Eötvös Loránd Tudományegyetem

Informatikai Kar

Programozáselmélet és Szoftvertechnológiai Tanszék

Shalendar

Témavezető:
Pintér Balázs
egyetemi adjunktus, PhD

Szerző:
Kertész János
programtervező informatikus BSc

Budapest, 2025

EÖTVÖS LORÁND TUDOMÁNYEGYETEM
INFORMATIKAI KAR

SZAKDOLGOZAT TÉMABEJELENTŐ

Hallgató adatai:
Név: Kertész János
Neptun kód: AM2VZ8

Képzési adatok:
Szak: programtervező informatikus, alapképzés (BA/BSc/BProf)
Tagozat : Nappali
Belső témavezetővel rendelkezem

Témavezető neve: Pintér Balázs
munkahelyének neve, tanszéke: **ELTE IK, Programozáselmélet és Szoftvertechnológia Tanszék**
munkahelyének címe: **1117, Budapest, Pázmány Péter sétány 1/C.**
beosztás és iskolai végzettsége: **egyetemi adjunktus, PhD**

A szakdolgozat címe: Shalendar

A szakdolgozat témája:
(A témavezetővel konzultálva adja meg 1/2 - 1 oldal terjedelemben szakdolgozat témájának leírását)

A dolgozat témája egy mindennapi életben használható time management szoftver megvalósítása. A projekt felépítése három fő felületre és egy bejelentkezési oldalra oszlik, az utóbbi lehetővé teszi több felhasználó számára a profil kezelését és az egymás közötti naptár megosztást.

Főoldal:
Az oldalon egy naptár található, mellette pedig minimum egy oszlop, amely a feladatkezelő szoftverekből ismert lista formátumot követi. A felhasználó igényei szerint több, saját tematikáinak megfelelő oszlopot is felvehet. Az oszlopban kártyák helyezhetők el, amelyeket a felhasználó szabadon hozzáadhat vagy törölhet. A kártyákon kötelezően megadható cím, valamint opcionálisan kezdeti dátumok, határidők és prioritások. Ezek a kártyák a naptár megfelelő napjaira húzhatók, így segítve a feladatok ütemezését.

Napi nézet:
A naptár adott napjára kattintva megjelenik a nap részletezése. Itt két lista található: Az egyik lista egy időjelző sávval rendelkező feladatlista, amelyben a naphoz tartozó, időponthoz kötött feladatok jelennek meg. A másik lista olyan teendőket tartalmaz, amelyek nem kötöttek időponthoz. Mindkét listában a feladatok „elvégzettnek” jelölhetők.

Több naptár kezelése:
A naptár mellett található egy plusz gomb, amely lehetővé teszi több naptár létrehozását és kezelését. Ezekhez a naptárakhoz további felhasználók is hozzáadhatók, valamint lehetőség nyílik a naptárakból egyes kártyák vagy teljes naptárak importálására a saját naptárba.

Budapest, 2024. 10. 05.

Tartalomjegyzék

1. fejezet

Bevezetés

2. fejezet

Felhasználói dokumentáció

3. fejezet

Fejlesztői dokumentáció - Megoldási terv

3.1. Rendszer architektúrája

Az alkalmazás a kliens-szerver modellt követve 3 fő komponensből áll. A Microsoft SQL Server adatbázis, az ASP.Net WebAPI alapú backend és a Vue.JS-alapú frontend. Az első két komponens a szervert, míg a harmadik a klienst képviseli.

A 3 komponenes tisztán szétválasztható, ezzel biztosítva a moduláris fejlesztést. A frontend és a backend API hívások segítségével kommunikál. A backend és az adatbázis pedig közvetlen kapcsolatban állnak az Entity Framework-nek köszönhetően.

3.1.1. Alrendszerek és rétegek szerepei és felelősségei

Frontend

Feladata a felhasználói interakciók kezelése valamint az üzleti logika megjelenítése. Az üzleti logika elérésére axios HTTP kéréseken keresztül történik. A kérés headerjébe automatikusan integrálja az autentikációhoz szükséges adatokat (JWT token) valamint a naptár azonosítóját amikor az releváns. Ezzel segítve, hogy a felhasználó csak számára elérhető adatokhoz férhessen hozzá.

Backend API

Felelősségei közé tartozik, hogy a JWT token megfelelően generálva legyen a bejelentkezés során, tartalmazza a felhasználó azonosítóját, email címét, JWT ID-ját (egyedi, véletlenszerű GUID) valamint az user naptárakhoz való engedélyeit a token megszokott jellemzői mellett. (Issuer, Audience, Expiration...)

Feladata továbbá, hogy kezelje a kliens által küldött HTTP kéréseket. Ahol szükséges ellenőriznie, hogy a felhasználó rendelkezik-e érvényes tokennel, valamint a naptárakkal kapcsolatos tevékenységes esetén figyelje az írási, olvasási vagy tulajdonosi engedélyek meglétét. A token ellenőrzését a beépített [Authorize] attribútummal végzi. Az engedély ellenőrzés pedig a tokenben szereplő engedélyek és a fejlécben kapott naptár azonosító összehasonlításával történik.

Amikor megtörtént az adatok validálása és az üzleti logika végrehajtása a backend feladata, hogy értesítse az összes klienst a változásról amely az adott naptár valamelyik nézetén tartózkodik .

Adatbázis

Az adatbázis szerepe, hogy hosszútávon, jól struktúráltan tárolják az adatokat a felhasználókról valamint azok naptárairól, engedélyeiről, kártyáiról.

Indexek, kulcsok és idegen kulcsok valamint megszorítások segítségével biztosítsja a következetességet és teljesítményt. Ezek tényleges kapcsolatát a *3.3. Adatbázis modell* című pont alatt tárgyaljuk.

3.1.2. Alkalmazott technológiák és eszközök

Programozási nyelvek, keretrendszerek

- **.NET 8.0 SDK** – A backend teljes projektje .NET 8.0 épül.
- **C#** – backend logika és API implementáció.
- **ASP.NET Core** – REST API keretrendszer.
- **Entity Framework Core** – ORM a relációs adatbázis kezelésére.
- **JavaScript** – frontend logika.
- **Vue.js** – JavaScript keretrendszer.

- **HTML, CSS** – struktúra és stílus.
- **Vite** – frontend build és hot-reload.

Fejlesztői eszközök

- **Visual Studio 2022** – backend fejlesztés, debug, teszt.
- **Visual Studio Code** – frontend fejlesztés, Vue komponensek.
- **Node.js** – frontend futtatási környezet.
- **Postman** – API-k kipróbálásához és manuális teszteléshez.
- **Git + GitHub** – verziókövetés.
- **SSMS** – SQL szerver kezelése, tesztelés, queryk írása.

Külső csomagok (NuGet és npm)

A projekt során több külső könyvtárat használtam, melyeket NuGet illetve npm segítségével kezeltem.

Backend (NuGet csomagok):

- **Microsoft.AspNetCore.Authentication.JwtBearer** (v8.0.0) – JWT tokenek feldolgozásához és hitelesítéshez.
- **Microsoft.AspNetCore.SignalR** (v1.2.0) – kliens oldali értesítések a backendből SignalR hubok felhasználásával
- **Microsoft.EntityFrameworkCore** (v9.0.2) – ORM réteg relációs adatbázisokhoz.
- **Microsoft.EntityFrameworkCore.SqlServer** (v9.0.2) – SQL Server-specifikus EF Core provider.
- **Microsoft.EntityFrameworkCore.Tools** (v8.0.11) – EF Core migrációs és scaffold eszközök; csak fejlesztési célra használva.
- **Microsoft.VisualStudio.Web.CodeGeneration.Design** (v8.0.7) – scaffold eszközök a WebAPI fejlesztéshez.

- **Swashbuckle.AspNetCore** (v6.4.0) – Swagger generálása és dokumentáció REST API-hoz.
- **System.IdentityModel.Tokens.Jwt** (v8.5.0) – JWT tokenek létrehozása és kezelése.

Frontend (npm csomagok):

- **vue** (v3.5.13) – A Vue.js 3 keretrendszer magja.
- **vue-router** (v4.5.0) – Oldalak közötti navigáció Vue-ban.
- **axios** (v1.7.9) – HTTP kliens API hívásokhoz.
- **@microsoft/signalr** (v8.0.7) – SignalR kliens valós idejű frissítésekhez. A backendelből érkező értesítések feldolgozásáh
- **jwt-decode** (v4.0.0) – JWT tokenek tartalmának frontend oldali dekódolása.
- **lucide-vue-next** (v0.479.0) – Ikonkészlet Vue 3-hoz.
- **vuedraggable** (v4.1.0) – Drag-and-drop funkcionalitás Vue komponensekhez.

Fejlesztői függőségek:

- **vite** (v6.0.11) – Build és hot-reload rendszer Vue-hoz.
- **@vitejs/plugin-vue** (v5.2.1) – Vue támogatás Vite-hez.
- **vite-plugin-vue-devtools** (v7.7.1) – Vue fejlesztői eszközök bővítménye.

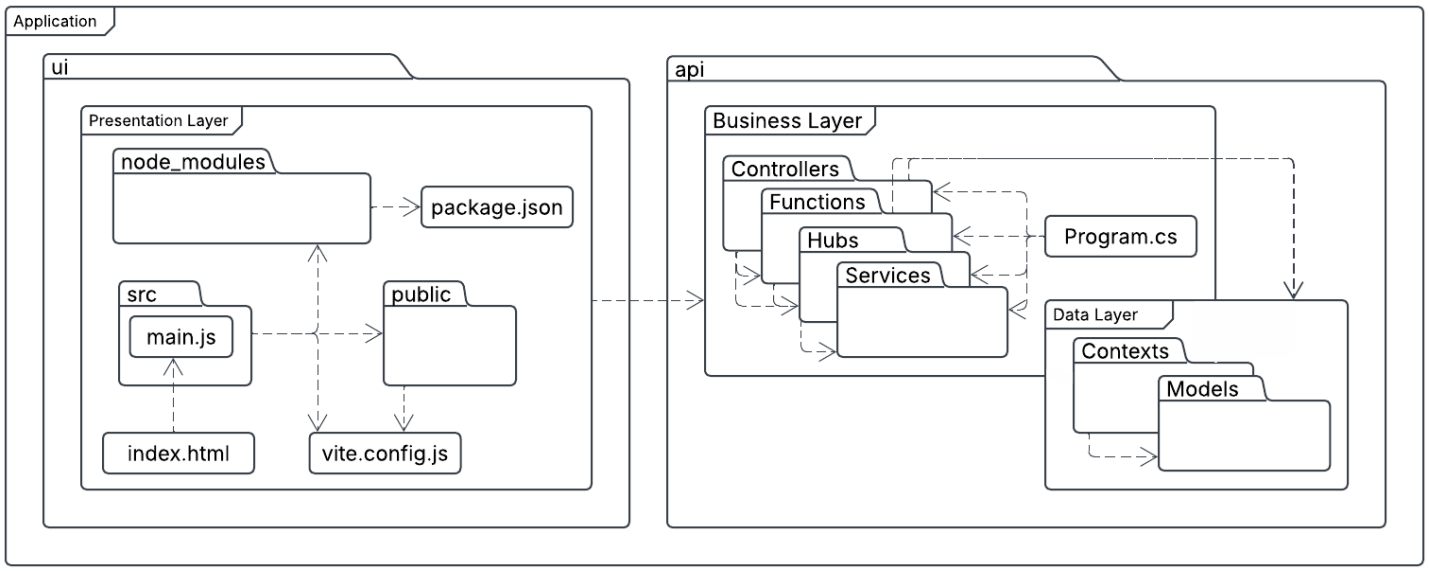
3.1.3. Fejlesztési módszertan

A fejlesztés során az első kitűzött cél egy működő MVP volt, csak funkcionalitást figyelve. Amint ez elkészült fokozatosan adtam hozzá funkciókat nézetek szerint csoportosítva. Amint a funkció elkészült manuális tesztelés, majd szükség esetén debugolás / refaktorálás után kezdtem a következő feladat implementálásába. A UI/UX dizájn megvalósításába a funkcionalitás 100%-os működése után kezdtem bele. Ezeket követte a unit majd integrációs tesztek megírása.

Ide még leírhatom hogy vettem fel issuekat gitHubon

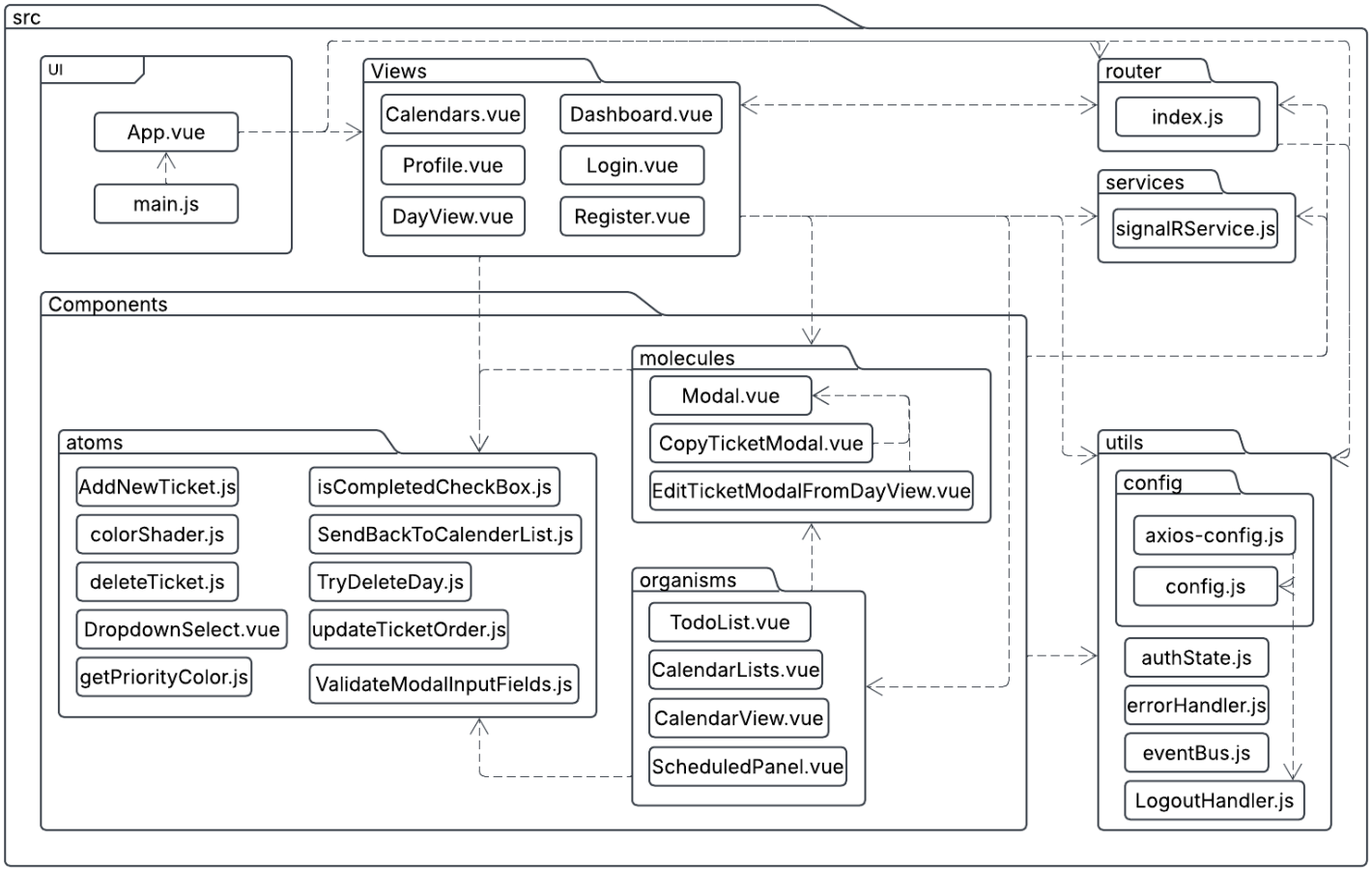
3.1.4. Csomag diagramok (UML)

A csomag diagrammokban amennyiben a mappa ikon látható, egy valós fizikai mappára utal, amennyiben a másik jelölés (a név a tárolón belül van, nem rajta) akkor pedig egy logikai egységről beszélünk. Minden egyéb jelölés valós fizikai fájlokat takar. A nyíl a importáló csomagtól közvetlenül az importált elemre mutat. Amennyiben nem egy elemre hanem egy logikai rétegre/mappára mutat az adott komponens mindegyike importálhatja azt.



3.1. ábra. Applikáció szintű csomag diagram

A program belépési pontja az index.html, amely a src/main.js mappáját importálja, a részletesebb megértés érdekében az alábbiakban látható az src mappa csomag diagramja is. Az api működése nem kerül bővebb kifejtésre ebben a pontban, mivel csomagok szempontjából annak működése jól leolvasható a fenti ábráról.



3.2. ábra. Src mappa csomag diagrammja

3.2. A rendszer működése

A működést 2 féle diagrammal mutatom be, az első hangsúlyt fektet a kommunikáció módjának bemutatására míg a másik a kommunikáció sorrendjét, valamint hibakezeléseket hivatott szemléltetni.

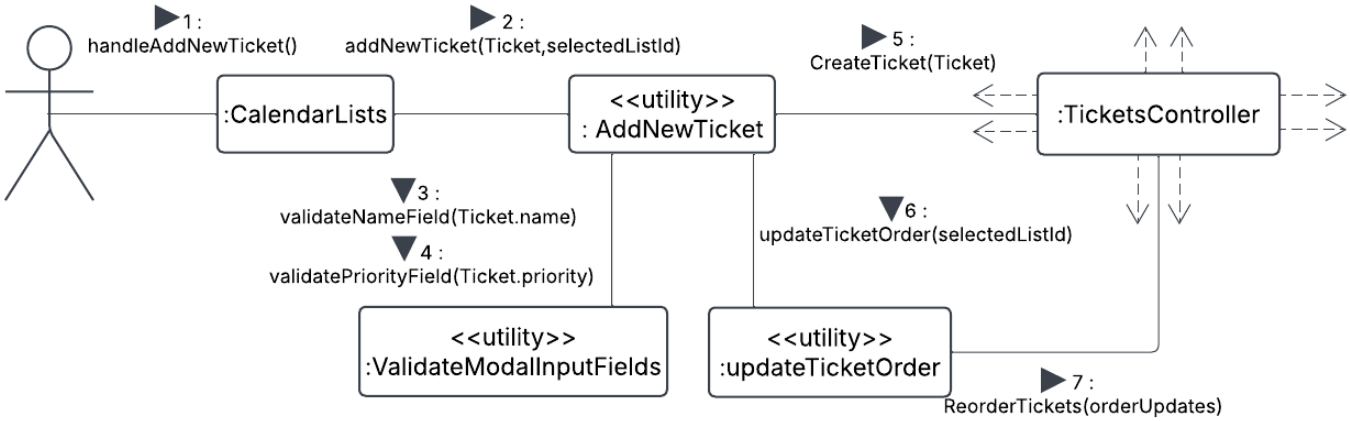
A frontend Vue 3 keretrendszeren alapuló implementációt reprezentál, amely a komponenseket JavaScript objektumokként valósítja meg. Mivel ez általában névtelen objektumokat generál ezért a komponensre annak a fájlnak a nevével fogunk hivatkozni, melyben található. A folyamatban részt vesznek fontos, objektumhoz nem kötött segédfüggvények is, melyeket «utility» sztereotípiával és a fent említett név adási konvenciókkal fogunk megjeleníteni a lent látható diagramokban.

A diagramok azt ábrázolják, hogy hogyan kezeli a program azt amikor a felhasználó egy új kártyát kíván felvenni valamelyik calendarList-be.

3.2.1. Kommunikációs diagram (UML)

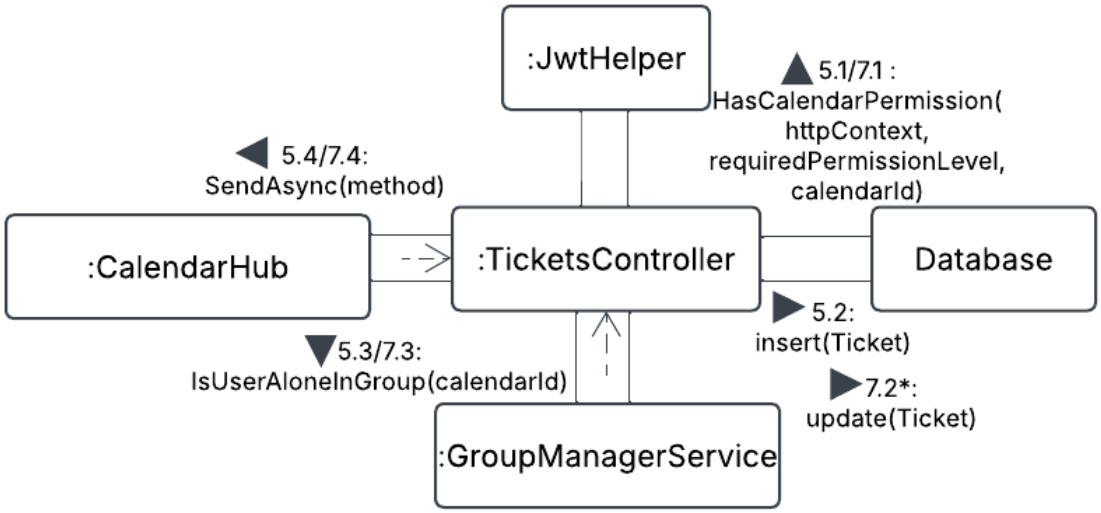
Az alábbi diagrammal bemutatok egy sikeres folyamat működését, a diagramnak hála jól láthatóak lesznek a kommunikációban részt vevő objektumok kapcsolatai és leolvasható lesz a kommunikáció sorrendje is.

A függvényhívások paraméterezése során csak a sikeres hívás, valamint az adat továbbítás szempontjából releváns információkat jelenítem meg, ezzel könnyítve a diagram átláthatóságát.



3.3. ábra. Kommunikációs diagram, kliens

A controllerek nagy része a lent látható logikát követve épül fel. A JwtHelper felel a naptárral kapcsolatos engedélyek ellenőrzéséért, a GroundManagerService valamint a CalendarHub pedig a kliens értesítéséről ha változás történik egy adott naptár azonosító szerinti csoportban.

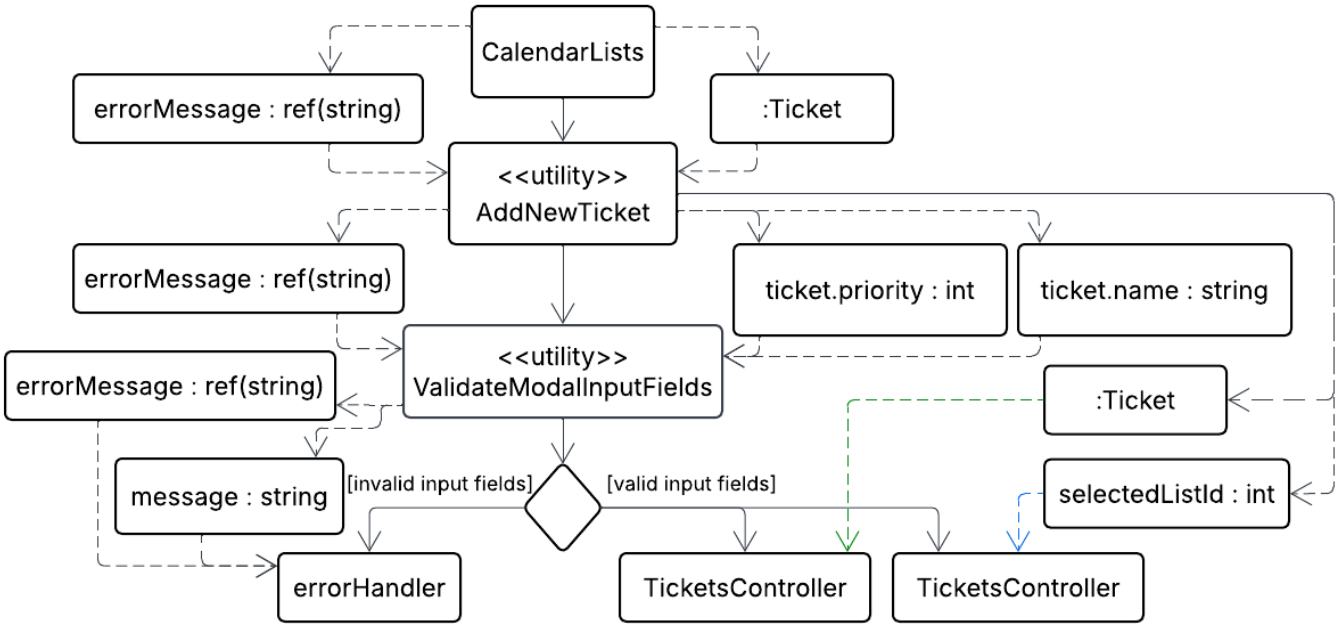


3.4. ábra. Kommunikációs diagram, szerver

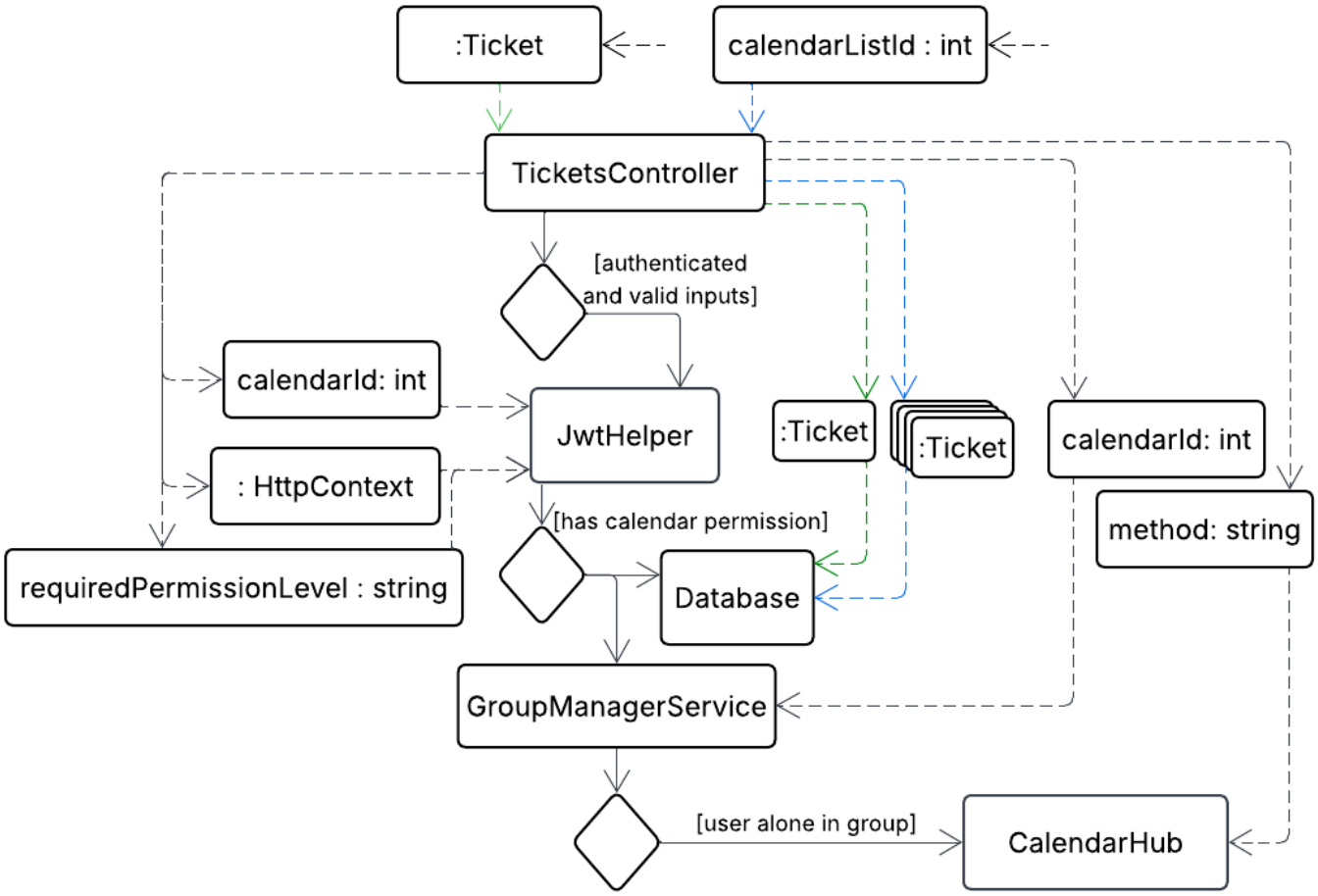
3.2.2. Tevékenységdiagram (UML)

Az alábbi diagramban ref(...) kulcsszó jelöli azon objektumokat melyek változása esetén frissül a felhasználói interfész is. Az olvashatóság érdekében nincs feltüntetve, de ilyen objektumok a calendarLists is, mely szintén frissül amikor hozzáadjuk a felhasználó által meghatározott kártyát.

Könnyebb követhetőség érdekében a TicketsController hívásokat kék és zöld színekkel jelöltem, hogy leolvasható legyen (a későbbi diagrammon is), hogy pontosan melyik tevékenység melyikhez tartozik.



3.5. ábra. Tevékenység diagram, kliens



3.6. ábra. Tevékenység diagram, szerver

3.3. Adatbázis modell

3.3.1. Az adatmodell áttekintése

ER-diagram vagy osztálydiagram a táblák és kapcsolatok bemutatására

3.3.2. Tábla-szintű leírás

Mezők: név, típus, kulcsok, indexek, alapértelmezett érték

3.3.3. Tárolt eljárások, triggerek, függvények

Amennyiben a rendszer tartalmaz ilyet

3.4. Modul- és osztályszerkezet

3.4.1. Backend modulok és rétegek

Controller, Service, Repository, Model felépítése

3.4.2. Osztálydiagram (UML)

Főbb osztályok és kapcsolataik (öröklés, asszociáció)

3.4.3. Főbb metódusok leírása

Módszerek bemenő paraméterei, kimenetei, működésük

3.5. A felhasználói felület terve

3.5.1. Képernyők listája és leírása

Pl. LoginView, CalendarView, DayView, TicketModal

3.5.2. Navigációs térkép

Képernyők közti kapcsolatok vizuális ábrázolása

3.5.3. Felhasználói események kezelése

Főbb interakciók leírása (pl. gombnyomás, drag-and-drop)

4. fejezet

Fejlesztői dokumentáció -Megvalósítás

4.1. Fejlesztés közbeni döntések

Adatábrázolás és struktúra

Felhasznált komponensek

Nyelvi eszközök alkalmazása és indoklása

4.2. Kiemelt kódrészletek

Hibakezelés, validálás, adatlekérés példái

4.3. Komponens terv és telepítés

Fizikai komponensek (frontend, backend, adatbázis)

Telepítési folyamat leírása (lokálisan és szerveren)

5. fejezet

Fejlesztői dokumentáció - Tesztelés

5.1. Tesztelési stratégia

Mehet egy techstack is ide, nugetekkel mint implMo

Modultesztek és rendszertesztek

Fekete- és fehérdoboz tesztelés

5.2. Tesztesetek

Bemenet, elvárt kimenet, tesztleírás

5.3. Tapasztalatok és módosítások

Implementációs döntések változtatása tesztek alapján

5.4. Nagy adatmennyiség melletti viselkedés

Teljesítménytesztek és értékelés

5.5. Eredmények és hatékonyság elemzése

Optimalizációs szempontok, erőforráskezelés

5.6. Manuális frontend teszt

Bejelentkezési oldal funkciói

Regisztráció

Given	A felhasználó megnyitotta az alkalmazást
When	A regisztrációs lehetőséget választja
Then	A rendszer belépteti a főoldalra, ahol láthatja a naptárát és a feladatlistáit.

Bejelentkezés

Given	A felhasználó megnyitotta az alkalmazást
When	A bejelentkezés lehetőséget választja, majd bejelentkezik
Then	A rendszer elvégzi a regisztrációt, majd a bejelentkezési oldalra irányítja a felhasználót.

Főoldal funkciói

Új feladatlista létrehozása

Given	A felhasználó a főoldalon van
When	A feladatlista hozzáadása opciót választja (+ gomb a feladatlisták felett) majd megfelelően paraméterezi a megnyíló modalt
Then	A feladatlista megjelenik a főoldalon.

Feladatlista módosítása

Given	A felhasználó a főoldalon van
When	Az adott feladatlistán a manage opciót választja, majd a felnyíló modalt megfelelően paraméterezi
Then	A feladatlista valamint a hozzá tartozó kártyák megváltoznak (a kártyák csak akkor ha változott a feladatlista színe).

Feladatlista törlése

Given	A felhasználó a főoldalon van
When	Az adott feladatlistán a manage opciót választja, majd a felnyíló modalon a törlés gombra nyom
Then	A feladatlista valamint a hozzá tartozó kártyák törlődnek.

Kártya létrehozása az adott feladatlistához

Given	A felhasználó a főoldalon van
When	A feladatlistán új kártya létrehozására kattint, majd megfelelően paraméterezi azt.
Then	A kártya megjelenik a kívánt oszlopban.

Kártya törlése az adott feladatlistáról

Given	A felhasználó a főoldalon van
When	A kártyát kitörli
Then	A kártya törlődik a feladatlistáról

Kártya másolása saját naptárba

Given	A felhasználó a főoldalon van
When	A naptár másolása gombra kattint majd kiválasztja a felugró modalon melyik naptárba szeretne másolni
Then	A rendszer lemásolja az adott kártyát feltéve hogy egy pontosan ilyen még nem létezik az adott naptárban

Feladatlista kártyáinak újrarendezése

Given	A felhasználó a főoldalon van
When	Az adott feladatlistán megfog egy kártyát majd a listán belül mozgatva új pozícióba teszi
Then	A feldatlistában szereplő kártyák sorrendje frissül

Kártya áthelyezése

Given	A felhasználó a főoldalon van és van egy oszlop egy meglévő kártyával.
When	A felhasználó a kártyát a naptár valamelyik mezőjére húzza. (balra időponthoz köti ami egy modal megfelelő paraméterezésével történik)
Then	A kártya átkerül a naptár adott napjára.

Naptár napjának megnyitása

Given	A felhasználó a főoldalon van
When	A naptár napjára kattint
Then	A naphoz tartozó napi nézet megnyílik.

További naptárak kezelése

Given	A felhasználó a főoldalon van
When	A naptáron lévő manage gombra kattint
Then	A további naptárak oldal megnyílik.

Naptár másolása saját naptárba

Given	A felhasználó a főoldalon van
When	A naptár másolása gombra kattint majd kiválasztja a felugró modalon melyik naptárba szeretne másolni
Then	A rendszer lemásolja a naptárhoz tartozó összes kártyát a duplikátumok szűrésére odafigyelve

Napi nézet funkciói

Kártya létrehozása

Given	A felhasználó a napi nézet oldalon van
When	Az új kártya létrehozása gombra (+) kattint, majd megfelelően paraméterezi azt.
Then	A kártya megjelenik a todo listában ha nem lett időponthoz kötve, amennyiben igen a scheduled listában lesz látható.

Kártya törlése

Given	A felhasználó a napi nézet oldalon van
When	A kártyát kitörli
Then	A kártya törlődik

Todo lista kártyáinak újrarendezése

Given	A felhasználó a napi nézet oldalon van
When	A todo listán megfog egy kártyát majd a listán belül mozgatva új pozícióba teszi
Then	A todo listában szereplő kártyák sorrendje frissül

Kártya megjelölése elvégzettként

Given	A felhasználó a napi nézet oldalán van
When	A kártyát megjelöli elvégzettként
Then	A kártya elvégzettként lesz megjelenítve

Kártya időpontra osztása

Given	A felhasználó a napi nézet oldalán van és van legalább 1 kártya az időponthoz nem kötött feladatlistában
When	A felhasználó a kártyát szerkesztésere nyitja, majd időponthoz köti
Then	A kártya a megfelelő időpontban megjelenik az időponthoz kötött feladatlistában

Kártya visszaküldése a főoldalra

Given	A felhasználó a napi nézet oldalán van és van legalább 1 kártya ami a főoldal valamelyik feladatlistájában volt.
When	A felhasználó a kártyát visszaküldi a főoldalra
Then	A kártya visszakerül az eredeti feladatlistájába

Kártya másolása saját naptárba

Given	A felhasználó a főoldalon van
When	A naptár másolása gombra kattint majd kiválasztja a felugró modalon melyik naptárba szeretne másolni
Then	A rendszer lemásolja az adott kártyát feltéve hogy egy pontosan ilyen még nem létezik az adott naptárban

Több naptár kezelése funkciói

Új naptár létrehozása

Given	A felhasználó a több naptár fülön van
When	Az új naptár létrehozása gombra kattint
Then	A rendszer létrehoz egy új naptárat, amelyhez további felhasználókat is hozzáadhat.

Naptár törlése vagy követésének megszüntetése

Given	A felhasználó a több naptár fülön van
When	A naptár törlése gombra kattint
Then	Amennyiben a felhasználó az utolsó owner a naptár és minden hozzá tartozó adat törlődik, ellenkező esetben csak az user hozzáférése

Profil funkciói

Jelszó változtatása

Given	A felhasználó a profil fülön van
When	A jelszó változtatása gombra kattint majd megfelelően paraméterezi a modalt
Then	A jelszava megváltozik

Profil törlése

Given	A felhasználó a profil fülön van
When	A profil törlése gombra kattint
Then	A felhasználó és minden hozzá kapcsolódó adat törlődik. A hozzá tartozó naptárak a naptár törlési szabályai szerint kerülnek kezelésre.

6. fejezet

Összegzés