Szent János Görögkatolikus Gimnázium és Szakképző Iskola

Sign-ey

beléptető rendszer

Készítette:

Lakatos Viktor

Stefán Levente

Edelény, 2025

1. Bevezetés

A Sign-ey egy innovatív, webalapú beléptetőrendszer, amelyet kifejezetten oktatási intézmények, például iskolák számára fejlesztettünk ki. A rendszer célja, hogy a diákok, tanárok és más intézményi dolgozók belépési és kilépési folyamatait automatizálja, miközben biztosítja a teljes körű nyomon követhetőséget és biztonságot. A Sign-ey rendszer minden diák és tanár számára egyedi belépőkártyát biztosít.

A projekt célja, hogy egy modern, könnyen használható rendszert hozzunk létre, amely megfelel az oktatási intézmények speciális igényeinek.

A Sign-ey nemcsak egy beléptetőrendszer, hanem egy átfogó megoldás, amely az intézmény mindennapi működését támogatja.

2. Projekt Célja és Küldetése

A Sign-ey projekt küldetése, hogy az oktatási intézmények számára egy olyan rendszert biztosítson, amely:

- **Automatizálja a beléptetési folyamatokat**: A diákok és dolgozók belépése és kilépése gyorsan és zökkenőmentesen történik.
- **Biztonságot nyújt**: A rendszer naplózza az összes belépési és kilépési eseményt, így bármikor visszakövethetők az események.
- **Egyszerűsíti az adminisztrációt**: Az intézmény vezetése valós idejű adatokat kap a jelenlétekről, hiányzásokról és egyéb eseményekről.
- **Támogatja a diákok fejlődését**: A rendszer integrálható más oktatási rendszerekkel, például tanulmányi nyilvántartásokkal vagy e-learning platformokkal.

A projekt célja, hogy az iskolák számára egy olyan eszközt biztosítson, amely nemcsak a beléptetést, hanem az intézmény egész működését hatékonyabbá és biztonságosabbá teszi.

3. Funkciók és Működés

Belépési és Kilépési Naplózás

A belépési és kilépési naplózás a rendszer egyik legfontosabb funkciója. Minden diák és dolgozó belépési és kilépési adatait automatikusan rögzíti a rendszer. Ez a funkció lehetővé teszi:

- **Valós idejű nyomon követést**: Az adminisztrátorok valós időben láthatják, hogy ki tartózkodik az intézmény területén.
- **Jelenléti adatok gyűjtését**: A rendszer automatikusan generál jelenléti adatokat, amelyeket a tanárok és az adminisztrátorok is elérhetnek.
- **Biztonsági események kezelését**: Ha egy diák vagy dolgozó nem rendelkezik érvényes belépési jogosultsággal, a rendszer figyelmeztetést küld.

Kártyakezelés

A kártyakezelés lehetővé teszi az adminisztrátorok számára, hogy új kártyákat adjanak hozzá a rendszerhez, vagy módosítsák a meglévő kártyák jogosultságait. A funkció főbb elemei:

- **Új kártyák regisztrálása**: Az adminisztrátorok új kártyákat rendelhetnek hozzá a diákokhoz vagy dolgozókhoz.

- **Kártyák deaktiválása**: Ha egy kártya elveszik vagy ellopják, az adminisztrátorok azonnal deaktiválhatják azt.

- **Jogosultságok kezelése**: A kártyákhoz különböző jogosultságok rendelhetők, például belépési zónák vagy időkorlátok.

Időzítés Beállítása

Az időzítés funkció lehetővé teszi az adminisztrátorok számára, hogy előre meghatározott időpontokban automatikusan nyissák vagy zárják az ajtókat. Ez a funkció különösen hasznos:

- **Rendezvények során**: Az időzítés segítségével az ajtók nyitva tarthatók egy adott esemény ideje alatt.

Vészhelyzeti Funkciók

A vészhelyzeti funkciók lehetővé teszik az adminisztrátorok számára, hogy gyorsan reagáljanak rendkívüli helyzetekre. A funkciók közé tartozik:

- **Vészhelyzeti értesítések küldése**: Az adminisztrátorok értesítéseket küldhetnek a tanároknak és diákoknak.
- **Ajtók automatikus zárása**: A rendszer automatikusan lezárhatja az ajtókat vészhelyzet esetén.

