

Work Breakdown Structure (Részletes feladatbontás)

A projekt 5 hetes ütemezésben, sprintszerűen halad. Az alábbi táblázat a fő fejlesztési feladatokat, hozzájuk tartozó User Story-kat, erőforrásokat, időbecslést, függőségeket és rövid leírásokat tartalmazza.

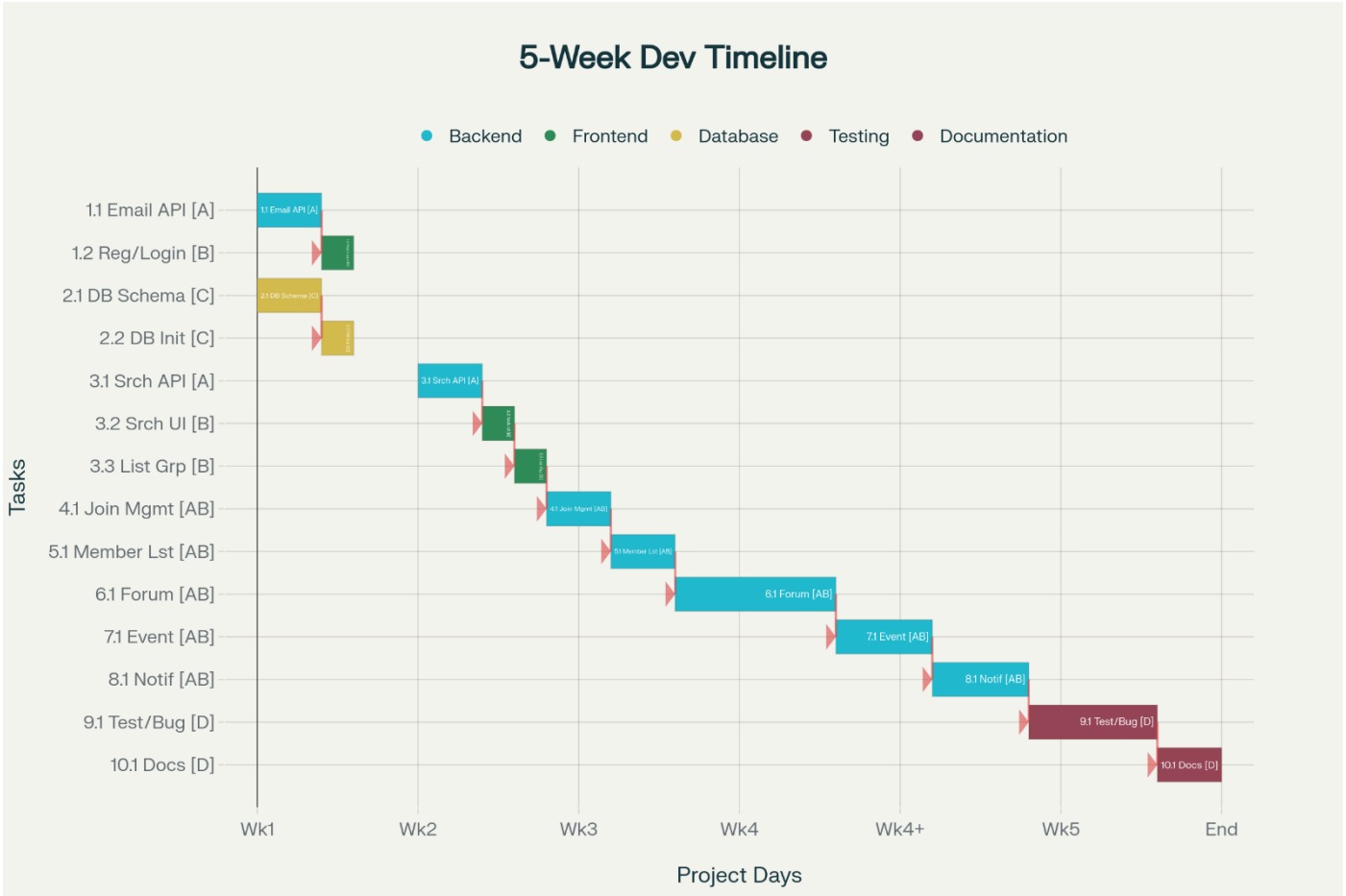
<u>Feladat</u> <u>megnevezése</u>	<u>Kapcsolódó</u> <u>Story</u>	<u>Felelős</u>	<u>Becsült</u> <u>idő</u> <u>(nap)</u>	<u>Hét</u>	<u>Függőségek</u>	<u>Rövid</u> <u>leírás</u>
Regisztráció és bejelentkezés ELTE-s emaillel	US1, US8		3	1		Felhasználók regisztrációja és hitelesítése ELTE email alapon
Backend: ELTE email validáció		Timi, Dávid, Tibi	2	1		ELTE emailcím ellenőrzés és validáció
Frontend: Regisztráció/ bejelentkezés UI		Dóri, Niki	1	1	Backend kész	Bejelentkezési és regisztrációs UI felület
Alap adatszerkezetek és adatbázis sémák készítése	US1, US6		3	1		Projekt adatbázis alapstruktúrájának tervezése
DB séma tervezése (Users, Groups, Posts, Events)		Timi, Dávid, Tibi	2	1		Users, Groups, Posts, Events, Comments táblák definiálása
DB inicializálás és migrációs szkriptek		Timi, Dávid, Tibi	1	1	DB séma kész	Adatbázis inicializálás, indexek, relációk
Tárgy alapú tanulócsoporthoz keresés magvalósítása	US1		4	2	DB kész	Tantárgyak szerinti csoportok keresése és megtalálása
Backend: Csoportkeresés API (tantárgy szerinti szűrés)		Timi, Dávid, Tibi	2	2		Keresés API: tantárgy, szint, vagy csoport név alapján
Frontend: Csoportkeresés UI komponensek		Dóri, Niki	1	2	Backend kész	Csoport megjelenítés és szűrési felület
Csoportok listázása és megjelenítése		Dóri, Niki	1	2	Frontend kész	Keresési eredmények listázása

