Kodolányi János Egyetem Informatika Tanszék

Eszközmonitorozó mobilalkalmazás és szerver-oldali rendszer fejlesztése

Készítette: Kovács János (Üzemmérnök-Informatikus), **Konzulens:** Dr. Pitlik László

Helyszín: 1117 Budapest, Prielle Kornélia u. 47-49.

Budapest, 2026. Január 21.

Problémafelvetés és célkitűzések

Probléma

A szülői képernyőidő monitorozás kihívásai.

A kereskedelmi megoldások (pl. Qustodio) költségesek, zártak és adatvédelmi aggályokat vethetnek fel (harmadik fél kezeli az adatokat).

Saját motiváció

Saját igény egy "self hosted", megbízható rendszerre.

Fő célkitűzés

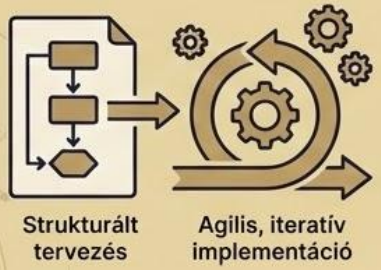
Egy költséghatékony, nyílt forráskódú, otthon üzemeltethető rendszer fejlesztése, amely tiszteletben tartja az adatvédelmet (pl. az adatok a családon belül maradnak).

A rendszer ingyenes, valós idejű push értesítésekkel támogassa a szülőket.

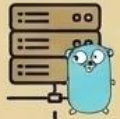
Alkalmazott módszertan és technológia

Fejlesztési metodológia

Hibrid modell, strukturált tervezés + agilis, iteratív implementáció.



Technológiai Stack



Backend: Go a statikus bináris miatt.

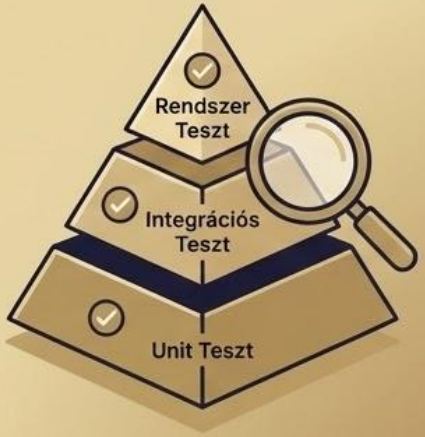
Mobilalkalmazás: Flutter (Dart) – a gyors UI fejlesztés és a jövőbeli bővíthetőség lehetősége miatt.

Adatbázis: SQLite3 – a "self-hosted" cél, egyszerű telepítés, 0-konfiguráció érdekében.

Értesítések: Firebase Cloud Messaging.

Validálás

Többszintű tesztelés (Unit, Integrációs, Rendszer).



Rendszertervezés és architektúra

Fő komponensek

Felügyelt Eszközök (Gyermek)

Hálózati Infrastruktúra (Router)

Flutter Mobilalkalmazás (Szülő)

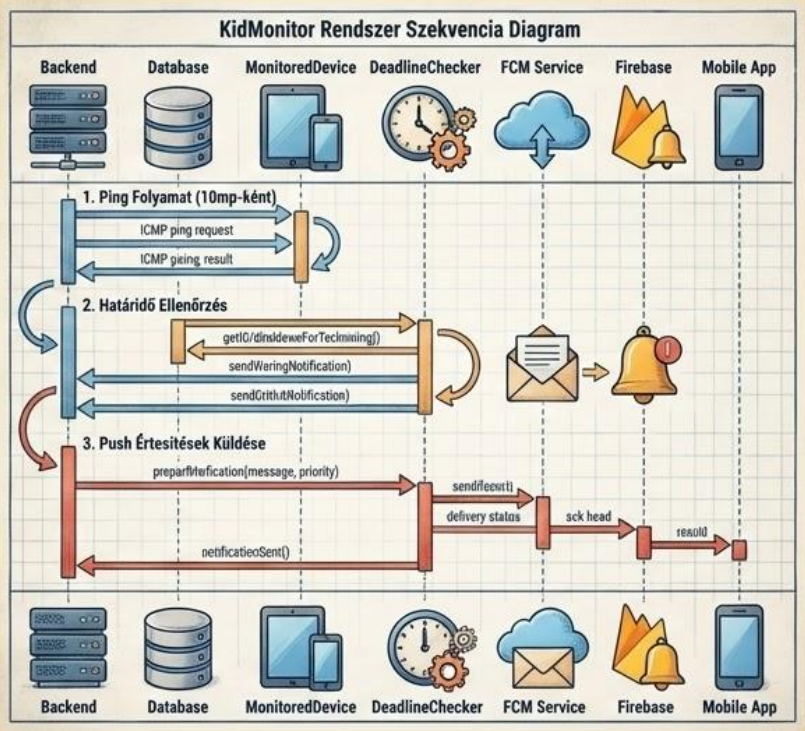
Go Backend (Linux)

Rendszertervezés és architektúra

Működési elv (Szekvencia)