Statisztikai Elemzések

A statisztikai elemzések lehetővé teszik az adminisztrátorok számára, hogy részletes jelentéseket készítsenek a belépési és kilépési adatok alapján. A funkciók közé tartozik:

- **Jelenléti statisztikák**: A rendszer megmutatja, hogy hány diák és dolgozó tartózkodik az intézményben egy adott időpontban.
- **Hiányzási adatok**: A rendszer automatikusan generál jelentéseket a hiányzásokról.
- **Trendek elemzése**: Az adminisztrátorok megtekinthetik a belépési és kilépési adatok hosszú távú trendjeit.

4. Hogyan Működik?

A Sign-ey rendszer alapja egy RFID-technológián alapuló belépőkártya, amelyet minden diák és dolgozó megkap. A kártya segítségével a következő funkciók érhetők el:

Az RFID (Radio Frequency Identification) kártyaolvasók olyan eszközök, amelyek rádiófrekvenciás technológiát használnak az adatok olvasására és írására RFID címkékről vagy kártyákról. Ezeket széles körben alkalmazzák beléptetési rendszerekben, mivel gyorsak, megbízhatóak és érintésmentes működést biztosítanak.

RFID működési elve:

Az RFID rendszer három fő komponensből áll:

1. **RFID címke (tag)**: Ez egy kis eszköz, amely tartalmaz egy mikrochipet és egy antennát. A mikrochip tárolja az adatokat (pl. egyedi azonosítót, azaz UID-t), míg az antenna lehetővé teszi a kommunikációt az olvasóval.
2. **RFID olvasó**: Az olvasó rádiófrekvenciás jeleket bocsát ki, amelyek energiát biztosítanak a passzív címkék számára, és lehetővé teszik az adatok kiolvasását vagy írását.
3. **Vezérlő rendszer**: Az olvasó által kiolvasott adatokat egy vezérlő rendszer (pl. számítógép, mikrokontroller) dolgozza fel, amely döntéseket hoz (pl. beléptetés engedélyezése vagy megtagadása).

RFID típusai:

Az RFID rendszerek különböző frekvenciasávokban működhetnek:

- **LF (Low Frequency)**: 125-134 kHz. Rövid hatótávolság (kb. 10 cm), lassabb adatátvitel. Gyakran használják állatok azonosítására.
- **HF (High Frequency)**: 13.56 MHz. Közepes hatótávolság (kb. 10-30 cm). Ez a leggyakoribb típus beléptetési rendszerekben (pl. MIFARE kártyák).
- **UHF (Ultra High Frequency)**: 860-960 MHz. Nagyobb hatótávolság (akár több méter), gyors adatátvitel. Logisztikában és raktározásban használják.
  RFID kártyaolvasók használata beléptetéshez

1. **Kártyaolvasás**: Az RFID olvasó folyamatosan rádiófrekvenciás jeleket bocsát ki. Amikor egy RFID kártyát az olvasó hatótávolságába helyeznek, a kártya antennája energiát kap, és válaszol az olvasónak az UID-jével vagy más tárolt adatokkal.
2. **Adatok feldolgozása**: Az olvasó továbbítja az adatokat a vezérlő rendszernek, amely összehasonlítja azokat egy adatbázissal vagy előre meghatározott engedélyezett listával.
3. **Döntéshozatal**: Ha a kártya azonosítója szerepel az engedélyezett listán, a rendszer engedélyezi a belépést (pl. egy ajtónyitó relé aktiválásával). Ha nem, a belépést megtagadja.

Előnyök:

- **Érintésmentes működés**: Nem szükséges fizikai érintkezés az olvasó és a kártya között.

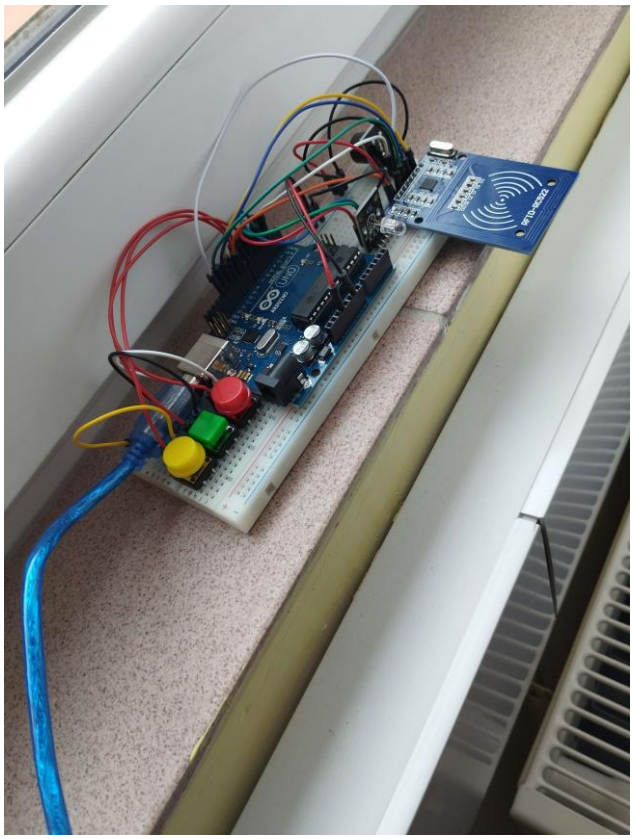
- **Gyors azonosítás**: Az RFID rendszerek másodpercek töredéke alatt képesek azonosítani a kártyákat.
- **Biztonság**: Az RFID kártyák titkosított adatokat is tárolhatnak, ami növeli a rendszer biztonságát.

Hátrányok:

- **Hatótávolság korlátozottsága**: A legtöbb beléptetési rendszerben használt RFID olvasók csak néhány centiméteres távolságból működnek.
- **Klónozás veszélye**: Az egyszerűbb RFID kártyák (pl. régebbi MIFARE típusok) könnyen klónozhatók, ha nincs megfelelő titkosítás.

Alkalmazási területek:

- **Beléptetési rendszerek**: Irodák, lakóépületek, parkolók.
- **Tömegközlekedés**: Elektronikus jegyrendszerek (pl. metrókártyák).
- **Logisztika**: Áruk nyomon követése.
- **Egészségügy**: Betegazonosítás.



Belépés az intézménybe:

a. A diákok és dolgozók a kártyájukat egy beléptető terminálhoz érintve léphetnek be az intézmény területére.
b. A rendszer automatikusan naplózza a belépés időpontját és a belépő személy azonosítóját.

5. Felhasználói Szerepkörök

A Sign-ey rendszer különböző szerepköröket támogat, amelyek mindegyike eltérő jogosultságokkal rendelkezik:

Diákok:

a. Belépés az intézménybe.
b. Órákra való bejelentkezés.

Tanárok:

c. Jelenléti adatok megtekintése.

Adminisztrátorok:

d. Felhasználók hozzáadása és kezelése.
e. Napló adatok megtekintése és exportálása.
f. Rendszerbeállítások módosítása.

Intézményvezetők:

g. Teljes körű hozzáférés a rendszerhez.
h. Statisztikák és jelentések megtekintése.
i. Biztonsági események kezelése.

6. Frontend Működése

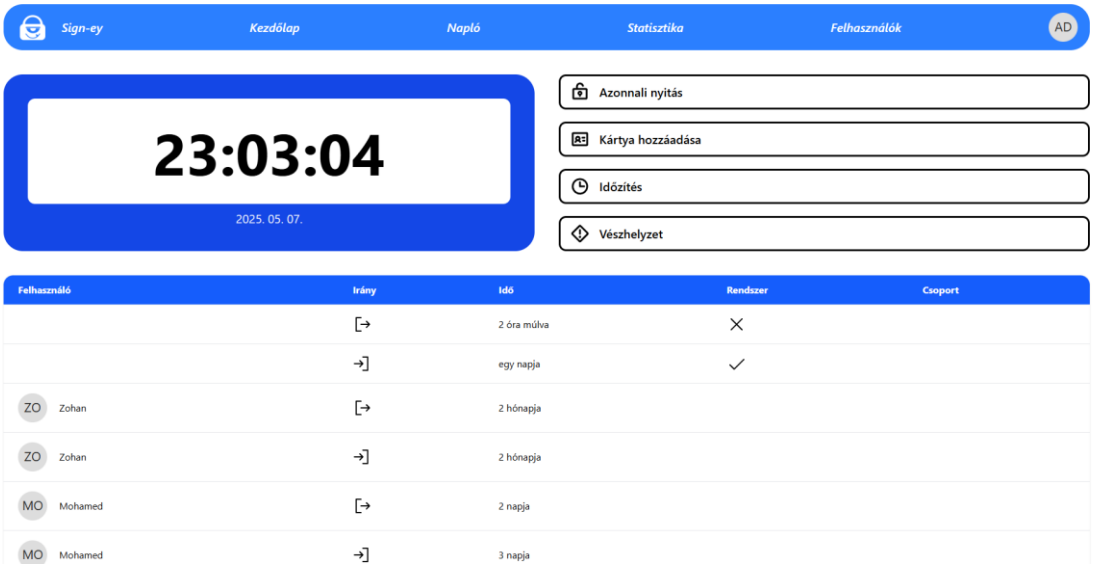
A frontend a felhasználói felületet biztosítja, amelyen keresztül a diákok, tanárok és adminisztrátorok elérhetik a rendszer funkcióit. A frontend főbb elemei:

- **Komponensalapú architektúra**: A React.js lehetővé teszi az újrafelhasználható komponensek létrehozását, például a navigációs sávot, a táblázatokat és a modal ablakokat.
- **Reszponzív dizájn**: A Tailwind CSS segítségével a rendszer minden eszközön, például mobiltelefonokon és táblagépeken is jól működik.

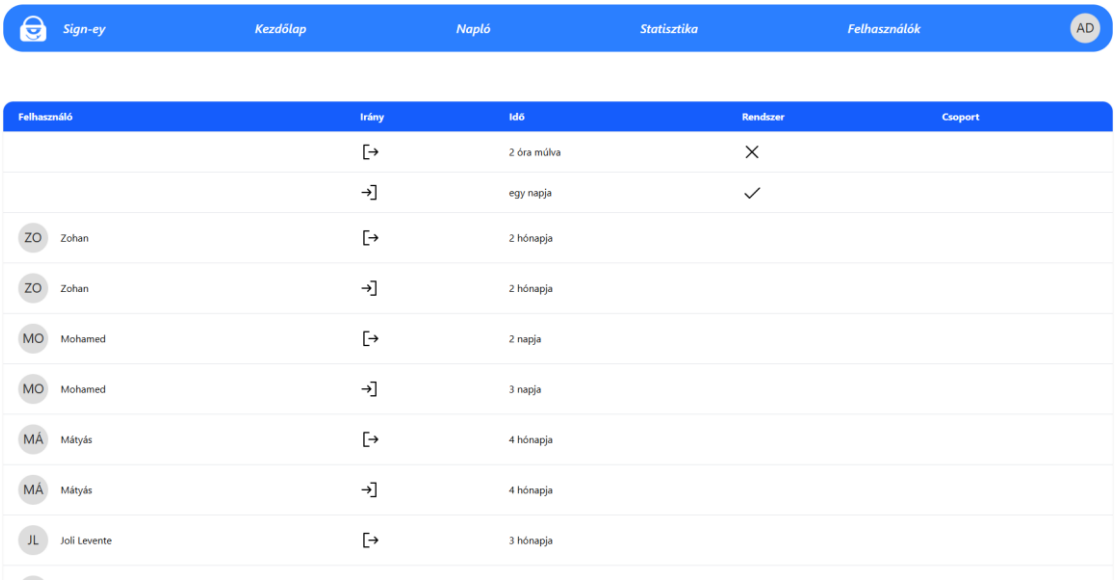
- **API-hívások kezelése**: A Fetch segítségével a frontend kommunikál a backenddel, és valós időben frissíti az adatokat.

BEJELENTKEZÉS

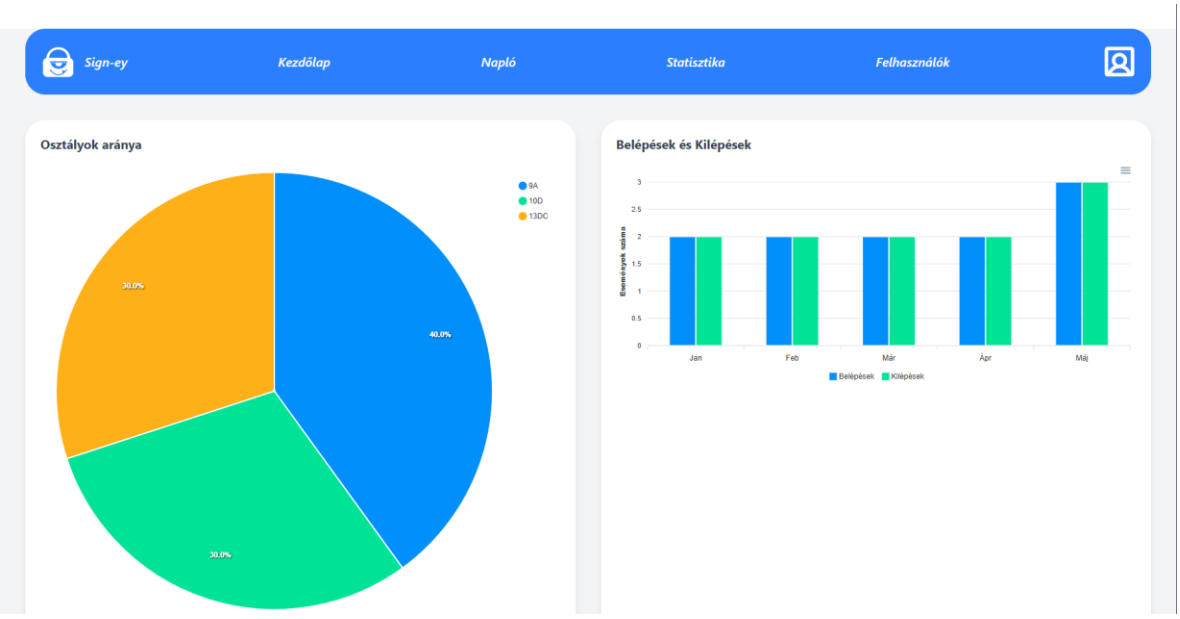
KEZDŐLAP



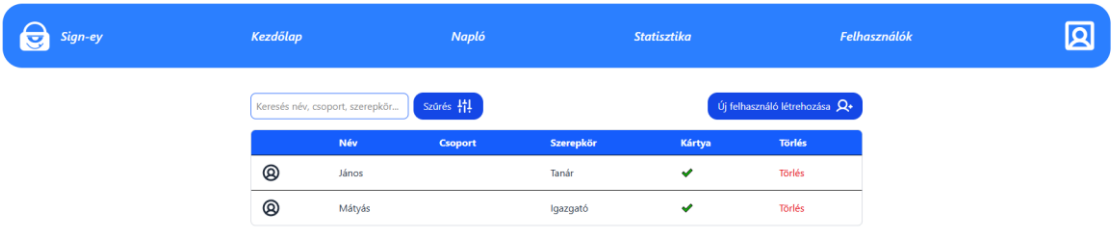
Napló

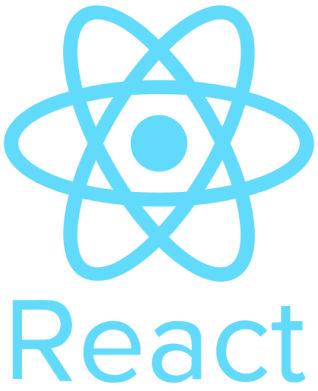


Statisztika



Felhasználó





React.js

Tailwind CSS

7. A Rendszer Különlegessége és Előnyei

A Sign-ey rendszer különlegességei:

- **Valós idejű működés**: Az aktuális események azonnal megjelennek a rendszerben.
- **Moduláris felépítés**: Könnyen bővíthető új funkciókkal.
- **Biztonság**: Az összes esemény naplózása és visszakövethetősége.
- **Felhasználóbarát felület**: Egyszerű és intuitív kezelőfelület.

8. Jövőbeli fejlesztések

Biometrikus azonosítás:

a. Ujjlenyomat-olvasók vagy arcfelismerő rendszerek integrálása.

