FACULTY OF ENGINEERING

COMPUTER ENGINEERING DEPARTMENT

CS 353

Database Systems Final Report

Media Services Data Management System

Watchflix

Group 6

Javid Haji-zada 21701336

Anar Huseynov 21603023

Leyla Hashimli 21701341

Leyla İsmayilova 21701299

Date 04.01.2021

1. Project Description

The aim of the project is to design a Media Services Provider Application. The system provides a variety of functionalities for the users. The main aim of this application is to provide users with the latest high-quality movies and series. The user is able to enjoy his/her favorite movies and series smoothly. The application has features like listing and searching movies and series based on their genre or name. Then, this application prepares suggestions based on the genre of the content in created channels by the user. Besides, like, dislike and comment functionalities are provided so that the users can give feedback to films and series they watched.

Each user has its own authorized account and a unique nickname. It is possible for users to add friends. Users are able to create multiple channels for movies to follow. In this way, they are able to categorize movies based on their choices. Also, users are able to join special groups with their friend to chat and watch a movie. There, they are able to discuss the ongoing movie or an episode of the series with one another. These chats are instantaneous. They are only able to see one another's reactions for a limited amount of time.

In our project, a database system is implemented to store media products, their details, necessary data about users and their activities. Our database handles user related queries, such as requests for authentication, group chatting and commenting. We keep users' nicknames, passwords and birthdates. The database is used to store and retrieve friends of the users with their activities, so that they can enjoy viewing one anothers' activities and invite them to watch together.

2. Final E/R Model

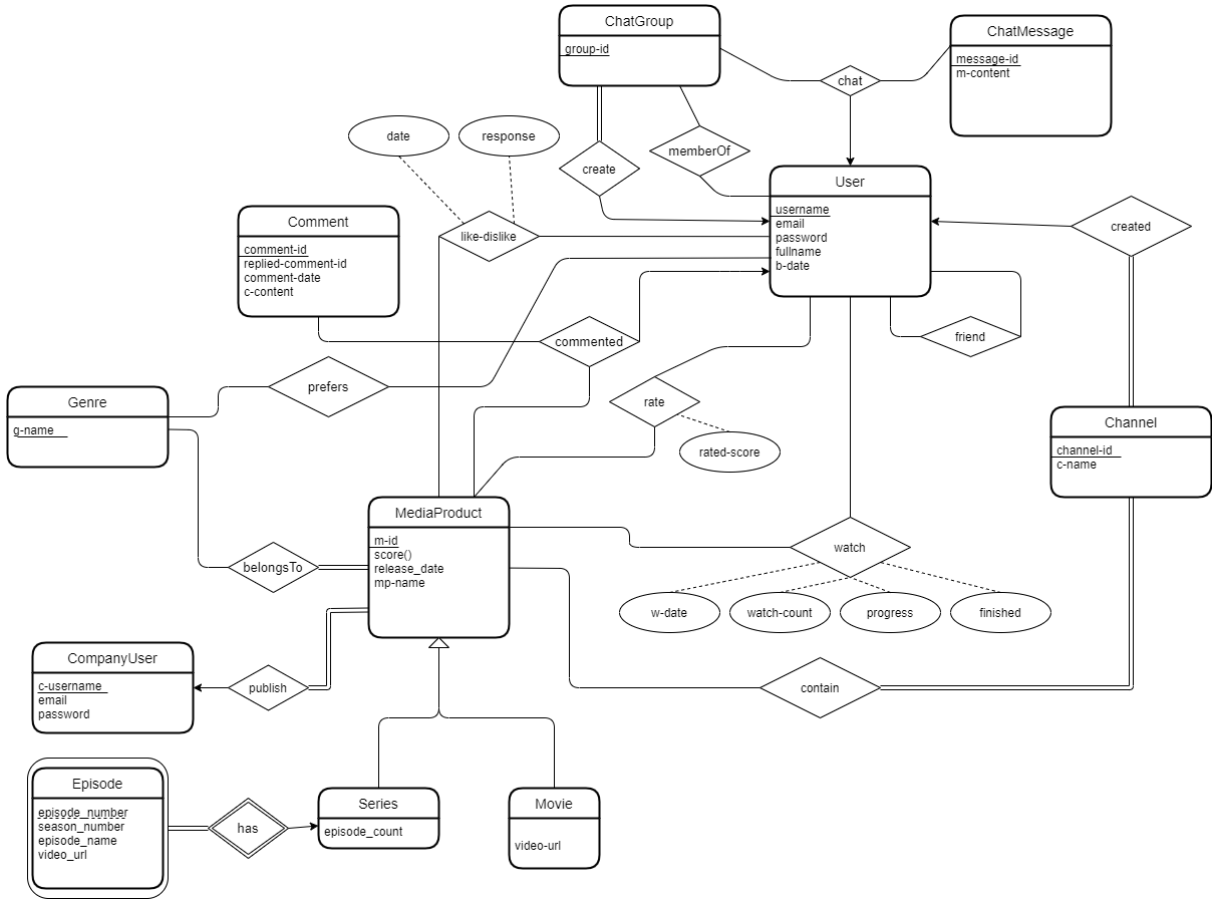

Figure 1. E/R Model

3. Relation Schemas

3.1 User

Relational Model

User(<u>username</u>, email, password, fullname, b-date)

3.2 CompanyUser

Relational Model

CompanyUser(<u>c-username</u>, email, password)

3.3 ChannelCreated

Relational Model

ChannelCreated(username, <u>channel-id</u>, c-name)

FOREIGN KEY: username REFERENCES User(username)

3.4 ChatMessageSent

Relational Model

ChatMessageSent(group-id, username, message-id, m-content)

FOREIGN KEY: group-id REFERENCES ChatGroup(group-id)

FOREIGN KEY: username REFERENCES User(username)

3.5 ChatGroup

Relational Model

ChatGroup(group-id, creatorName)

FOREIGN KEY: creatorName REFERENCES User(username)

3.6 CommentPosted

Relational Model

CommentPosted(comment-id, replied-comment-id, m-id, username, comment-date, c-content)

FOREIGN KEY: username REFERENCES User(username)

FOREIGN KEY: replied-comment-id REFERENCES CommentPosted(comment-id)

3.7 MediaProduct

Relational Model

MediaProduct(m-id, publisher, release_date, mp-name)

FOREIGN KEY: publisher REFERENCES CompanyUser(username)

3.8 Series

Relational Model

Series(m-id, episode_count)

FOREIGN KEY: m-id REFERENCES MediaProduct(m-id)

3.9 Episode

Relational Model

Episode(m-id, episode_number, season_number, episode_name, video_url)

FOREIGN KEY: m-id REFERENCES MediaProduct(m-id)

3.10 Movie

Relational Model

Movie(m-id, video-url)

FOREIGN KEY: m-id REFERENCES MediaProduct(m-id)

3.11 Genre

Relational Model

Genre(g-name)

3.12 watch

Relational Model

watch(m-id, username, w-date, watch-count, progress, finished)

FOREIGN KEY: m-id REFERENCES MediaProduct(m-id)

