Mobil&Fullstack Témalabor — Picture Team

Fotó Portál

Záró Dokumentáció

Résztvevő Neve	Neptun Kód	E-mail Cím	Technológia
Kálvin Tamás	PMZ90Y	kalvintamasendre@gmail.com	Android (Kotlin)
Ócsai Dávid			Web (React)
Tamás Ferenc	WEC6UB	tamasf97@outlook.com	Back-end (Rust)

Tartalomjegyzék

1 Eredeti Specifikáció

1.1 Feladat

Feladatunk egy fotó portál készítése volt, melyben felhasználók fotókat tölthettek fel kategóriák alapján, nézhettek meg és értékelhettek. A portál áll egy back-end alkalmazásból, egy Android kliensből és egy webes front-end kliensből.

1.2 Aktorok

1.2.1 Admin

A kategóriák kezeléséért felelős.

1.2.2 Felhasználó

A portál funkcióit használja az Android kliensen keresztül.

1.3 Funkciók

1.3.1 Web Kliens

- **Regisztráció:** Regisztráció e-maillel és jelszóval.
- **Belépés:** Adminok számára belépés e-maillel és jelszóval.
- **Kategóriák kezelése:** Kategóriák létrehozása, módosítása, törlése.

1.3.2 Android Alkalmazás

- **Belépés:** Felhasználók számára belépés e-maillel és jelszóval.
- **Képek feltöltése:** Képek feltöltése metaadatokkal:
  - Kép neve
  - Kép leírása
  - Kép kategóriái
- **Képek Listázása:** Opcionálisan kategóriák alapján szűrve.
- **Képek Keresése:** Képek keresése címben és leírásban adott szöveg alapján.
- **Képek Értékelése:** Képek értékelése (1-5).

1.3.3 Back-end

REST-szerű API nyújtása az web kliensnek és az Android alkalmazásnak.

2 Megvalósítás

A projekt egésze megtalálható GitHubon itt.

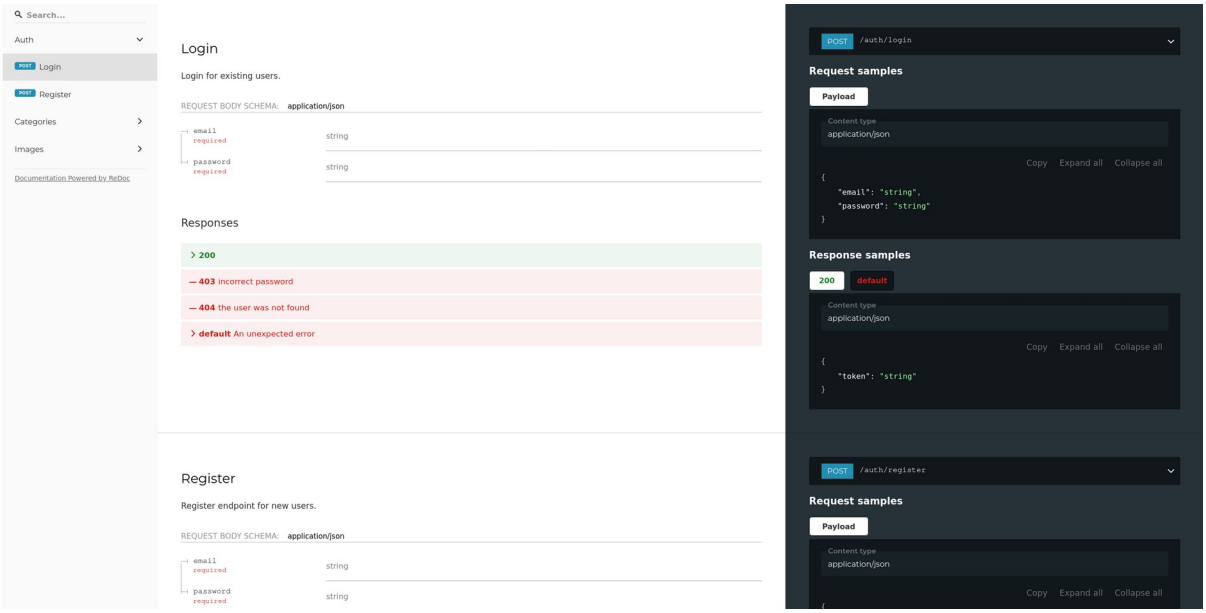
2.1 Back-end (Rust)

A serverhez Rust nyelvet használtunk, azon belül az Actix Web könyvtárat a HTTP REST API-ra.

Az adatok nagy részét PostgreSQL adatbázisban tároltuk. Kivéve magukat a felhasználók által feltöltött képeket, amiket a helyi fájlrendszerben mentettünk el.