Push értesítések:

b. Az adminisztrátorok értesítése vészhelyzet esetén.

Integráció más rendszerekkel:

c. Kamerarendszerek vagy riasztórendszerek integrálása.

Mobilalkalmazás:

d. Egy dedikált mobilalkalmazás fejlesztése a rendszer távoli vezérléséhez.

Felhasználói jogosultságok:

e. Részletesebb jogosultságkezelés, például különböző szintű hozzáférések biztosítása.

9. Miért fontosak ezek a fejlesztések?

- **Biztonság növelése**: A biometrikus azonosítás és a push értesítések növelik a rendszer biztonságát.
- **Felhasználói élmény javítása**: A mobilalkalmazás és az integrációk kényelmesebbé teszik a rendszer használatát.
- **Skálázhatóság**: Az új funkciók lehetővé teszik a rendszer bővítését nagyobb intézmények számára.

BACKEND

1. Áttekintés

Ez a beléptetőrendszer egy Node.js alapú alkalmazás, amely egy soros porton keresztül kommunikál egy hardveres eszközzel (pl. kártyaolvasó). Az alkalmazás az Express.js keretrendszert használja az API végpontok kezelésére, a Sequelize ORM-et az adatbázis-kezeléshez, és támogatja a felhasználói hitelesítést, valamint a belépési/kilépési események naplózását.

2. Főbb Funkciók

Felhasználók kezelése: Felhasználók létrehozása, csoportokhoz rendelése, és belépési/kilépési jogosultságok kezelése.

Belépési/kilépési események naplózása: Minden esemény rögzítése az adatbázisban.

Soros port kommunikáció: Hardveres események kezelése (pl. ajtó nyitása/zárása).

API végpontok: Adatok lekérdezése és kezelése REST API-n keresztül.

Hitelesítés és munkamenet-kezelés: Felhasználói munkamenetek kezelése express-session és passport segítségével.

3. Fejlesztői szoftverek

NODE.JS

EXPRESS.JS

SEQUELIZE

MySQL

PASSPORT.JS

DOTENV

POSTMAN

- **Backend**: Node.js, Express.js
- **Adatbázis-kezelés**: Sequelize ORM (MySQL vagy más SQL-alapú adatbázis)
- **Soros port kommunikáció**: serialport könyvtár
- **Hitelesítés**: Passport.js
- **Környezetváltozók kezelése**: dotenv
- **Middleware-ek**: express-session, cors

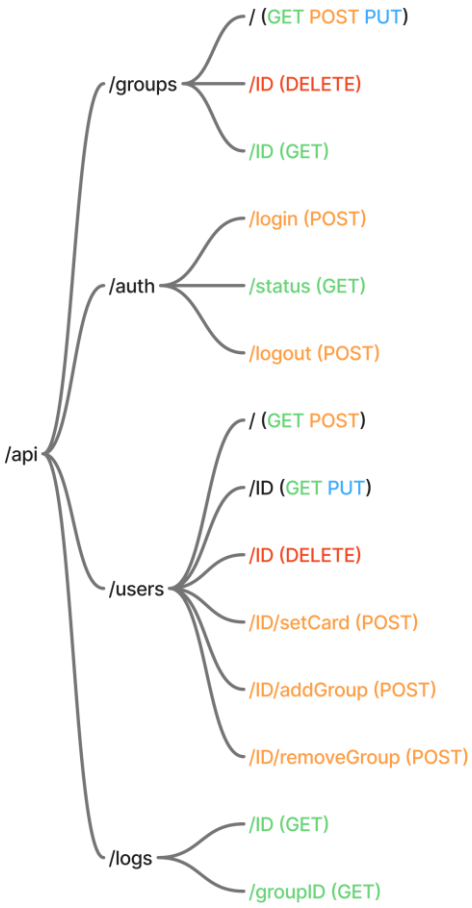
4. Adatbázis Struktúra

Az adatbázis a következő főbb táblákat tartalmazza:

User (Felhasználók)

a. userID: Egyedi azonosító.
b. email: Felhasználó e-mail címe.
c. password: Titkosított jelszó.
d. name: Felhasználó neve.

e. phone: Telefonszám (opcionális).
f. groupID: A felhasználóhoz rendelt csoport azonosítója.
g. cardNumber: A felhasználóhoz rendelt kártyaszám.



