

WEBTECH 2. BEADANDÓ

Bartók-Balogh Gábor(QVTQO8)



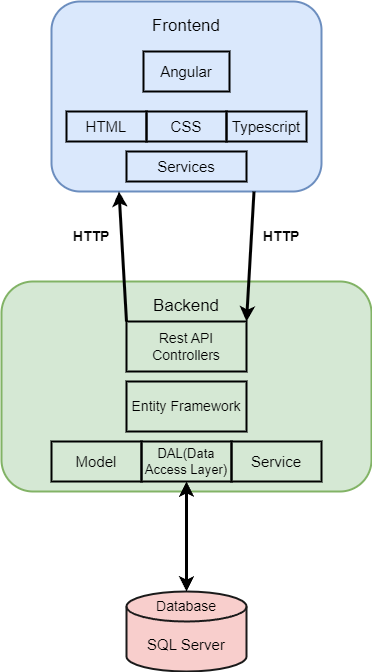
2023. január 1.

**Téma**

Webtechnológia 2. beadandómnak a szakdolgozatomat választottam.

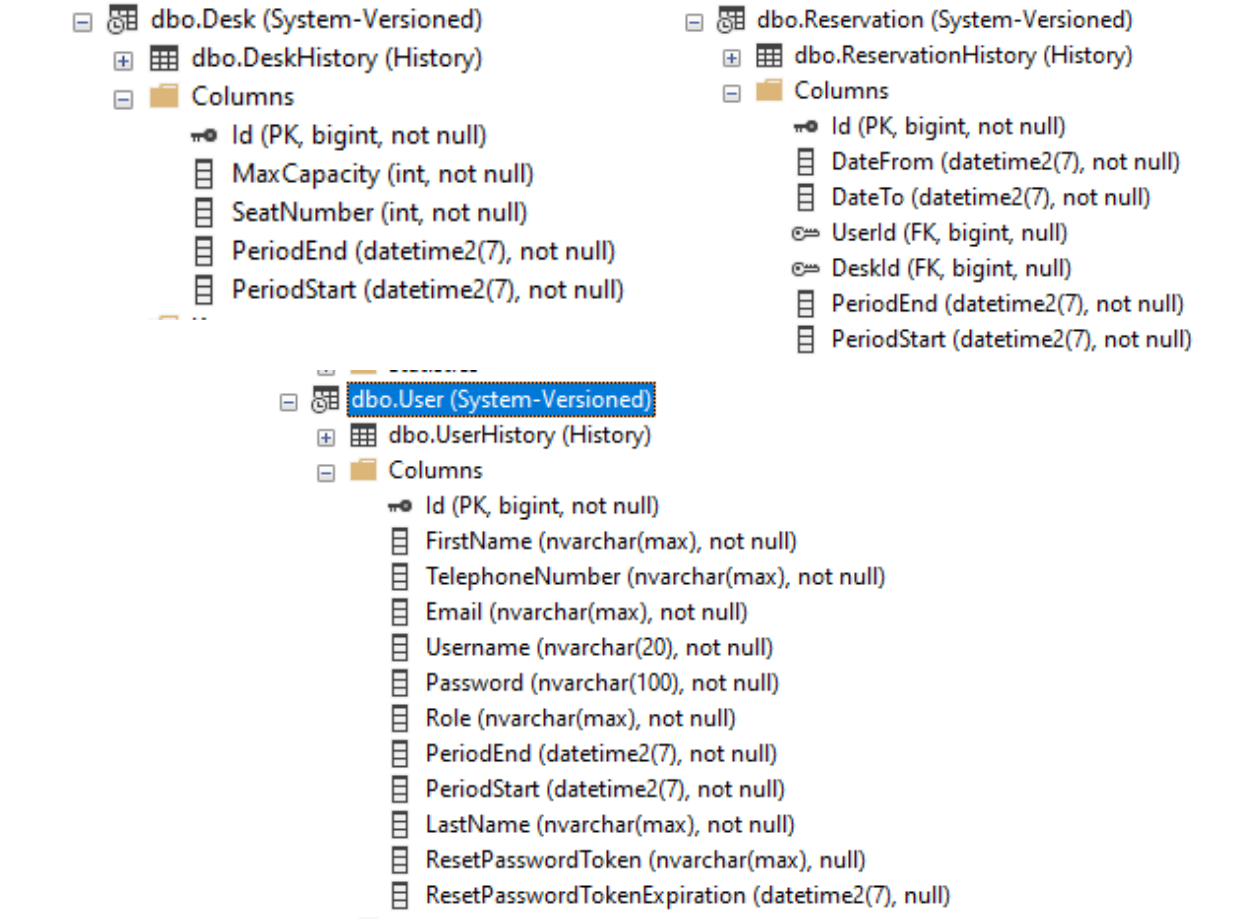
A SeatManagement rendszer egy háromrétegű architektúrát használ, amely magában foglalja a frontendet, a backendet és az adatbázist. Az alkalmazás egy komplex rendszer, amely lehetővé teszi a helyfoglalások kezelését és megjelenítését, különböző komponensei és algoritmusai biztosítják a hatékony működést és a felhasználók által elvárt funkcionalitást.