Csoporthoz való csatlakozás funkció	US8		2	2	Tárgy alapú keresés kész	Felhasználók csatlakozhatnak a meglévő csoportokhoz
Backend és Frontend: Csatlakozás és tagsági kezelés		Timi, Dávid, Tibi, Dóri, Niki	2	2		Csatlakozási logika, tagság kezelése
Csoporttagok listázása	US6		2	2-3	Csatlakozás funkció kész	Csoport tagjainak listázása és profiljai
Backend és Frontend: Taglista és tagprofil megjelenítése		Timi, Dávid, Tibi, Dóri, Niki	2	2-3		Taglista UI megjelenítés, alapvető profil info
Aszinkron fórum és üzenetek implementációja	US3, US6		5	3-4	Taglista kész	Aszinkron kommunikáció : posztok és kommentek
Backend: Posztok, kommentek API		Timi, Dávid, Tibi	5	3-4		Fórum API
Frontend: Fórum UI		Dóri, Niki			Backend kész	Fórum UI
Esemény-naptár fejlesztés	US6		3	4		Közös események kezelése és naptári megjelenítés
Backend: Esemény CRUD		Timi, Dávid, Tibi	3	4	Fórum kész	Esemény létrehozás, szerkesztés, naptár view
Frontend: naptár megjelenítés		Dóri, Niki				Naptár megjelenések az integrálása
Felhasználói felület, értesítések beállítása	US5		3	4	Core funkciók kész	Zavaró értesítésektől mentes UI, beállítások
Backend és Frontend: Értesítési beállítások és megjelenítés		Timi, Dávid, Tibi, Dóri, Niki	3	4		Értesítési preferenciák, jelzések kezelése
Szükséges adminisztráció és tesztelés			4	4-5	Összes fő modul	Teljes rendszer tesztelése, integrációs vizsgálatok
Integrációs tesztek, bug fixek, teljes körű tesztelés		Timi, Dávid, Tibi,	4	4-5		Unit, integrációs tesztek futtatása
Dokumentáció, telepítési útmutató		Mindenki	2	5	Összes fő modul	Teljes dokumentáció , deployment útmutató
API dokumentáció,		Mindenki	2	5	Összes fő modul	API docs, fejlesztői és

fejlesztői útmutató						telepítési útmutatók
---------------------	--	--	--	--	--	----------------------

