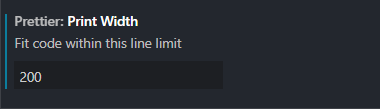
Codey fejlesztői dokumentáció

# Fejlesztői környezet

A fejlesztés Visual Studio Code-ban (1.66.2 verzió vagy újabb) ajánlott, de bármilyen IDE felhasználható, amely támogatja Prettier kód formázót. A fejlesztők közötti formázott és egységes kód érdekében a következő beállításokat kell megváltoztatni:



A képen szöveg látható

Automatikusan generált leírás

A fejlesztés megkezdése előtt a ’*.env.example’* fájlt át kell nevezni *’.env’*-re fájlok mezőit ki kell tölteni mint a backend és frontend könyvtárban.

Az alkalmazás a front- és backend JavaScript-ben íródott. A backend és API a skálázhatóság érdekében NodeJS, a frontend pedig EJS renderelést használ és JQuery segítségével kap funkcionalitást. Az alkalmazás REST architektúrát követ, tehát POST vagy GET kéréseket küldünk és JSON formátumú adatok közlekednek. Az adatbázis SQL és Sequelize ORM.

# Futtatás

Futtatáshoz szükséges szoftver verziók:

* node v16.14.0 vagy újabb
* npm 8.7.0 vagy újabb

A futtatáshoz két terminálra van szükség két útvonalon:

1. %*projekt mappa útvonala*%/Codey/App/\_API
2. %*projekt mappa útvonala*%/Codey/App/frontend

node modulok telepítése után a következő paranccsal indulnak a szerverek:

npm run dev

## Node modulok

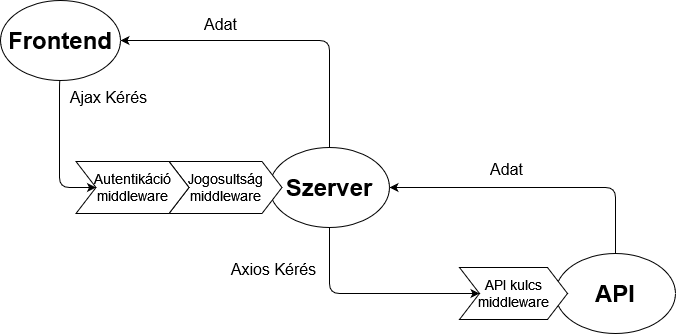
* [bcrypt](https://github.com/kelektiv/node.bcrypt.js#readme) – 5.0.1
* [dotenv](https://github.com/motdotla/dotenv#readme) – 16.0.0
* [express](http://expressjs.com/en/4x/api.html) – 4.17.2
* [express-session](https://github.com/expressjs/session#readme) – 1.17.2
* [mysql2](https://github.com/sidorares/node-mysql2#readme) – 2.3.3
* [protonmail-api](https://github.com/justinkalland/protonmail-api) – 2.2.0
* [sequelize](https://sequelize.org/) – 6.15.0
* [axios](https://axios-http.com/docs/intro) – 0.25.0
* [ejs](https://ejs.co/#docs) – 3.1.6

## Fejlesztői modulok

* [nodemon](https://github.com/remy/nodemon#nodemon) – 2.0.15

Az akadálymentes működéshez a megjelölt verzió vagy újabb használandó

# Felépítés



Frontendről érkezik egy kérés, ez átkerül egy autentikáció middleware-en, amely megvizsgálja, hogy a session, amiből a kérés érkezik be van-e jelentkezve, illetve Jogosultság middleware-en, amely megnézi a session jogosultságát. Ha ezek megfelelnek egy új http kérés kap egy API kulcsot az API irányába. Mielőtt a kérés meg érkezne az API-ba egy harmadik middleware megvizsgálja, hogy helyes-e ez az API kulcs (erre azért van szükség, hogy ne lehessen szabadon API endpoint-okat meghívni) és csak ezután történik adatbázis lekérdezés.

Jogosultságok  
-Adminisztrátor – Teljes hozzáférés  
-Moderátor – Adminisztrátori funkciók közül csak blogok engedélyezése  
-Felhasználó – Alap funkciók elérése

# Frontend

A felület megjelenítésért a szerver felel és ejs fájlok beszúrásával épül fel az alkalmazás oldalainak elrendezése. Az ejs fájlok HTML tartalmon kívül a JQuery scriptet is egyaránt tartalmazzák.

A képen szöveg látható

Automatikusan generált leírás

A fájlok helyét a navigációs menüben elfoglalt kategória határozza meg. A views könyvtárban a lapokat felépítő sablonfájlok találhatók.

A többször felhasználandó metódusokat a myTools.ejs tárolja. A metódusok rövid leírása kommentekben található.