FOREIGN KEY: username REFERENCES User(username)

3.13 prefers

Relational Model

prefers(g-name, username)

FOREIGN KEY: g-name REFERENCES Genre(g-name)

FOREIGN KEY: username REFERENCES User(username)

3.14 like-dislike

Relational Model

like-dislike(m-id, username, date, response)

FOREIGN KEY: m-id REFERENCES MediaProduct(m-id)

FOREIGN KEY: username REFERENCES User(username)

3.15 friend

Relational Model

friend(username1, username2)

FOREIGN KEY: username1 REFERENCES User(username)

FOREIGN KEY: username2 REFERENCES User(username)

3.16 memberOf

Relational Model

memberOf(group-id, username)

FOREIGN KEY: group-id REFERENCES ChatGroup(group-id)

FOREIGN KEY: username REFERENCES User(username)

3.17 belongsTo

Relational Model

belongsTo(g-name, m-id)

FOREIGN KEY: g-name REFERENCES genre(g-name)

FOREIGN KEY: m-id REFERENCES MediaProduct(m-id)

3.18 contain

Relational Model

contain(channel-id, m-id)

FOREIGN KEY: channel-id REFERENCES ChannelCreated(channel-id)

FOREIGN KEY: m-id REFERENCES MediaProduct(m-id)

3.19 rate

Relational Model

rate(username, m-id, rated-score)

FOREIGN KEY: username REFERENCES User(username)

FOREIGN KEY: m-id REFERENCES MediaProduct(m-id)

4. Implementation Details

Implementation details. State the environment, framework, languages you used explicitly. You have to provide a couple of problems you faced and describe how you solved them. Also, indicate which group member is responsible for which functionalities for implementation.

Our database management system consists of two parts, database and a website. MySQL was used as a database service. We hosted our database on a remote server and used Node.js to establish communication. We created necessary tables through SQL statements written by us, inside our Node.js server file.

As per front-end we used React.js framework. Most of the components were prepared by use of pure HTML, CSS implementation. We mainly used flexbox, which is a CSS3 web layout model. Additionally, we used the React-Bootstrap framework to provide better user interface design; version 1.4.0.

We only used JavaScript during the implementation of our database management system. On the back-end side we had a single “server.js” file where we included all of the API related functions. We also had a “sql.js” file where we kept all of the SQL statements regarding table creation. We implemented API routing, GET, POST methods inside “server.js” file. On the front-end side we used Fetch API which provides an interface for accessing and manipulating parts of the HTTP pipeline, such as requests and responses. We had several server classes which maintained communication between our API on localhost:5000 and our app on localhost:3000. During all of the backend calls we used Promises to handle responses, errors and loading states.

5. Advanced Database Components

5.1 Reports

FILTERING:

Inputs: `@genre, @score, @release-date`

SQL Statements:

```
SELECT *

FROM MediaProduct

WHERE score >= @score AND release-date >= @release-date AND
EXISTS(SELECT *
```

```
FROM belongsTo

WHERE belongsTo.g-name = @genre AND belongsTo.m-id =
MediaProduct.m-id);
```

This is a filtering sql for finding movies with specified combination score, release_date and genre. Firstly movies with specified score and release dates are chosen. Then, it is checked in the belongTo table for genre. EXISTS command is used to decide if the pre-filtered products are from specified genre.

5.2 Triggers

```
CREATE TRIGGER accept_friend AFTER INSERT ON friend
    FOR EACH ROW
    BEGIN
     DELETE FROM request WHERE username1=NEW.username1 AND
     username2=NEW.username2;
    END
```

This trigger is executed when a user accepts a friend request of another user. It is used to delete the request from the “request” relation in the database.

6. User Manual

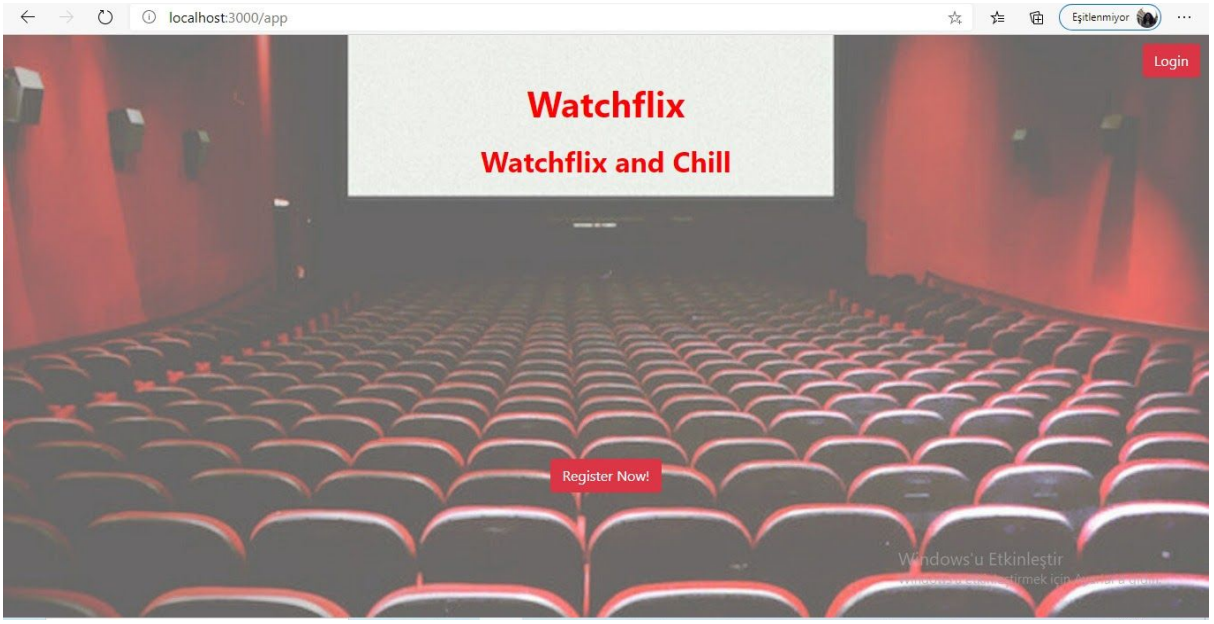

Figure 2. Home page

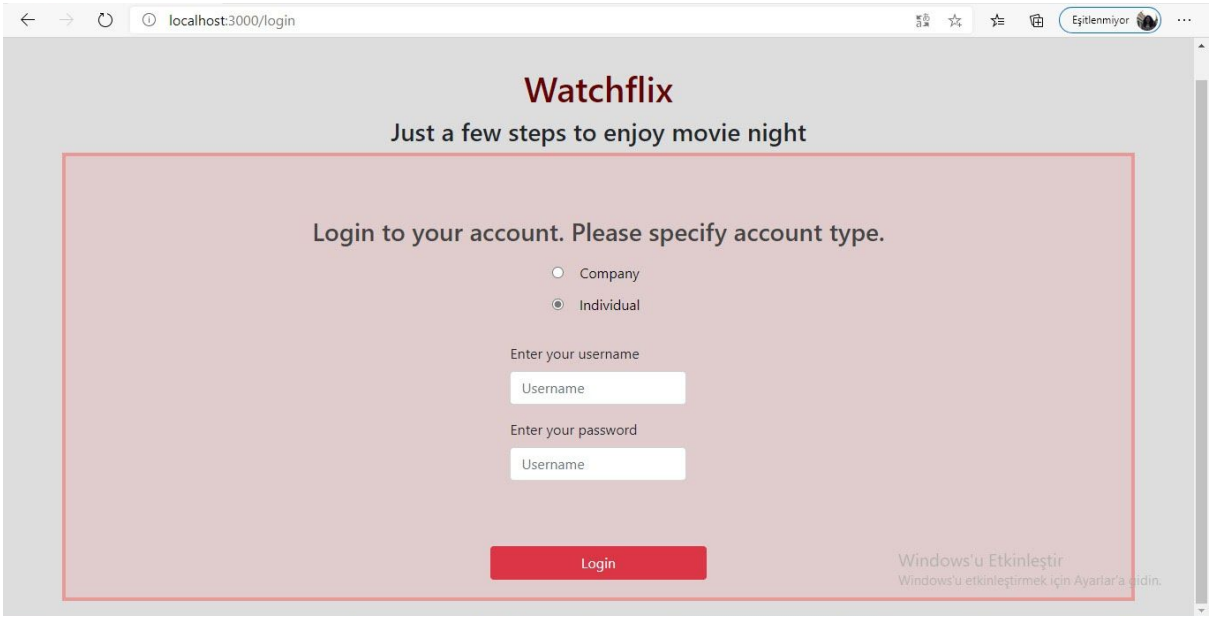

Figure 3. Login page

Users are directed to this page when the “Login” button clicked on the home page. Here, users should choose “Company” user choice to upload movies and series. Users should choose the “Individual” option, to enjoy media products.

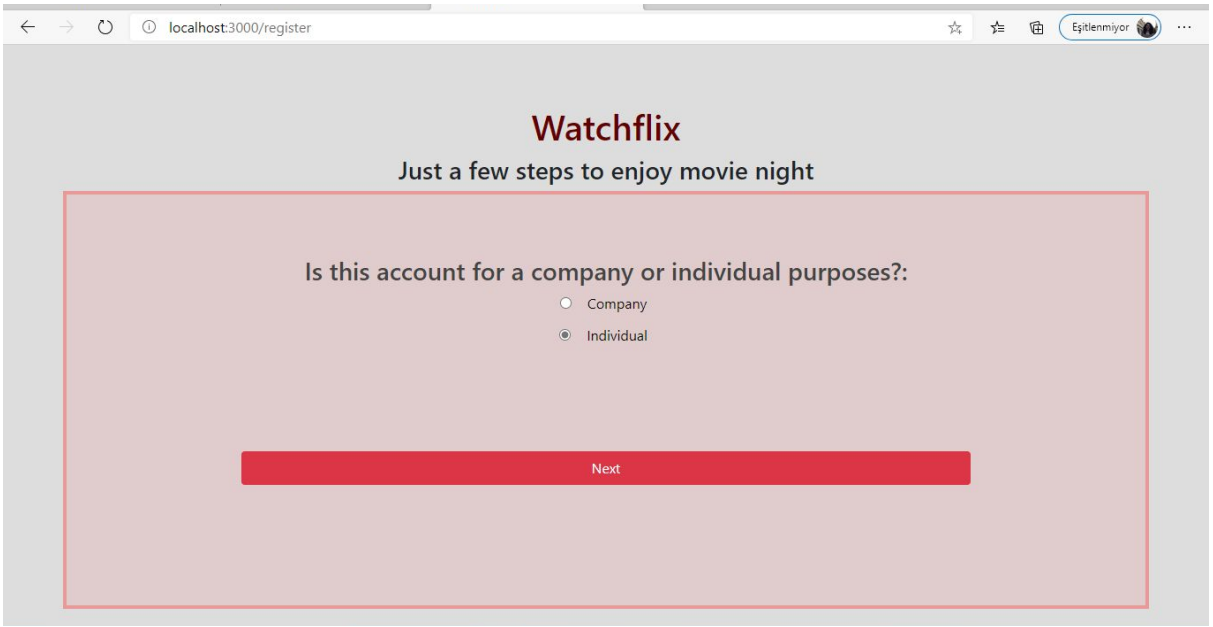

Figure 4. User Registration Step #1

In this step, the user should choose the type of his account.

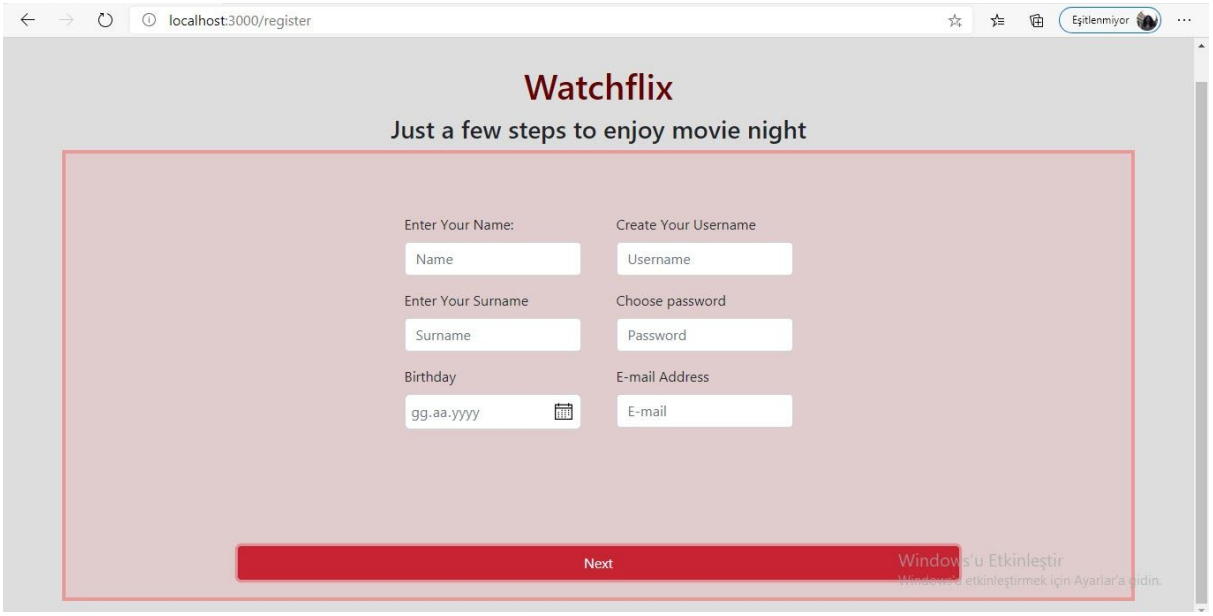

Figure 5. User Registration Step #2

User is directed to this screen when the “Register Now!” button clicked on the home page. Here user should insert necessary data as shown in the figure in order to proceed with “Step #3”. Username should be unique, otherwise an error message shown above the “Next!” button.