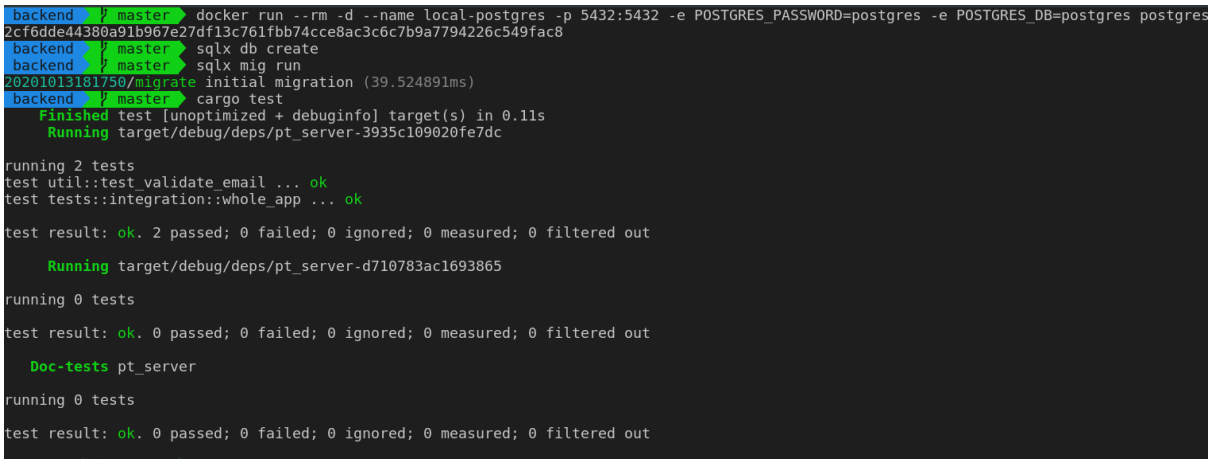
Az authentikációhoz pedig JSON Web Tokeneket használtunk, amit a kliensek az *Authorization* HTTP feljécben küldtek.

Az API dokumentációjának generálásához egy saját könyvtárat (aide) használtunk, és ReDoc felületet a megjelenítéshez.



API Dokumentáció

A teszteléshez a beépített Rust lehetőségeket használtam, egy rövid zöld-utas integrációs tesztre.



Tesztelés folyamata

A kész applikáció manuálisan egy Kubernetes clusterre volt deployolva, CI/CD nem volt.

Forráskód (GitHub)

A Forráskód elérhető itt.

2.2 Webes Kliens (React)

A deployment ugyanarra a Kubernetes clusterre történt, ahol a back-end is volt, itt viszont már GitHub Actions CI/CD segítségével.

Forráskód (GitHub)

A Forráskód elérhető itt.

2.3 Android Alkalmazás (Kotlin)

TODO

Forráskód (GitHub)

A Forráskód elérhető itt.

2.4 Sprint Eltérítés

2.1 Új Funkcionalitás – Top Lista

Az Android alkalmazásban implementálni kellett egy top lista nézetet a felhasználókról, ahol rangsorolva voltak a képeikre kapott értékelések alapján.

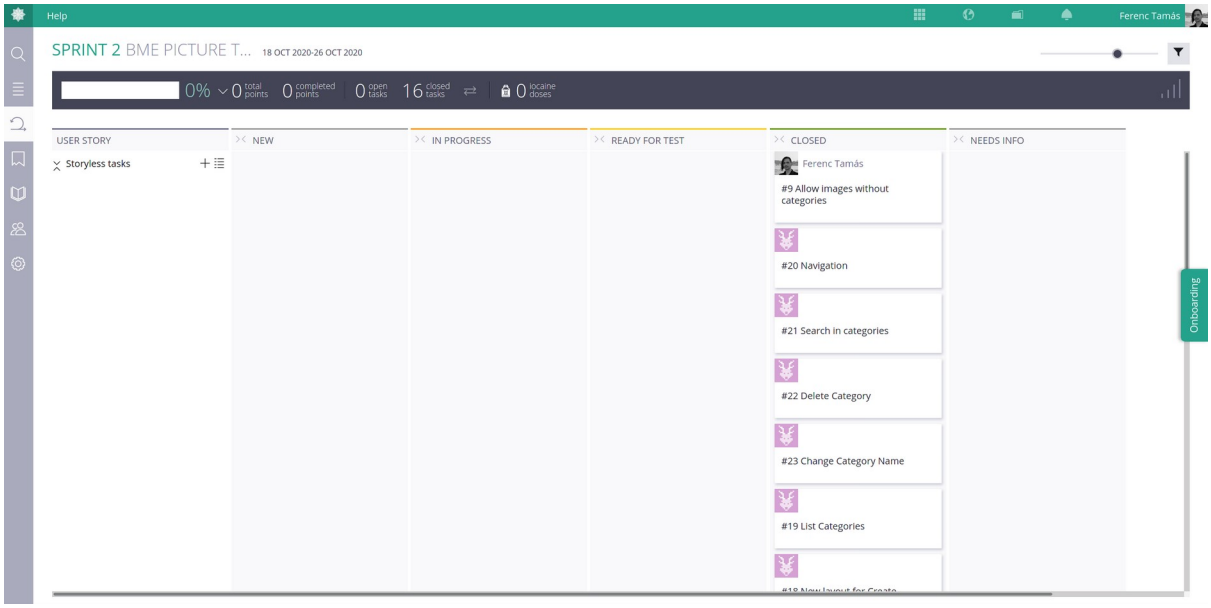
Ehhez változtatni kellett az Android alkalmazáson is és a Back-end oldalon is szükség volt ennek a kezelésére.

