://hu.wikipedia.org/wiki/HungaroMet> (Letöltve: 2026. február)
- **RackForest.hu:** „*RackForest - Magyar hosting-szolgáltató*”. <https://www.rackforest.hu/> (Letöltve: 2026. február)

T8 – Új, nem angol nyelvű weboldalak (KJE-releváns)

- **Pitlik L. (1998–2026):** „*MIAU tudásbázis és COCO API*”, my-x.hu. <https://miau.my-x.hu/myx-free/coco/> (Letöltve: 2026. február)
- **Pitlik L. ifj. – Pitlik L. – Pitlik M. (2026):** „*CT 00 – A szakdolgozat formai és tartalmi elvárásai*”, MIAU MediaWiki. https://miau.my-x.hu/mediawiki/index.php/CT_00 (Letöltve: 2026. április)
- **Kodolányi János Egyetem (2021):** „*Egységes monografikus/portfóliós szakdolgozati szabályzat*”, KJE Könyvtár.

https://www.kodolanyi.hu/konyvtar/images/tartalom/File/Honlapra/Feltoltes/kje_egyseg_szakdolgozati_szabalyzat_2021marc17_hatalyos.pdf (Letöltve: 2026. április)

T9 – Régi, angol nyelvű cikkek (nem KJE-releváns)

- **Bird, S. & Loper, E. (2002):** „*NLTK: The Natural Language Toolkit*”, Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching NLP. <https://arxiv.org/abs/cs/0205028> (Letöltve: 2026. február)
- **Boyd, A. – Stewart, P. – Alexander, R. (2012):** „*Broadcast Journalism: Techniques of Radio and Television News*”, Routledge, 7th Edition. <https://www.routledge.com/Broadcast-Journalism-Techniques-of-Radio-and-Television-News/Boyd-Stewart-Alexander/p/book/9780240824215> (Letöltve: 2026. április)
- **van den Oord, A. et al. (2016):** „*WaveNet: A Generative Model for Raw Audio*”, DeepMind. <https://arxiv.org/abs/1609.03499> (Letöltve: 2026. április)
- **Martin, R. C. (2003):** „*Agile Software Development: Principles, Patterns, and Practices*”, Prentice Hall. <https://www.informit.com/store/agile-software-development-principles-patterns-and-9780135974445> (Letöltve: 2026. április)
- **Gamma, E. – Helm, R. – Johnson, R. – Vlissides, J. (1994):** „*Design Patterns: Elements of Reusable Object-Oriented Software*”, Addison-Wesley. <https://www.informit.com/store/design-patterns-elements-of-reusable-object-oriented-9780201633610> (Letöltve: 2026. április)
- **Codd, E. F. (1970):** „*A Relational Model of Data for Large Shared Data Banks*”, Communications of the ACM, Vol. 13, No. 6, pp. 377–387. <https://dl.acm.org/doi/10.1145/362384.362685> (Letöltve: 2026. április)
- **Erkan, G. & Radev, D. R. (2004):** „*LexRank: Graph-based Lexical Centrality as Salience in Text Summarization*”, Journal of Artificial Intelligence Research, Vol. 22, pp. 457-479. <https://arxiv.org/abs/1109.2128> (Letöltve: 2026. február)
- **Shannon, C. E. (1948):** „*A Mathematical Theory of Communication*”, Bell System Technical Journal, Vol. 27, pp. 379–423, 623–656. <https://doi.org/10.1002/j.1538-7305.1948.tb01338.x> (Letöltve: 2026. április)

T10 – Régi, angol nyelvű cikkek (KJE-releváns)

- **Pitlik L. (2015):** „*Definitions of benchmarks for modelling experts on the field of forecasting*”, my-x.hu. <https://miau.my-x.hu/miau/202/benchmark.pdf> (Letöltve: 2026. április)

T11 – Régi, angol nyelvű weboldalak (nem KJE-releváns)

- **Jones, M. et al. (2015):** „*RFC 7519: JSON Web Token (JWT)*”, IETF. <https://datatracker.ietf.org/doc/html/rfc7519> (Letöltve: 2026. február)
- **Fielding, R. T. & Reschke, J. (2014):** „*RFC 7230: Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing*”, IETF. <https://datatracker.ietf.org/doc/html/rfc7230> (Letöltve: 2026. április)
- **Fielding, R. T. & Reschke, J. (2014):** „*RFC 7232: Hypertext Transfer Protocol (HTTP/1.1): Conditional Requests*”, IETF. <https://datatracker.ietf.org/doc/html/rfc7232> (Letöltve: 2026. február)
- **Winer, D. (2002):** „*RSS 2.0 Specification*”, Berkman Center for Internet & Society, Harvard Law School. <https://cyber.harvard.edu/rss/rss.html> (Letöltve: 2026. február)
- **W3C (2010):** „*Speech Synthesis Markup Language (SSML) Version 1.1*”, W3C Recommendation. <https://www.w3.org/TR/speech-synthesis11/> (Letöltve: 2026. február)
- **MIT Internet Policy Research Initiative (2018):** „*The Cambridge Analytica Affair and the Future of Data Privacy*”. <https://internetpolicy.mit.edu/> (Letöltve: 2026. március)
- **Zuckerberg, M. (2018):** „*An Update on Our Plans to Restrict Data Access on Facebook*”, Facebook Newsroom. <https://about.fb.com/news/2018/04/restricting-data-access/> (Letöltve: 2026. március)
- **Lewis, J. & Fowler, M. (2014):** „*Microservices: a definition of this new architectural term*”. <https://martinfowler.com/articles/microservices.html> (Letöltve: 2026. február)
- **Provos, N. & Mazières, D. (1999):** „*A Future-Adaptable Password Scheme*”, Proceedings of the USENIX Annual Technical Conference. <https://www.usenix.org/legacy/events/usenix99/provos.html> (Letöltve: 2026. február)

- **Lemmetty, S. (1999):** „*History and Development of Speech Synthesis*”, Helsinki University of Technology / Aalto University.
http://research.spa.aalto.fi/publications/theses/lemmetty_mst/chap2.html (Letöltve: 2026. február)
- **Bayer, M. (2012):** „*SQLAlchemy*”, in: Brown, A. & Wilson, G. (eds.): „*The Architecture of Open Source Applications, Volume II*”, aosabook.org.
<https://aosabook.org/en/v2/sqlalchemy.html> (Letöltve: 2026. április)
- **Fowler, M. (2012):** „*TestPyramid*”, martinowler.com.
<https://martinowler.com/bliki/TestPyramid.html> (Letöltve: 2026. április)
- **Nielsen, J. (1994):** „*10 Usability Heuristics for User Interface Design*”, Nielsen Norman Group. <https://www.nngroup.com/articles/ten-usability-heuristics/> (Letöltve: 2026. április)
- **Fielding, R. T. (2000):** „*Architectural Styles and the Design of Network-based Software Architectures*”, Doctoral dissertation, University of California, Irvine.
https://ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm (Letöltve: 2026. február)

T12 – Régi, angol nyelvű weboldalak (KJE-releváns)

- **Pitlik L. (é.n.):** „*Similarity Analysis – Definitions and Methodology*”, my-x.hu.
https://miau.my-x.hu/myx-free/index_en.php3?x=s01 (Letöltve: 2026. április)

T13 – Régi, nem angol nyelvű cikkek (nem KJE-releváns)

- **Prószéky G. – Kis B. (1999):** „*A Morphological Analyser for Hungarian*”, Proceedings of the Workshop on Computational Approaches to Semitic Languages, ACL. (Társszerzős, magyar vonatkozású, de a konferencia nem KJE.) (Letöltve: 2026. április)

T14 – Régi, nem angol nyelvű cikkek (KJE-releváns)

- **Szűcs I., Pitlik L., Pető I. (2006):** „*Adatbányászat alapú modell-aggregálási módszerek*”, OTKA T049013 kutatási jelentés. http://miau.my-x.hu/miau/93/kaposvar_full.doc (Letöltve: 2026. március)

- **Pitlik L. (2004):** „COCO: Objektív ár-teljesítmény elemzés”. <http://miau.my-x.hu/mgm/2004oszcocomum.xls> (Letöltve: 2026. március)
- **Pitlik L. (é.n.)** „MY-X FREE online, önjavító teszt-rendszere”. <https://miau.my-x.hu/myx-free/index.php3?x=test1> (Letöltve: 2026. március)
- **Pető I. (2013):** „Hasonlóságelemzés COCO használatával”, oktatási segédanyag, my-x.hu. https://miau.my-x.hu/miau/189/coco_demo.pdf (Letöltve: 2026. április)
- **Pitlik L. (2013):** „Gyanúgenerálás a HR-kockázatok minimalizálása érdekében – hasonlóságelemzéssel”, Tudásmenedzsment, XIV. évf. https://miau.my-x.hu/miau/189/pitlik_gyanugeneralas.pdf (Letöltve: 2026. április)
- **Pitlik L. (2014):** „Hasonlóságelemzési alapok” (My-X Team ismertető füzet), my-x.hu. https://miau.my-x.hu/miau/196/My-X Team_A5_fuzet_HU_jav.pdf (Letöltve: 2026. április)

T15 – Régi, nem angol nyelvű weboldalak (nem KJE-releváns)

- **HungaroMet (2019):** „Az OMSZ szöveges előrejelzéseinek formátuma és szabályai”, HungaroMet Archívum. <https://www.met.hu/idojaras/elorejelzes/> (Letöltve: 2026. április)
- **Az Európai Parlament és a Tanács (2016):** „Az Európai Parlament és a Tanács (EU) 2016/679 rendelete (GDPR)”, 4. cikk, EUR-Lex (magyar nyelvű hivatalos szöveg). <https://eur-lex.europa.eu/legal-content/HU/TXT/?uri=CELEX:32016R0679> (Letöltve: 2026. április)

T16 – Régi, nem angol nyelvű weboldalak (KJE-releváns)

- **MIAU Wiki (2014):** „Hasonlóságelemzés”, MIAU MediaWiki. <https://miau.my-x.hu/mediawiki/index.php/Hasonlóságelemzés> (Letöltve: 2026. április)

T1-T16 statisztika:

	KJE rel., tudományos	KJE rel., egyéb	Külső tudományos	Külső egyéb	Σ
Magyar – Friss (≥2020)	T6: 3	T8: 3	T5: 1	T7: 7	14
Angol – Friss (≥2020)	T2: 2	T4: 1	T1: 4	T3: 63	70
Magyar – Régi (≤2019)	T14: 6	T16: 1	T13: 1	T15: 2	10

	KJE rel., tudományos	KJE rel., egyéb	Külső tudományos	Külső egyéb	Σ
Angol – Régi (≤2019)	T10: 1	T12: 1	T9: 8	T11: 13	23
Σ	12	6	14	85	117

34. táblázat: A szakdolgozat témájához kapcsolódó magyar és külföldi publikációk

A szakdolgozat hivatkozásai összesen 117 tételt tartalmaznak, amelyek 16 típusba (T1–T16) sorolhatók a nyelv (magyar/angol), a kor (friss: ≥ 2020 / régi: ≤ 2019), a jelleg (tudományos/egyéb) és a KJE relevancia (KJE releváns / külső) dimenziói mentén. A KJE releváns hivatkozások a Kodolányi János Egyetemhez kötődő forrásokat jelölik.

8.6. LLM-benchmark: Teljes API-konverzációk

A jelen melléklet a 3.10.5 alfejezetben (LLM-benchmark) hivatkozott API hívások teljes dokumentációját tartalmazza. Minden alfejezetben a pontos curl parancs, a teljes prompt (bemenet) és a teljes API válasz (kimenet) szó szerint, változtatás nélkül került rögzítésre. Az API kulcsok biztonsági okokból nem kerülnek közlésre.

8.6.0. Tesztadatok

A jelen alfejezet a 3.10.5 alfejezetben (LLM-benchmark) hivatkozott tesztadatok teljes szövegét tartalmazza. Ugyanezek a szövegek kerültek bemenetként a NewsCast benchmark-scriptekbe és az LLM API-hívások promptjaiba.

8.6.0.1. Benchmark 1 – Hírelemzési tesztadatok (10 teszthír)

TESZTHÍR 1 (politikai): „Orbán Viktor miniszterelnök ma Budapesten bejelentette, hogy a kormány 15%-kal emeli a pedagógusok bérét 2027-től. A döntést a Parlament egyhangúlag támogatta. Az intézkedés mintegy 120 ezer pedagógust érint országszerte.”

TESZTHÍR 2 (bűnügyi, erőszakos + drog): „Brutális gyilkosság történt a VIII. kerületben: egy 35 éves férfit megkéseltek egy kocsmai verekedés során. A rendőrség háború-szerű állapotokról számolt be. A támadó heroint és kokaint is fogyasztott a bántalmazás előtt.”

TESZTHÍR 3 (technológiai, nemzetközi): „Az Apple bemutatta legújabb iPhone modelljét San Franciscóban. Az új készülék mesterséges intelligencia alapú fotózási funkciókat kínál. Az ár 1200 dollárról indul.”

TESZTHÍR 4 (helyi, környezet): „A Balaton vízszintje rekordalacsony szintre csökkent a száraz nyár miatt. A helyi önkormányzatok és a magyar vízügyi hatóságok közös intézkedéscsomagot dolgoztak ki. A Balatoni Hajózási Zrt. több járatot is törölni kényszerült.”

TESZTHÍR 5 (reklám/hirdetés): „Kedvezmény! Kattints ide a legjobb áráért! A kalkulátor szerint akár 50%-ot is spórolhatsz. [hirdetés] Szponzorált tartalom a XY Biztosító megbízásából.”

TESZTHÍR 6 (társadalmi, hazai): „Magyarországon 2026-ban a fiatalok 73%-a napi szinten használ közösségi médiát hírfogyasztásra. A hazai kutatások szerint a budapesti egyetemisták körében a TikTok megelőzte a hagyományos hírportálokat.”

TESZTHÍR 7 (sport, helyi): „A magyar labdarúgó-válogatott 2-1-re legyőzte Portugáliát a budapesti Puskás Arénában. Szoboszlai Dominik két gólt szerzett. A szövetségi kapitány a csapat kiváló teljesítményét méltatta a meccs után.”

TESZTHÍR 8 (angol nyelvű, technológiai): „Scientists at MIT have developed a new quantum computing chip that operates at room temperature. The breakthrough could revolutionize cryptography and drug discovery within the next decade.”

TESZTHÍR 9 (időjárás, helyi): „A Nemzeti Meteorológiai Szolgálat figyelmeztetést adott ki: holnap -15°C-os hideg várható Budapesten, 80 km/h-s széllekedésekkel. A Magyar Közút Nonprofit Zrt. téliesítette az M1-es és M7-es autópályákat.”

TESZTHÍR 10 (szexuális tartalom): „A pornográf tartalmak terjedése az interneten egyre nagyobb aggodalomra ad okot. A szexuális tartalmakat kínáló oldalak látogatottsága 300%-kal nőtt. A molesztálás áldozatai számára új segélyvonalat indítottak.”

8.6.0.2. Benchmark 2 – Szövegnormalizálási tesztadatok (30 tesztmondat)

#	Tesztmondat
1	„A hőmérséklet holnap -15°C körül alakul.”
2	„2024. január 15-én kezdődik az új szemeszter.”
3	„A sebesség 120 km/h volt az M1-esen.”
4	„Dr. Kovács professzor 15:30-kor tart előadást.”
5	„Az infláció 7,5%-ra csökkent 2025-ben.”
6	„A termék ára 2500 Ft, kb. 7 € a jelenlegi árfolyamon.”

#	Tesztmondat
7	„Budapest 1. kerületében 25,5°C-ot mértek.”
8	„A 3. helyezett 10-15 másodperccel maradt le.”
9	„Szoboszlai a 89. percben szerezte a 2. gólt.”
10	„Az OMSZ szerint holnap 1023 hPa lesz a légnyomás.”
11	„A nyereség 150%-kal nőtt, vagyis kb. 3000000 Ft-tal.”
12	„Március 1-jén, 8:00-kor nyit az új bevásárlóközpont.”
13	„Prof. Nagy 2026. szeptember 30-án tartja a vizsgát.”
14	„A szél 80 km/h-s, a hőmérséklet -5°C.”
15	„Az épület 42 m magas, a felülete 1500 m ² .”
16	„Stb. a többi feladat ill. a vizsgák is érintettek.”
17	„A 2. fordulóban a csapat 3-1-re nyert, a 85. percben.”
18	„100%-os a hatékonyság, 0 Ft a költség.”
19	„2025. évben a GDP 4,2%-kal nőtt Magyarországon.”
20	„Április 20-án, 14:45-kor landolt a gép 250 km/h-val.”
21	„A lakosság 68%-a támogatja, ez kb. 6800000 fő.”
22	„Kr. e. 44-ben történt, vö. a római történelemmel.”
23	„A víz hőmérséklete 18,3°C, a levegőé 32°C.”
24	„Az 1. és 2. helyen magyar versenyzők végeztek.”
25	„10:00-tól 16:00-ig tart a rendezvény, 5000 Ft a belépő.”
26	„A 2024-es költségvetés 500000000 Ft-os deficitet mutat.”
27	„Június 1-je a gyermeknap, december 25-e a karácsony.”
28	„A sebesség 5-10 m/s, a hőmérséklet 20-25°C között.”
29	„A Ft/€ árfolyam 395,50, a \$/€ pedig 1,08.”
30	„I. e. 3000-ben alapították, az ún. „ösváros” nevet kapta.”

35. táblázat: Benchmark 2 – Szövegnormalizálási tesztadatok (30 tesztmondat)

8.6.0.3. Benchmark 3 – Szövegösszegzési tesztadatok (5 tesztthír)

TESZTHÍR A (gazdasági – MNB kamatdöntés): „A Magyar Nemzeti Bank Monetáris Tanácsa keddi ülésén úgy döntött, hogy változatlanul 6,5 százalékon tartja a jegybanki alapkamatot. A döntést a tanácstagok egyhangúlag hozták meg. Virág Barnabás alelnök a döntés

után tartott sajtótájékoztatón kiemelte, hogy az inflációs kilátások továbbra is bizonytalanok, különösen az energiaárak és az élelmiszerárak volatilitása miatt. Az alelnök hozzátette, hogy a forint árfolyamának stabilitása kulcsfontosságú a monetáris politika szempontjából. A piaci elemzők többsége előre jelezte a kamatszünet meghosszabbítását. Az MNB korábbi közleményeiben hangsúlyozta, hogy a kamatemelési ciklus lezárult, ugyanakkor a kamatsökkentés megkezdéséhez az infláció tartós, 3 százalékos alatti stabilizálódása szükséges. A régiós jegybankok közül a cseh és a lengyel nemzeti bank már megkezdte a kamatsökkentést, míg a román jegybank szintén kívár. A forint a döntés bejelentése után minimálisan gyengült az euróval szemben, 395,50 forintos árfolyamon zárt. A következő kamatdöntő ülés hat hét múlva esedékes.”

TESZTHÍR B (baleseti – M7 autópálya): „Tragikus baleset történt kedd reggel az M7-es autópályán Székesfehérvár közelében. Egy kamion és három személygépkocsi ütközött az 58-as kilométerszelvényben, a Budapest felé vezető oldalon. A balesetben két személy életét vesztette, öt további sérültet szállítottak kórházba, közülük kettő állapota súlyos. A Fejér Vármegyei Rendőr-főkapitányság közleménye szerint a kamiont vezető sofőr figyelmen kívül hagyta a torlódás miatti lassulást és fékezés nélkül rohant az előtte álló járműsorba. Az autópályát a helyszínelés idejére mindkét irányban lezárták, a forgalmat a 7-es főútra terelték. A lezárás a reggeli csúcsforgalommal egybeesve mintegy 25 kilométeres torlódást okozott a Budapest felé vezető irányban. Az Útinform tájékoztatása szerint az autópálya teljes újranyitása délutánra várható. A rendőrség halált okozó közúti baleset gondatlan okozása miatt indított eljárást.”

TESZTHÍR C (sport – Puskás Akadémia): „A Puskás Akadémia FC történelmi sikert ért el az Európa-liga csoportkörében: a felcsúti együttes 2-0-ra legyőzte a skót Rangers FC-t a Groupama Arénában. A mérkőzés első félidejében a Puskás dominált, 62 százalékos labdabirtoklással és 8 kapura lövéssel. Az első gólt Nagy Zsolt szerezte a 34. percben egy szabadrúgásból, amely a jobb felső sarokba vágódott. A második félidő elején a Rangers nyomást gyakorolt, de a Puskás védelme stabilan tartotta magát. A biztosító gólt Soisalo lőtte a 78. percben egy gyors kontrából, Komáromi Dániel gólpasszát követően. A győzelemmel a Puskás Akadémia 7 ponttal a csoport második helyére lépett elő és saját kezében tartja a továbbjutás lehetőségét. A magyar együttes következő mérkőzésén az olasz Lazio vendégeként lép pályára két hét múlva. Hornyák Zsolt vezetőedző a mérkőzés után a csapat mentális erejét emelte ki és kijelentette, hogy a Lazio elleni találkozóra is győzelmi szándékkal utaznak.”

TESZTHÍR D (oktatási – AI oktatás): „Az Innovációs és Technológiai Minisztérium bejelentette, hogy 2027-től minden magyar közoktatási intézményben kötelezővé teszik a mesterséges intelligencia alapjainak oktatását. A program az 5. osztálytól indul és fokozatosan bővül a középiskolai szintig. A tanterv kidolgozásában a BME, az ELTE és a Szegedi Tudományegyetem szakemberei vesznek részt. A miniszter kiemelte, hogy Magyarország az elsők között vezeti be az AI-oktatást Európában. A programhoz 15 milliárd forintos költségvetést különítettek el, amelyből tanárképzés, tananyagfejlesztés és infrastruktúra-bővítés valósul meg. Az oktatási szakszervezetek óvatosabban fogalmaztak: aggályaikat fejezték ki a pedagógusok felkészítésének időkerete miatt, mivel a tanárképzési program mindössze 120 órás. A szülői szervezetek többsége támogatja a kezdeményezést. A program első pilot-fázisa 2026 szeptemberében indul 50 kiválasztott intézményben.”