A következő részekben nézzük át, hogy hogyan is épül fel.



**Adatbázis kialakítása**

Az adatbázis létrehozása a már említett Entity Framework segítségével lett megvalósítva, 3 tábla került megalkotásra ami a felhasználó(User), egység(Desk) és a foglalást(Reservation) modellezi nekünk. A kódban látható adat annotációk segítenek az adatmodellekben lévő tulajdonságokra vonatkozó különböző korlátozások, szabályok és viselkedés meghatározásában.

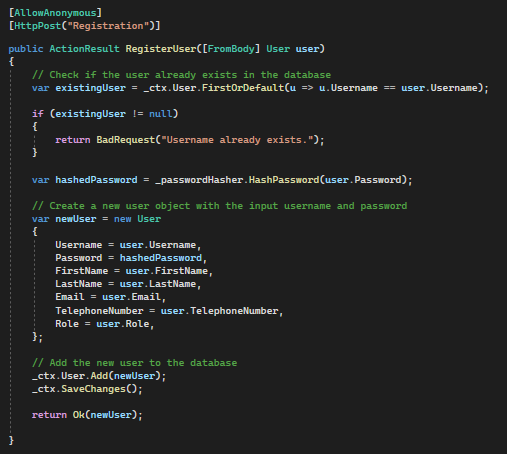
A migrálásnál a táblák temporal table-ként vannak kiexportálva az adatbázisban lévő adatok könnyebb verzió követése és módosítása.

**Backend**

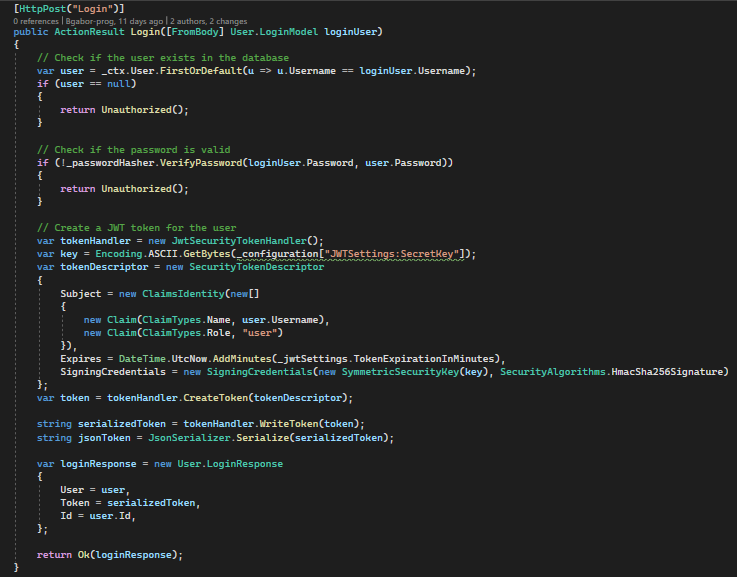
A már említett Backend lesz a felelős számunkra, ami tartani fogja a kapcsolatot a frontend-el Api végpontokon keresztül. Vegyük szemügyre a fontosabb végpontokat amelyek szükségesek a foglalások és felhasználó kezelés szempontjából.

