Jegyzőkönyv

Webtech 2

Komponens Nyilvántartó

Készítette: **Regecz Márk**

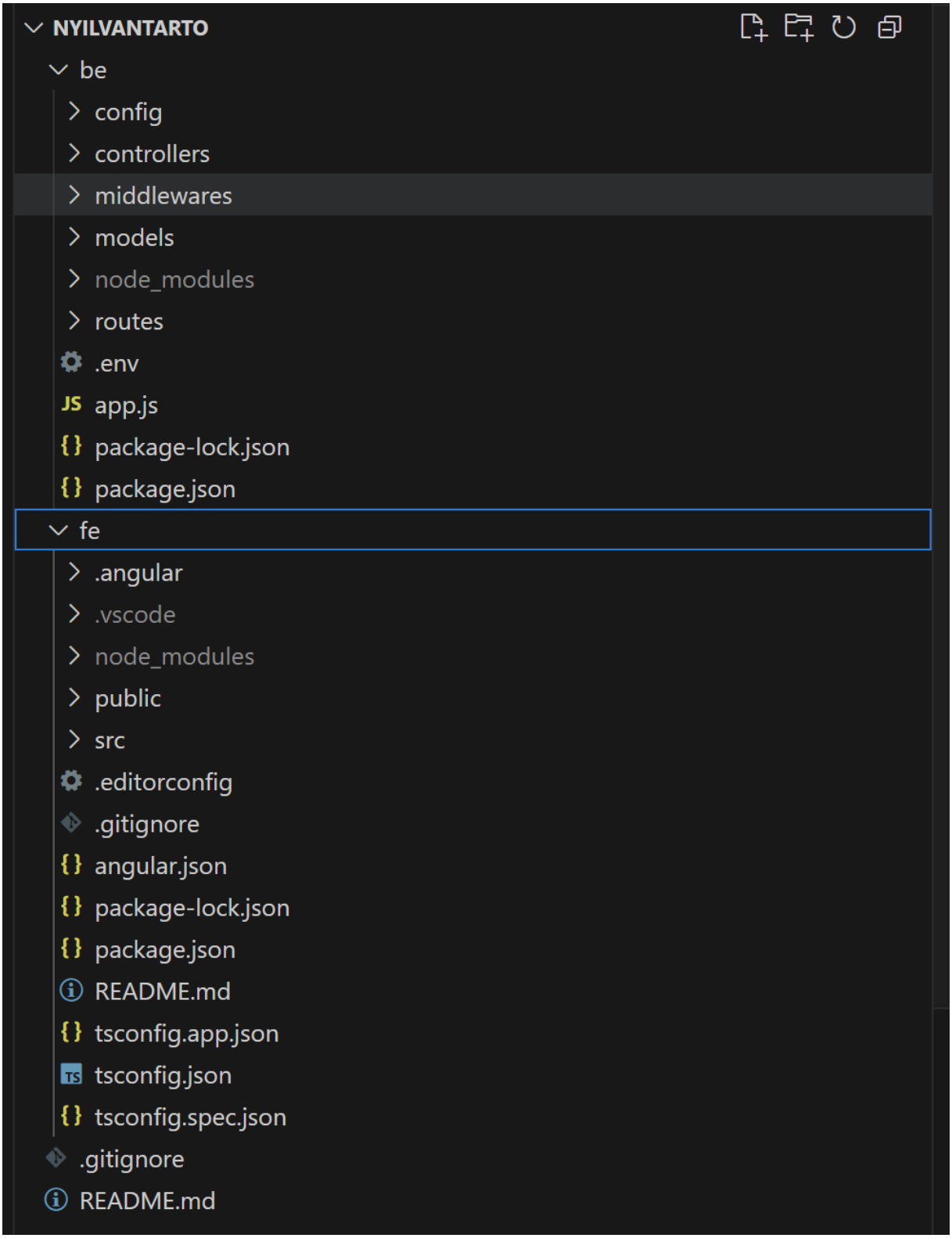
Neptun: **CNGDZ3**

Tartalom

Bevezetés

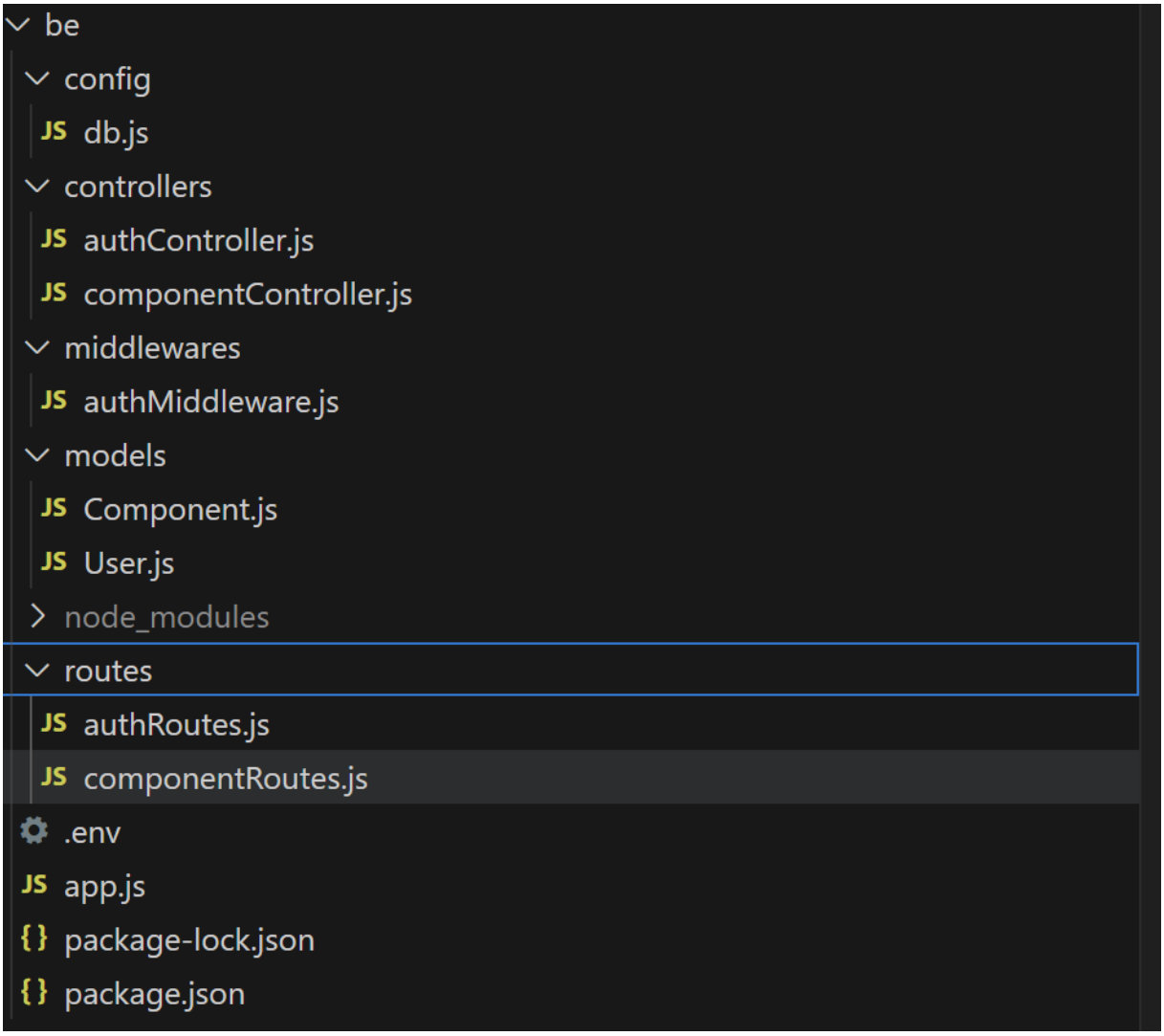
A szoftverem célja számítógép komponensekről adatokat eltárolni. Munkahelyemen amennyiben egy számítógépben alkatrészt cserélünk, azt az adott területre költségeljük el. Az alkatrészeket, amiket rendelünk, annak az élet útját hivatott lekövetni, hogy nyomon tudjuk követni, hogy egy adott komponens milyen típusú, ki rendelte, hova stb. Kereshetők az alkatrészek. A szoftver képes külön kezelni felhasználókat. Fejlesztésem során Node.js-t használtam Backend rész fejlesztéséhez, illetve Angular-t a Frontend részére.

Mappa és fájlok stuktúrája



A „be” mappában található a Backend programrész, amelyben Node.js alkalmazás található, Express.js-el. Az „fe” mappában található a Frontend része a programnak, amely Angular projektet tartalmaz.

Backend



A fájlok elnevezéséből egyértelműen következnek, hogy melyik fájl, melyik részt valósítja meg a programban.

db.js

Mongoose segítségével csatlakozok az adatbázishoz.

authController.js

A modul gondoskodik a biztonságos felhasználói hitelesítésről és session kezelésről, amely alapvető az alkalmazás védett erőforrásainak eléréséhez.