Időterv

Az 5 hetes projekt ütemtervét az alábbi diagram mutatja, ahol az egyes feladatok hetenkénti eloszlása, a fejlesztők munkaterhelése, valamint a függőségek egyértelműen láthatók.



Jelölések a diagramban:

- Kék sáv: Backend fejlesztési feladatok
- Zöld sáv: Frontend fejlesztési feladatok
- Narancssárga sáv: Adatbázis feladatok
- Lila sáv: Tesztelés és dokumentáció
- Nyilak: Feladatok közötti függőségek

Kritikus útvonal (Critical Path):

1. hét: Infrastruktúra (DB séma + Auth API)
2. hét: Csoportkezelés alapja (keresés, csatlakozás, taglista)
- 3-4. hét: Fórum megvalósítása (függ a taglista készülségétől)
4. hét: Naptár és értesítések (párhuzamosan)
- 4-5. hét: Tesztelés és dokumentáció

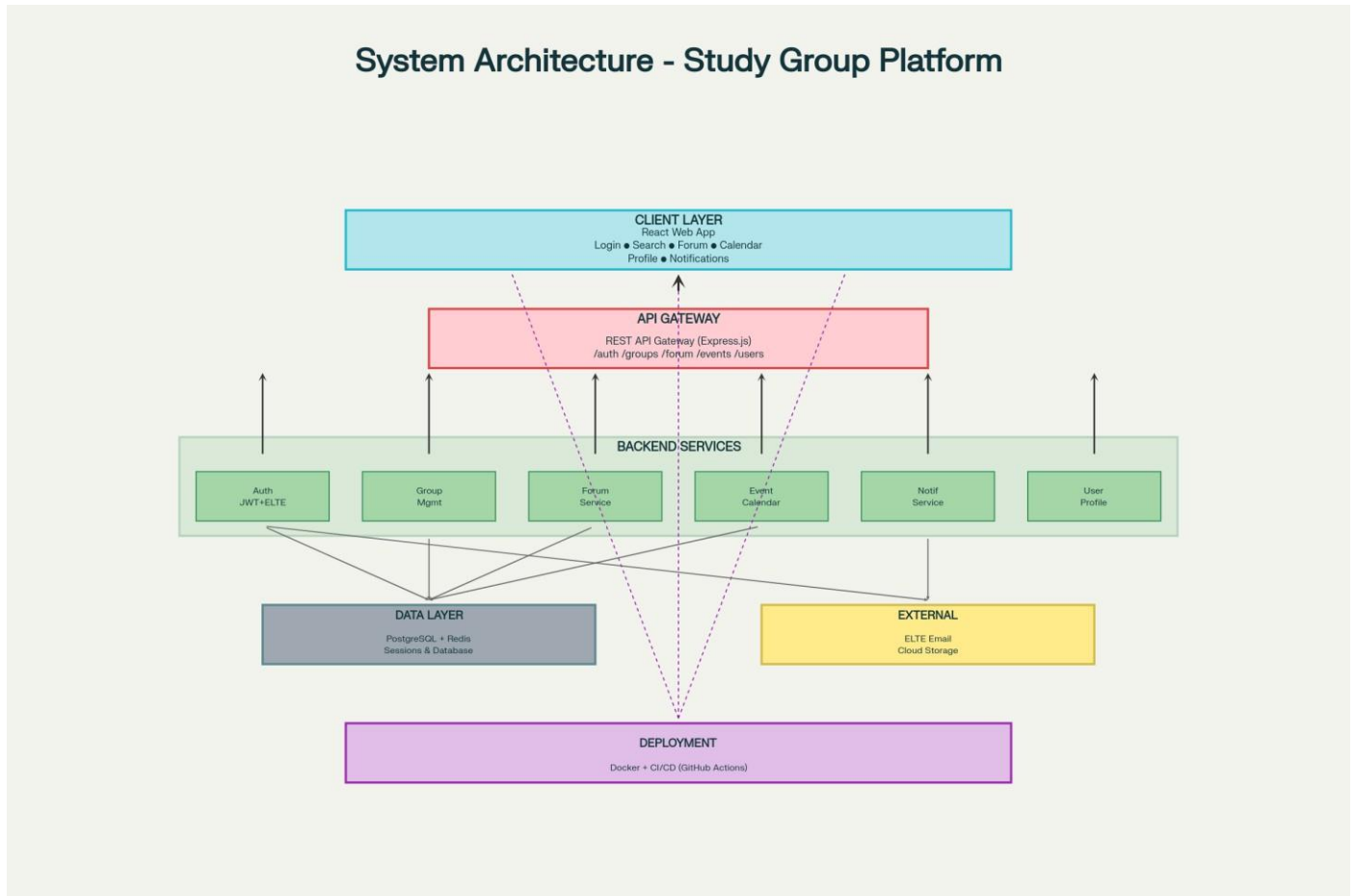
Párhuzamos fejlesztések:

- A különböző fejlesztők egymástól független feladatokon dolgoznak az egyes hetekben, ezzel maximalizálva a csapat produktivitását
- Fejlesztő 1: Autentikáció → Csatlakozás funkció
- Fejlesztő 2: Adatbázis → Taglista
- Fejlesztő 3: Keresés → Fórum
- Fejlesztő 4: Naptár
- Fejlesztő 5: Értesítések

Architektúrális terv - Tanulócsoport kereső platform

Rendszer architektúra - Rétegzett felépítés

A platform egy klasszikus 3-rétegű (3-tier) architektúrát követ, ahol az üzleti logika, prezentáció és adatkezelés elválasztva működik egymástól:



Rendszer architektúra - Tanulócsoport platform rétegzett felépítése

Frontend réteg (Prezentációs réteg)

Technológia: React 18.x + JavaScript, Redux Toolkit, React Router v6

Fő komponensek:

- Login/Register módok
- Csoportkeresés és szűrés
- Csoport adatlap és tagok nézete
- Fórum oldal (posztok, kommentek)
- Naptár és eseménynézet
- Felhasználói profil
- Értesítési beállítások

Deployment: Vercel, Netlify vagy egyéb statikus webhely hosztingszolgáltatás

API Gateway réteg

Technológia: Python (Flask), Port: 5000 (fejlesztésben)

Fő API végpontok:

- POST /auth/register - Regisztráció
- POST /auth/login - Bejelentkezés
- GET /groups - Csoportok listázása
- GET /groups/search?q=... - Csoportok keresése
- POST /groups/:id/members - Csatlakozás csoporthoz
- GET /groups/:id/members - Csoport tagjainak listája
- POST /posts - Új poszt létrehozása
- POST /posts/:id/comments - Komment hozzáadása
- POST /events - Esemény létrehozása
- GET /notifications - Felhasználó értesítései

Backend szolgáltatások réteg

Authentication Service: JWT token alapú autentikáció, ELTE email validáció (@elte.hu domain check).

Group Management Service: Csoportok CRUD műveleti, tantárgy alapú keresés és szűrés, tagok kezelése

Forum Service: Posztok és kommentek kezelése, aszinkron kommunikáció, threaded discussions

Event/Calendar Service: Közös események kezelése, naptári megjelenítés, emlékeztetők

Notification Service: Értesítési típusok (új poszt, komment, esemény), beállítható preferenciák, csend mód

User Profile Service: Felhasználói profil szerkesztése, profilkép feltöltés, privacy beállítások