**Regisztráció**

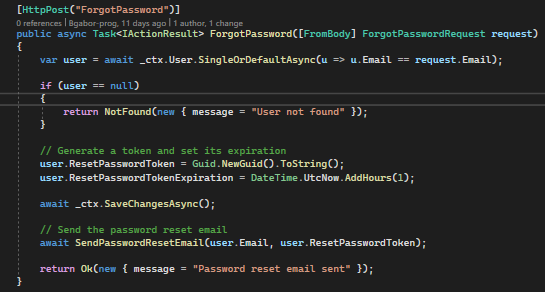
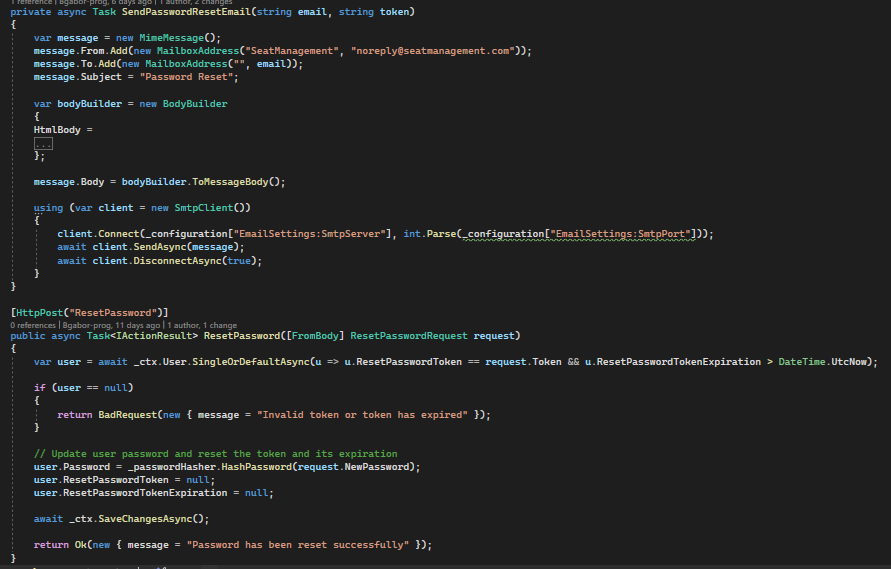
A regisztráció API végpontja egy HttpPost kérést fogad amely megvizsgálja, hogy a felhasználó név létezik e már az adatbázisban egy LinQ lekérés segítségével, majd ennek megfelelően elvégzi a szükséges műveletet, hozzáadja vagy BadRequestet küld vissza a Frontend számára.