- Regisztráció: Új felhasználók létrehozása jelszóhash-eléssel
- Bejelentkezés: Felhasználók hitelesítése és session létrehozása
- Kijelentkezés: Session törlése, cookie tisztítása
- Felhasználói adatok lekérése: Session alapján azonosított felhasználó adatainak visszaadása
- Biztonsági mechanizmusok: Jelszavak biztonságos tárolása, session kezelés
- Hibakezelés: Megfelelő hibaüzenetek és HTTP státuszkódok biztosítása

componentController.js

IT eszköznyilvántartó rendszer komponenseinek kezelését végzi:

- **Komponensek lekérdezése:** Listázás lapozással, keresés különböző mezők alapján (név, márka, státusz)
- **Komponens részleteinek lekérdezése:** Egy komponens adatainak részletes megjelenítése
- **Komponens létrehozása:** Új eszköz/alkatrész felvétele a rendszerbe, automatikus dátumbélyeggel
- **Komponens módosítása:** Meglévő komponens adatainak frissítése
- **Komponens törlése:** Eszköz eltávolítása a nyilvántartásból
- **Státuszkezelés:** Komponensek állapotának módosítása (raktáron, használatban, selejtezve)
- **Felhasználói kapcsolat:** Eszközök összekapcsolása a felelős/megrendelő felhasználókkal

A controller biztosítja a komponensek teljes életciklus-kezelését és integrálja a felhasználói rendszerrel, amely lehetővé teszi az eszközpark hatékony nyomon követését és menedzselését.

Component.js

IT alkatrészek adatbázismodelljét definiálja Mongoose sémával:

- **Adatszerkezet:** A rendszerben tárolt komponensek/alkatrészek struktúráját határozza meg
- **Fő mezők:**
    - compName: Komponens neve (kötelező)
    - compType: Komponens típusa (pl. processzor, memória, monitor)
    - brand: Gyártó/márka (pl. Dell, HP, Intel)
    - description: Részletes leírás a komponensről
    - serial: Sorozatszám egyedi azonosításhoz
    - status: Komponens állapota (pl. raktáron, használatban, selejtezve)
    - orderedBy: Kapcsolat a User modellhez (ki rendelte/felelős érte)
    - createdAt: Létrehozás időpontja
    - updatedAt: Utolsó módosítás időpontja
- **Kapcsolatok:** Más modellek (User) kapcsolódási pontjai
- **Validáció:** Kötelező mezők és adattípus-ellenőrzés
- **Indexek:** Hatékony kereséshez optimalizált adatbázis-indexek

User.js

A felhasználói adatbázismodellt definiálja Mongoose sémával:

- **Adatszerkezet:** A rendszerben regisztrált felhasználók struktúráját határozza meg
- **Fő mezők:**
    - username: Egyedi felhasználónév bejelentkezéshez (kötelező)
    - password: Hash-elt jelszó (kötelező, nem tárolódik közvetlenül)
    - email: Felhasználó e-mail címe (opcionálisan validált)
    - role: Felhasználói szerepkör (pl. admin, user) jogosultságkezeléshez
    - createdAt: Regisztráció időpontja
    - lastLogin: Utolsó bejelentkezés időpontja
    - isActive: Felhasználói fiók aktív állapota
- **Módszerek:**
    - Jelszó összehasonlító metódus bejelentkezéshez
    - Virtuális mezők és getter/setter metódusok
- **Validáció:** Egyedi felhasználónév ellenőrzése, jelszó komplexitás validálása
- **Biztonság:** Jelszó soha nem kerül lekérdezésre, csak hash-elt formában tárolódik

authMiddleware.js

Express middleware komponens, amely a felhasználói hitelesítésért és jogosultságkezelésért felelős. Elsődleges feladata annak ellenőrzése, hogy a beérkező kérések hitelesített felhasználótól származnak-e, így biztosítva a védett erőforrások és API végpontok biztonságát.

authRoutes.js

A felhasználói autentikációval kapcsolatos műveleteket tartalmazza. Felelős a regisztrációért, bejelentkezésért, kijelentkezésért. Megvalósul benne az új felhasználó létrehozása ellenőrzött módon (megadott felhasználónév nem-e foglalt még), jelszót hash-elve kezeli, session-t, cookie-t kezel.

componentRoutes.js

Az alkatrész (komponensek) kezelésével kapcsolatos API végpontokat definiálja, valamint hibákat kezel.