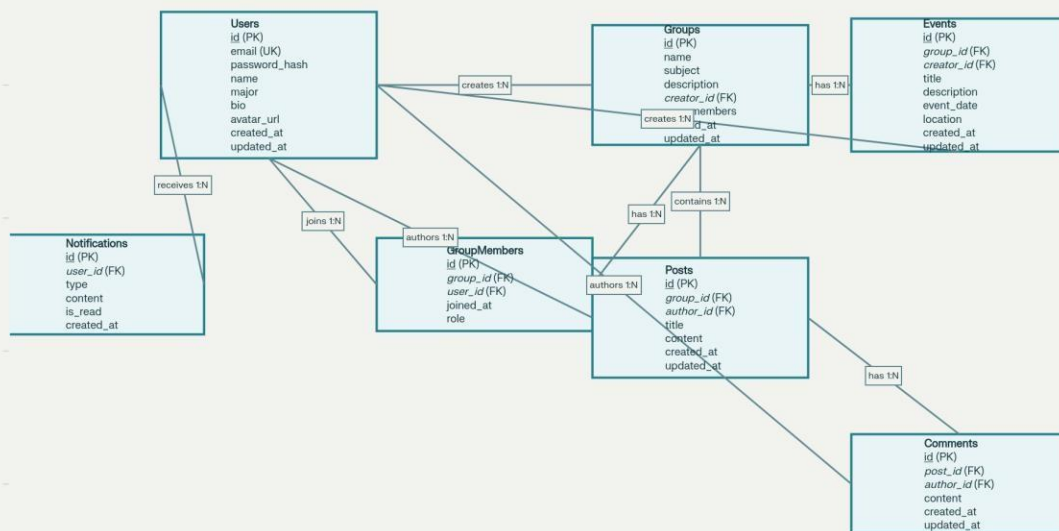
Adatréteg (Data Layer)

PostgreSQL relációs adatbázis

- Verzió: MySQL 8.0+
- Táblák: Users, Groups, GroupMembers, Posts, Comments, Events, Notifications

Adatbázis séma (Entity-Relationship)

Study Group DB ER Diagram



Adatbázis séma - Entity-Relationship diagram

Telepítési topológia

Fejlesztői környezet

Docker Compose-sal egyszerűen elindítható:

- Frontend (React) → localhost:3000
- Backend API (Flask) → localhost:5000
- MySQL → localhost:5432

Üzemi (Production) környezet

Komponens	Megoldás
Frontend	Vercel vagy Netlify
Backend	Railway, Render vagy AWS EC2
Adatbázis	AWS RDS MySQL vagy Supabase
Storage	AWS S3 vagy Azure Blob Storage
CDN	CloudFlare vagy AWS CloudFront
CI/CD	GitHub Actions

Biztonsági intézkedések

- JWT token alapú autentikáció (access token: 15 perc, refresh token: 7 nap)
- HTTPS/TLS titkosítás minden kommunikációban
- SQL injection védelem (Prepared Statements, ORM)
- CSRF token az URL-módosító kérésekhez
- GDPR compliance: Adattörlés, adattovábbítás támogatása
- Rate limiting: 100 request/10 percenként API-nként
- DDoS védelem: CloudFlare

Monitorozás és Naplózás

- Sentry: Valós idejű hibakövetés
- Winston/Bunyan: Strukturált backend naplózás
- Lighthouse CI: Frontend performance monitoring
- New Relic/DataDog: Backend performance monitoring

Skálázhatóság

- Horizontális: Több backend instance Kubernetes alatt
- Vertikális: Adatbázis erőforrások növelése igény szerint
- Load Balancer: Nginx vagy AWS ALB
- Automatikus biztonsági mentés: Napi, 30 napig őrzött

Technológiai stack összefoglalása

Réteg	Technológia
Frontend	React 18.x, Redux, Material-UI, React Router
Backend	Python 3.x, Flask, JavaScript
Adatbázis	MySQL 8.0+
DevOps	Docker, Docker Compose
CI/CD	GitHub Actions
Tesztelés	Jest, Supertest, Cypress
Monitoring	Sentry, Winston
Kódminőség	ESLint, Prettier, SonarQube

A teljes, részletes architektúrális dokumentáció letölthető: