



Készítette:

Császár Ákos, Informatika III

Molnár Hunor, Informatika III

Tasnádi Norbert, Informatika III

Váncza Tibor, Informatika III

Tartalomjegyzék

1.	Bevezető	3
2.	Célkitűzések	4
3.	Specifikus követelmények	5
3.1	Funkcionális követelmények	5
3.2	Nem funkcionális követelmények	6
3.3	Szoftver követelmények	7
4.	Rendszer architektúra	7
5.	Projekt management	7
6.	Projekt előkészítése	7
7.	Részletes projektstruktúra	8
7.1	Felhasznált technológiák	8
7.2	Adatbázis	8
7.3	Api	9
7.3.1	Users controller	10
7.3.2	Movies controller	11
7.3.3	Ratings controller	11
7.3.4	Watchlist kontroller	12
7.4	Frontend	12
7.4.1	Login/Register:	12
7.4.2	Forgot Password:	13
7.4.3	Filmek listázása:	13
7.4.4	Film adatok:	14

7.4.5	Filmek értékelése:.....	Error! Bookmark not defined.
7.4.6	Watchlist(megnézendő filmek):	14
7.5	Tesztek	15
7.5.1	Tesztelési eszköz	15
7.5.2	Megvalósított Unit tesztek.....	15
8.	Diagrammok	17
8.1	Folyamat diagrammok	17
8.1.1	Login	17
8.1.2	Register.....	17
9.	Jövőbeli tervek elképzelések.....	18
10.	Összegzés	19

1. Bevezető

A projektünk célja egy olyan webalkalmazás létrehozása volt, amelyen a felhasználók filmeket tudnak böngészni kategóriák, és értékelések alapján, ezenkívül pedig tippeket adni, hogy melyik filmet érdemes neki megnézni. Napjainkban rengeteg film készül és eldönteni, hogy melyiket szeretnénk nézni, vagy jegyet váltani a moziba, igazán nehéz feladat. Ebben nyújt segítséget a RuntimeTerror webalkalmazás ahol ezek a funkciók mellett a felhasználók a közönség értékelése alapján dönthetik el, hogy egy filmet megnéznék, vagy sem.

2. Célkitűzések

A webalkalmazás elkészítésének ötlete mellett döntöttünk, mivel platformfüggetlen és mindenki el tudja érni eszköztől és operációs rendszertől függetlenül. A legfőbb szempont az alkalmazással kapcsolatban az volt, hogy legyen egy beléptető rendszer, így létrehozhat, és bejelentkezhet az általa készített fiókba, és így kedve szerint értékelheti, és megnézendők közé teheti az általa kedvelt filmet. Mindehhez tartozik még egy API ami kapcsolatot létesít egy adatbázissal ahol el vannak tárolva a felhasználók adatai és a filmek. A fontosabb célok a következők voltak:

- Belépés és Regisztráció
- Filmek listázása
- Filmek Megnézendő listába tévése
- Filmek értékelése
- Átlátható, jó kinézetű UI

3. Specifikus követelmények

3.1 Funkcionális követelmények

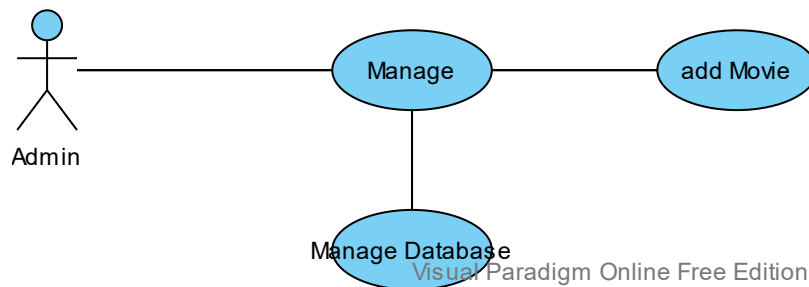
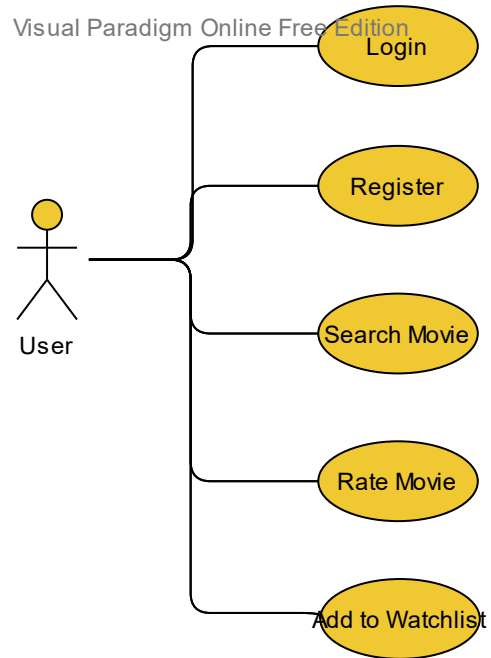


Figure 1 Funkcionális követelmények

User:

- Register/Login: A felhasználó be kell jelentkezzen ahhoz hogy később megnézendő filmekhez hozzá tudjon adni újabbakat.
- Search Movie: A felhasználó rákereshet filmekre a film címe alapján
- Rating: A felhasználó értékelheti a filmeket
- Watchlist: A felhasználó listát készít a kedvenc filmjeiről

Admin:

- Manage database: Az adminisztrátor kezeli és karban tartja az adatbázist
- Add Movie: az adminisztrátor új filmeket tölt fel az adatbázisba

3.2 Nem funkcionális követelmények

A legfontosabb nem funkcionális követelmények a következők a webalkalmazás esetében:

- Az adatbázisban tárolt felhasználói adatokhoz (felhasználónév + jelszó) csak, és kizárólag a felhasználói fiók tulajdonosa fér hozzá.
- A karbantarthatósága könnyű, filmek törlésére, és feltöltésére is adatbázison keresztül egyszerű módon nyílik lehetősége az adminisztrátornak.

3.3 Szoftver követelmények

- Internet elérhetőség

4. Rendszer architektúra

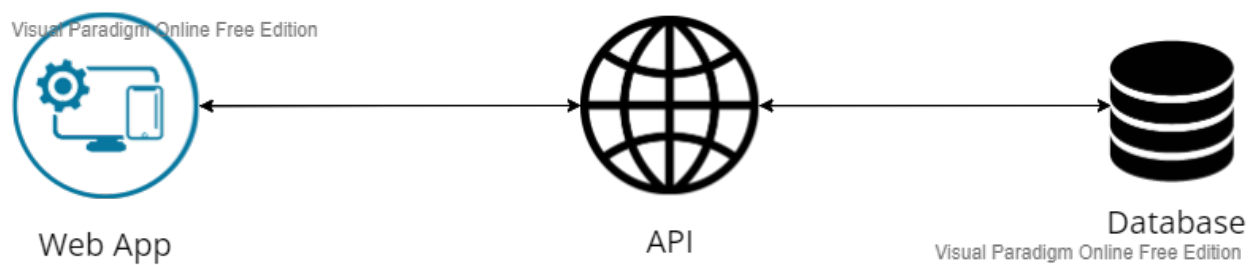


Figure 2 Rendszer Architektúra

5. Projekt management

A fejlesztés fázisai, és lépései a Trello, és Github segítségével voltak dokumentálva. Trello segítségével a taskokat osztottuk szét egymás között, míg a GitHub-ra töltöttük fel a már meglévő részeket és emelett verzió követésre is használtuk.

6. Projekt előkészítése

- Frontend - Váncza Tibor, Császár Ákos
- Backend – Tasnádi István- Norbert, Molnár Hunor

Verzió követésre GitHubot használtunk. A fő projekt 3 részre lett osztva frontend, backend és végül a Unit tesztek.

7. Részletes projektstruktúra

7.1 Felhasznált technológiák

- Angular
- .Net 6.0
- SSMS

7.2 Adatbázis

Az adatbázist SQL Server Management Studio -ban lett elkészítve melynek táblái a

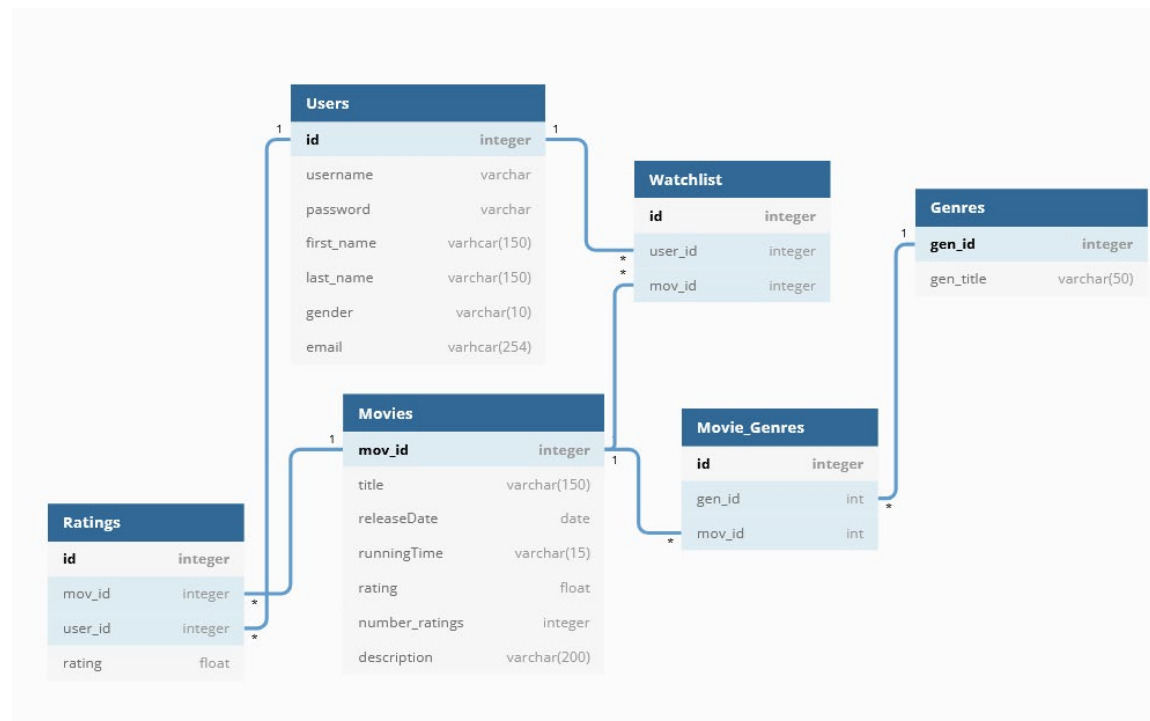


Figure 3 Adatbázis táblái

következők:

Users: ebben a táblában vannak eltárolva a felhasználók információi

Movies: ebben a táblában vannak eltárolva a filmek adatai

Genres: ebben a táblában vannak eltárolva a műfajok nevei

Movie_Genres : ebben a táblában a filmek kategóriákhoz vannak besorolva

Ratings : ebben a táblában a felhasználó által értékelt filmjei vannak eltárolva

Watchlist: ebben a táblában a felhasználó megnézendő filmjei vannak eltárolva

7.3 API

Az API keretén belül megvalósítottunk különböző endpointokat, melyekkel majd a weboldal fog kommunikálni. Számos kontrollert hoztunk létre mindegyik fő funkcióhoz, ezek a kontrollerek a következők:

- Users
- Movies
- Ratings
- Watchlist

7.3.1 Users controller

Ez a kontroller felelős a felhasználók adatainak feldolgozására és továbbítására a kliens

Users		
GET	/api/Users/get-all-users	This retrieves all users and their data from database
GET	/api/Users/get-user-by-id	This gets a user's data by its ID
GET	/api/Users/get-user-by-name	This gets a user(s) data by its name
POST	/api/Users/add-new-user	This adds a new user to database
PUT	/api/Users/update-user-profile	Updates user profile data
PUT	/api/Users/update-user-password	Updates user's current password with the new password.
POST	/api/Users/login	This is the login feature for the website

Figure 5 User kontroller endpointjai

félhez, ahol a választ a kliens JSON formátumban fogja megkapni.

7.3.2 Movies controller

Ez a kontroller felelős a filmek adatainak feldolgozására és továbbítására a kliens felhez,

Movies		
GET	/api/Movies/get-all-movies	Retrieves all movies and their data from database
GET	/api/Movies/get-movie-by-id	This gets movie's data by its ID
GET	/api/Movies/get-movies-by-title	This is the search feature of the web

Figure 6 Movies controller endpointjai

ahol a választ a kliens JSON formátumban fogja megkapni.

7.3.3 Ratings kontroller

Ez a kontroller felelős a felhasználók által értékelt filmek adatainak feldolgozására és továbbítására a kliens felhez, ahol a választ a kliens JSON formátumban fogja megkapni.

Ratings		
GET	/api/Ratings/get-all-ratings	This gets all ratings from database
POST	/api/Ratings/add-new-rating	This adds a new rating to given movie by given user by their IDs
GET	/api/Ratings/get-ratings-by-userid	This gets given user's ratings by its UserID
GET	/api/Ratings/get-ratings-by-movieid	This gets given movie's ratings by its MovieID
PUT	/api/Ratings/update-rating	This updates user's rating to given movie by their IDs
DELETE	/api/Ratings/delete-rating	This deletes user's rating to given movie by their IDs

Figure 7 Ratings kontroller endpointjai

7.3.4 Watchlist controller

Ez a kontroller felelős a felhasználók megnézendő filmlistájának feldolgozására és

Watchlist		
GET	/api/Watchlist/get-all-users-watchlist	This gets all user's watchlist from database as a list
GET	/api/Watchlist/get-user-watchlist-by-id	This gets user's watchlist by its UserID
GET	/api/Watchlist/get-user-watchlist-by-username	This gets user's watchlist by its Username
POST	/api/Watchlist/add-movie-to-watchlist	This adds the given movie to the given user's watchlist by their IDs
DELETE	/api/Watchlist/delete-movie-from-watchlist	This deletes given movie from given user's watchlist by their IDs

Figure 8 Watchlist kontroller endpointjai

továbrbítására a kliens felé, ahol a választ a kliens JSON formátumban fogja megkapni.

7.4 Frontend

7.4.1 Login / Register:

Minden egyes felhasználónak lehetősége van saját accountot létrehozni amelybe be tud jelentkezni. Amíg a felhasználó nem tölti ki a megadott mezőket megfelelő adattal addig nem válik elérhetővé a gomb lenyomása

LOGIN

Please log in

Username: *

Password: *

LOGIN

REGISTER

Forgot password

REGISTER

Please fill out the following:

Username: *

Password : *

Email : *

First Name :

Last Name :

REGISTER

7.4.2 Elfelejtett jelszó:

PASSWORD RESET

✕

Reset your password

Username:

*

Email :





*

Password:

*

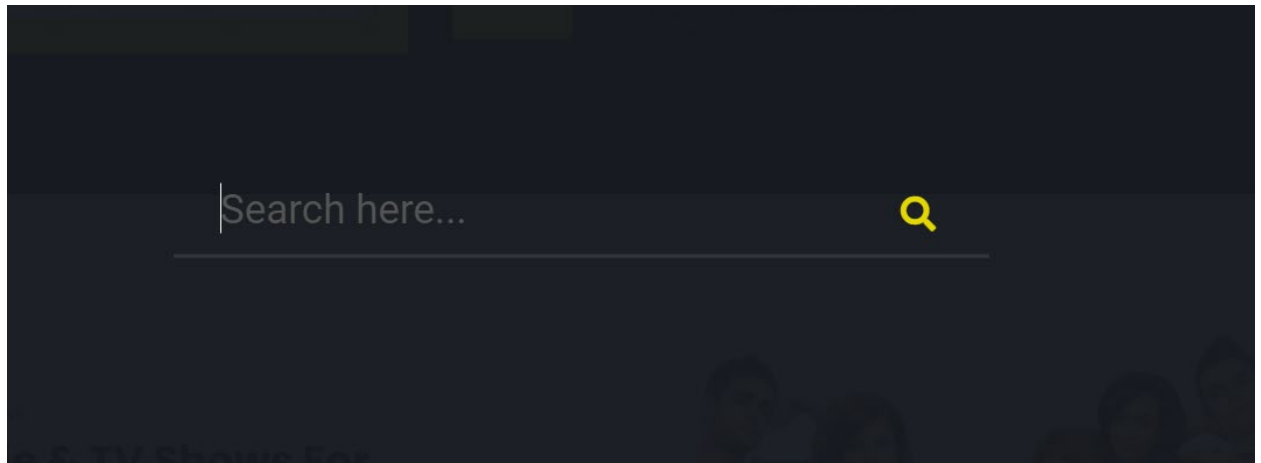
RESET PASSWORD

7.4.3 Filmek listázása:

			
Guardians of the Galaxy	Avengers: Age of Ultron	Ant-Man	Captain America: Civil War
2014	2015	2015	2016
HD 121 min 8.1	4K 141 min 7.3	HD 117 min 7.3	HD 147 min 7.8

7.4.4 Filmek keresése:

A felhasználó rákereshet konkrét filmekre melyből a legelső találatot meg is jeleníti a weboldal számára.




7.4.5 Film adatok:

A felhasználó megnézheti, hogy mi az adott film értékelése, mikor jött ki, milyen hosszú és mi a leírása.



7.4.6 Watchlist (megnézendő filmek):

A felhasználó bejelentkezés után listát készíthet a megnézni kívánt filmekről. Az "Add To Watchlist" gomb lenyomásával hozzáadhatja a kívánt filmet a saját listájához, a gomb újboli lenyomásával leveheti a listájáról a kívánt filmet.



Guardians of the Galaxy

PG 18

HD


Action, Adventure, Comedy

📅 2014


🕒 121 min

Rating: **8.1**

An action-packed, epic space adventure, Marvel's Guardians of the Galaxy expands the Marvel Cinematic Universe into the cosmos, where brash adventurer Peter Quill finds himself the object of an unrelenting bounty hunt after stealing a mysterious orb coveted by Ronan, a powerful villain with ambitions that threaten the entire universe. To evade the ever-persistent Ronan, Quill is forced into an uneasy truce with a quartet of disparate misfits--Rocket, a gun-toting raccoon; Groot, a tree-like humanoid; the deadly and enigmatic Gamora; and the revenge-driven Drax the Destroyer. But when Quill discovers the true power of the orb and the menace it poses to the cosmos, he must do his best to rally his ragtag rivals for a last, desperate stand--with the galaxy's fate in the balance.

 Share

Watch later?
Save this movie

 ADD TO WATCHLIST

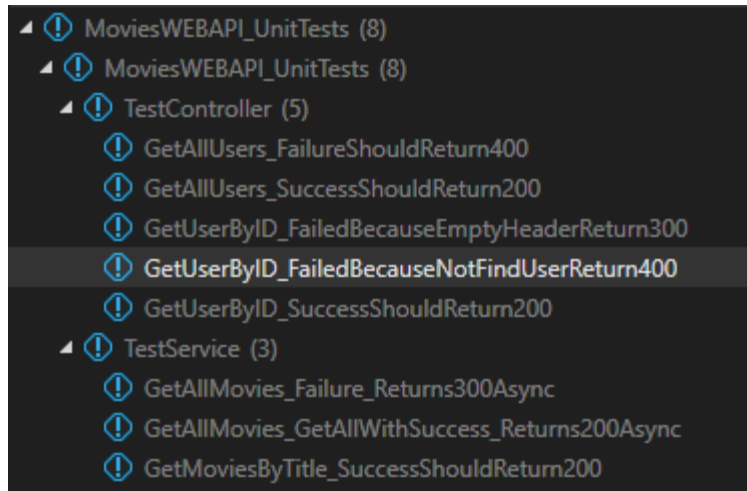
7.5 Tesztek

7.5.1 Tesztelési eszköz

A tesztelés során Unit teszteket végeztünk az XUnit nyílt forráskódú tesztelői eszköz segítségével. Ez a csomag támogatja a .Net 6.0-át, szóval a tesztelési folyamatra tökéletesen megfelelt.

7.5.2 Megvalósított Unit tesztek

Összesen 8 Unit tesztet valósítottunk meg, ezek közül 5-öt a Controllerben található metódusokon, 3-at pedig a Serviceben.



Ezek segítségével tudjuk ellenőrizni a függvényeink helyes működését, kimeneti értékek ellenőrzését. Az elvégzett tesztjeink során egyes függvények összes lehetséges kimenetét teszteltük. Ennek szemléltetésére láthatjuk az alábbi példát:

✓	GetAllUsers_SuccessShouldReturn200	< 1 ms
✓	GetUserByID_FailedBecauseEmptyHeaderReturn300	< 1 ms
✓	GetUserByID_FailedBecauseNotFindUserReturn400	5 ms
✓	GetUserByID_SuccessShouldReturn200	5 ms

Egy függvény tesztelésére vegyük az alábbi példát:

```
[Fact]
0 references
public async Task GetAllUsers_SuccessShouldReturn200()
{
    //Arrange - Változók létrehozása, melyekre szükségünk van tesztjeink során
    AllUsersResponse dummyResponse = new AllUsersResponse();
    dummyResponse.UserList = MockData.GetUsers();
    var userService = new Mock<IUserService>();

    //Act - Tesztelésnél használt változók felruházása
    // egy általunk elképzelt forgatókönyvvel.
    userService.Setup(_ => _.GetAllUsers()).ReturnsAsync(dummyResponse);
    var sut = new UsersController(userService.Object);
    var result = (OkObjectResult) await sut.GetAllUsers();

    //Assert - A forgatókönyv tesztelése a visszatérített értékkel.
    result.StatusCode.Should().Be(200);
}
```


Az “AAA” tervezési mintát alkalmaztuk, ahol minden betűnek megfelel a tervezés egy fázisa:

Arrange, Act, Assert. Ezen fázisok leírása látható a fenti, szemléltetésre szolgáló képen.