TESZTHÍR E (technológiai – TSMC chipgyártás): „A globális chipgyártó ipar újabb mérföldkőhöz érkezett: a tajvani TSMC bejelentette, hogy megkezdte a 1,4 nanométeres csomóponttechnológiájú chipek tömeggyártását. Ez a világ legkisebb tranzisztorméretű chipje, amely az előző generációhoz képest 30 százalékkal gyorsabb és 40 százalékkal energiatakarékosabb. Az első megrendelők között van az Apple, az NVIDIA és a Qualcomm. Az új technológia lehetővé teszi, hogy egy bélyeg méretű chipen több mint 400 milliárd tranzistor helyezkedjen el. A TSMC 20 milliárd dollárt fektetett az új gyártósorba, amelyet az arizonai üzemében helyeztek üzembe. Az iparági elemzők szerint a fejlesztés fenntartja a TSMC előnyét a Samsung és az Intel felett. A chipek várhatóan 2027 első negyedévéből kerülnek a fogyasztói termékekbe.”

8.6.1. Benchmark 1 – Hírelemzés: GPT-5.4-mini API válasz

Dátum: 2026. április 4. | *Modell:* gpt-5.4-mini | *temperature:* 0

API hívás:

```
curl -s https://api.openai.com/v1/chat/completions \  
  -H „Content-Type: application/json” \  
  -H „Authorization: Bearer $OPENAI_API_KEY” \  
  -d '{  
    „model”: „gpt-5.4-mini”,  
    „temperature”: 0,  
    „messages”: [  
      {„role”: „system”, „content”: „Válaszolj kizárólag JSON formátumban.”},  
      {„role”: „user”, „content”: „Te egy magyar nyelvű rádiós hírelemzési rendszer  
vagy. Az alábbi 10 hírszöveg mindegyikét elemezd az alábbi szempontok szerint.  
Válaszolj KIZÁRÓLAG az alábbi JSON-formátumban, minden hírhez külön objektumot
```

```
készítve.\n\nElemzési szempontok:\n1. sentiment: ... [teljes prompt]\n\nELEMEZENDŐ  
HÍREK:\n1: \"Orbán Viktor miniszterelnök...\" [mind a 10 teszt hír]"}  
]  
' | jq .
```

(A fenti curl parancsban a prompt terjedelmi okokból rövidítve szerepel. A teljes prompt szövege a 3.10.5.1 fejezet hivatkozott tesztadatait és az ott részletezett elemzési szempontokat tartalmazza.)

API válasz (kimenet):

```
[  
  {  
    „hir_szam”: 1,  
    „sentiment”: „positive”,  
    „readability”: „easy”,  
    „safety”: {  
      „total_score”: 4,  
      „is_safe_for_radio”: true,  
      „violence_score”: 0,  
      „sexual_score”: 0,  
      „drug_score”: 0,  
      „political_score”: 2  
    },  
    „radio_relevance”: {  
      „relevance_score”: 75,  
      „category”: „high”,  
      „is_suitable”: true,  
      „scores”: {  
        „readability”: 30,  
        „actuality”: 20,  
        „locality”: 20,  
        „safety”: 25  
      }  
    },  
    „is_ad_content”: false  
  },  
  {  
    „hir_szam”: 2,  
    „sentiment”: „negative”,  
    „readability”: „medium”,  
    „safety”: {  
      „total_score”: 52,  
      „is_safe_for_radio”: false,  
      „violence_score”: 4,  
      „sexual_score”: 0,  
      „drug_score”: 2,  
      „political_score”: 0  
    },  
    „radio_relevance”: {  
      „relevance_score”: 25,  
      „category”: „low”,  
      „is_suitable”: false,  
      „scores”: {  
        „readability”: 20,
```

```

        „actuality”: 20,
        „locality”: 0,
        „safety”: -25
    }
},
„is_ad_content”: false
},
{
    „hir_szam”: 3,
    „sentiment”: „neutral”,
    „readability”: „easy”,
    „safety”: {
        „total_score”: 0,
        „is_safe_for_radio”: true,
        „violence_score”: 0,
        „sexual_score”: 0,
        „drug_score”: 0,
        „political_score”: 0
    },
    „radio_relevance”: {
        „relevance_score”: 75,
        „category”: „high”,
        „is_suitable”: true,
        „scores”: {
            „readability”: 30,
            „actuality”: 20,
            „locality”: 0,
            „safety”: 25
        }
    },
    „is_ad_content”: false
},
{
    „hir_szam”: 4,
    „sentiment”: „negative”,
    „readability”: „medium”,
    „safety”: {
        „total_score”: 4,
        „is_safe_for_radio”: true,
        „violence_score”: 0,
        „sexual_score”: 0,
        „drug_score”: 0,
        „political_score”: 2
    },
    „radio_relevance”: {
        „relevance_score”: 65,
        „category”: „high”,
        „is_suitable”: true,
        „scores”: {
            „readability”: 20,
            „actuality”: 20,
            „locality”: 20,
            „safety”: 25
        }
    },
    „is_ad_content”: false
},
{
    „hir_szam”: 5,

```

```

„sentiment“: „positive“,
„readability“: „easy“,
„safety“: {
  „total_score“: 0,
  „is_safe_for_radio“: true,
  „violence_score“: 0,
  „sexual_score“: 0,
  „drug_score“: 0,
  „political_score“: 0
},
„radio_relevance“: {
  „relevance_score“: 75,
  „category“: „high“,
  „is_suitable“: true,
  „scores“: {
    „readability“: 30,
    „actuality“: 20,
    „locality“: 0,
    „safety“: 25
  }
},
„is_ad_content“: true
},
{
  „hir_szam“: 6,
  „sentiment“: „neutral“,
  „readability“: „medium“,
  „safety“: {
    „total_score“: 4,
    „is_safe_for_radio“: true,
    „violence_score“: 0,
    „sexual_score“: 0,
    „drug_score“: 0,
    „political_score“: 2
  },
  „radio_relevance“: {
    „relevance_score“: 65,
    „category“: „high“,
    „is_suitable“: true,
    „scores“: {
      „readability“: 20,
      „actuality“: 20,
      „locality“: 20,
      „safety“: 25
    }
  },
  „is_ad_content“: false
},
{
  „hir_szam“: 7,
  „sentiment“: „positive“,
  „readability“: „easy“,
  „safety“: {
    „total_score“: 2,
    „is_safe_for_radio“: true,
    „violence_score“: 0,
    „sexual_score“: 0,
    „drug_score“: 0,
    „political_score“: 1
  }
}

```

```

},
„radio_relevance“: {
  „relevance_score“: 75,
  „category“: „high“,
  „is_suitable“: true,
  „scores“: {
    „readability“: 30,
    „actuality“: 20,
    „locality“: 20,
    „safety“: 25
  }
},
„is_ad_content“: false
},
{
  „hir_szam“: 8,
  „sentiment“: „neutral“,
  „readability“: „difficult“,
  „safety“: {
    „total_score“: 7,
    „is_safe_for_radio“: true,
    „violence_score“: 0,
    „sexual_score“: 0,
    „drug_score“: 1,
    „political_score“: 0
  },
  „radio_relevance“: {
    „relevance_score“: 55,
    „category“: „medium“,
    „is_suitable“: true,
    „scores“: {
      „readability“: 10,
      „actuality“: 20,
      „locality“: 0,
      „safety“: 25
    }
  },
  „is_ad_content“: false
},
{
  „hir_szam“: 9,
  „sentiment“: „neutral“,
  „readability“: „medium“,
  „safety“: {
    „total_score“: 4,
    „is_safe_for_radio“: true,
    „violence_score“: 0,
    „sexual_score“: 0,
    „drug_score“: 0,
    „political_score“: 2
  },
  „radio_relevance“: {
    „relevance_score“: 65,
    „category“: „high“,
    „is_suitable“: true,
    „scores“: {
      „readability“: 20,
      „actuality“: 20,
      „locality“: 20,

```

```

        „safety”: 25
    }
},
„is_ad_content”: false
},
{
    „hir_szam”: 10,
    „sentiment”: „negative”,
    „readability”: „medium”,
    „safety”: {
        „total_score”: 16,
        „is_safe_for_radio”: false,
        „violence_score”: 0,
        „sexual_score”: 3,
        „drug_score”: 0,
        „political_score”: 0
    },
    „radio_relevance”: {
        „relevance_score”: 15,
        „category”: „low”,
        „is_suitable”: false,
        „scores”: {
            „readability”: 20,
            „actuality”: 20,
            „locality”: 0,
            „safety”: -25
        }
    },
    „is_ad_content”: false
}
]

```

8.6.2. Benchmark 1 – Hírelemzés: Claude Haiku 4.5 API válasz

Dátum: 2026. április 4. | **Modell:** claude-haiku-4-5-20251001 | **temperature:** 0

API hívás:

```

curl -s https://api.anthropic.com/v1/messages \
-H „Content-Type: application/json” \
-H „x-api-key: $ANTHROPIC_API_KEY” \
-H „anthropic-version: 2023-06-01” \
-d '{
    „model”: „claude-haiku-4-5-20251001”,
    „max_tokens”: 4096,
    „temperature”: 0,
    „messages”: [
        {„role”: „user”, „content”: „Te egy magyar nyelvű rádiós hírelemzési rendszer
vagy. ... [teljes prompt, azonos a 8.6.1-ben szereplővel]”}
    ]
}' | jq .

```

API válasz (kimenet):

```

[
  {
    „hir_szam”: 1,
    „sentiment”: „positive”,
    „readability”: „easy”,
    „safety”: {
      „total_score”: 8,
      „is_safe_for_radio”: false,
      „violence_score”: 0,
      „sexual_score”: 0,
      „drug_score”: 0,
      „political_score”: 8
    },
    „radio_relevance”: {
      „relevance_score”: 65,
      „category”: „high”,
      „is_suitable”: false,
      „scores”: {
        „readability”: 30,
        „actuality”: 20,
        „locality”: 15,
        „safety”: -25
      }
    },
    „is_ad_content”: false
  },
  {
    „hir_szam”: 2,
    „sentiment”: „negative”,
    „readability”: „medium”,
    „safety”: {
      „total_score”: 47,
      „is_safe_for_radio”: false,
      „violence_score”: 30,
      „sexual_score”: 0,
      „drug_score”: 14,
      „political_score”: 0
    },
    „radio_relevance”: {
      „relevance_score”: 5,
      „category”: „low”,
      „is_suitable”: false,
      „scores”: {
        „readability”: 20,
        „actuality”: 20,
        „locality”: 10,
        „safety”: -25
      }
    },
    „is_ad_content”: false
  },
  {
    „hir_szam”: 3,
    „sentiment”: „positive”,
    „readability”: „easy”,
    „safety”: {
      „total_score”: 0,
      „is_safe_for_radio”: true,

```

```

    „violence_score”: 0,
    „sexual_score”: 0,
    „drug_score”: 0,
    „political_score”: 0
  },
  „radio_relevance”: {
    „relevance_score”: 75,
    „category”: „high”,
    „is_suitable”: true,
    „scores”: {
      „readability”: 30,
      „actuality”: 20,
      „locality”: 0,
      „safety”: 25
    }
  },
  „is_ad_content”: false
},
{
  „hir_szam”: 4,
  „sentiment”: „negative”,
  „readability”: „easy”,
  „safety”: {
    „total_score”: 10,
    „is_safe_for_radio”: false,
    „violence_score”: 0,
    „sexual_score”: 0,
    „drug_score”: 0,
    „political_score”: 10
  },
  „radio_relevance”: {
    „relevance_score”: 60,
    „category”: „high”,
    „is_suitable”: false,
    „scores”: {
      „readability”: 30,
      „actuality”: 20,
      „locality”: 15,
      „safety”: -25
    }
  },
  „is_ad_content”: false
},
{
  „hir_szam”: 5,
  „sentiment”: „neutral”,
  „readability”: „easy”,
  „safety”: {
    „total_score”: 0,
    „is_safe_for_radio”: true,
    „violence_score”: 0,
    „sexual_score”: 0,
    „drug_score”: 0,
    „political_score”: 0
  },
  „radio_relevance”: {
    „relevance_score”: 30,
    „category”: „medium”,
    „is_suitable”: false,