Group (Csoportok)

h. groupID: Egyedi azonosító.
i. name: Csoport neve.
j. enterTimeRange: Belépési időintervallum (pl. "08:00-18:00").
k. exitTimeRange: Kilépési időintervallum (pl. "08:00-18:00").

Log (Napló)

l. logID: Egyedi azonosító.
m. userID: A felhasználó azonosítója (ha van).
n. entryTime: Belépési idő.
o. exitTime: Kilépési idő.
p. action: Az esemény típusa (pl. "OPEN", "CLOSE", "RESET").

5. Főbb Fájlok és Funkcióik

index.js

a. Az alkalmazás belépési pontja.
b. Inicializálja az Express.js szervert, az adatbázist, és a soros port kommunikációt.
c. Kezeli az alapértelmezett végpontokat (pl. / és /api).

schema/index.js

d. Definiálja az adatbázis-modelleket (User, Group, Log).
e. Beállítja a modellek közötti kapcsolatokat (pl. User és Group közötti kapcsolat).

Serial.js

f. Kezeli a soros port kommunikációt.
g. Inicializálja a soros portot és a ReadlineParser-t.

router/logs/index.js

h. API végpontokat biztosít a naplózott események lekérdezéséhez.
i. Támogatja a szűrést (pl. felhasználó, csoport, dátum alapján).

schema/User.js

j. Definiálja a User modellt.
k. Tartalmazza a jelszó titkosításához és ellenőrzéséhez szükséges metódusokat.

6. API Végpontok

GET /

a. **Leírás**: Az API verzióját adja vissza.
b. **Válasz**: API <API_VERSION>.

GET /api/logs

c. **Leírás**: Naplózott események lekérdezése.
d. **Query paraméterek**:
    I. userID: Felhasználó azonosítója.
    II. groupID: Csoport azonosítója.
    III. manual: Manuális események szűrése.
    IV. fromDate: Kezdő dátum.
    V. toDate: Záró dátum.
    VI. limit: Eredmények maximális száma.
e. **Válasz**: Naplózott események listája.

7. Soros Port Kommunikáció

A soros porton érkező adatok feldolgozása a következőképpen történik:

Események:

a. "NYITAS-MANUAL": Ajtó manuális nyitása.
b. "ZARAS-MANUAL": Ajtó manuális zárása.
c. "RESET": Rendszer alaphelyzetbe állítása.
d. "KOD:<kártyaszám>": Kártyaszám alapján belépési/kilépési jogosultság ellenőrzése.

Feldolgozás:

e. A kártyaszám alapján a rendszer megkeresi a felhasználót az adatbázisban.
f. Ellenőrzi a belépési/kilépési időintervallumokat.
g. Ha jogosult, az ajtó nyitása/zárása megtörténik, és az esemény naplózásra kerül.

8. Környezetváltozók

Az alkalmazás működéséhez szükséges környezetváltozók:

- DB: Az adatbázis kapcsolati URL-je.
- PORT: A soros port elérési útvonala.
- SERVER_PORT: Az Express.js szerver portja.

- FRONTEND_URL: A frontend URL-je.
- API_VERSION: Az API verziója.

9. Telepítés és Futtatás

Követelmények:

a. Node.js (ajánlott verzió: 16.x vagy újabb)
b. MySQL vagy más SQL-alapú adatbázis

Telepítés:

npm install

Környezetváltozók beállítása: Hozz létre egy .env fájlt az alábbi tartalommal:

DB=mysql://user:password@localhost:3306/database
PORT=COM3
SERVER_PORT=3000
FRONTEND_URL=http://localhost:4200
API_VERSION=1.0

Futtatás:

node index.js

10. Fejlesztési Lehetőségek

- **További végpontok**: Felhasználók és csoportok kezelésére szolgáló CRUD végpontok.
- **Tesztelés**: Egység- és integrációs tesztek hozzáadása.
- **Biztonság**: HTTPS támogatás és erősebb hitelesítési mechanizmusok.
- **Frontend integráció**: A rendszerhez tartozó felhasználói felület fejlesztése.