Definiált útvonalak:

1. Komponensek lekérdezése (GET /components)

- Összes komponens listázása szűréssel és lapozással
- Támogatja a keresést különböző mezők alapján (név, márka, státusz)
- Limit és offset paraméterekkel lapozást biztosít

- A komponensekhez a felelős/rendelő felhasználót is betölti (populate)

2. Egyedi komponens lekérdezése (GET /components/:id)

- Egy adott komponens részletes adatainak lekérdezése azonosító alapján
- A komponenshez kapcsolódó felhasználói adatokat is betölti

3. Új komponens létrehozása (POST /components)

- Új komponens/alkatrész hozzáadása a nyilvántartáshoz
- Validálja a kötelező mezőket
- A bejelentkezett felhasználót állítja be rendelőként/felelősként
- Eltárolja a létrehozás és utolsó módosítás időpontját

4. Komponens módosítása (PUT /components/updateComponent/:id)

- Meglévő komponens adatainak frissítése
- Ellenőrzi, hogy a komponens létezik-e
- Frissíti a módosítás időpontját
- Csak a megadott mezőket frissíti

5. Komponens törlése (DELETE /components/:id)

- Komponens eltávolítása a nyilvántartásból

6. Státusz módosítása (PUT /components/status/:id)

- Komponens státuszának módosítása (pl. "raktáron", "használatban", "selejtezve")

Frontend

Alkalmazás áttekintés

Egy Angular alapú single-page alkalmazás, amely IT komponensek/eszközök nyilvántartására szolgál:

- **Technológiai stack:** Angular, Angular Material
- **Adatkezelés:** HTTP kliensekkel kommunikál a backend API-val

Főbb modulok

AppModule

- Az alkalmazás gyökérmodulja
- Alapvető Angular szolgáltatások konfigurációja
- Routing és közös szolgáltatások beállítása

AuthModule

- Felhasználói hitelesítésért felelős komponensek
- Login és regisztrációs felületek
- Session és token kezelés

ComponentsModule

- IT komponensek/eszközök kezelésének modulja
- Listázás, létrehozás, módosítás, törlés funkciók
- Szűrési és keresési lehetőségek

Fontosabb komponensek

LoginComponent

- Bejelentkezési felület
- Felhasználónév és jelszó validáció
- Hibakezelés és visszajelzés a felhasználónak

AllComponentsComponent

- Eszközök/komponensek listázása
- Lapozás és keresési funkciók
- Táblázatos megjelenítés szűrési lehetőségekkel

LayoutComponent

- Az alkalmazás fő elrendezése
- Navigációs sáv és menü

- Bejelentkezett felhasználó kezelése

Fő szolgáltatások (Services)

AuthService

- Felhasználói bejelentkezés és regisztráció kezelése
- Session állapot menedzselése
- BehaviorSubject az aktuális felhasználói állapot követésére
- HTTP kérések a backend auth végpontokhoz

ComponentService

- Komponensek CRUD műveleteinek kezelése
- HTTP kommunikáció a backend API-val
- Adat transzformáció és formázás

UserService

- Felhasználói profilok és beállítások kezelése
- Felhasználói adatok cache-elése
- Adatok továbbítása a komponensek között

Routing felépítés

Az alkalmazás routing rendszere a következő fő útvonalakat tartalmazza:

- /login - Bejelentkezési képernyő
- /components - Komponensek listázása
- /components/add - Új komponens hozzáadása
- /components/:id - Komponens részletes nézet
- /components/:id/edit - Komponens szerkesztése
- /profile - Felhasználói profil kezelése

Űrlapkezelés

- Reaktív form megközelítés az Angular FormModule használatával
- Validátorok a bementi mezők ellenőrzéséhez
- Mat-form-field komponensek a Material designhoz
- Hibakezelés és felhasználói visszajelzések

Dialógusok és felugró ablakok

UpdateComponentDialogComponent

- Komponens adatainak módosítására szolgál

- Reaktív űrlapot használ az adatok kezeléséhez
- Validálja a beírt adatokat mentés előtt

DeleteConfirmDialogComponent

- Törlési műveletek megerősítésére szolgál
- Visszajelzést ad a törlés sikerességéről
- Megelőzi a véletlen adattörlést

Biztonság és hitelesítés

- JWT token vagy session cookie alapú hitelesítés
- Route guard-ok védik az engedély nélküli hozzáférést
- XSRF/CSRF védelem a módosító kérésekhez
- Automatikus kijelentkezés inaktivitás esetén

UI/UX sajátosságok

Material Design komponensek

- Mat-table a komponensek listázásához
- Mat-card a részletes nézetekhez
- Mat-form-field az űrlapelemekhez