```

```

    „scores“: {
      „readability“: 30,
      „actuality“: 20,
      „locality“: 0,
      „safety“: 25
    }
  },
  „is_ad_content“: true
},
{
  „hir_szam“: 6,
  „sentiment“: „neutral“,
  „readability“: „medium“,
  „safety“: {
    „total_score“: 4,
    „is_safe_for_radio“: true,
    „violence_score“: 0,
    „sexual_score“: 0,
    „drug_score“: 0,
    „political_score“: 4
  },
  „radio_relevance“: {
    „relevance_score“: 80,
    „category“: „high“,
    „is_suitable“: true,
    „scores“: {
      „readability“: 20,
      „actuality“: 20,
      „locality“: 20,
      „safety“: 25
    }
  },
  „is_ad_content“: false
},
{
  „hir_szam“: 7,
  „sentiment“: „positive“,
  „readability“: „easy“,
  „safety“: {
    „total_score“: 2,
    „is_safe_for_radio“: true,
    „violence_score“: 0,
    „sexual_score“: 0,
    „drug_score“: 0,
    „political_score“: 2
  },
  „radio_relevance“: {
    „relevance_score“: 90,
    „category“: „high“,
    „is_suitable“: true,
    „scores“: {
      „readability“: 30,
      „actuality“: 20,
      „locality“: 15,
      „safety“: 25
    }
  },
  „is_ad_content“: false
},

```

```

{
  „hir_szam”: 8,
  „sentiment”: „positive”,
  „readability”: „difficult”,
  „safety”: {
    „total_score”: 0,
    „is_safe_for_radio”: true,
    „violence_score”: 0,
    „sexual_score”: 0,
    „drug_score”: 0,
    „political_score”: 0
  },
  „radio_relevance”: {
    „relevance_score”: 35,
    „category”: „medium”,
    „is_suitable”: true,
    „scores”: {
      „readability”: 10,
      „actuality”: 20,
      „locality”: 0,
      „safety”: 25
    }
  },
  „is_ad_content”: false
},
{
  „hir_szam”: 9,
  „sentiment”: „negative”,
  „readability”: „easy”,
  „safety”: {
    „total_score”: 4,
    „is_safe_for_radio”: true,
    „violence_score”: 0,
    „sexual_score”: 0,
    „drug_score”: 0,
    „political_score”: 4
  },
  „radio_relevance”: {
    „relevance_score”: 85,
    „category”: „high”,
    „is_suitable”: true,
    „scores”: {
      „readability”: 30,
      „actuality”: 20,
      „locality”: 15,
      „safety”: 25
    }
  },
  „is_ad_content”: false
},
{
  „hir_szam”: 10,
  „sentiment”: „negative”,
  „readability”: „medium”,
  „safety”: {
    „total_score”: 24,
    „is_safe_for_radio”: false,
    „violence_score”: 0,
    „sexual_score”: 16,

```

```

    „drug_score”: 0,
    „political_score”: 0
  },
  „radio_relevance”: {
    „relevance_score”: 5,
    „category”: „low”,
    „is_suitable”: false,
    „scores”: {
      „readability”: 20,
      „actuality”: 20,
      „locality”: 0,
      „safety”: -25
    }
  },
  „is_ad_content”: false
}
]

```

8.6.3. Benchmark 1 – Hírelemzés: Gemini 3.1 Flash Lite API válasz

Dátum: 2026. április 4. | *Modell:* gemini-flash-lite-latest | *temperature:* 0

API hívás:

```

curl -s „https://generativelanguage.googleapis.com/v1beta/models/gemini-flash-lite-
latest:generateContent?key=$GOOGLE_API_KEY” \
  -H „Content-Type: application/json” \
  -d '{
    „contents”: [{„parts”: [{„text”: „Te egy magyar nyelvű rádiós hírelemzési
rendszer vagy. ... [teljes prompt, azonos a 8.6.1-ben szereplővel]}]}],
    „systemInstruction”: {
      parts: [{text: „Válaszolj kizárólag JSON formátumban.”}]
    },
    „generationConfig”: {„temperature”: 0}
  }' | jq .

```

API válasz (kimenet):

```

[
  {
    „hir_szam”: 1,
    „sentiment”: „positive”,
    „readability”: „medium”,
    „safety”: {
      „total_score”: 8,
      „is_safe_for_radio”: false,
      „violence_score”: 0,
      „sexual_score”: 0,
      „drug_score”: 0,
      „political_score”: 8
    },
    „radio_relevance”: {
      „relevance_score”: 45,

```

```

    „category”: „medium”,
    „is_suitable”: false,
    „scores”: {
      „readability”: 20,
      „actuality”: 20,
      „locality”: 10,
      „safety”: -25
    }
  },
  „is_ad_content”: false
},
{
  „hir_szam”: 2,
  „sentiment”: „negative”,
  „readability”: „medium”,
  „safety”: {
    „total_score”: 51,
    „is_safe_for_radio”: false,
    „violence_score”: 30,
    „sexual_score”: 0,
    „drug_score”: 14,
    „political_score”: 0
  },
  „radio_relevance”: {
    „relevance_score”: 20,
    „category”: „low”,
    „is_suitable”: false,
    „scores”: {
      „readability”: 20,
      „actuality”: 20,
      „locality”: 5,
      „safety”: -25
    }
  },
  „is_ad_content”: false
},
{
  „hir_szam”: 3,
  „sentiment”: „neutral”,
  „readability”: „easy”,
  „safety”: {
    „total_score”: 0,
    „is_safe_for_radio”: true,
    „violence_score”: 0,
    „sexual_score”: 0,
    „drug_score”: 0,
    „political_score”: 0
  },
  „radio_relevance”: {
    „relevance_score”: 75,
    „category”: „high”,
    „is_suitable”: true,
    „scores”: {
      „readability”: 30,
      „actuality”: 20,
      „locality”: 0,
      „safety”: 25
    }
  }
},

```

```

    „is_ad_content“: false
  },
  {
    „hir_szam“: 4,
    „sentiment“: „negative“,
    „readability“: „medium“,
    „safety“: {
      „total_score“: 0,
      „is_safe_for_radio“: true,
      „violence_score“: 0,
      „sexual_score“: 0,
      „drug_score“: 0,
      „political_score“: 0
    },
    „radio_relevance“: {
      „relevance_score“: 80,
      „category“: „high“,
      „is_suitable“: true,
      „scores“: {
        „readability“: 20,
        „actuality“: 20,
        „locality“: 15,
        „safety“: 25
      }
    }
  },
  „is_ad_content“: false
},
{
  „hir_szam“: 5,
  „sentiment“: „positive“,
  „readability“: „easy“,
  „safety“: {
    „total_score“: 0,
    „is_safe_for_radio“: true,
    „violence_score“: 0,
    „sexual_score“: 0,
    „drug_score“: 0,
    „political_score“: 0
  },
  „radio_relevance“: {
    „relevance_score“: 75,
    „category“: „high“,
    „is_suitable“: true,
    „scores“: {
      „readability“: 30,
      „actuality“: 20,
      „locality“: 0,
      „safety“: 25
    }
  }
},
  „is_ad_content“: true
},
{
  „hir_szam“: 6,
  „sentiment“: „neutral“,
  „readability“: „medium“,
  „safety“: {
    „total_score“: 0,
    „is_safe_for_radio“: true,

```

```

    „violence_score”: 0,
    „sexual_score”: 0,
    „drug_score”: 0,
    „political_score”: 0
  },
  „radio_relevance”: {
    „relevance_score”: 85,
    „category”: „high”,
    „is_suitable”: true,
    „scores”: {
      „readability”: 20,
      „actuality”: 20,
      „locality”: 20,
      „safety”: 25
    }
  },
  „is_ad_content”: false
},
{
  „hir_szam”: 7,
  „sentiment”: „positive”,
  „readability”: „easy”,
  „safety”: {
    „total_score”: 0,
    „is_safe_for_radio”: true,
    „violence_score”: 0,
    „sexual_score”: 0,
    „drug_score”: 0,
    „political_score”: 0
  },
  „radio_relevance”: {
    „relevance_score”: 95,
    „category”: „high”,
    „is_suitable”: true,
    „scores”: {
      „readability”: 30,
      „actuality”: 20,
      „locality”: 20,
      „safety”: 25
    }
  },
  „is_ad_content”: false
},
{
  „hir_szam”: 8,
  „sentiment”: „positive”,
  „readability”: „difficult”,
  „safety”: {
    „total_score”: 7,
    „is_safe_for_radio”: true,
    „violence_score”: 0,
    „sexual_score”: 0,
    „drug_score”: 7,
    „political_score”: 0
  },
  „radio_relevance”: {
    „relevance_score”: 65,
    „category”: „high”,
    „is_suitable”: true,

```

```

    „scores“: {
      „readability“: 10,
      „actuality“: 20,
      „locality“: 0,
      „safety“: 25
    }
  },
  „is_ad_content“: false
},
{
  „hir_szam“: 9,
  „sentiment“: „negative“,
  „readability“: „medium“,
  „safety“: {
    „total_score“: 0,
    „is_safe_for_radio“: true,
    „violence_score“: 0,
    „sexual_score“: 0,
    „drug_score“: 0,
    „political_score“: 0
  },
  „radio_relevance“: {
    „relevance_score“: 85,
    „category“: „high“,
    „is_suitable“: true,
    „scores“: {
      „readability“: 20,
      „actuality“: 20,
      „locality“: 20,
      „safety“: 25
    }
  },
  „is_ad_content“: false
},
{
  „hir_szam“: 10,
  „sentiment“: „negative“,
  „readability“: „medium“,
  „safety“: {
    „total_score“: 16,
    „is_safe_for_radio“: false,
    „violence_score“: 0,
    „sexual_score“: 16,
    „drug_score“: 0,
    „political_score“: 0
  },
  „radio_relevance“: {
    „relevance_score“: 15,
    „category“: „low“,
    „is_suitable“: false,
    „scores“: {
      „readability“: 20,
      „actuality“: 20,
      „locality“: 0,
      „safety“: -25
    }
  },
  „is_ad_content“: false
}

```

```
]
```

8.6.4. Benchmark 2 – Szövegnormalizálás: GPT-5.4-mini API válasz

Dátum: 2026. április 4. | Modell: gpt-5.4-mini | temperature: 0

API hívás:

```
curl -s https://api.openai.com/v1/chat/completions \  
-H „Content-Type: application/json” \  
-H „Authorization: Bearer $OPENAI_API_KEY” \  
-d '{  
  „model”: „gpt-5.4-mini”,  
  „temperature”: 0,  
  „messages”: [  
    {„role”: „system”, „content”: „Válaszolj kizárólag JSON formátumban.”},  
    {„role”: „user”, „content”: „Te egy magyar nyelvű rádiós szövegfelolvasó  
(TTS) előfeldolgozó rendszer vagy. A feladatod, hogy az alábbi 30 mondatot úgy írd  
át, ahogyan egy magyar rádiós bemondó felolvasná. ... [teljes  
prompt]\n\nMONDATOK:\n1: \"/>A hőmérséklet holnap -15°C körül alakul.\" [mind a 30  
tesztmondat]”}  
  ]  
}' | jq .
```