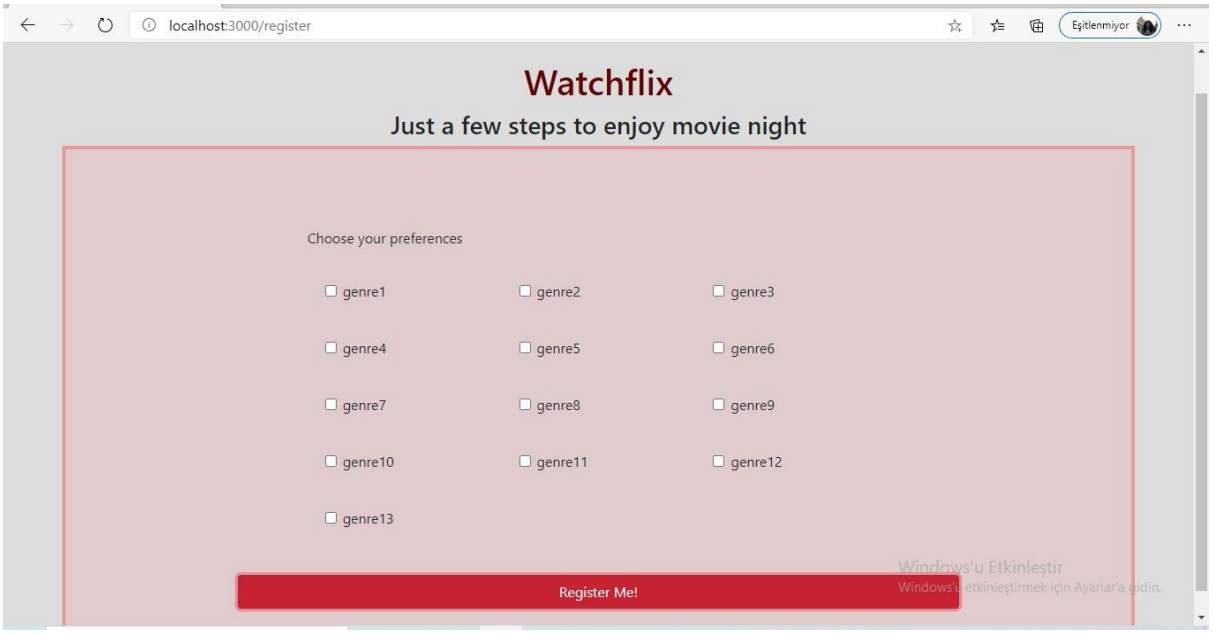

Figure 6. Genre preferences selection

This page is the third step of the registration process for individual user types. User should choose his/her preferred genres.

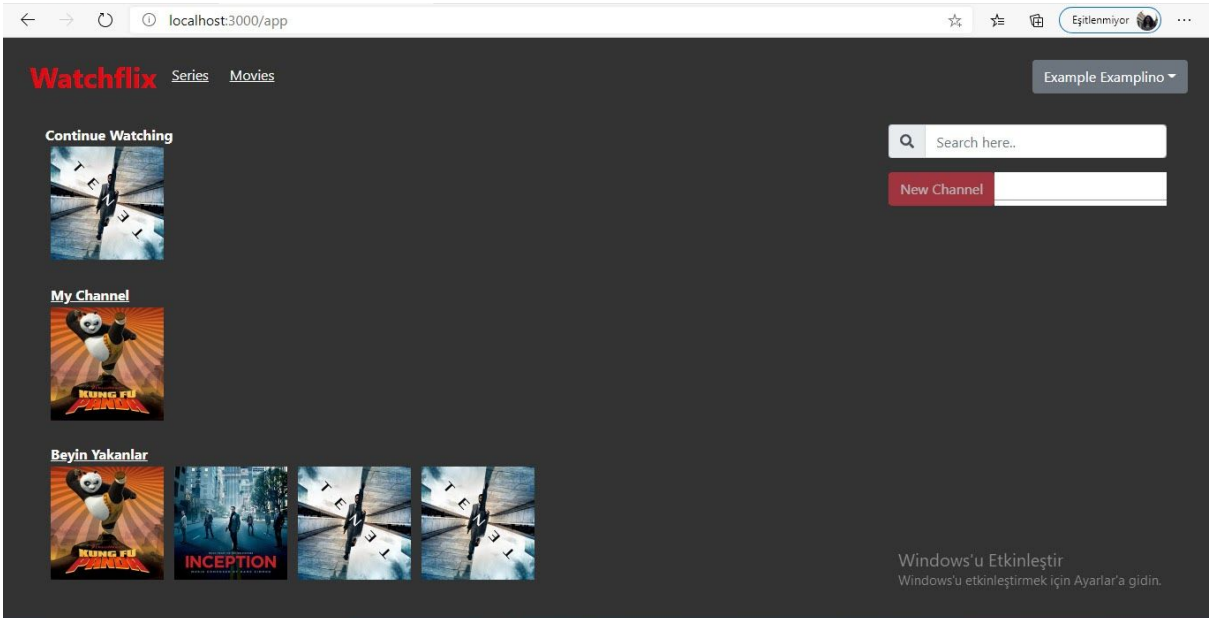

Figure 7. Logged in User Main page

Users are directed to this page after successfully completing registration, or sign in. Here they are provided with the channels and movies/series under each channel. If the user has watched anything before, last watched media is shown under “Continue Watching”. On this page users can either create a new channel by inserting channel name and pressing “New Channel” button or they can navigate to series-only or movies-only pages.

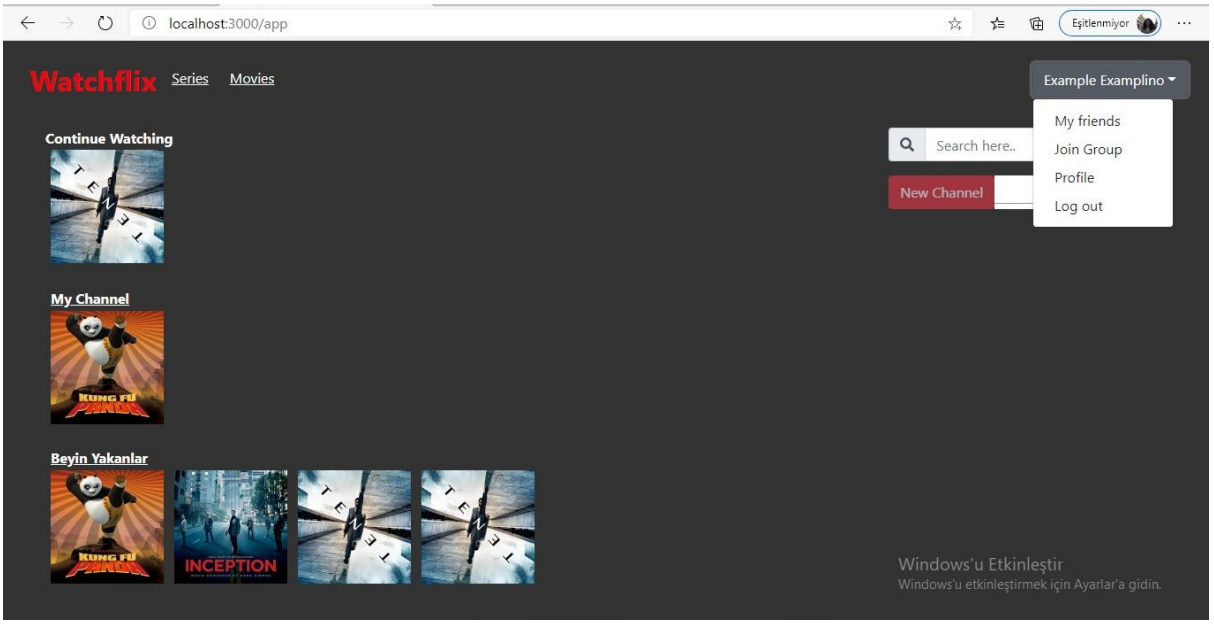

Figure 8. Dropdown Menu

Users can see certain functionalities by pressing their name on the top right side of the main page. Here, they can go to their friends list, profile settings, join a group or log out from the system.

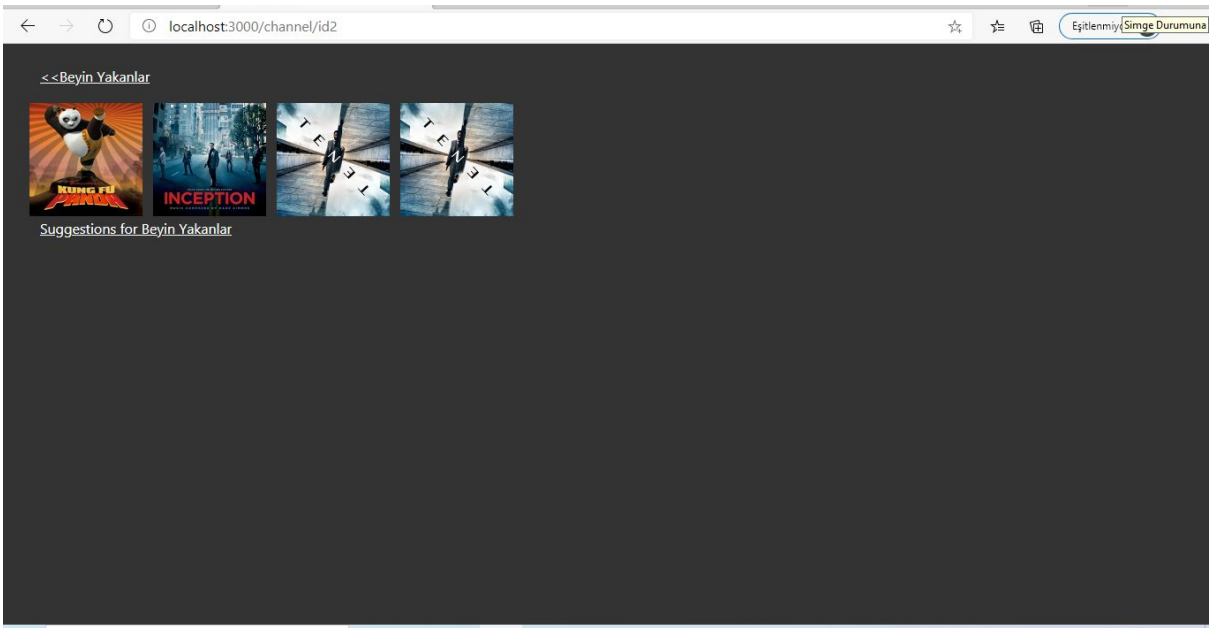

Figure 9. Channel Insights

When a user clicks on the channel name on the main page, he/she is redirected to this channel. Here, user can see all of the media products inside the corresponding channel.

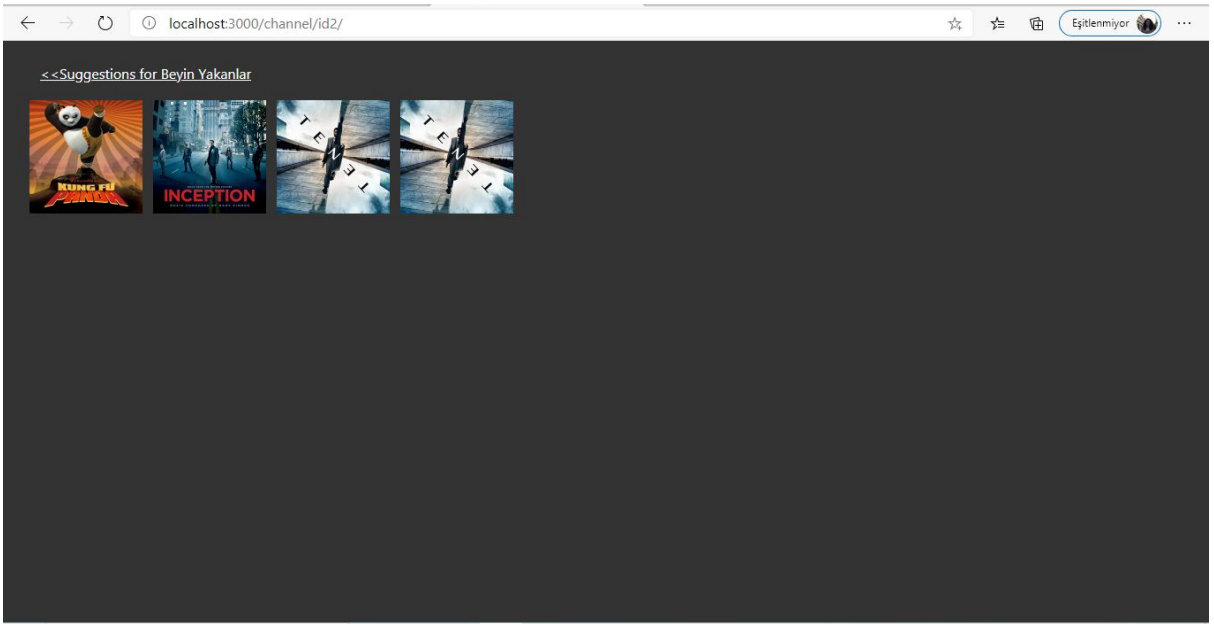

Figure 10. Suggestions for channel

When users press “Suggestions for <channel_name>” inside the Channel Insights page, they are redirected to the suggestions page. Suggestions are provided based on the genres of the media products under the given channel.

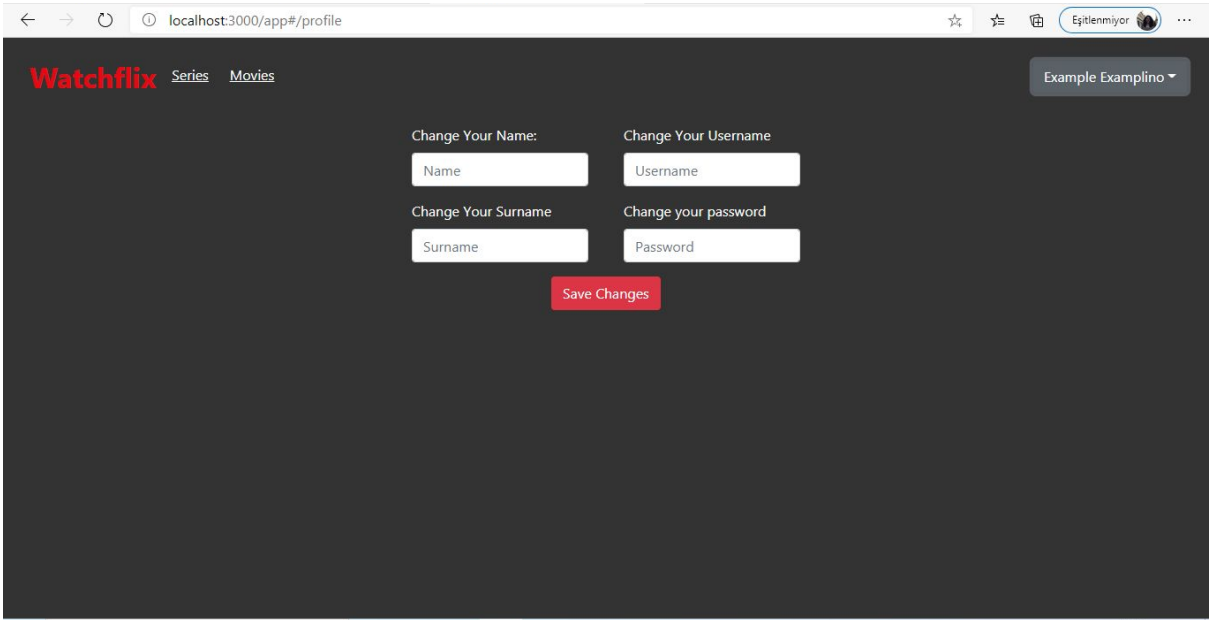

Figure 11. Profile Settings

Users can change the above fields of their profiles. Username should be unique, otherwise an error is prompted under the “Save Changes” button.

Figure 12. Friend List Page

Users can add new friends by simply writing their usernames and clicking the “Add Friend” button. An error is prompted if the given username doesn’t exist or they are already friend. Users can also accept or reject friend requests on the same page.

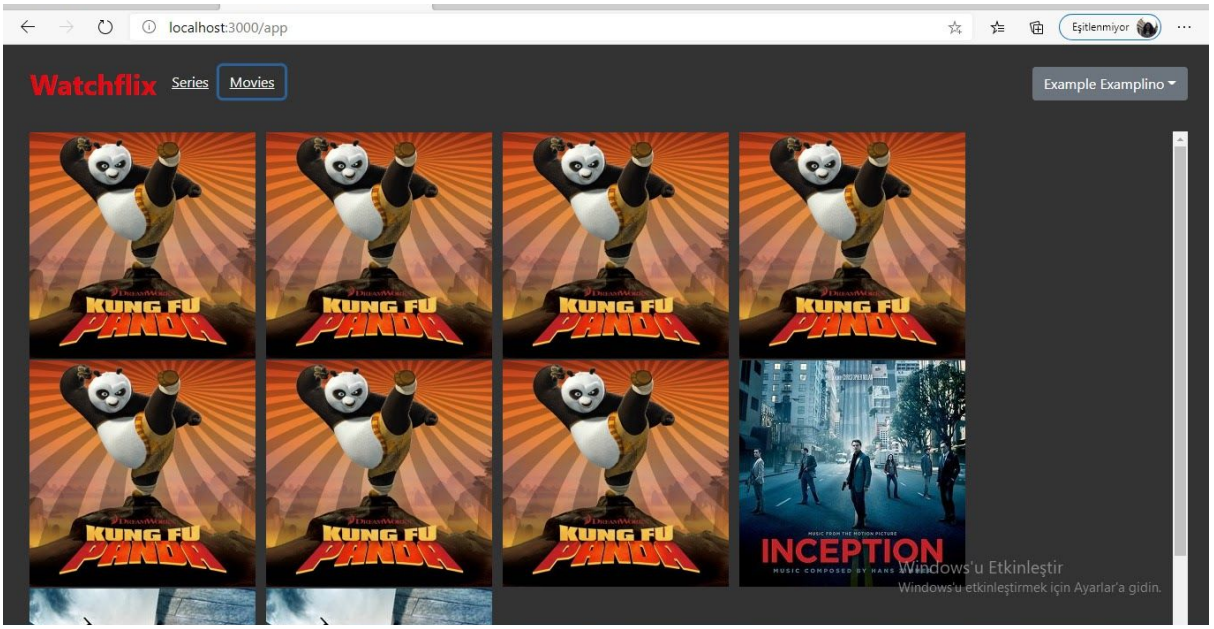

Figure 13. Movies page

After being directed from the main page, users can see available movies on this page, filtered based on their release date.

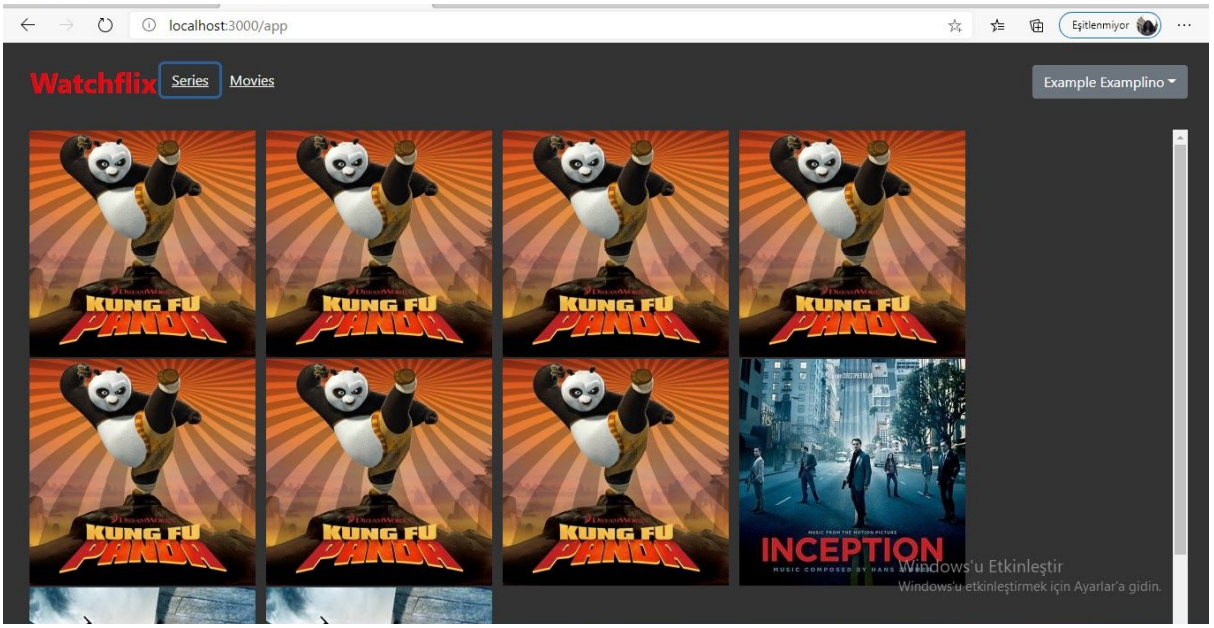

Figure 14. Series Page

After being directed from the main page, users can see available series on this page, filtered based on their release date.