****

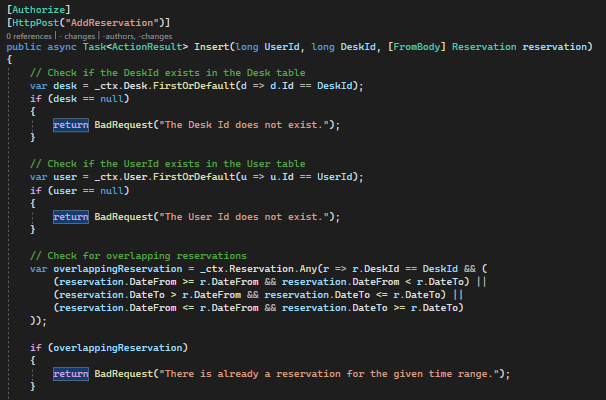
**Bejelentkezés**

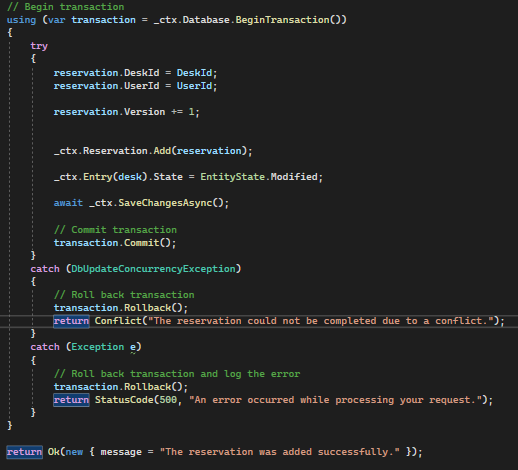
****A bejelentkezés végpontja jelentkezteti be a felhasználót a rendszerbe. Itt már több dolog is történik de nézzük a legérdekesebbeket. A felhasználó által megadott adatok ellenőrzés alá kerülnek, megnézzük, hogy létezik e a rendszerben a megadott felhasználónév.

**Jelszó visszaállítás**

A felhasználó kezelésnél elengedhetetlen a jelszó visszaállításnak a lehetősége, ezért erre is született egy megoldás, igaz csak egy "hamis" SMTP levelezési rendszer került kialakításra így hivatalosan nem küld valódi levelet az adott email címre, azonban lokális használatnál bőven megfelelt a rendszer tesztelésének céljából. Ezt pedig a Papercut SMTP server segítségével végeztem.

**Foglalás bevitel**

****Érdemes megnézni,hogy amikor beviszünk egy foglalást a rendszerbe az időpontokat le kell kezelni, azaz nem lehet olyan foglalást teljesíteni aminek időintervalluma megegyezik más foglalás idő tartományával. Ezért is mielőtt egy foglalás teljesülne átmegy egy dátum tartomány ellenőrzésen amely megvizsgálja, hogy van e bármilyen olyan foglalás a kiválasztott egységhez amelynek időintervalluma egybe esik a foglaláshoz megadott intervallummal.

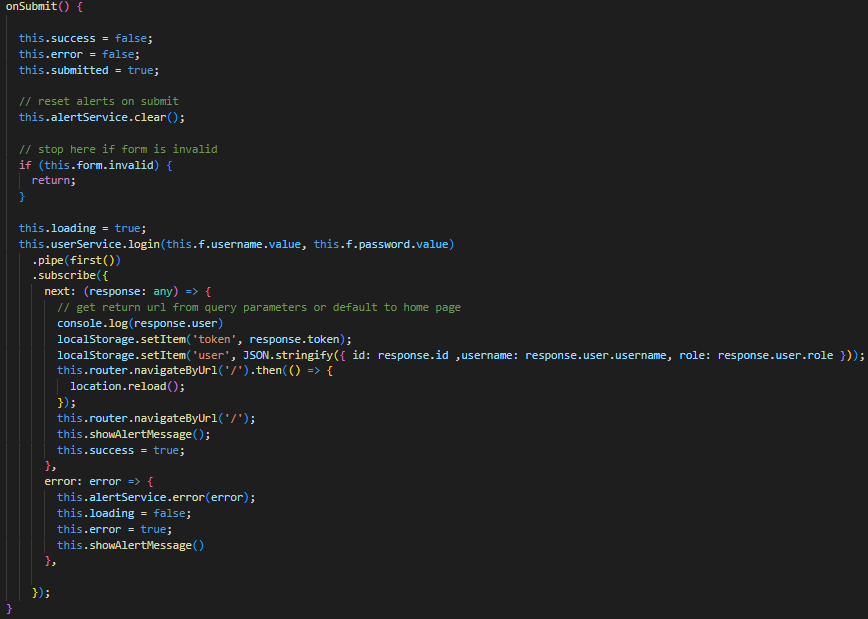
****

**Frontend**

Tekintsük át a Frontend kialakításának és megvalósításának fontosabb részeit amelyek lehetővé teszik a foglalások és egyéb funkciók működését.