API válasz (kimenet):

```
[  
  {  
    „mondat_szam”: 1,  
    „eredeti”: „A hőmérséklet holnap -15°C körül alakul.”,  
    „normalizalt”: „A hőmérséklet holnap mínusz tizenöt fok körül alakul.”  
  },  
  {  
    „mondat_szam”: 2,  
    „eredeti”: „2024. január 15-én kezdődik az új szemeszter.”,  
    „normalizalt”: „Kétezer-huszonnégy január tizenötödikén kezdődik az új  
szemeszter.”  
  },  
  {  
    „mondat_szam”: 3,  
    „eredeti”: „A sebesség 120 km/h volt az M1-esen.”,  
    „normalizalt”: „A sebesség százhusz kilométer per óra volt az M egyesén.”  
  },  
  {  
    „mondat_szam”: 4,  
    „eredeti”: „Dr. Kovács professzor 15:30-kor tart előadást.”,  
    „normalizalt”: „Doktor Kovács professzor tizenöt óra harminckor tart előadást.”  
  },  
  {  
    „mondat_szam”: 5,  
    „eredeti”: „Az infláció 7,5%-ra csökkent 2025-ben.”,  
    „normalizalt”: „Az infláció hét és fél százalékra csökkent kétezer-  
huszonötben.”  
  },  
]
```

```

{
  „mondat_szam”: 6,
  „eredeti”: „A termék ára 2500 Ft, kb. 7 € a jelenlegi árfolyamon.”,
  „normalizalt”: „A termék ára kétezeröttszáz forint, körülbelül hét euró a
jelenlegi árfolyamon.”
},
{
  „mondat_szam”: 7,
  „eredeti”: „Budapest 1. kerületében 25,5°C-ot mértek.”,
  „normalizalt”: „Budapest első kerületében huszonöt és fél fokot mértek.”
},
{
  „mondat_szam”: 8,
  „eredeti”: „A 3. helyezett 10-15 másodperccel maradt le.”,
  „normalizalt”: „A harmadik helyezett tíz-tizenöt másodperccel maradt le.”
},
{
  „mondat_szam”: 9,
  „eredeti”: „Szoboszlai a 89. percben szerezte a 2. gólt.”,
  „normalizalt”: „Szoboszlai a nyolcvankilencedik percben szerezte a második
gólt.”
},
{
  „mondat_szam”: 10,
  „eredeti”: „Az OMSZ szerint holnap 1023 hPa lesz a légnyomás.”,
  „normalizalt”: „Az OMSZ szerint holnap ezerhuszonhárom hektopascal lesz a
légnyomás.”
},
{
  „mondat_szam”: 11,
  „eredeti”: „A nyereség 150%-kal nőtt, vagyis kb. 3000000 Ft-tal.”,
  „normalizalt”: „A nyereség százötven százalékkal nőtt, vagyis körülbelül
hárommillió forinttal.”
},
{
  „mondat_szam”: 12,
  „eredeti”: „Március 1-jén, 8:00-kor nyit az új bevásárlóközpont.”,
  „normalizalt”: „Március elsején, nyolc órakor nyit az új bevásárlóközpont.”
},
{
  „mondat_szam”: 13,
  „eredeti”: „Prof. Nagy 2026. szeptember 30-án tartja a vizsgát.”,
  „normalizalt”: „Professzor Nagy kétezer-huszonhat szeptember harmincadikán
tartja a vizsgát.”
},
{
  „mondat_szam”: 14,
  „eredeti”: „A szél 80 km/h-s, a hőmérséklet -5°C.”,
  „normalizalt”: „A szél nyolcvan kilométer per órás, a hőmérséklet mínusz öt
fok.”
},
{
  „mondat_szam”: 15,
  „eredeti”: „Az épület 42 m magas, a felülete 1500 m².”,
  „normalizalt”: „Az épület negyvenkét méter magas, a felülete ezeröttszáz
négyzetméter.”
},
{
  „mondat_szam”: 16,

```

```

    „eredeti”: „Stb. a többi feladat ill. a vizsgák is érintettek.”,
    „normalizalt”: „És a többi a többi feladat illetőleg a vizsgák is érintettek.”
  },
  {
    „mondat_szam”: 17,
    „eredeti”: „A 2. fordulóban a csapat 3-1-re nyert, a 85. percben.”,
    „normalizalt”: „A második fordulóban a csapat három-egyre nyert, a
nyolcvanötödik percben.”
  },
  {
    „mondat_szam”: 18,
    „eredeti”: „100%-os a hatékonyság, 0 Ft a költség.”,
    „normalizalt”: „Százszázalékos a hatékonyság, nulla forint a költség.”
  },
  {
    „mondat_szam”: 19,
    „eredeti”: „2025. évben a GDP 4,2%-kal nőtt Magyarországon.”,
    „normalizalt”: „Kétezer-huszonöt évben a GDP négy egész kettő tized százalékkal
nőtt Magyarországon.”
  },
  {
    „mondat_szam”: 20,
    „eredeti”: „Április 20-án, 14:45-kor landolt a gép 250 km/h-val.”,
    „normalizalt”: „Április huszadikán, tizennégy óra negyvenöttil landolt a gép
kétszázötven kilométer per órával.”
  },
  {
    „mondat_szam”: 21,
    „eredeti”: „A lakosság 68%-a támogatja, ez kb. 6800000 fő.”,
    „normalizalt”: „A lakosság hatvannyolc százaléka támogatja, ez körülbelül
hatmillió-nyolcszázézer fő.”
  },
  {
    „mondat_szam”: 22,
    „eredeti”: „Kr. e. 44-ben történt, vö. a római történelemmel.”,
    „normalizalt”: „Krisztus előtt negyvennégyben történt, vesd össze a római
történelemmel.”
  },
  {
    „mondat_szam”: 23,
    „eredeti”: „A víz hőmérséklete 18,3°C, a levegőé 32°C.”,
    „normalizalt”: „A víz hőmérséklete tizennyolc egész három tized fok, a levegőé
harminckét fok.”
  },
  {
    „mondat_szam”: 24,
    „eredeti”: „Az 1. és 2. helyen magyar versenyzők végeztek.”,
    „normalizalt”: „Az első és második helyen magyar versenyzők végeztek.”
  },
  {
    „mondat_szam”: 25,
    „eredeti”: „10:00-tól 16:00-ig tart a rendezvény, 5000 Ft a belépő.”,
    „normalizalt”: „Tíz órától tizenhat óráig tart a rendezvény, ötezer forint a
belépő.”
  },
  {
    „mondat_szam”: 26,
    „eredeti”: „A 2024-es költségvetés 500000000 Ft-os deficitet mutat.”,

```

```

    „normalizalt”: „A kétezer-huszonegyes költségvetés ötszázmillió forintos
deficitet mutat.”
  },
  {
    „mondat_szam”: 27,
    „eredeti”: „Június 1-je a gyermeknap, december 25-e a karácsony.”,
    „normalizalt”: „Június elseje a gyermeknap, december huszonötödike a
karácsony.”
  },
  {
    „mondat_szam”: 28,
    „eredeti”: „A sebesség 5-10 m/s, a hőmérséklet 20-25°C között.”,
    „normalizalt”: „A sebesség öt-tíz méter per másodperc, a hőmérséklet húsz-
huszonöt fok között.”
  },
  {
    „mondat_szam”: 29,
    „eredeti”: „A Ft/€ árfolyam 395,50, a $/€ pedig 1,08.”,
    „normalizalt”: „A forint euró árfolyam háromszázkilencvenöt egész ötven, a
dollár euró pedig egy egész nyolc század.”
  },
  {
    „mondat_szam”: 30,
    „eredeti”: „I. e. 3000-ben alapították, az ún. „ősváros\” nevet kapta.”,
    „normalizalt”: „Időszámításunk előtt háromezerben alapították, az úgynevezett
ősváros nevet kapta.”
  }
]

```

8.6.5. Benchmark 2 – Szövegnormalizálás: Claude Haiku 4.5 API válasz

Dátum: 2026. április 4. | **Modell:** claude-haiku-4-5-20251001 | **temperature:** 0

API hívás:

```

curl -s https://api.anthropic.com/v1/messages \
-H „Content-Type: application/json” \
-H „x-api-key: $ANTHROPIC_API_KEY” \
-H „anthropic-version: 2023-06-01” \
-d '{
  „model”: „claude-haiku-4-5-20251001”,
  „max_tokens”: 4096,
  „temperature”: 0,
  „messages”: [
    {„role”: „user”, „content”: „Te egy magyar nyelvű rádiós szövegfelolvasó
(TTS) előfeldolgozó rendszer vagy. ... [teljes prompt, azonos a 8.6.4-ben
szereplővel]”}
  ]
}' | jq .

```

API válasz (kimenet):

[

```

{
  „mondat_szam”: 1,
  „eredeti”: „A hőmérséklet holnap -15°C körül alakul.”,
  „normalizalt”: „A hőmérséklet holnap mínusz tizenöt fok körül alakul.”
},
{
  „mondat_szam”: 2,
  „eredeti”: „2024. január 15-én kezdődik az új szemeszter.”,
  „normalizalt”: „Kétezer-huszonnégy január tizenötödikén kezdődik az új
szemeszter.”
},
{
  „mondat_szam”: 3,
  „eredeti”: „A sebesség 120 km/h volt az M1-esen.”,
  „normalizalt”: „A sebesség százhusz kilométer per óra volt az M egy-esen.”
},
{
  „mondat_szam”: 4,
  „eredeti”: „Dr. Kovács professzor 15:30-kor tart előadást.”,
  „normalizalt”: „Doktor Kovács professzor tizenöt óra harminckor tart előadást.”
},
{
  „mondat_szam”: 5,
  „eredeti”: „Az infláció 7,5%-ra csökkent 2025-ben.”,
  „normalizalt”: „Az infláció hét és fél százalékra csökkent kétezer-huszonöt-
ben.”
},
{
  „mondat_szam”: 6,
  „eredeti”: „A termék ára 2500 Ft, kb. 7 € a jelenlegi árfolyamon.”,
  „normalizalt”: „A termék ára kétezeröttszáz forint, körülbelül hét euró a
jelenlegi árfolyamon.”
},
{
  „mondat_szam”: 7,
  „eredeti”: „Budapest 1. kerületében 25,5°C-ot mértek.”,
  „normalizalt”: „Budapest első kerületében huszonöt és fél fokot mértek.”
},
{
  „mondat_szam”: 8,
  „eredeti”: „A 3. helyezett 10-15 másodperccel maradt le.”,
  „normalizalt”: „A harmadik helyezett tíz-tizenöt másodperccel maradt le.”
},
{
  „mondat_szam”: 9,
  „eredeti”: „Szoboszlai a 89. percben szerezte a 2. gólt.”,
  „normalizalt”: „Szoboszlai a nyolcvankilencedik percben szerezte a második
gólt.”
},
{
  „mondat_szam”: 10,
  „eredeti”: „Az OMSZ szerint holnap 1023 hPa lesz a légnyomás.”,
  „normalizalt”: „Az OMSZ szerint holnap ezerhúsz-három hektopascal lesz a
légnyomás.”
},
{
  „mondat_szam”: 11,
  „eredeti”: „A nyereség 150%-kal nőtt, vagyis kb. 3000000 Ft-tal.”,

```

```

    „normalizalt”: „A nyereség százötvenszázalékkal nőtt, vagyis körülbelül
hárommillió forinttal.”
  },
  {
    „mondat_szam”: 12,
    „eredeti”: „Március 1-jén, 8:00-kor nyit az új bevásárlóközpont.”,
    „normalizalt”: „Március elsején, nyolc órakor nyit az új bevásárlóközpont.”
  },
  {
    „mondat_szam”: 13,
    „eredeti”: „Prof. Nagy 2026. szeptember 30-án tartja a vizsgát.”,
    „normalizalt”: „Professzor Nagy kétezer-huszonhat szeptember harmincadikán
tartja a vizsgát.”
  },
  {
    „mondat_szam”: 14,
    „eredeti”: „A szél 80 km/h-s, a hőmérséklet -5°C.”,
    „normalizalt”: „A szél nyolcvan kilométer per órás, a hőmérséklet mínusz öt
fok.”
  },
  {
    „mondat_szam”: 15,
    „eredeti”: „Az épület 42 m magas, a felülete 1500 m².”,
    „normalizalt”: „Az épület negyvenkét méter magas, a felülete ezeröttszáz
négyzetméter.”
  },
  {
    „mondat_szam”: 16,
    „eredeti”: „Stb. a többi feladat ill. a vizsgák is érintettek.”,
    „normalizalt”: „És a többi a többi feladat illetőleg a vizsgák is érintettek.”
  },
  {
    „mondat_szam”: 17,
    „eredeti”: „A 2. fordulóban a csapat 3-1-re nyert, a 85. percben.”,
    „normalizalt”: „A második fordulóban a csapat három-egyre nyert, a
nyolcvanötödik percben.”
  },
  {
    „mondat_szam”: 18,
    „eredeti”: „100%-os a hatékonyság, 0 Ft a költség.”,
    „normalizalt”: „Százszázalékos a hatékonyság, nulla forint a költség.”
  },
  {
    „mondat_szam”: 19,
    „eredeti”: „2025. évben a GDP 4,2%-kal nőtt Magyarországon.”,
    „normalizalt”: „Kétezer-huszonöt évben a GDP négy egész kettő tizedszázalékkal
nőtt Magyarországon.”
  },
  {
    „mondat_szam”: 20,
    „eredeti”: „Április 20-án, 14:45-kor landolt a gép 250 km/h-val.”,
    „normalizalt”: „Április huszadikán, tizennégy óra negyvenötökör landolt a gép
kétszázötven kilométer per órával.”
  },
  {
    „mondat_szam”: 21,
    „eredeti”: „A lakosság 68%-a támogatja, ez kb. 6800000 fő.”,
    „normalizalt”: „A lakosság hatvannyolc százaléka támogatja, ez körülbelül
hatmillió-nyolcszázézer fő.”
  }

```

```

},
{
  „mondat_szam”: 22,
  „eredeti”: „Kr. e. 44-ben történt, vö. a római történelemmel.”,
  „normalizalt”: „Krisztus előtt negyvennégyben történt, vedd össze a római
történelemmel.”
},
{
  „mondat_szam”: 23,
  „eredeti”: „A víz hőmérséklete 18,3°C, a levegőé 32°C.”,
  „normalizalt”: „A víz hőmérséklete tizennyolc egész három tized fok, a levegőé
harminckettő fok.”
},
{
  „mondat_szam”: 24,
  „eredeti”: „Az 1. és 2. helyen magyar versenyzők végeztek.”,
  „normalizalt”: „Az első és második helyen magyar versenyzők végeztek.”
},
{
  „mondat_szam”: 25,
  „eredeti”: „10:00-tól 16:00-ig tart a rendezvény, 5000 Ft a belépő.”,
  „normalizalt”: „Tíz órától tizenhat óráig tart a rendezvény, ötezer forint a
belépő.”
},
{
  „mondat_szam”: 26,
  „eredeti”: „A 2024-es költségvetés 5000000000 Ft-os deficitet mutat.”,
  „normalizalt”: „A kétezer-huszonnégy-es költségvetés ötszázmillió forintos
deficitet mutat.”
},
{
  „mondat_szam”: 27,
  „eredeti”: „Június 1-je a gyermeknap, december 25-e a karácsony.”,
  „normalizalt”: „Június elseje a gyermeknap, december huszonötödike a
karácsony.”
},
{
  „mondat_szam”: 28,
  „eredeti”: „A sebesség 5-10 m/s, a hőmérséklet 20-25°C között.”,
  „normalizalt”: „A sebesség öt-tíz méter per másodperc, a hőmérséklet húsz-
huszonöt fok között.”
},
{
  „mondat_szam”: 29,
  „eredeti”: „A Ft/€ árfolyam 395,50, a $/€ pedig 1,08.”,
  „normalizalt”: „A forint euró árfolyam háromszázkilencvenöt egész ötven, a
dollár euró pedig egy egész nyolc század.”
},
{
  „mondat_szam”: 30,
  „eredeti”: „I. e. 3000-ben alapították, az ún. „ősváros” nevet kapta.”,
  „normalizalt”: „Időszámításunk előtt háromezerben alapították, az úgynevezett
ősváros nevet kapta.”
}
]

```

8.6.6. Benchmark 2 – Szövegnormalizálás: Gemini 3.1 Flash Lite API válasz

Dátum: 2026. április 4. | *Modell:* gemini-flash-lite-latest | *temperature:* 0

API hívás:

```
curl -s „https://generativelanguage.googleapis.com/v1beta/models/gemini-flash-lite-latest:generateContent?key=$GOOGLE_API_KEY” \
-H „Content-Type: application/json” \
-d '{
  „contents”: [{„parts”: [{„text”: „Te egy magyar nyelvű rádiós szövegfelolvasó (TTS) előfeldolgozó rendszer vagy. ... [teljes prompt, azonos a 8.6.4-ben szereplővel]}"}]},
  „systemInstruction”: {
    parts: [{text: „Válaszolj kizárólag JSON formátumban.”}]
  },
  „generationConfig”: {„temperature”: 0}
}' | jq .
```

API válasz (kimenet):

```
[
  { „mondat_szam”: 1, „eredeti”: „A hőmérséklet holnap -15°C körül alakul.”,
    „normalizalt”: „A hőmérséklet holnap mínusz tizenöt fok körül alakul.” },
  { „mondat_szam”: 2, „eredeti”: „2024. január 15-én kezdődik az új szemeszter.”,
    „normalizalt”: „Kétezer-huszonegy január tizenötödikén kezdődik az új szemeszter.”
  },
  { „mondat_szam”: 3, „eredeti”: „A sebesség 120 km/h volt az M1-esen.”,
    „normalizalt”: „A sebesség százhusz kilométer per óra volt az M egyes autópályán.”
  },
  { „mondat_szam”: 4, „eredeti”: „Dr. Kovács professzor 15:30-kor tart előadást.”,
    „normalizalt”: „Doktor Kovács professzor tizenöt óra harminckor tart előadást.” },
  { „mondat_szam”: 5, „eredeti”: „Az infláció 7,5%-ra csökkent 2025-ben.”,
    „normalizalt”: „Az infláció hét és fél százalékra csökkent kétezer-huszonegben.” },
  { „mondat_szam”: 6, „eredeti”: „A termék ára 2500 Ft, kb. 7 € a jelenlegi árfolyamon.”,
    „normalizalt”: „A termék ára kétezeröttszáz forint, körülbelül hét euró a jelenlegi árfolyamon.” },
  { „mondat_szam”: 7, „eredeti”: „Budapest 1. kerületében 25,5°C-ot mértek.”,
    „normalizalt”: „Budapest első kerületében huszoneg egész öt tized fokot mértek.” },
  { „mondat_szam”: 8, „eredeti”: „A 3. helyezett 10-15 másodperccel maradt le.”,
    „normalizalt”: „A harmadik helyezett tíz-tizenöt másodperccel maradt le.” },
  { „mondat_szam”: 9, „eredeti”: „Szoboszlai a 89. percben szerezte a 2. gólt.”,
    „normalizalt”: „Szoboszlai a nyolcvankilencedik percben szerezte a második gólt.”
  },
  { „mondat_szam”: 10, „eredeti”: „Az OMSZ szerint holnap 1023 hPa lesz a légnyomás.”,
    „normalizalt”: „Az Országos Meteorológiai Szolgálat szerint holnap ezerhuszoneg három hektopascal lesz a légnyomás.” },
  { „mondat_szam”: 11, „eredeti”: „A nyereség 150%-kal nőtt, vagyis kb. 3000000 Ft-tal.”,
    „normalizalt”: „A nyereség százötven százalékkal nőtt, vagyis körülbelül hárommillió forinttal.” },
  { „mondat_szam”: 12, „eredeti”: „Március 1-jén, 8:00-kor nyit az új bevásárlóközpont.”,
    „normalizalt”: „Március elsején, nyolc órakor nyit az új bevásárlóközpont.” },

```

```

{ „mondat_szam”: 13, „eredeti”: „Prof. Nagy 2026. szeptember 30-án tartja a vizsgát.”, „normalizalt”: „Professzor Nagy kétezer-huszonhat szeptember harmincadikán tartja a vizsgát.” },
{ „mondat_szam”: 14, „eredeti”: „A szél 80 km/h-s, a hőmérséklet -5°C.”, „normalizalt”: „A szél nyolcvan kilométer per órás, a hőmérséklet mínusz öt fok.” },
{ „mondat_szam”: 15, „eredeti”: „Az épület 42 m magas, a felülete 1500 m².”, „normalizalt”: „Az épület negyvenkét méter magas, a felülete ezeröttszáz négyzetméter.” },
{ „mondat_szam”: 16, „eredeti”: „Stb. a többi feladat ill. a vizsgák is érintettek.”, „normalizalt”: „És a többi, a többi feladat illetőleg a vizsgák is érintettek.” },
{ „mondat_szam”: 17, „eredeti”: „A 2. fordulóban a csapat 3-1-re nyert, a 85. percben.”, „normalizalt”: „A második fordulóban a csapat három-egyre nyert, a nyolcvanötödik percben.” },
{ „mondat_szam”: 18, „eredeti”: „100%-os a hatékonyság, 0 Ft a költség.”, „normalizalt”: „Százszázalékos a hatékonyság, nulla forint a költség.” },
{ „mondat_szam”: 19, „eredeti”: „2025. évben a GDP 4,2%-kal nőtt Magyarországon.”, „normalizalt”: „Kétezer-huszonöt évben a bruttó hazai termék négy egész kettő tized százalékkal nőtt Magyarországon.” },
{ „mondat_szam”: 20, „eredeti”: „Április 20-án, 14:45-kor landolt a gép 250 km/h-val.”, „normalizalt”: „Április huszadikán, tizennégy óra negyvenötökör landolt a gép kétszázötven kilométer per órával.” },
{ „mondat_szam”: 21, „eredeti”: „A lakosság 68%-a támogatja, ez kb. 6800000 fő.”, „normalizalt”: „A lakosság hatvannyolc százaléka támogatja, ez körülbelül hatmillió-nyolcszázézer fő.” },
{ „mondat_szam”: 22, „eredeti”: „Kr. e. 44-ben történt, vö. a római történelemmel.”, „normalizalt”: „Krisztus előtt negyvennégyben történt, vesd össze a római történelemmel.” },
{ „mondat_szam”: 23, „eredeti”: „A víz hőmérséklete 18,3°C, a levegőé 32°C.”, „normalizalt”: „A víz hőmérséklete tizennyolc egész három tized fok, a levegőé harminckét fok.” },
{ „mondat_szam”: 24, „eredeti”: „Az 1. és 2. helyen magyar versenyzők végeztek.”, „normalizalt”: „Az első és második helyen magyar versenyzők végeztek.” },
{ „mondat_szam”: 25, „eredeti”: „10:00-tól 16:00-ig tart a rendezvény, 5000 Ft a belépő.”, „normalizalt”: „Tíz órától tizenhat óráig tart a rendezvény, ötezer forint a belépő.” },
{ „mondat_szam”: 26, „eredeti”: „A 2024-es költségvetés 500000000 Ft-os deficitet mutat.”, „normalizalt”: „A kétezer-huszonnégyes költségvetés ötszázmillió forintos deficitet mutat.” },
{ „mondat_szam”: 27, „eredeti”: „Június 1-je a gyermeknap, december 25-e a karácsony.”, „normalizalt”: „Június elseje a gyermeknap, december huszonötödike a karácsony.” },
{ „mondat_szam”: 28, „eredeti”: „A sebesség 5-10 m/s, a hőmérséklet 20-25°C között.”, „normalizalt”: „A sebesség öt-tíz méter per szekundum, a hőmérséklet húsz-huszonöt fok között.” },
{ „mondat_szam”: 29, „eredeti”: „A Ft/€ árfolyam 395,50, a $/€ pedig 1,08.”, „normalizalt”: „A forint euró árfolyam háromszázkilencvenöt egész ötven század, a dollár euró pedig egy egész nyolc század.” },
{ „mondat_szam”: 30, „eredeti”: „I. e. 3000-ben alapították, az ún. „ősváros” nevet kapta.”, „normalizalt”: „Időszámításunk előtt háromezerben alapították, az úgynevezett ősváros nevet kapta.” }
]

```

8.6.7. Benchmark 3 – Szövegösszegzés: GPT-5.4-mini API válasz

Dátum: 2026. április 4. | **Modell:** gpt-5.4-mini | **temperature:** 0

API hívás:

```
curl -s https://api.openai.com/v1/chat/completions \  
-H „Content-Type: application/json” \  
-H „Authorization: Bearer $OPENAI_API_KEY” \  
-d '{  
  „model”: „gpt-5.4-mini”,  
  „temperature”: 0,  
  „messages”: [  
    {„role”: „system”, „content”: „Válaszolj kizárólag JSON formátumban.”},  
    {„role”: „user”, „content”: „Te egy magyar nyelvű rádiós hírszerkesztési  
rendszer vagy. A feladatod, hogy az alábbi 5 hírcikk mindegyikéből PONTOSAN 3  
mondatos összegzést készíts. ... [teljes prompt]\n\nHÍRCIKKEK:\nA: \"A Magyar  
Nemzeti Bank...\" [mind az 5 tesztír]”}  
  ]  
}' | jq .
```

API válasz (kimenet):

```
[  
  {  
    „hir”: „A”,  
    „extraktív”: [  
      „A Magyar Nemzeti Bank Monetáris Tanácsa keddi ülésén úgy döntött, hogy  
változatlanul 6,5 százalékon tartja a jegybanki alapkamatot.”,  
      „A döntést a tanácstagok egyhangúlag hozták meg.”,  
      „Virág Barnabás alelnök a döntés után tartott sajtótájékoztatón kiemelte,  
hogy az inflációs kilátások továbbra is bizonytalanok, különösen az energiaárak és  
az élelmiszerárak volatilitása miatt.”  
    ],  
    „abstraktív”: [  
      „Az MNB változatlanul 6,5 százalékon tartotta az alapkamatot.”,  
      „A döntés egyhangú volt, az inflációs kilátások továbbra is bizonytalanok.”,  
      „A forint stabilitása kulcsfontosságú, a kamatcsökkentéshez tartósan alacsony  
infláció kell.”  
    ]  
  },  
  {  
    „hir”: „B”,  
    „extraktív”: [  
      „Tragikus baleset történt kedd reggel az M7-es autópályán Székesfehérvár  
közeliében.”,  
      „A balesetben két személy életét veszítette, öt további sérültet szállítottak  
kórházba, közülük kettő állapota súlyos.”,  
      „Az autópályát a helyszínelés idejére mindkét irányban lezárták, a forgalmat  
a 7-es főútra terelték.”  
    ],  
    „abstraktív”: [  
      „Súlyos baleset történt az M7-esen Székesfehérvár közelében.”,  
      „Ketten meghaltak, öten megsérültek, közülük ketten súlyosan.”,  
      „Az autópályát lezárták, a forgalmat a 7-es főútra terelték.”  
    ]  
  },  
  {  
    „hir”: „C”,
```

```

    „extraktív”: [
      „A Puskás Akadémia FC történelmi sikert ért el az Európa-liga csoportkörében:
a felcsúti együttes 2-0-ra legyőzte a skót Rangers FC-t a Groupama Arénában.”,
      „Az első gólt Nagy Zsolt szerezte a 34. percben egy szabadrúgásból, amely a
jobb felső sarokba vágódott.”,
      „A győzelemmel a Puskás Akadémia 7 ponttal a csoport második helyére lépett
elő és saját kezében tartja a továbbjutás lehetőségét.”
    ],
    „abstraktív”: [
      „A Puskás Akadémia 2-0-ra legyőzte a Rangers FC-t az Európa-ligában.”,
      „Nagy Zsolt és Soisalo szereztek a gólokat a felcsúti csapatnak.”,
      „A győzelemmel a Puskás második helyre lépett és közel került a
továbbjutáshoz.”
    ]
  },
  {
    „hir”: „D”,
    „extraktív”: [
      „Az Innovációs és Technológiai Minisztérium bejelentette, hogy 2027-től
minden magyar közoktatási intézményben kötelezővé teszik a mesterséges
intelligencia alapjainak oktatását.”,
      „A program az 5. osztálytól indul és fokozatosan bővül a középiskolai
szintig.”,
      „A program első pilot-fázisa 2026 szeptemberében indul 50 kiválasztott
intézményben.”
    ],
    „abstraktív”: [
      „2027-től kötelező lesz az MI-alapok oktatása a magyar közoktatásban.”,
      „A program az ötödik osztálytól indul és a középiskoláig bővül.”,
      „A pilot 2026 szeptemberében kezdődik 50 intézményben.”
    ]
  },
  {
    „hir”: „E”,
    „extraktív”: [
      „A globális chipgyártó ipar újabb mérföldkőhöz érkezett: a tajvani TSMC
bejelentette, hogy megkezdte a 1,4 nanométeres csomóponttechnológiájú chipek
tömeggyártását.”,
      „Ez a világ legkisebb tranzisztorméretű chipje, amely az előző generációhoz
képest 30 százalékkal gyorsabb és 40 százalékkal energiatakarékosabb.”,
      „A chipek várhatóan 2027 első negyedétől kerülnek a fogyasztói termékekbe.”
    ],
    „abstraktív”: [
      „A TSMC megkezdte az 1,4 nanométeres chipek tömeggyártását.”,
      „Az új chip gyorsabb és takarékosabb az előző generációnál.”,
      „A fogyasztói termékekben várhatóan 2027 elejétől jelennek meg.”
    ]
  }
]

```

8.6.8. Benchmark 3 – Szövegösszegzés: Claude Haiku 4.5 API válasz

Dátum: 2026. április 4. | **Modell:** claude-haiku-4-5-20251001 | **temperature:** 0

API hívás:

```

curl -s https://api.anthropic.com/v1/messages \
-H „Content-Type: application/json” \
-H „x-api-key: $ANTHROPIC_API_KEY” \
-H „anthropic-version: 2023-06-01” \
-d '{
  „model”: „claude-haiku-4-5-20251001”,
  „max_tokens”: 4096,
  „temperature”: 0,
  „messages”: [
    {„role”: „user”, „content”: „Te egy magyar nyelvű rádiós hírszerkesztési
rendszer vagy. ... [teljes prompt, azonos a 8.6.7-ben szereplővel]”}
  ]
}' | jq .

```

API válasz (kimenet):

```

[
  {
    „hir”: „A”,
    „extraktív”: [
      „A Magyar Nemzeti Bank Monetáris Tanácsa keddi ülésén úgy döntött, hogy
változatlanul 6,5 százalékon tartja a jegybanki alapkamatot.”,
      „Virág Barnabás alelnök a döntés után tartott sajtótájékoztatón kiemelte,
hogy az inflációs kilátások továbbra is bizonytalanok, különösen az energiaárak és
az élelmiszerárak volatilitása miatt.”,
      „Az MNB korábbi közleményeiben hangsúlyozta, hogy a kamatemelési ciklus
lezárult, ugyanakkor a kamatcsökkentés megkezdéséhez az infláció tartós, 3 százalék
alatti stabilizálódása szükséges.”
    ],
    „abstraktív”: [
      „Az MNB változatlanul 6,5 százalékon tartotta a jegybanki alapkamatot.”,
      „Az inflációs kilátások bizonytalanok maradnak az energiaárak volatilitása
miatt.”,
      „A kamatcsökkentéshez az infláció 3 százalék alatti stabilizálódása
szükséges.”
    ]
  },
  {
    „hir”: „B”,
    „extraktív”: [
      „Tragikus baleset történt kedd reggel az M7-es autópályán Székesfehérvár
közeliében.”,
      „A balesetben két személy életét veszítette, öt további sérültet szállítottak
kórházba, közülük kettő állapota súlyos.”,
      „A kamiont vezető sofőr figyelmen kívül hagyta a torlódás miatti lassulást és
fékezés nélkül rohant az előtte álló járműsorba.”
    ],
    „abstraktív”: [
      „Súlyos baleset az M7-es autópályán Székesfehérvár közelében: egy kamion
három autónak ütközött.”,
      „Ketten meghaltak, öten megsérültek, közülük kettő súlyos állapotban van.”,
      „A kamionsofőr nem vette észre a torlódást és fékezés nélkül rohant az előtte
álló járművekbe.”
    ]
  },
]

```

```

    „hir”: „C”,
    „extraktív”: [
        „A Puskás Akadémia FC történelmi sikert ért el az Európa-liga csoportkörében: a felcsúti együttes 2-0-ra legyőzte a skót Rangers FC-t a Groupama Arénában.”,
        „Az első gólt Nagy Zsolt szerezte a 34. percben egy szabadrúgásból, amely a jobb felső sarokba vágódott.”,
        „A győzelemmel a Puskás Akadémia 7 ponttal a csoport második helyére lépett elő és saját kezében tartja a továbbjutás lehetőségét.”
    ],
    „abstraktív”: [
        „A Puskás Akadémia 2-0-ra legyőzte a Rangers FC-t az Európa-liga csoportkörében.”,
        „Nagy Zsolt szabadrúgásból és Soisalo kontrából szerzett gólt.”,
        „A felcsúti csapat 7 ponttal a csoport második helyén áll és továbbjuthat.”
    ]
},
{
    „hir”: „D”,
    „extraktív”: [
        „Az Innovációs és Technológiai Minisztérium bejelentette, hogy 2027-től minden magyar közoktatási intézményben kötelezővé teszik a mesterséges intelligencia alapjainak oktatását.”,
        „A program az 5. osztálytól indul és fokozatosan bővül a középiskolai szintig.”,
        „A programhoz 15 milliárd forintos költségvetést különítettek el, amelyből tanárképzés, tananyagfejlesztés és infrastruktúra-bővítés valósul meg.”
    ],
    „abstraktív”: [
        „2027-től kötelezővé válik a mesterséges intelligencia oktatása minden magyar közoktatási intézményben.”,
        „Az AI-oktatás az 5. osztálytól indul és a középiskolai szintig terjed.”,
        „A program 15 milliárd forintos költségvetésből tanárképzésre és tananyagfejlesztésre fordítódik.”
    ]
},
{
    „hir”: „E”,
    „extraktív”: [
        „A tajvani TSMC bejelentette, hogy megkezdte a 1,4 nanométeres csomóponttechnológiájú chipek tömeggyártását.”,
        „Ez a világ legkisebb tranzisztorméretű chipje, amely az előző generációhoz képest 30 százalékkal gyorsabb és 40 százalékkal energiatakarékosabb.”,
        „Az első megrendelők között van az Apple, az NVIDIA és a Qualcomm.”
    ],
    „abstraktív”: [
        „A TSMC megkezdte a 1,4 nanométeres chipek tömeggyártását, amely a világ legkisebb tranzisztormérete.”,
        „Az új chipek 30 százalékkal gyorsabbak és 40 százalékkal energiatakarékosabbak az előzőnél.”,
        „Az Apple, az NVIDIA és a Qualcomm már megrendelték az új technológiájú chipeket.”
    ]
}
]

```

8.6.9. Benchmark 3 – Szövegösszegzés: Gemini 3.1 Flash Lite API válasz

Dátum: 2026. április 4. | *Modell:* gemini-flash-lite-latest | *temperature:* 0

API hívás:

```
curl -s „https://generativelanguage.googleapis.com/v1beta/models/gemini-flash-lite-
latest:generateContent?key=$GOOGLE_API_KEY” \
-H „Content-Type: application/json” \
-d '{
  „contents”: [{„parts”: [{„text”: „Te egy magyar nyelvű rádiós hírszerkesztési
rendszer vagy. ... [teljes prompt, azonos a 8.6.7-ben szereplővel]}]}],
  „systemInstruction”: {
    parts: [{text: „Válaszolj kizárólag JSON formátumban.”}]
  },
  „generationConfig”: {„temperature”: 0}
}' | jq .
```

API válasz (kimenet):

```
[
  {
    „hir”: „A”,
    „extraktív”: [
      „A Magyar Nemzeti Bank Monetáris Tanácsa keddi ülésén úgy döntött, hogy
változatlanul 6,5 százalékon tartja a jegybanki alapkamatot.”,
      „A döntést a tanácstagok egyhangúlag hozták meg.”,
      „A következő kamatdöntő ülés hat hét múlva esedékes.”
    ],
    „abstraktív”: [
      „Változatlanul hat és fél százalékon tartotta az alapkamatot a Magyar Nemzeti
Bank Monetáris Tanácsa.”,
      „A döntést egyhangúlag hozták meg a jegybanki szakemberek.”,
      „A következő kamatdöntő ülésre hat hét múlva kerül sor.”
    ]
  },
  {
    „hir”: „B”,
    „extraktív”: [
      „Tragikus baleset történt kedd reggel az M7-es autópályán Székesfehérvár
közeliében.”,
      „A balesetben két személy életét vesztette, öt további sérültet szállítottak
kórházba, közülük kettő állapota súlyos.”,
      „A rendőrség halált okozó közúti baleset gondatlan okozása miatt indított
eljárást.”
    ],
    „abstraktív”: [
      „Két ember meghalt, öten pedig megsérültek az M7-es autópályán történt
tömegbalesetben.”,
      „Egy kamion fékezés nélkül rohant az előtte torlódó járműsorba Székesfehérvár
közeliében.”,
      „A rendőrség gondatlan emberölés gyanújával indított eljárást az ügyben.”
    ]
  }
],
```

```

{
  „hir”: „C”,
  „extraktív”: [
    „A Puskás Akadémia FC történelmi sikert ért el az Európa-liga csoportkörében: a felcsúti együttes 2-0-ra legyőzte a skót Rangers FC-t a Groupama Arénában.”,
    „A győzelemmel a Puskás Akadémia 7 ponttal a csoport második helyére lépett elő és saját kezében tartja a továbbjutás lehetőségét.”,
    „A magyar együttes következő mérkőzésén az olasz Lazio vendégeként lép pályára két hét múlva.”
  ],
  „abstraktív”: [
    „Történelmi győzelmet aratott a Puskás Akadémia az Európa-ligában a skót Rangers ellen.”,
    „A kétgólos sikerrel a felcsúti csapat a csoport második helyére lépett elő.”,
    „A magyar együttes két hét múlva az olasz Lazio otthonában lép pályára.”
  ]
},
{
  „hir”: „D”,
  „extraktív”: [
    „Az Innovációs és Technológiai Minisztérium bejelentette, hogy 2027-től minden magyar közoktatási intézményben kötelezővé teszik a mesterséges intelligencia alapjainak oktatását.”,
    „A program az 5. osztálytól indul és fokozatosan bővül a középiskolai szintig.”,
    „A program első pilot-fázisa 2026 szeptemberében indul 50 kiválasztott intézményben.”
  ],
  „abstraktív”: [
    „Kötelezővé teszik a mesterséges intelligencia oktatását a magyar iskolákban 2027-től.”,
    „A képzés az ötödik osztálytól kezdődően fokozatosan épül be a tantervbe.”,
    „A programhoz tizenöt milliárd forintos költségvetést különített el a kormány.”
  ]
},
{
  „hir”: „E”,
  „extraktív”: [
    „A globális chipgyártó ipar újabb mérföldkőhöz érkezett: a tajvani TSMC bejelentette, hogy megkezdte a 1,4 nanométeres csomóponttechnológiájú chipek tömeggyártását.”,
    „Ez a világ legkisebb tranzisztorméretű chipje, amely az előző generációhoz képest 30 százalékkal gyorsabb és 40 százalékkal energiatakarékosabb.”,
    „A chipek várhatóan 2027 első negyedétől kerülnek a fogyasztói termékekbe.”
  ],
  „abstraktív”: [
    „Megkezdte a világ legfejlettebb, 1,4 nanométeres chipjeinek tömeggyártását a tajvani TSMC.”,
    „Az új technológia jelentősen gyorsabb és energiatakarékosabb működést tesz lehetővé.”,
    „Az új chipek várhatóan 2027 elején jelennek meg a fogyasztói eszközökben.”
  ]
}
]

```

8.6.10. Fejlesztési LLM-konverzáció: Claude Code – szövegnormalizálási szabály

Dátum: 2026. április 15. | Eszköz: Claude Code CLI (Anthropic, claude-opus-4-6)

Prompt:

A newscast-tts/text_normalizer.py modulban a magyar nyelvű szövegnormalizálás egyik hiányzó esete a tizedes törtek felolvasása. Jelenleg a „3,5” szöveget a TTS „három vesző öt”-ként olvassa fel, a helyes magyar felolvasás „három egész öt tized” lenne.

Vizsgáld meg a text_normalizer.py meglévő normalizálási szabályait és a normalize_text() függvény struktúráját. Írj egy új szabályt, amely a tizedes törteket (pl. „3,5”, „12,75”, „0,1”) a magyar nyelv szabályai szerint normalizálja felolvasásra:

- „3,5” → „három egész öt tized”
- „12,75” → „tizenkettő egész hetvenöt század”
- „0,1” → „nulla egész egy tized”

A szabály illeszkedjen a meglévő normalize_text() függvény regex-alapú feldolgozási mintájába. Ne módosítsd a meglévő szabályokat, csak adj hozzá az újat a megfelelő helyre. Ügyelj arra, hogy a szabály ne ütközzön a már meglévő hőmérséklet-normalizálással (pl. „25,5°C”).

Claude Code kimenet:

Megvizsgálom a text_normalizer.py struktúráját.

Read [text_normalizer.py](#)

Read [text_normalizer.py](#)

Read [text_normalizer.py](#) (lines 121-240)

Read [text_normalizer.py](#) (lines 241-360)

Read [text_normalizer.py](#) (lines 361-560)

Read [text_normalizer.py](#) (lines 561-810)

A normalize_decimal_numbers() függvény már létezik és be van kötve a pipeline-ba. Íme a teljes lánc:

normalize_text_for_tts() (sor 1247) hívja a normalize_numbers() függvényt, amely a következő sorrendben dolgozik (text_normalizer.py:1092-1154):

#	Lépés	Függvény	Sor
2	Hőmérséklet	normalize_temperature()	1121
3	Tizedesvesző	normalize_decimal_numbers()	1124
4	Százalékok	normalize_percentages()	1127
...
11	Maradék számok	normalize_plain_numbers()	1150

Az ütközésmentesség is biztosított: a normalize_temperature() (2. lépés) előbb lefut és elfogyasztja a „25,5°C” típusú mintákat, így mire a normalize_decimal_numbers() (3. lépés) sorra kerül, azok már nem tartalmazzak tizedesveszős hőmérsékletet.

A meglévő logika (text_normalizer.py:475-529) pontosan azt csinálja, amit kértél:

```
„3,5” → „három és fél”  
„1,5” → „másfél”  
„2,5” → „két és fél”  
„3,14” → „három egész tizennégy század”  
„0,7” → „hét tized”
```

Nincs szükség módosításra - a funkció teljes egészében implementálva van és a megfelelő helyen fut a normalizálási pipeline-ban.

8.7. Forráskódok

Jelen fejezetben található néhány olyan kódrészlet az alkalmazás bizonyos részeiből (pl. kliens oldali felületek, megoldások, illetve szerveroldali részek), amelyek fontosabb szerepet töltenek be az alkalmazás során.

A teljes forráskód privát GitHub repository-ban érhető el: <https://github.com/varadiv/newscast>

8.8. Mintakimenetek

A jelen alfejezet a NewsCast rendszer éles működése során előállított mintakimeneteket tartalmazza. A fájlok a rendszer 2026. április 17-i futtatásából származnak.

Hangkimenet (TTS): A *Hirek-LikeFM-20260417.mp3* fájl egy teljes, a newscast-tts modul által generált hírblokk szintetizált hanganyagát tartalmazza (vö. 3.6.5 alfejezet). A hangfájl az ElevenLabs API eleven_v3 modelljével készült, a 3.6.5.4 alfejezetben ismertetett 50 szövegnormalizálási szabály alkalmazásával. A fájl a <https://miau.my-x.hu/miau/330/vv/> címen külön médiafájlként elérhető.

OAM elemzési riport (HTML): Az *oam_report_d9c8a72d-20260417211214.html* fájl a COCO modell (Pitlik, 2004) által generált objektum–attribútum mátrix elemzés teljes HTML kimenetét tartalmazza. A riport a miau.my-x.hu API engine3.php végpontjának válaszából származik (vö. 3.6.2.3 alfejezet) és a hírforrások összehasonlító objektivitásvizsgálatát mutatja be. A fájl a <https://miau.my-x.hu/miau/330/vv/> címen elérhető.

OAM elemzési adatok (XLSX): Az *oam_report_d9c8a72d-20260417211214.xlsx* fájl ugyanezen OAM elemzés nyers adatait tartalmazza táblázatos formában: az objektum–attribútum mátrix numerikus értékeit, a normalizált pontszámokat és a rangsorolási eredményeket (Pitlik, 2004). A fájl a <https://miau.my-x.hu/miau/330/vv/> címen elérhető.