# Tartalomjegyzék

[Tartalomjegyzék 1](#_Toc195260775)

[Bevezetés 3](#_Toc195260776)

[Témaválasztás 3](#_Toc195260777)

[Feladat specifikáció 4](#_Toc195260778)

[A program célja és lényegesebb funkciói 4](#_Toc195260779)

[Futtatási környezet 4](#_Toc195260780)

[A program fő funkcióinak leírása 5](#_Toc195260781)

[Drótvázak 7](#_Toc195260782)

[Tervezés 11](#_Toc195260783)

[Fejlesztői környezet 11](#_Toc195260784)

[Adatbázis 11](#_Toc195260785)

[Leírás 11](#_Toc195260786)

[Funkciók 12](#_Toc195260787)

[Adatbázis terv 13](#_Toc195260788)

[Egyed kapcsolat diagram 14](#_Toc195260789)

[Technikai táblák 15](#_Toc195260790)

[Táblák szerkezete a tábla szintű megszorításokkal 16](#_Toc195260791)

[Kapcsolati ábra: 19](#_Toc195260792)

[Hasznos lekérdezések 20](#_Toc195260793)

[Beépítendő megszorítások 21](#_Toc195260794)

[Scriptek 22](#_Toc195260795)

[Backend 23](#_Toc195260796)

[Frontend 27](#_Toc195260797)

[Reszponzivitás 27](#_Toc195260798)

[Implementáció 28](#_Toc195260799)

[Program struktúra 28](#_Toc195260800)

[Backend 28](#_Toc195260801)

[Frontend 28](#_Toc195260802)

[Tesztelés 29](#_Toc195260803)

[Adatbázis 29](#_Toc195260804)

[Backend 29](#_Toc195260805)

[Frontend 29](#_Toc195260806)

[Adatbázis 30](#_Toc195260807)

[Fejlesztési lehetőségek 30](#_Toc195260808)

[Irodalomjegyzék 31](#_Toc195260809)

# Bevezetés

## Témaválasztás

A szakdolgozat témájának a magánklinika online foglalási rendszer fejlesztését választottuk, mert ez egy valós problémára ad digitális megoldást, illetve javíthatja a klinikák működését és a páciensek élményét.

Az egészségügyi szektor egyre inkább digitalizálódik, és egy hatékony, felhasználóbarát foglalási rendszer nagy előnyt jelenthet a magánklinikák számára. Az emberek hozzászoktak az online időpontfoglaláshoz, és elvárják, hogy gyorsan és kényelmesen intézhessék ügyeiket.

Szakdolgozatunk egy valós életben is használható rendszert mutat be, amelyet egy valódi klinika is bevezethetne.

A projekt során többféle technológiát alkalmazunk, például:

* Backend fejlesztés (adatbázis-kezelés, API-k, autentikáció)
* Frontend fejlesztés (felhasználóbarát UI, reszponzivitás)
* Mobilbarát megoldások (mivel a páciensek sok esetben mobilról foglalnak)
* Tesztelési folyamatok (automatizált és manuális tesztelés a megbízhatóság érdekében)

Ezek hozzájárulnak ahhoz, hogy egy komplex, fejlesztői és tesztelői kihívást oldjunk meg.

# Feladat specifikáció

## A program célja és lényegesebb funkciói

A feladat célja egy orvos-páciens menedzsment rendszer létrehozása, amely lehetőséget biztosít a látogatóknak a regisztrációra, az orvosi vizsgálatok közötti keresésre, valamint a vizsgálatok időpontfoglalására és kezelésére.

A páciensek bejelentkezés után egyszerűen megtekinthetik, módosíthatják és lemondhatják a jövőbeni időpontjaikat, valamint kiválaszthatják a szabad időpontok közül azokat, amelyek az általuk kívánt eljárásokhoz elérhetőek, tehát a páciens egy szakon belül lévő vizsgálatra kiválaszt egy szabad időpontot, és látja a hozzá tartozó orvos(oka)t. Az orvos kiválasztását követően le tudja foglalni a vizsgálatot.

Az orvosok saját profiljukon keresztül kezelhetik rendelési idejüket és nyomon követhetik a páciensek értékeléseit. Egy orvoshoz egy szak tartozhat.

Az adminisztrátorok hozzáféréssel rendelkeznek a felhasználói nem szenzitív adatokhoz és a foglalások kezeléséhez, emellett orvosokat regisztrálhatnak.

## Futtatási környezet

Az ötletelés fázisában gyorsan felmerült, hogy időpontfoglaló applikáció készítése illeszkedik leginkább az elképzelésekhez, amelyet webalkalmazás formájában kívánunk véghez vinni. Ennek oka, hogy kiemelten fontosnak tartottuk a felhasználóbarátságot, és résztvevők sokrétűsége miatti általános megtekinthetőséget. Ezáltal telefonon, vagy nagy képernyő előtt számítógépen ugyanúgy megtekinthető a felület külön alkalmazás telepítése nélkül, amely bizonyos felhasználók esetén nehézséget okozna. Emellett fontosnak tartottuk a platform független működést, így a felhasználó számára is egyszerűbb egy weboldal megtekintése, mint egy alkalmazás futtatása (esetlegesen frissítést követően), emellett nem követel meg átlagon felüli sávszélességet. A megtekintésre a webböngészők friss verziói a legalkalmasabbak (Microsoft Edge, Google Chrome, Mozilla Firefox, Opera).

## A program fő funkcióinak leírása

A program fő célja:

A szakdolgozat keretében fejlesztett rendszer egy magánklinika online időpontfoglaló és menedzsment rendszerének létrehozása, amelynek célja, hogy egyszerűsítse és digitalizálja a klinika működésével kapcsolatos legfontosabb folyamatokat.

A rendszer fő funkciói a következők:

1. Könnyű és gyors időpontfoglalás
   * A páciensek online, néhány kattintással foglalhatnak időpontot a megfelelő szakterület orvosához.
   * Az elérhető időpontokat valós időben frissíti a rendszer.
2. Orvosok és adminisztrátorok munkájának megkönnyítése
   * Az orvosok megadhatják rendelési idejüket, szabadságukat, és kezelhetik a foglalásokat.
   * Az adminisztrátorok szükség esetén módosíthatják az időpontokat és kezelhetik a felhasználókat.
3. Felhasználóbarát és mobilbarát felület
   * A rendszer reszponzív dizájnnal készül, így asztali gépen és mobiltelefonon is könnyen használható.
   * Letisztult és intuitív felület biztosítja a gördülékeny navigációt.
4. Értesítések és emlékeztetők
   * A páciensek automatikus értesítést kapnak foglalásukról és annak közelgő időpontjáról.
   * Az értesítések csökkentik az elfelejtett időpontok számát, optimalizálva az orvosok időbeosztását.
5. Biztonságos adatkezelés és jogosultságkezelés
   * A rendszer különböző szerepköröket használ (páciens, orvos, admin), hogy mindenki csak a számára releváns adatokhoz férjen hozzá.
   * Az érzékeny egészségügyi adatok védelméről modern titkosítási és biztonsági eljárások gondoskodnak.
6. Statisztikák és jelentések
   * Az adminisztrátorok és orvosok betekintést nyerhetnek a foglalási statisztikákba, például a leggyakoribb szolgáltatásokba vagy a kihasználtsági arányokba.
   * Az adatelemzés segíthet a klinika működésének optimalizálásában.

# Drótvázak

Főoldal:

A képen szöveg, képernyőkép, Téglalap, tervezés látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.

Bejelentkezés:

A képen szöveg, képernyőkép, szoftver, képernyő látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.

Regisztráció páciensek esetén:

A képen szöveg, képernyőkép, szoftver, képernyő látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.

Regisztráció orvosok esetén:

A képen szöveg, képernyőkép, képernyő, szoftver látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.

Doktorok listája:

A képen szöveg, képernyőkép, Webhely, Weblap látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.

Időpontok orvosok szerint:

A képen szöveg, képernyőkép, szoftver, képernyő látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.

Időpontok kezelések szerint:

A képen szöveg, képernyőkép, képernyő, szoftver látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.

# Tervezés

## Fejlesztői környezet

A létrehozott magánklinika alkalmazás szerver-kliens alapon működik, segítve az erőforrás optimálisabb használatát. Adatbázis szempontjából Microsoft SQL Servert alkalmaztunk a tervezés során. A kivitelezés MySQL segítségével történt. Szerver oldalon PHP programozási nyelvet használtunk, illetve RESTful mentén Laravel keretrendszert használtunk. Kliens oldali megvalósítás Javascript segítségével történt React használatával. A fejlesztést Visual Studio Code-ban végeztünk, verziókövetésre GitHub-ot használtunk.

## Adatbázis

### Leírás

A Magánklinika adatbázis a klinika betegfoglalásait, orvosi szakrendeléseit és felhasználóit kezeli.

Az adatbázisban kezeljük a felhasználók adatait, akik szerepkörök alapján vannak besorolva. Ezen szerepkörök határozzák meg a rendszerbeli jogosultságaikat, amik 3 részre válnak szét. Lehetnek páciensek, orvosok vagy adminisztrátorok.

A tárolt felhasználók azon adatait, amik megegyeznek az orvosok és páciensek esetén, egy közös táblában kezelnénk. A nem közös adatokat pedig külön az orvosoknak és pácienseknek létrehozandó táblában tárolnánk. Az adminisztrátoroknak nem tervezünk külön táblát, mivel nekik speciális adatra nincs szükség, ami a felhasználóknak létrehozandó táblában nem tárolható.

Minden orvos rendelkezik egy adott szakosodással, amit későbbiekben külön táblában szeretnénk tárolni.

A rendszerben a kezelések és azok időtartama is nyilvántartva kell, hogy legyen. Minden kezeléshez tartoznia kell egy adott szakosod ásnak, amely biztosítja, hogy csak megfelelő szaktudással rendelkező orvosok végezhetik el az adott beavatkozásokat.

Mivel a páciensek időpontokat szeretnének foglalni az orvosokhoz, ezért ezen időpontokat szeretnénk táblában tárolni. A beazonosíthatóság miatt egy foglalás tartalmazza az orvos azonosítóját, a beteg azonosítóját, a foglalás kezdeti időpontját, a kiválasztott kezelést és annak állapotát. Továbbá egy mezőt, ami tartalmazza a doktor által megírt kezelés zárójelentését, Emellett egy értékelés mező is bekerül, amely segíti a pácienseket a megfelelő szakember kiválasztásában. Az időpontokat az orvosok szabad kapacitásai alapján lehet lefoglalni.

### Funkciók

- Felhasználók (páciensek, orvosok, adminisztrátorok) felvétele, módosítása, törlése

- Páciensek időpontfoglalása, módosítása, törlése

- Orvosi szakrendelések kezelése

- Kezelések nyilvántartása, módosítása, törlése

- Orvosok szakosodásának kezelése

- Navigációs menüpontok és jogosultságok kezelése

- Adminisztrátori műveletek: felhasználók, kezelések és foglalások menedzselése

- Orvosi értékelések kezelése és megjelenítése

- Statisztikák és riportok generálása

### Adatbázis terv

Jelölés:   
Tábla magyar neve (TÉNYLEGES TÁBLA NEVE)  
(kulcs, leíró, *kapcsolathordozó*)  
Rövid leírás

#### Adattáblák leírása

Felhasználói szerepkör

{jogosultság azonosító, név}  
A rendszerben minden felhasználó egy meghatározott szerepkörbe tartozik, amely meghatározza a jogosultságait. Például az adminisztrátorok kezelhetik a felhasználókat, míg az orvosok csak a saját időpontjaikat láthatják és módosíthatják. A szerepkörök előre definiáltak, és minden egyes felhasználónak pontosan egy szerepkör van hozzárendelve.

Felhasználó  
{azonosító, név, telefonszám, email, jelszó, *jogosultság azonosító,* profilkép}

Ez a tábla tárolja a rendszer összes regisztrált felhasználóját, függetlenül attól, hogy páciensekről, orvosokról vagy adminisztrátorokról van szó. Minden felhasználónak meg kell adnia a személyes adatait, beleértve az elérhetőségeit és egy biztonságos jelszót. A rendszer biztosítja, hogy egy felhasználónak mindig legyen érvényes szerepköre. Tovább a felhasználóhoz opcionálisan feltölthető a default profilképtől eltérő kép.

Páciens  
{felhasználó azonosító, TAJ szám, születési idő, cím}  
Ez a tábla tartalmazza a klinikára regisztrált páciensek adatait. Minden páciens rendelkezik egy egyedi azonosítóval, amelyet a rendszerben való azonosításra használunk. Az egészségügyi ellátás biztosítása érdekében itt tároljuk a társadalombiztosítási számukat, a születési dátumukat és lakcímüket is.

Orvosi szakterület  
{specializáció azonosító, specializáció név}  
A klinikán dolgozó orvosok különböző szakterületekre specializálódnak, például belgyógyászat, sebészet vagy ortopédia. Ebben a táblában vannak felsorolva ezek a szakterületek.

Orvos  
{felhasználó azonosító*, specializáció azonosító*}  
Az orvosok adatai ebben a táblában vannak tárolva. Minden orvos egyedi azonosítóval rendelkezik, amely a rendszer többi táblájában is használható azonosításra. Az orvosok csak olyan kezeléseket végezhetnek, amelyek a saját szakterületükhöz tartoznak.

Kezelés  
{kezelés azonosító, *specializáció azonosító,* kezelés neve, kezelés hossza, ár}  
A klinika által nyújtott különböző kezelések listája ebben a táblában található. Minden kezeléshez meghatározott időtartam és ár tartozik, valamint egy szakirány, amely biztosítja, hogy csak a megfelelő szaktudással rendelkező orvosok végezhessék el az adott beavatkozást.

Orvosi időpontfoglalás  
{időpont azonosító, *orvos azonosító*, kezdés ideje, *páciens azonosító,* *kezelés azonosító*, leírás, státusz, értékelés}

A páciensek a rendszerben foglalhatnak időpontokat az orvosokhoz. Minden foglalás tartalmazza az érintett orvos és páciens azonosítóját, valamint az időpont kezdetét. A páciens azonosítója akkor kerül a táblába, ha a foglalás megtörténik, ezzel a státusza „b” (foglalt) lesz. Az időpont státusza lehet „v” (szabad), „b” (foglalt), „d” (elvégzett), „c” (orvos által törölt) vagy „p” (páciens által törölt). A rendszer lehetőséget biztosít a kezelések értékelésére is, így a páciensek visszajelzést adhatnak az ellátás minőségéről. Emellett az érintett kezelés aktuális időpontjára az orvos leírást tud adni.

### Egyed kapcsolat diagram

A képen szöveg, diagram, képernyőkép, Betűtípus látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.

### Technikai táblák

A Navigációs menüpont és Menüpont-hozzáférési jogosultság táblák kezelik a rendszer menüpontjait és azt, hogy egy adott szerepkör milyen felületeket érhet el. Az adminisztrátorok jogosultak kezelni a felhasználókat, orvosokat, kezeléseket és a páciensek foglalásait.

Az Doktor licensz tábla tartalmazza a licensz számát, illetve hogy már aktiválásra került-e.

#### Technikai táblák leírása

Navigációs menüpont  
{navigáció azonosító, név, URL, komponens neve, szülő}

A rendszer különböző funkcióit menüpontokon keresztül érhetik el a felhasználók. Minden menüpontnak van egy neve és egy URL-címe, amely meghatározza, hogy melyik oldalra vezet. Egyes menüpontok önállóan léteznek, míg mások egy főmenü alá tartoznak.

Menüpont-hozzáférési jogosultság

{navigációs szerepkör azonosító, sorszám, *joosultság azonosító*, *navigáció azonosító*}

Nem minden menüpont érhető el minden felhasználói szerepkör számára. Ebben a táblában van meghatározva, hogy mely szerepkörök milyen menüpontokat láthatnak és használhatnak. Így például a páciensek nem férhetnek hozzá az adminisztrációs funkciókhoz, míg az adminisztrátorok teljes hozzáféréssel rendelkeznek.

Doktor licensz

{licensz azonosító, licensz sorszám, aktivált}

Elképzelésünk egy izolált tábla, amely az adatbázis szerkezeten kívül helyezkedik el. Ez a tábla az orvos regisztrációjához szükséges.

#### A képen szöveg, képernyőkép, Betűtípus, sor látható Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.Technikai táblák egyed kapcsolat diagram

A felhasználói szerepkör tábla ugyan nem technikai, de közvetlen kacsolatban van a menüpont-hozzáérési jogosultság táblával. Az áttekinthetőség érdekében a Felhasználói szerepkör táblát is szerepeltetjük ezen ábrán.

### Táblák szerkezete a tábla szintű megszorításokkal

#### Adattáblák

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SZEREP (ROLE)** | | | | |
| Mező magyarul | Mező valós | Adattípus | Szerep | Korlátozás |
| jogosultság azonosító | role\_id | bigint(20) | K |  |
| név | name | varchar(255) | kötelező, egyedi |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **FELHASZNÁLÓ (USER)** | | | | |
| Mező magyarul | Mező valós | Adattípus | Szerep | Korlátozás |
| azonosító | id | bigint(20) | K |  |
| név | name | varchar(255) | kötelező |  |
| telefonszám | phone\_number | varchar(255) | kötelező, egyedi |  |
| email | email | varchar(255) | kötelező, egyedi |  |
| jelszó | password | varchar(255) | kötelező |  |
| jogosultság azonosító | role\_id | bigint(20) | KK(Role) | role\_id > 0 & role\_id < role tála méret |
| profil kép | profile\_picture | varchar(255) |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SPECIALIZÁCIÓ (SPECIALIZATION)** | | | | |
| Mező magyarul | Mező valós | Adattípus | Szerep | Korlátozás |
| specializáció azonosító | specialization\_id | bigint(20) | K |  |
| specializáció név | specializaton\_name | varchar(255) | kötelező, egyedi |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ORVOS (DOCTOR)** | | | | |
| Mező magyarul | Mező valós | Adattípus | Szerep | Korlátozás |
| felhasználó azonosító | user\_id | bigint(20) | K, KK (USER) |  |
| specializáció azonosító | specialization\_id | bigint(20) | KK (SPECIALIZATION) |  |

További megszorítások:

* Egy orvos csak egy szakkal rendelkezhet

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **PÁCIENS (PATIENT)** | | | | |
| Mező magyarul | Mező valós | Adattípus | Szerep | Korlátozás |
| felhasználó azonosító | user\_id | bigint(20) | K, KK (USER) |  |
| tajszám | taj\_number | int(11) | kötelező, egyedi | taj\_number > 0 |
| születési idő | birth\_date | date | kötelező | birth\_date > mai dátum |
| cím | address | varchar(255) | kötelező |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **KEZELÉS (TREATMENT)** | | | | |
| Mező magyarul | Mező valós | Adattípus | Szerep | Korlátozás |
| kezelés azonosító | treatment\_id | bigint(20) | K |  |
| specializáció azonosító | specialization\_id | bigint(20) | KK (SPECIALIZATION) |  |
| kezelés neve | treatment\_name | varchar(255) | kötelező |  |
| kezelés hossza | treatment\_length | time | kötelező | kezelés hossza > 0 |
| ár | price | int(10) | kötelező | ár >=0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ORVOSI IDŐPONTFOGLALÁSOK (DOCTOR\_APPOINTMENT)** | | | | |
| Mező magyarul | Mező valós | Adattípus | Szerep | Korlátozás |
| időpont azonosító | id | bigint(20) | K |  |
| orvos azonosító | doctor\_id | bigint(20) | KK része (DOCTOR) |  |
| kezdés ideje | start\_time | datetime | kötelező | kezdés idő > mai nap |
| páciens azonosító | patient\_id | bigint(20) | KK (PATIENT) |  |
| kezelés azonosító | treatment\_id | bigint(20) | KK (TREATMENT) |  |
| leírás | description | varchar(255) |  |  |
| státusz | status | enum | default vacant |  |
| értékelés | rating | int(11) |  | rating > 0 & rating < 6 |

További megszorítások:

* ugyanarra a kezelésre adott páciensnek egy 'booked'(foglalt) státuszú kezelése legyen
* ha a kezelés státusza nem "d", akkor nem lehetséges értékelni
* ugyanazon kezelésre adott páciens csak egy értékelést tudjon adni
* adott orvoshoz, csupán a hozzá tartozó kezelések közül tudjon a páciens foglalni időpontot

#### Technikai táblák

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **NAVIGÁCIÓ (NAVIGATION)** | | | | |
| Mező magyarul | Mező valós | Adattípus | Szerep | Korlátozás |
| navigáció azonosító | navigation\_id | bigint(20) | K |  |
| megnevezés | name | varchar(255) | kötelező |  |
| URL | URL | varchar(255) | kötelező |  |
| komponens neve | component\_name | varchar(255) | kötelező |  |
| szülő | parent | int(11) |  |  |

További megszorítások:

* hurokvizsgálat (parent)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **NAVIGÁCIÓS SZEREPKÖRÖK (NAVIGATION\_ROLE)** | | | | |
| Mező magyarul | Mező valós | Adattípus | Szerep | Korlátozás |
| navigációs szerepkör azonosító | navigationRole\_id | bigint(20) | K |  |
| sorszám | ranking | int(11) | kötelező | ranking > 0 |
| jogosultság azonosító | role\_id | bigint(20) | KK (ROLE) |  |
| navigáció azonosító | navigation\_id | bigint(20) | KK (NAVIGATION) |  |

További megszorítások:

* szerepkörönként nem ismétlődhet a sorszám

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DOKTOR LICENSZ (DOCTOR\_LICENCE)** | | | | |
| Mező magyarul | Mező valós | Adattípus | Szerep | Korlátozás |
| licensz azonosító | licence\_id | bigint(20) | K |  |
| licensz sorszám | licence\_number | int(11) | kötelező |  |
| használatban | isUsed | boolean |  |  |

### Kapcsolati ábra:

A képen szöveg, képernyőkép, Betűtípus, diagram látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.

### Hasznos lekérdezések

1. Adott orvosnak hány elvégzett kezelése volt az előző hónapban?

Ez a lekérdezés egy adott orvos által az előző hónapban végzett sikeres kezelések számát adja vissza. Csak azokat az időpontokat veszi figyelembe, amelyek sikeresen lezajlottak (status = 'done'). Az időintervallum meghatározására a DATEADD függvényeket használja, így mindig az aktuális hónaphoz képest előző hónapot vizsgálja.

1. Adott Páciensnek hány kezelése volt az előző hónapban?

Ez a lekérdezés egy adott páciens számára az előző hónapban elvégzett kezelések számát határozza meg. A páciens egyedi TAJ-száma alapján keres, és csak a lezárt, sikeres kezeléseket számolja (status = 'done'). Az időtartomány az előző hónap teljes időszakára vonatkozik.

1. Hány Páciensnek volt elvégzett kezelése?

Ez a lekérdezés azt határozza meg, hogy hány különböző páciens vett részt legalább egy lezárt (status = 'done') kezelésen. A COUNT(DISTINCT) operátor biztosítja, hogy minden páciens csak egyszer szerepeljen a számlálásban, még akkor is, ha több kezelése volt.

1. Adott orvosnak hány törölt időpontja volt az adott hónapban/évben?

Ez a lekérdezés egy adott orvos által törölt időpontok számát adja vissza egy adott hónapban vagy évben. A törölt időpontokat a status mezőben tárolt adatok alapján lehet azonosítani (status = 'cancelled'). A lekérdezés szűrhető egy adott hónapra vagy évre, így az orvos munkájának elemzésében segíthet.

1. Melyik három orvos kapta a legtöbb értékelést tavaly?

Ez a lekérdezés megmutatja, hogy melyik három orvos kapta a legtöbb visszajelzést (rating IS NOT NULL) az előző évben. Az orvosok értékelései az appointments táblában találhatók, és a COUNT(\*) segítségével összesítjük, hány értékelés érkezett az adott orvoshoz. Az eredmény csökkenő sorrendben jelenik meg, hogy a legtöbb értékelést kapott orvosok kerüljenek előre.

1. Melyik három orvos kapta a legrosszabb értékelést tavaly?

Ez a lekérdezés az előző év legrosszabbul értékelt három orvosát azonosítja. Az orvosok értékeléseinek átlaga (AVG(da.rating)) alapján sorba rendezi az orvosokat növekvő sorrendben, így az első három eredmény a leggyengébben értékelt orvosokat mutatja.

1. Mely userek nem lépett be a rendszerbe több mint 3 éve?

Ez a lekérdezés azokat a felhasználókat keresi, akik legalább három éve nem jelentkeztek be a rendszerbe. Az utolsó bejelentkezési dátum (last\_login) alapján ellenőrzi, hogy a dátum több mint három évvel ezelőtti-e. Ez segíthet az inaktív felhasználók kiszűrésében vagy figyelmeztetés küldésében.

1. Mely szakon történt a legtöbb elvégzett vizsgálat/kezelés?

Ez a lekérdezés azonosítja, hogy melyik orvosi szakterületen végezték a legtöbb sikeres kezelést (status = 'done'). Az egyes kezelésekhez kapcsolódó szakterületeket (specialisation\_name) összesíti, és az eredményt csökkenő sorrendben rendezi, hogy a legnépszerűbb szakterület jelenjen meg elsőként.

1. Mely páciensek mondták le az időpontjaikat legalább 10 alkalommal tavaly?

Ez a lekérdezés azoknak a pácienseknek a listáját adja vissza, akik az előző évben legalább tíz alkalommal lemondták a foglalt időpontjaikat (status = 'cancelled\_by\_patient'). Az eredmény a páciensek egyedi TAJ-számával és nevükkel együtt jelenik meg, lehetőséget biztosítva a rendszer adminisztrátorainak a gyakori időpontlemondások elemzésére.

1. Mely Vizsgálatok hozták a legtöbb pénzt adott évben vagy hónapban?

Ez a lekérdezés az adott időszakban (évben vagy hónapban) legtöbb bevételt termelő vizsgálatokat és kezeléseket határozza meg. A SUM(t.price) segítségével az egyes kezelésekből származó teljes bevételt számolja össze, és az eredményt csökkenő sorrendben rendezi, így az első néhány sor a legnagyobb bevételt hozó kezeléseket mutatja.

1. Melyik páciens volt, aki a volt vizsgálaton, de nem értékelt?

Ez a lekérdezés azoknak a pácienseknek a listáját adja vissza, akik részt vettek egy vizsgálaton, de nem adtak értékelést (rating IS NULL). Az eredményben szerepel a páciens neve, elérhetősége, a kapott kezelés neve és az időpont dátuma, lehetőséget biztosítva arra, hogy az intézmény visszajelzést kérjen a páciensektől.

### Beépítendő megszorítások

* patients.taj\_number 9 számjegy hosszú kell, hogy legyen
* appointments.status csak előre meghatározott értékeket vehet fel
* appointments.rating csak 1 és 5 közötti szám lehet
* treatments.price > 0 (A kezelés ára nem lehet negatív vagy nulla)
* appointments.start\_time nem lehet múltbeli időpont az új foglalásoknál
* appointments.end\_time nagyobb kell, hogy legyen, mint appointments.start\_time
* users.last\_login nem lehet nagyobb, mint az aktuális dátum
* appointments.status alapértelmezett értéke 'scheduled'
* users.role\_id alapértelmezett értéke '3' (ha nincs megadva, akkor páciens lesz)
* appointments\_rating alapértelmezett értéke NULL, mivel a páciens nem ad rögtön értékelést
* Egy adott orvos (doctor\_id) egy adott időpontra (start\_time) csak olyan kezelést (treatment\_id) adhat ki, amely az ő szakterületéhez (specialization\_id) tartozik.
* Az orvos csak akkor tudjon regisztrálni, ha a doctor\_licence táblában található isUsed állapot false.

### Scriptek

Az adatbázis scriptjei gyorshivatkozásként láthatóak az alábbiakban.

Táblák scriptje:  
[privateclinic\_table.sql](https://szamalkszalezihu-my.sharepoint.com/:u:/g/personal/bordacs_bela_diak_szamalk-szalezi_hu/EfZ95R_QS5ZHoDiyxQn9dPQB941qmTIQwcL4b1sereHvcg?e=18bh0b)  
  
Példa adatok scriptje:  
[privateclinic\_teszt\_adatok.sql](https://szamalkszalezihu-my.sharepoint.com/:u:/g/personal/bordacs_bela_diak_szamalk-szalezi_hu/Ea9uKNG08n9IhE7qPtk5ilQBSpDgydALNMlHyDn27rbA-A?e=GAlwgM)

Hasznos lekérdezések scriptje:  
[privateclinic\_lekerdezesek.sql](https://szamalkszalezihu-my.sharepoint.com/:u:/g/personal/bordacs_bela_diak_szamalk-szalezi_hu/EZ7m75_9Y2VNkEOgASHskdsBhkEAqS0dZ8NbsdFwNIObug?e=y8KYNr)

## Backend

Az adatbázis megtervezése után a következő lépés a backend, tehát a háttérrendszer kialakítása. Ez felel az üzleti logika megvalósításáért és az adatok kezeléséért. A mai modern webalkalmazások nagyrésze ehhez API-kat használ, amelyek lehetővé teszik a frontend számára, hogy struktúrált módon, jellemzően JSON formátumban kommunikáljon a backenddel.

A szakdolgozatunk készítése során ilyen API végpontokat hoztunk létre, amelyhez Laravel keretrendszert használtunk. Ez a módszer az MVC (Model – View – Controller) tervezési mintát követi.

A Model kezeli az egyes erőforrásokkal kapcsolatos adatokat.  
A View réteg a megjelenésért, és a felület kinézetéért felelős. Emiatt erre bővebben a Frontend esetén fogunk kitérni.  
A Controller réteg köti össze a Modelt és Viewt. Így folyamatos kommunikációt végez a Model és View rétegekkel, azokat vezérelve.

Backend esetén a route-ok segítségével határoztuk meg, hogy az éppen aktuális kérés feldolgozásáért melyik Controller, és azon belül melyik metódus felelős.

#### Modellek és Controllerek

Modellek az alábbiakban láthatóak VSCode alapján:

A képen szöveg, képernyőkép, Betűtípus, tervezés látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.

Conrollerek megjelenése VSCode-ban:

A képen szöveg, képernyőkép, Betűtípus, tervezés látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.

#### API végpontok

Az API végpontok listáját a következőkben szétbontjuk a különböző jogosultsági körökre (middleware):  
  
Jogosultság nélkül elérhető végpontok:

|  |  |  |
| --- | --- | --- |
| Kérés Típusa | URI | LEÍRÁS |
| GET | /nav-items | Lekéri a navigációs elemeket |
| GET | /get-avg-ratings-by-doctors | Lekéri a doktorok átlag értékelését |
| GET | /treatments | Kilistázza az összes kezelést |
| GET | /specializations | Visszaadja az összes specializációt |
| GET | /checkLicenceById/{id} | Ellenőrzi, hogy az orvosi licensz létezik-e, vagy már használatban van, vagy szabad |
| GET | /get-available-appointments | Kilistázza a szabad időpontok listáját |

Páciens jogosultsággal rendelkező végpontok:

|  |  |  |
| --- | --- | --- |
| **Kérés Típusa** | **URI** | **LEÍRÁS** |
| GET | /get-appointments-by-doctor | Lekéri az adott orvoshoz tartozó időpontokat |
| GET | /get-treatments-by-specialization | Lekéri a specializációhoz tartozó kezeléseket |
| GET | /test-get-tbs | Az összes specializációhoz tartozó kezeléseket adja vissza |
| GET | /appointments | Kilistázza az összes időpontot |
| GET | /doctors-with-spec | Visszaadja a doktorokat és hozzájuk tartozó specializációkat |
| GET | /get-treatment/{id} | Lekéri az adott azonosítójú kezelést |
| GET | /user-data | A bejelentkezett user tábla összekapcsolásáért felel a páciens vagy orvos táblával, a jogosultság alapján |
| GET | /get-appointments-by-patients/{id} | Lekéri a megadott páciens időpontjait |
| PUT | /profile/rate/{id} | Értékelést ad az adott időponthoz |
| PUT | /booking-appointment/{id} | Lefoglal egy létező és foglalható időpontot |
| GET | /get-all-appointment-by-doctor | Lekéri az adott orvos összes időpontját |
| PUT | /cancel-appointment-by-patient/{id} | Az adott beteg törli az adott időpontfoglalását |
| PUT | /change-user-info/{id} | Módosítja a felhasználó adatait |
| POST | /file-upload | A profilkép feltöltéséért felel |

Orvos jogosultsággal rendelkező végpontok:

|  |  |  |
| --- | --- | --- |
| **Kérés Típusa** | **URI** | **LEÍRÁS** |
| POST | /create-appointments | Új időpontokat hoz létre orvos számára |
| GET | /get-all-patients-with-name | Lekéri az összes páciens nevét és egyéb adatait |
| GET | /get-patients-to-auth-doctor | Lekéri az autentikált orvoshoz tartozó betegeket |
| GET | /get-appointments-count | Lekéri az időpontok számát páciensenként |
| PUT | /appointment-delete-by-doctor/{id} | Az orvos törli az adott időpontot |
| PUT | /appointment-cancel-delete-by-doctor/{id} | Az orvos visszavonja az általa törölt időpontot |
| GET | /get-today-appointments/{id} | Lekéri a megadott orvos mai időpontjait |
| GET | /get-patient-data/{id} | Lekéri a megadott beteg részletes adatait |
| PUT | /finish-appointment/{id} | Az adott időpontot lezárja, befejezettnek jelöli |

Admin jogosultsággal rendelkező végpontok:

|  |  |  |
| --- | --- | --- |
| **Kérés Típusa** | **URI** | **LEÍRÁS** |
| GET | /roles | Kilistázza az összes szerepkört |
| PUT | /update-nav | Frissíti a navigációs elemek sorrendjét |
| GET | /get-nav-items-with-roles | Lekéri a navigációs elemeket a szerepkörökkel együtt |
| GET | /navs | Kilistázza az összes navigációs elemet |
| POST | /add-nav-to-role | Hozzáad egy navigációs elemet a szerepkörhöz |
| POST | /check-nav-assigned-to-role | Ellenőrzi, hogy adott szerepkörhöz már van-e rendelve ilyen navigációs elem |
| DELETE | /remove-nav-from-role/{id} | Eltávolítja a navigációs elemet a szerepkörből |
| GET | /users | Kilistázza a felhasználókat és azok szerepköreit |
| PUT | /update-user-role/{id} | Frissíti a felhasználó szerepkörét |

#### Authentikáció / Hitelesítés

A projektünk során használunk gyárilag beépített végpontokat is a hitelesítéshez, jelszó visszaállításhoz, regisztrációt követő verifikációs email küldéséhez is:

|  |  |  |
| --- | --- | --- |
| **Kérés Típusa** | **URI** | **LEÍRÁS** |
| POST | /register | Regisztrál egy új felhasználót |
| POST | /login | Bejelentkezik egy felhasználó |
| POST | /forgot-password | Elküldi a jelszóhelyreállító linket a megadott email címre |
| POST | /reset-password | Beállítja az új jelszót a megadott adatokat használva |
| GET | /verify-email/{id}/{hash} | Ellenőrzi a felhasználó email címét a megadott azonosítóval |
| POST | /email/verification-notification | Új email megerősítő linket küld a felhasználónak |
| POST | /logout | Kijelentkezik a felhasználó |

#### Létrehozott Trigger

Az általunk létrehozott Trigger az időpont létrehozásának ütközéseit vizsgálja.  
Minden esetben az új időpont beszúrása előtt fut le, és amennyiben időpont ütközést talál, nem engedi beszúrni az új rekordot.   
Ehhez elsőként lekéri a kezelés hosszát percben egy adott ’treatment\_id’ alapján, majd kiszámolja az új időpont végét.   
Ezt követően lekérdezi, hogy az orvosnak van-e másik időpontja, ami ütközik a jelenlegivel (meglévő időpont eleje vagy vége beleesik az új időpontba, illetve a meglévő teljesen lefedi-e az új időpontot).   
Ha talál ütközést, hibaüzenetet küld, és megakadályozza a beszúrást.

A képen szöveg, képernyőkép, menü, Betűtípus látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.

#### A Triggerhez tartozó időpont létrehozás

A DoctorAppointmentController.php fájlban található metódus végzi az időpontok létrehozását.  
Elsőként megtörténik a bemenetek validációja. Ezt követően lekéri a bejelentkezett orvos azonosítóját, majd megkeresi az adatbázisban a megadott nevű kezelést (amennyiben nem találja 404-es hibakóddal ’Kezelés nem található’ üzenetet kapunk).  
Megtörténik a kezdő és záró időpontok Carbon objektumokká alakítása. A kezelések hosszát percekben átszámolja, majd ciklus segítségével időpontokat generál, a végükön 10 perc szünetekkel. A ciklus addig fut, amíg a kezdő időpont + kezelés hossza túl nem lépi a megadott végidőt. Ha az utolsó elem túlnyúlna, eltávolítja a hozzáadandó tömbből. Végül beszúrja az összes időpontot az adatbázisba, és visszaad egy üzenetet 200-as kóddal ’Kezelések sikeresen létrehozva’ szöveggel.

A képen szöveg, képernyőkép, szoftver látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.

## Frontend

A háttérrendszer megvalósítását követően (backend), a következő lépés a felhasználói felület kialakítása, amelyet a frontend testesít meg. Ezen szakaszban az MVC-ből a View réteg kap fő szerepet. Célunk, hogy a backend által létrehozott adatok alapán a felhasználók számára esztétikus és megfelelő formában jelenjenek meg a kívánt adatok.

A frontend fejlesztéséhez React JavaScript könyvtárat használtunk, ami biztosította számunkra a dinamikus, komponens alapú felületek létrehozását. A backenddel való megfelelő szinkronizációt az Axios használatával értük el, ami az aszinkron http kérések optimális és hatékony kezelését biztosítja.

### Axios

Az Axioson keresztül történik a végpontok meghívása, amelyek az Api.php-ban találhatóak.

Létrehoztunk az Axios fájlban egy myAxios változót, amelyben eltároltuk a végpont elérési útját. A későbbiekben ezen myAxios változón keresztül tudjuk elérni a különböző Backend végpontokat.

Axios fájlunk az alábbi útvonalon érhető el:  
/src/api/Axios.jsx

### Contextek

A React beépített eszközei közé tartoznak.

A contextek globális állapotot biztosítanak a React alkalmazáson belül, tehát lehetővé teszik, hogy egy adatot a teljes alkalmazáson keresztül elérhetővé tegyük anélkül, hogy paramétereken keresztül minden szinten tovább kellene őket adni. A végpont hívások nagy része a contextekben kerül megfogalmazásra.

#### Auth Context

Ezen fájlban olyan állapotokat tárolunk, amik az autentikációhoz és a fő adatok backendről való lekérésére szolgál. Célja, hogy biztosítsa a hitelesítéssel, regisztrációval, és bejelentkezett felhasználó adataival kapcsolatos funkciókat.

Elérési útvonala:  
/src/contexts/AuthContext.jsx

Elérhető metódusok:

|  |  |  |
| --- | --- | --- |
| **Metódus neve** | **Végpont elérési útvonal** | **Leírás** |
| csrf | /sanctum/csrf-cookie | Lekéri a CSRF cookie-t a biztonságos űrlapküldéshez. |
| getUser | /api/user | Lekéri a bejelentkezett felhasználó alapadatait. |
| getUserData | /api/user-data | Részletes felhasználói adatokat kér le a felhasználói profilhoz. |
| login | /login | Bejelentkezteti a felhasználót az e-mail és jelszó alapján. |
| logout | /logout | Kijelentkezteti a felhasználót a rendszerből. |
| reg | /register | Regisztrálja az új felhasználót a megadott adatokkal. |
| fetchNavigation | /api/nav-items | Lekéri a szerepkör alapú navigációs elemeket. |
| fetchSpecializations | /api/specializations | Lekéri az elérhető orvosi szakterületeket. |
| fetchAVGDoctorsRatings | /api/get-avg-ratings-by-doctors | Visszaadja az orvosok értékelési átlagait. |
| fetchEmailStatus | /api/user/email-status | Ellenőrzi, hogy a felhasználó emailje hitelesítve van-e. |
| checkDoctorLicence | /api/checkLicenceById/{licence} | Lekérdezi, hogy egy adott orvosi licenc érvényes-e. |
| fetchUploadProfilePicture | /api/file-upload (GET) | Előkészíti a profilkép-feltöltéshez szükséges adatokat. |
| fetchHandleUpload | /api/file-upload (POST) | Feltölti a kiválasztott profilképet az adatbázisba. |
| changeUserInfo | /api/change-user-info/{id} | Módosítja a felhasználó adatait a megadott id alapján. |

A képen szöveg, képernyőkép, szoftver, Betűtípus látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.Jelen képen látható egy bejelentkezés. Lekérjük a CSRF tokent, amely elengedhetetlen a bejelentkezéshez. Meghívjuk myAxios POST-al a ’/login’ végpontot, átadva paraméternek az adattípust (autentikációs adatok).

Ennek eredményét elmentjük a response változóba. Meghívjuk késleltetve a getUser metódust, majd ha sikeresen megtörént a user lekérése. Ha jók voltak a hitelesítő adatok, bejelentkezteti a usert, és elnavigál a főoldalra. Egyéb esetben egy változóban eltároljuk a hibát.

#### Patient Context

Célja, hogy a páciensekhez kapcsolódó adatokat és műveleteket kezelje. Többek között biztosítja a kezelések és időpontok lekérdezését, a foglalások kezelését (lefoglalás és foglalás törlése), illetve a foglalásokhoz tartozó értékelést.

Elérési útvonala:  
/src/contexts/PatientContext.jsx

Elérhető metódusok:

|  |  |  |
| --- | --- | --- |
| **Metódus neve** | **Végpont elérési útvonal** | **Leírás** |
| fetchAppointments | /api/appointments | Lekéri az összes páciens időpontot, és dátum alapján csoportosítja őket. |
| fetchAvailableAppointments | /api/get-available-appointments?treatment\_id={id} | Visszaadja az elérhető időpontokat egy adott kezelés alapján. |
| fetchDoctorAppointments | /api/get-appointments-by-doctor?doctor\_id={id} | Lekéri az adott orvoshoz tartozó időpontokat. |
| groupAppointmentsByDate | – (belső segédfüggvény) | Csoportosítja az időpontokat dátum szerint. |
| fetchTreatmentsBySpecialization | /api/get-treatments-by-specialization | Lekéri a szakterületekhez tartozó kezeléseket. |
| fetchTreatments | /api/get-treatments | Lekéri az összes kezelés elérhető opcióit. |
| fetchPatientData | – (összetett metódus, több API-hívással) | Elindítja az összes szükséges adat betöltését. |
| bookingAppointment | /api/booking-appointment/{id} | Lefoglal egy kiválasztott időpontot a páciens részére. |
| fetchAppointmentRating | /api/profile/rate/{id} | Értékelést küld az adott időponthoz. |
| fetchCancelAppointmentByPatient | /api/cancel-appointment-by-patient/{id} | Lemondja a páciens által foglalt időpontot. |

#### Doctor Context

Ezen Context biztosítja az orvosok számára a betegek és időpontok kezelését,

Elérési útvonala:  
/src/contexts/DoctorContext.jsx

Elérhető metódusok:

|  |  |  |
| --- | --- | --- |
| **Metódus neve** | **Végpont elérési útvonal** | **Leírás** |
| appointmentDeleteByDoctor | /api/appointment-delete-by-doctor/{id} | Törli az orvos által kezelt időpontot. |
| appointmentCancelDeleteByDoctor | /api/appointment-cancel-delete-by-doctor/{id} | Visszavonja az időpont törlését. |
| fetchAllPatients | /api/get-all-patients-with-name | Lekéri az összes beteget név, és egyéb adatok szerint. |
| fetchAppointmentByPatients | /api/get-appointments-by-patients/{pat\_id} | Lekéri a betegekhez tartozó időpontokat. |
| fetchAppointmentsCount | /api/get-appointments-count | Lekéri az elvégzett időpontok számát páciensenként csoportosítva. |
| fetchPatientsByAppointments | /api/get-patients-to-auth-doctor | Lekéri a bejelentkezett orvoshoz tartozó pácienseket. |
| fetchDoctorsWithSpec | /api/doctors-with-spec | Lekéri az orvosokat szakirány szerint. |
| fetchDoctorData | – (összetett metódus, több API-hívással) | Lekéri a doktor adatait bejelentkezést követően. |
| fetchAppontmentsByDoctor | /api/get-appointments-by-doctor | Lekéri az orvoshoz tartozó foglalható időpontokat. |
| fetchAllAppontmentsByDoctor | /api/get-all-appointment-by-doctor | Lekéri az összes időpontot egy orvoshoz. |
| fetchGetTodayAppointments | /api/get-today-appointments/{id} | Lekéri az orvos számára a mai napra vonatkozó időpontokat. |
| fetchGetPatientData | /api/get-patient-data/{id} | Lekéri a beteg adatokat. |
| fetchFinishAppointment | /api/finish-appointment/{id} | Lezárja az időpontot egy leírás hozzáadásával. |

#### Admin Context

Ezen Context biztosítja az Admin jogosultsággal rendelkező felhasználók számára a navigációs elemek, felhasználók és szerepkörök kezelését, valamint admin joghoz tartozó alap adatok lekérdezését.

Elérési útvonala:  
/src/contexts/AdminContext.jsx

Elérhető metódusok:

|  |  |  |
| --- | --- | --- |
| **Metódus neve** | **Végpont elérési útvonal** | **Leírás** |
| getUsers | /api/users | Lekéri az összes felhasználó adatát. |
| fetchNavRoleInfo | /api/get-nav-items-with-roles | Lekéri a navigációs elemeket és azok szerepköreit. |
| fetchAdminData | /api/roles, /api/navs, /api/users, /api/get-nav-items-with-roles | Lekéri az adminisztrációs adatokat, beleértve a szerepköröket, navigációs elemeket és felhasználókat. |
| setFilteredUsersList | – (belső segédfüggvény) | Beállítja a szűrt felhasználók listáját. |

#### Contextek összekapcsolása

Az oldalak betöltésére a React useEffect-et használ, amelyet lehet paraméterezni, és ezen paraméter a frissítés gyakoriságát szolgálja.

A képen szöveg, képernyőkép, szoftver látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.Sorrendben haladva a képen látható useEffektek esetén az első kimenti a session-ből, hogy van-e bejelentkezett felhasználó. Amennyiben van, újra lekérjük az adatait. Ez a useEffect egyszer fut le az oldal betöltésekor.

A második arra szolgál, hogy ha van felhasználó, de nincs megerősítve az e-mail, akkor átnavigáljon az e-mail megerősítő oldalra. Ez akkor fut le, ha változik az ’isVerified’ változó.

A harmadik useEffect a jogosultságok vizsgálatára szolgál. Ha van user, és annak szerepköre 3 vagy az alatti, illetve megerősített e-mail címe van, akkor lefutnak a pácienshez tartozó metódusok. Ezt követően hasonló vizsgálattal megnézzük, hogy a szerepkör 2 vagy az alatti, majd az adott orvos szerepkörhöz tartozó metódusok futnak le. Végül ellenőrizzük, hogy a szerepekör értéke 1, és betölt az admin jogosultsághoz tarotozó metódus készlet.

Ha van user, de nincs megerősítve, akkor átnavigáljuk az e-mail visszaigazoló oldalra.

Amennyiben van user, de a szerepköre 4, akkor átnavigáljuk az inaktivált felhasználók oldalra. Egyéb esetben töltse be a mindenki számára elérhető funkciókat.

Ez a user változó változásakor fut le minden alkalommal.

### Routolás és navigáció

A navigáció az adatbázisból kerül betöltésre. Ennek célja, hogy szerepkörönként dinamikusan szabályozhassuk az elérhető menüpontok listáját.

A React Router DOM egy olyan extension, amely lehetővé teszi a React alapú weboldalon az aloldalak kezelését anélkül, hogy az egész oldal újratöltődésére lenne szükség. Segítségével gyorsabb az oldal betöltődése, valamint kevesebb az adatmozgás. Ezen keresztül lehetséges többek között a navigáció létrehozása.

A képen szöveg, képernyőkép, képernyő, szoftver látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.A képen látható VendegLayout kezeli a gyermek komponenseket. A route-ok között az ’index’ a Föoldal töltődik be alapértelmezetten. Néhányat manuálisan adtunk meg, a többi pedig backend oldalról töltünk be. Ehhez segítségül szükséges volt egy ComponentsMap.jsx fájl létrehozása, melyben egy lista található a komponensek neveivel. Ezeket a neveket használjuk fel a dinamikus route feltöltésekhez.

A képen szöveg, képernyőkép látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.Ezen a képen látható, hogy a backendtől kapott listán megyünk végig, majd egyesével betöltjük az egyesével kapott navigációs elemeket.

### Reszponzivitás

A stílusok kialakításához Bootstrap keretrendszert használtuk, ami segített a reszponzív és gyorsan kialakítható kinézet létrehozásában.

# Implementáció

## Program struktúra

## Backend

## Frontend

# Tesztelés

## Adatbázis

## Backend

## Frontend

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Az összes teszteset** | **Tesztelési egység** | | | |
|  | | | |
| Teszteset / Bemeneti paraméterek | Várt eredmény | Kapott eredmény | Státusz |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## Adatbázis

# Fejlesztési lehetőségek

# Irodalomjegyzék