# Elrendezés



Az oldal a 300px alatti felbontásokat nem támogatja



# Szerver

A szerver végzi az Endpointok meghívását és az ejs fájlok megjelenítését, valamint itt történik a session kezelés és a jogosultság kezelés. A fájlstruktúra MVC mintát követ. A szerver nem tartalmaz adatmodelleket mivel az adatbázis lekéréseket az API végzi.

A képen szöveg látható

Automatikusan generált leírás

A kontrollerek(controllers) felelősek a lapok betöltéséért és az API endpointok meghívásáért. A routing(routes) a navigációs menü pontjai, illetve a felhasználói és adminisztrátori fiókspecifikus funkciók (jelszó módosítás, blog engedélyezés stb.) alapján lett kialakítva.

A képen szöveg látható

Automatikusan generált leírás

A képen szöveg látható

Automatikusan generált leírás

# API

Az API nem végez jogosultság vagy session kezelést, kizárólag adatbázis műveleteket. Művelet esetén visszajelzést küld a szerver felé egy *„{success: false / true}”* objektum által, ezáltal a szerver tudja, hogy a kérés végbement vagy sem, illetve hiba üzeneteket vagy adatokat társít a kérés mellé szükségszerűen. A fájlstruktúra MVC mintát követ. A szerverrel ellentétben nem tartalmaz nézeteket, helyette az adatmodelleket tartalmazza.

A képen szöveg látható

Automatikusan generált leírás

A kontrollerek(controllers) és a routing(routes) funkciónként vannak csoportosítva, ha egy funkciónak egy bizonyos jogosultsághoz specifikus akkor külön kontrollerhez kerül.

A képen szöveg látható

Automatikusan generált leírás

A képen szöveg látható

Automatikusan generált leírás

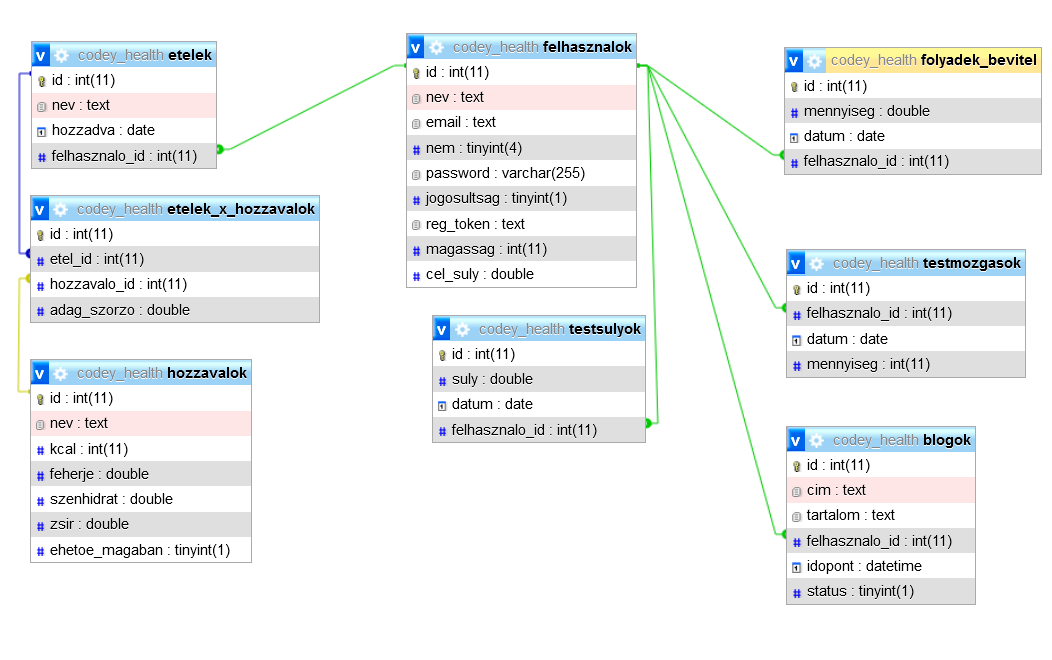
Az adatmodellek adatbázis táblánként szervezendők, a Sequelize működéséhez elengedhetetlenek.  
Sequelize dokumentáció: <https://sequelize.org/docs/v6/>

A képen szöveg látható

Automatikusan generált leírás

# Adatbázis

Az adatok SQL adatbázisban tárolódnak, Sequelize ORM felhasználásával a komplikált lekérések elkerülése érdekében.



# Tesztelés

## Frontend funkcionális tesztek:

*„Codey/Dokumentumok/Frontend\_tesztek.docx”* fájlban találhatók

## Backend egységtesztek

Az API tesztelése automatizált. Lehetőség szerint egy GitBash terminálra és a következő parancsok futtatására lesz szükség

1. cd %*projekt mappa útvonala*%/Codey/Dokumentumok/jest
2. npm i -g jest
3. npm i
4. jest

jest dokumentáció: <https://jestjs.io/docs/getting-started>