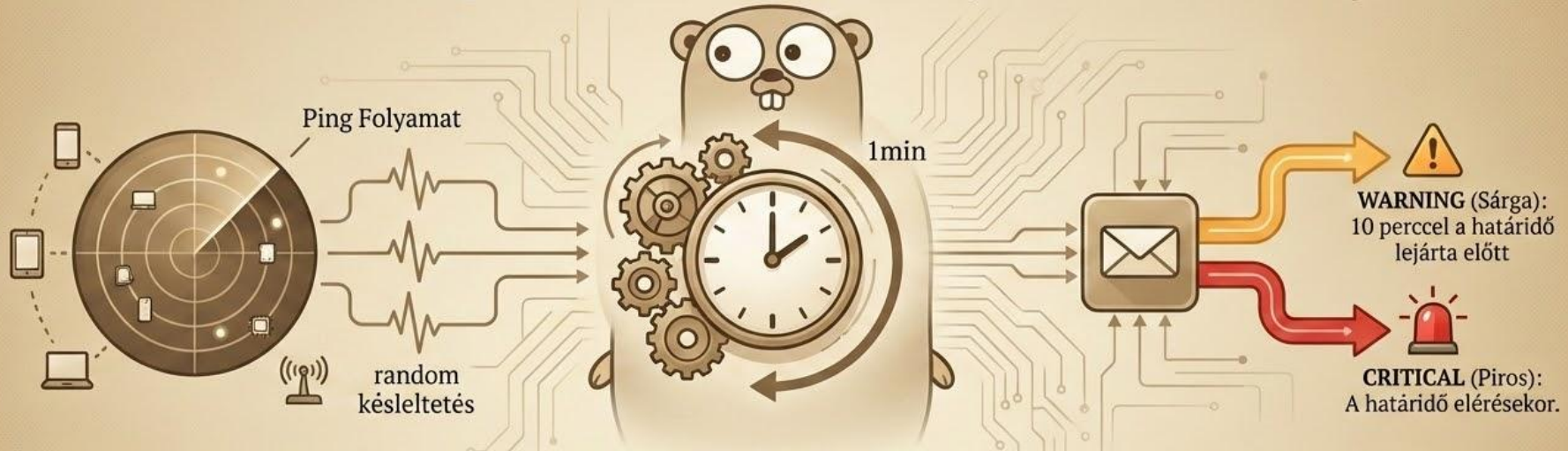
A Go Backend 10 másodpercenként pingeli a helyi hálózaton lévő regisztrált eszközöket.

Ha egy eszköz aktív a tiltott időszakban, a DeadlineChecker FCM-en keresztül push üzenetet küld a Szülő Flutter alkalmazásának.



A Backend implementáció, nyelv: GO

KODOLÁNYI UNIVERSITY



Ping Monitor

A rendszer "szíve". Párhuzamosan (goroutine-ok) ellenőrzi az összes eszközt, a hálózati terhelés elosztására random késleltetéssel.

Deadline Checker

Percenként futó modul, ez felel a szabályok kiértékeléséért.

FCM Modul

Kétlépcsős riasztási logikát kezel:

Mobilalkalmazás

Cél

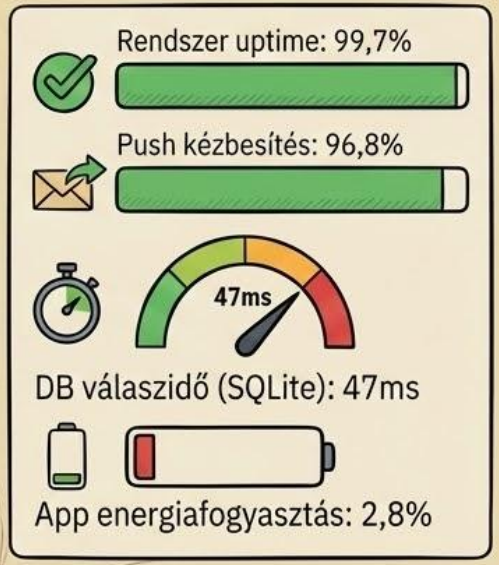
Minimalista UI, fókusz a funkcionalitáson (értesítések fogadása).

Android Notification Channels

A riasztások priorizálása (pl. a “Critical” felülbírálja a “Néma” módot) külön csatornákon történik.

Eredmények, vita és következtetések

Eredmények



Önkritika

Nem skálázható (SQLite)

Csak Androidot támogat

Terheléses tesztek hiányoznak

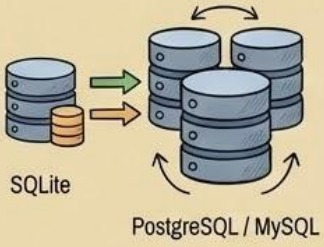
Következtetés

A projekt egy "self-hosted" Proof of Concept volt, nem egy SaaS-termék. Az SQLite tudatos és megfelelő döntés volt. A koncepció sikeresen validálva.

Jövőkép és fejlesztési lehetőségek

A jelenlegi rendszer egy sikeres prototípus. A következő lépések az azonosított korlátok megszüntetésére fókuszálnak:

1. Architektúra-váltás (Skálázhatóság):



Az SQLite adatbázis cseréje egy robusztusabb, párhuzamos írásra képes rendszerre (pl. PostgreSQL vagy MySQL).

2. iOS Verzió Kiadása (Platformfüggetlenség):

A Flutter keretrendszer valódi erejének kihasználása. Az alkalmazás lefordítása és publikálása iOS-re.

3. Funkcionális Bővítés (Aktív Menedzsment):

Elmozdulás a passzív monitorozástól az aktív beavatkozás felé. Funkciók: Alkalmazás-szintű blokkolás, alkalmazás-specifikus időkvóták.

4. Szülői Dashboard Fejlesztése (Adatvizualizáció):

Egy részletes statisztikai felület (webes vagy mobilos) fejlesztése. Funkciók: Grafikonok a képernyőidő-trendekről, szabályok vizuális kezelése.

Kérdések

Köszönöm a figyelmet!