**Bejelentkezés és Regisztráció komponens**

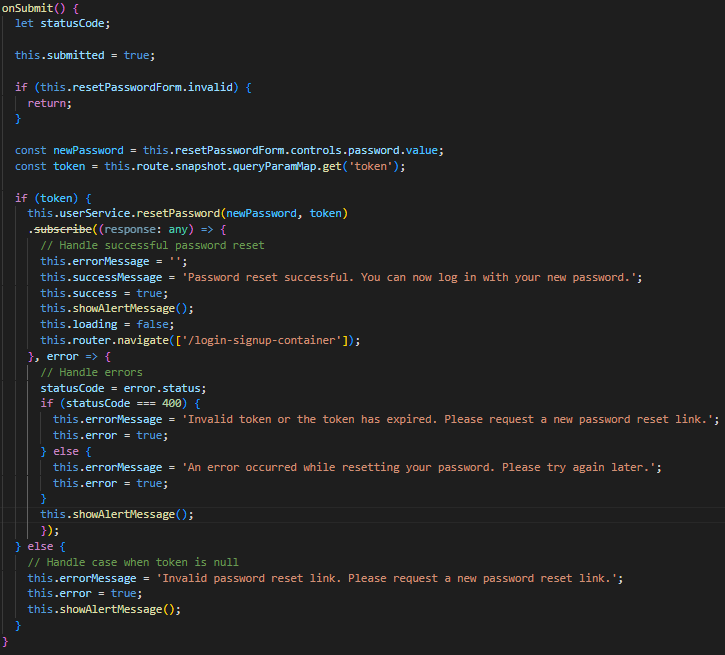
Ezen két komponens fogja nekünk elvégezni a felhasználók felvételét és hitelesítését a rendszerünkben egy UserService segítségével.

A kód részletben látható, hogy a UserService login függvényét használva a subscribe metód ussal feliratkozunk a kérésre amit a végpontnak küldünk majd annak válasza szerint eljárunk. Sikeresség esetén bejelentkeztetjük a felhasználót elmentve a JWT token-t és azokat az adatokat a felhasználóról amire szükségünk van az alkalmazásban. A regisztráció hasonló módon működik azzal a különbséggel, hogy ott nincs szükség felhasználói adatok mentésére. Ezeket a backend végzi el.

**Elfelejtett-jelszó komponens**

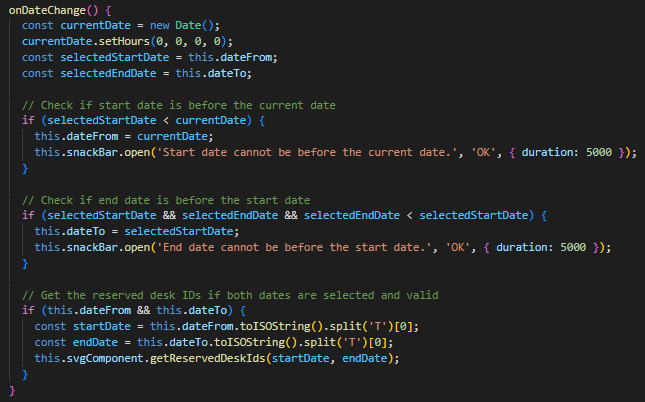
A felhasználó a bejelentkezés fülnél tud jelszó visszaállítást kérni. Létező email cím alapján amely az adatbázisba regisztrálva lett kiküld egy levelet amely tartalmaz egy linket.