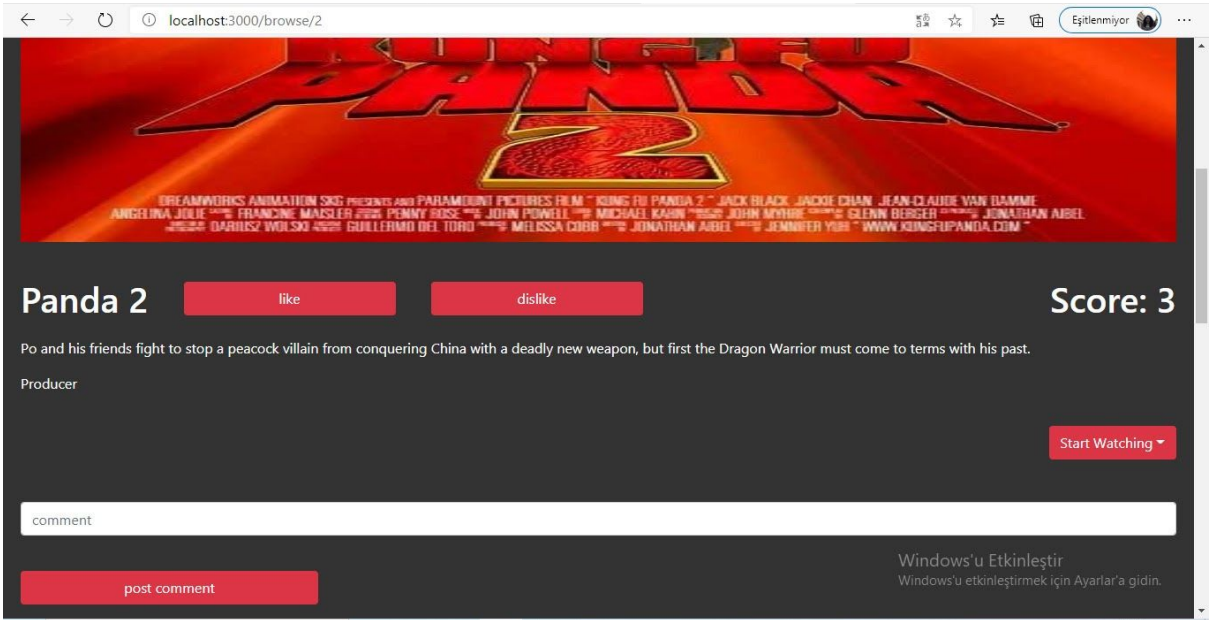

Figure 15. Product details page

Users are directed to this page when the product icons are clicked. Name, description, score and publisher of the product are mentioned. there are also like, dislike and post comment options for giving feedback to the product. Under the “Start Watching” option, user can watch the product or create chat group before starting to watch.

Figure 16. Product details page, comments section

In the bottom of the “product details” page, comments are located. username and content of the comment are mentioned. Users can also reply to comments.