8. Diagrammok

8.1 Folyamat diagrammok

8.1.1 Login

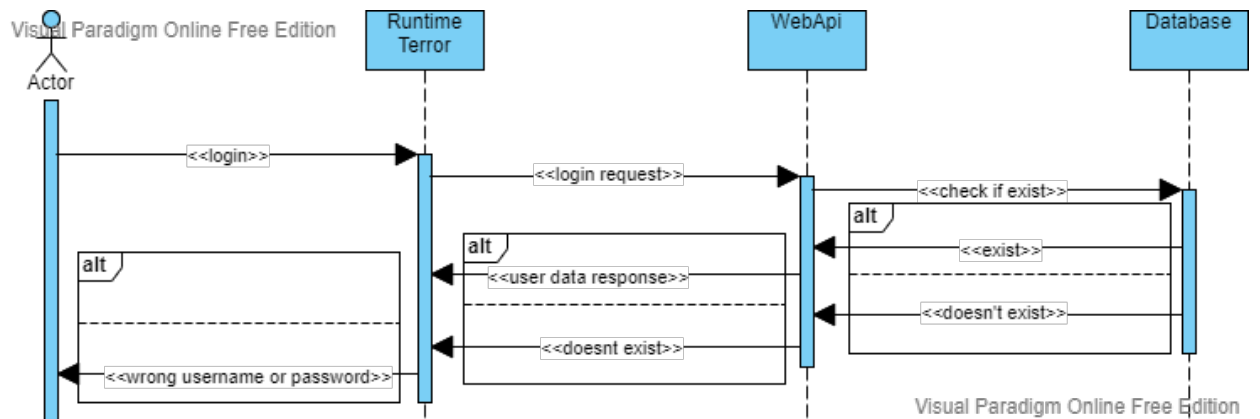


Figure 9 Login Folyamat Diagramm

8.1.2 Register

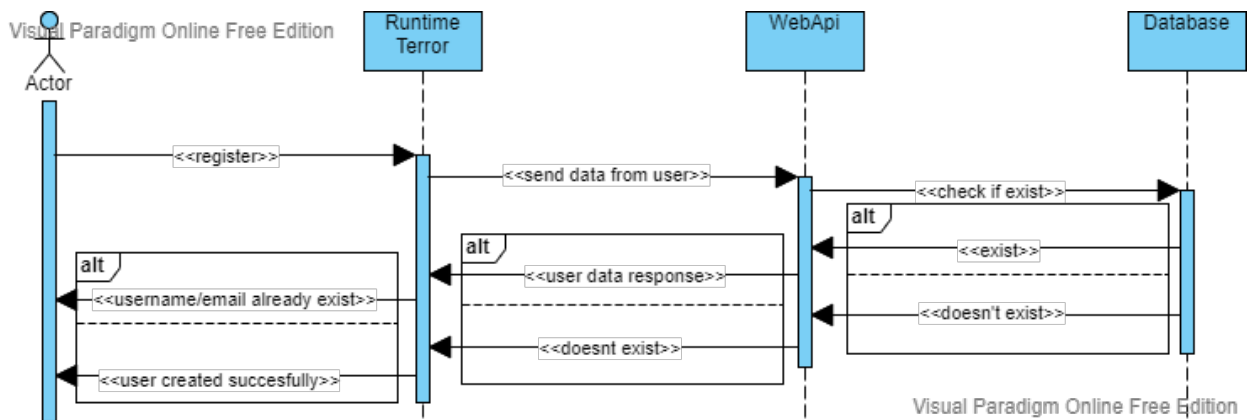


Figure 10 Register Folyamat Diagramm

9. Jövőbeli tervek elképzelések

Az alkalmazás egy lehetőséget nyújt arra, hogy kisebb baráti kör felhasználja saját célra.

Lehetne itt új filmeket keresni, és naplószerűen elmenteni, hogy milyen filmeket szeretnénk megnézni a későbbiekben.

Hátralévő funkcionalitások megvalósítására is szükség lenne, és a jelenlegi funkcionalitások bővítését is tervezzük a közeljövőben.

10. Összegzés

Összegzésként elmondhatjuk, hogy a projekt megvalósítása rengeteg tanulási lehetőséget nyújtott, és bebizonyosodott, hogy amit eddig csak másoktól hallottunk, de eddig nem tapasztaltuk: a .Net környezet nagyon jó eszközöket ad a kezünkbe egy szerver oldali rész megvalósításához.

Ennek a projektnek a megvalósítása sok kihívást jelentett számunkra, mivel olyan problémákkal találkoztunk, amikkel eddig ritkán, vagy egyáltalán nem találkoztunk:

- Hogy kell Unit teszteket készíteni, és futtatni.
- Hogy kell AngularJS keretrendszert használni.
- Idő hiányában hogy állítsuk fel a prioritást a funkciók között, és melyeket valósítsuk meg legelőször?
- A feladatok leosztására 4 ember esetén mi a leghatékonyabb mód.
- Egyes részfeladatok megvalósítására mennyi időre van szükség.

Végzőként ahhoz, hogy egy szoftver jól nézzen ki, nélkülözhetetlen a minőségi csapatmunka, tesztek készítése a munkafolyamat közben, akár teszt vezérelt fejlesztési folyamat alkalmazásának előtérbe helyezése, és elengedhetetlen hogy mindezek mellett a végeredményről legyen egy dokumentáció is.