A változtatás nem volt nehéz a kliens és a server oldalon sem, egy új endpointra volt szükség a server oldalon, és egy queryre az adatbázisból (tesztet ehhez nem írtunk). Az android alkalmazásban meg csupán egy új nézetet kellett létrehozni, és meg kellett hívni a servert.

2.5 Együttműködési Platform

2.5.1 Taiga.io

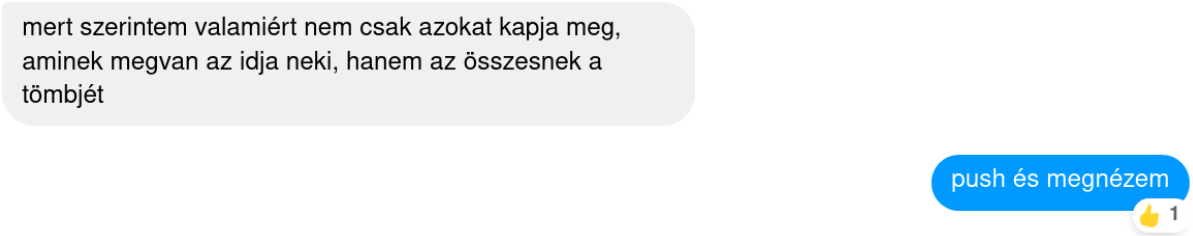
Kipróbáltuk a Taiga nevű Agile managementnek hirdetet platformot, de a végén nem nagyon használtuk végül. A választás azért esett erre, mert pár éve én (*Tamás Ferenc*) nézegettem self-hosted project management platformokat, és ez az egyiket volt azoknak. Viszont akkor egy túlkomplikált, lassú és (szubjektíven) ronda felületet biztosító alkalmazás volt. Arra voltam kíváncsi, hogy mennyit változott, de úgy tűnik, hogy nem sokat.



Taiga Sprint

2.5.2 Messenger

Egy ekkora projekthez bőven elég volt a Facebook által nyújtott Messenger, ez mindenkinek volt, és mindenki többnyire elérhető volt rajta.



Messenger Chat Példa

2.6 Nem Megvalósított Funkciók

Az Android alkalmazásban nem lehetséges kategóriánként szűrni a képeket, szimplán azért mert elfelejtettük, hogy ez a funkció létezik a specifikációban.

3 Végső értékelés – Megjegyzések

3.1 Kálvin Tamás

3.2 Ócsai Dávid

3.3 Tamás Ferenc

Mindent összevetve pozitív élmény volt számomra, ha ez a projekt nincs, valószínűleg sosem írom meg ezt a könyvtárat, amit már régebben elkezdtem. Ráadásul mindig is nézegettem a Rust Actix és SQLx könyvtárakat, de a hello worldöt és néhány kisebb saját projektet leszámítva nem írtam velük még semmi párszáz sornál nagyobb alkalmazást.

Most utólag nézve kicsit máshogy struktúrálnám a forráskódot, valamiért minden projekt elején jó ötletnek tűnik típusonként tagolni a dolgokat (models, services, stb..), de mindig felesleges fájlok közötti ugrálás és keresgélés lesz a vége. Valamint az architektúrában is servicek helyett aktor modellt használnék.

A Taiga-ra is kíváncsi voltam, amit korábban említettünk, sajnos ismét csalódtam benne.

3.4 Összesítés

Szerintünk mindenki ugyanannyi időt áldozott a projekre, és nagyjából együtt dolgoztunk mindvégig, úgyhogy fejenként mindenki a feladtok *33.3333333334%*-át kapta meg és teljesítette.