Az elfejtett jelszónál a kiküldött emailben található link tartalmaz egy token-t amit a backend generál. Ez a token teszi lehetővé nekünk, hogy a felhasználó jelszót tudjon változtatni. Ez a token egyszer használatos. Ha felhasználó jelszót váltott nincs több lehetősége a változtatásra csak új kérés esetén.

****

**Foglalás komponens**

Nézzük át, hogyan is működik a foglalás komponens. Itt található az a panel amelyen a foglalást le lehet bonyolítani. Kettő input mező található amely az egység azonosítóját és a bejelentkezett felhasználó azonosítóját jelzi, ezek mind rendszer által vezérelt mezők. Az időintervallum kiválasztás is itt történik. A dátumok kiválasztását a MatDatePicker komponens biztosítja melyet az '@angular/material' könyvtár biztosít. Ezek segítségével megadható az az intervallum amiben leakarjuk foglalni az egységet. Minden dátum változtatásnál ellenőrizzük, hogy

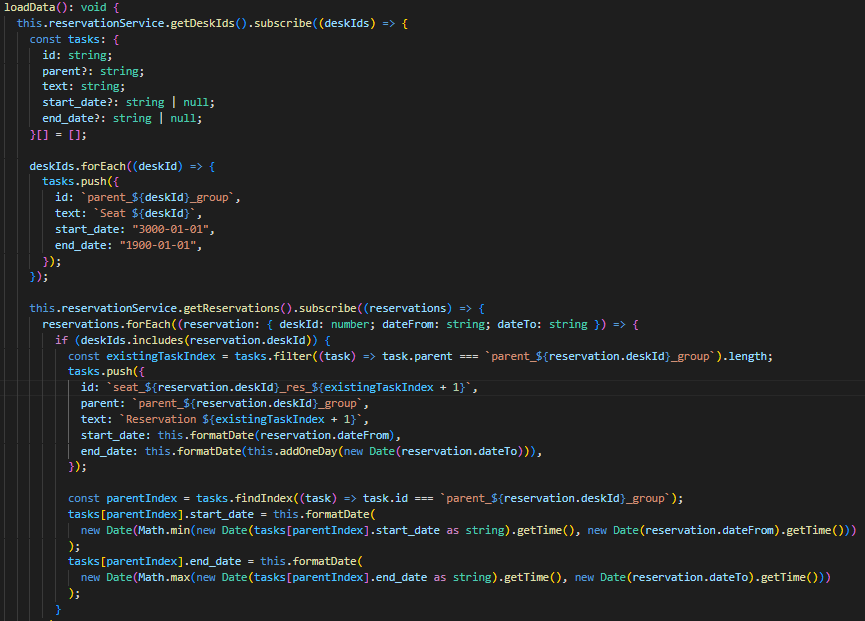
a kezdő időpont ne lehessen a végdátum után és a kezdő időpont se lehessen a mai dátum előtt. 

**Naptár nézet**

A naptár nézet megalkotására a már említett DHTMLX-Gantt könyvtár lett használva. Ezt alaposan átszabva, hogy a megfelelő megjelenítést adja lett kialakítva. Állítható benne a nézet(napi,heti,havi) valamint tudunk váltani az adott nézetben, hogy az előző vagy a következő periódust lássuk lásd.

Maga a diagram úgy működik, hogy egy DeskService segítségével lekér minden egységet az adatbázisból ezeket pedig task tömbként menti majd pedig lekérjük a foglalt egységeket a ReservationService segítségével.

Ha egy foglalt asztal azonosítója szerepel a foglalható asztalok azonosítói között, hozzáad egy új objektumot a tasks tömbhöz, amely tartalmazza a foglalás adatait és a szülő asztal azonosítóját.

****

****

**Összefoglalás**

Az izgalmasabb részei a projektnek bemutatásra kerültek. Azonban még így is számtalan részt lehetne bemutatásra adni viszont a forrás kódban minden elérhető.