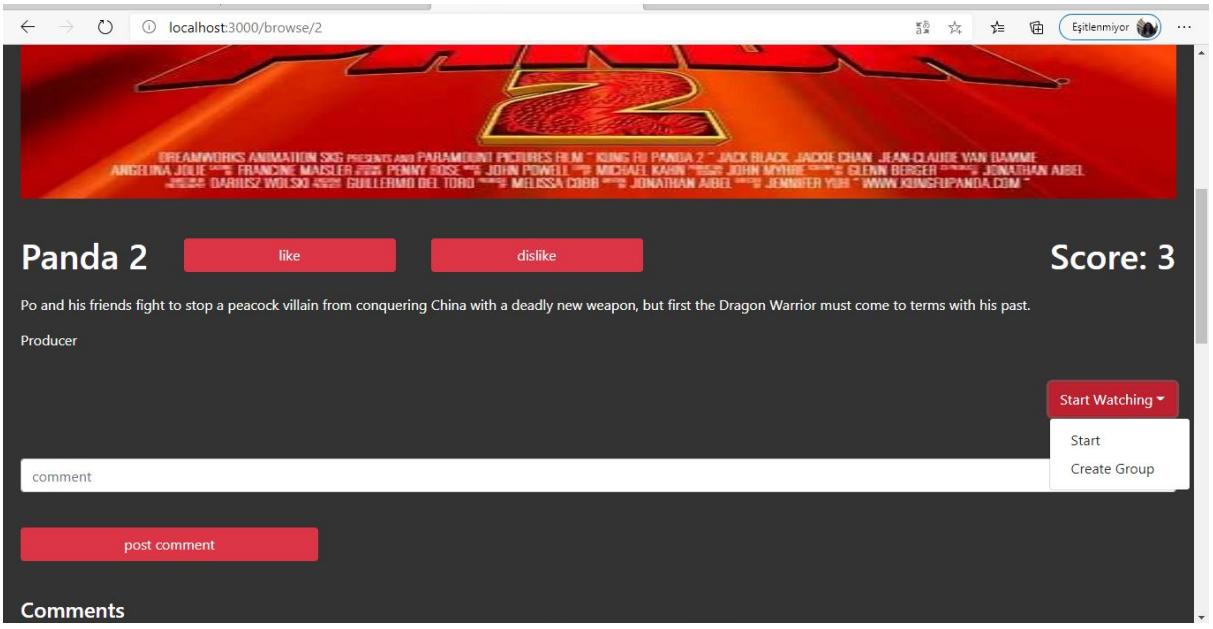

Figure 17. Dropdown Menu on start watching

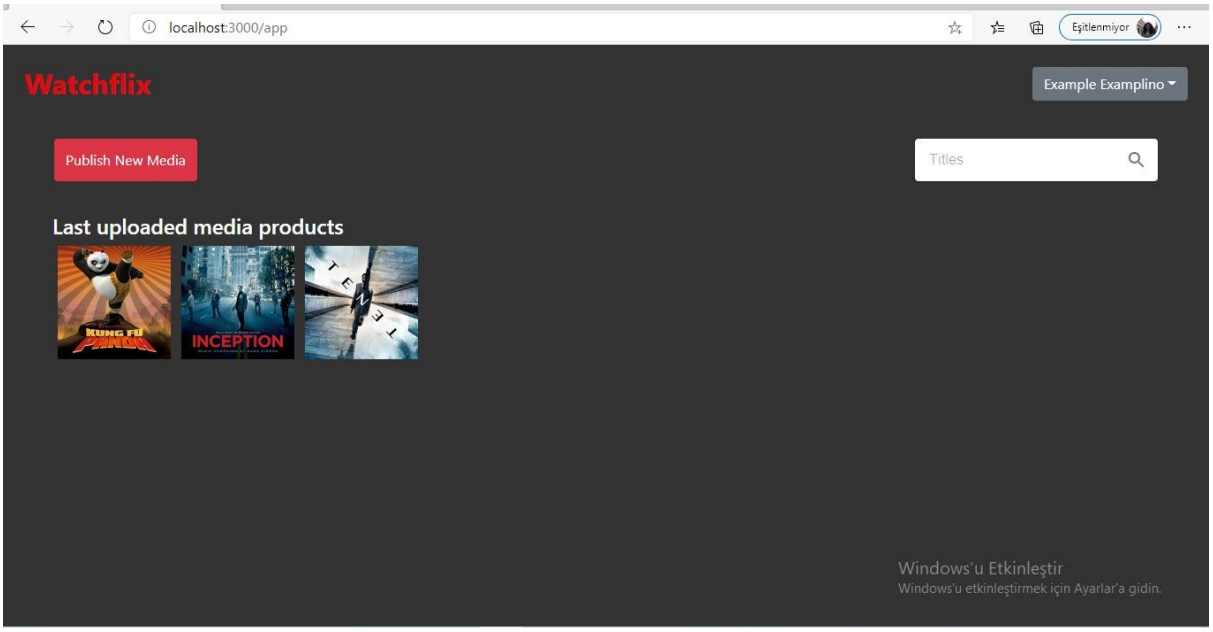

Figure 18. Company User Main Page

Company User sees uploaded media products on this page. They can also create new media from this page.

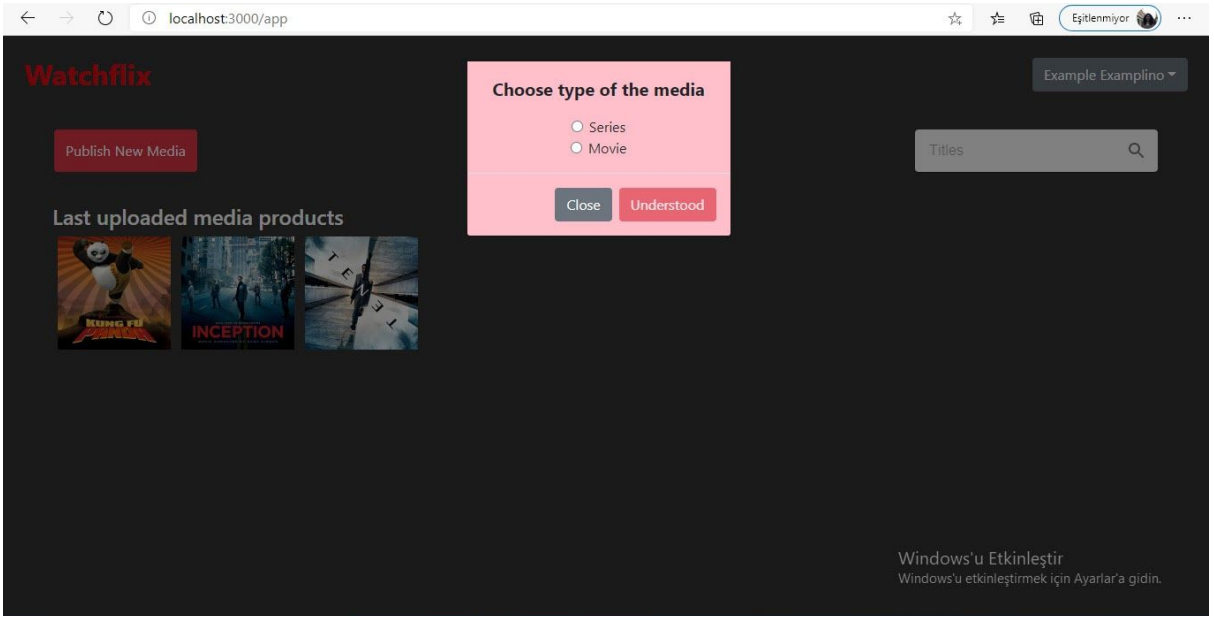

Figure 19. Pop up for choosing type of media to upload

When the user clicks to “Publish New Media” Button, this view pops up. Here user can choose whether to upload a series or movie.

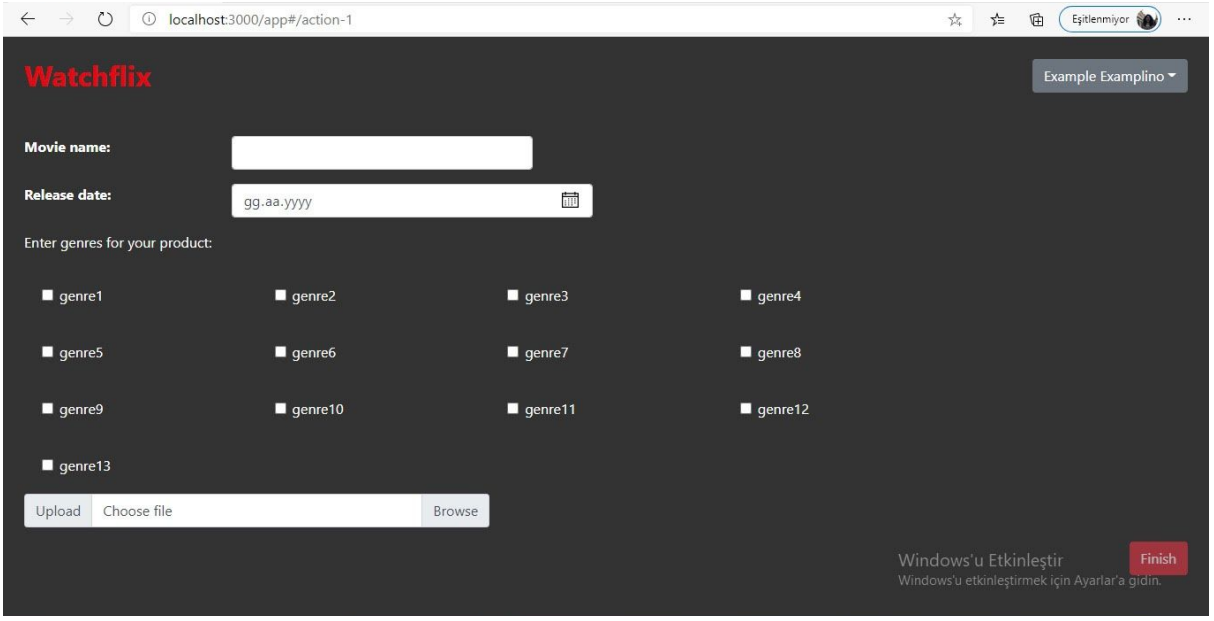

Figure 20. Movie uploading page

If a user has chosen “Movie” as media type, he/she is directed to this page. Here, users should fill forms including name, release date , genres and upload the product. When the user clicks the finish button, the product will be saved and the user will be directed to the main page.

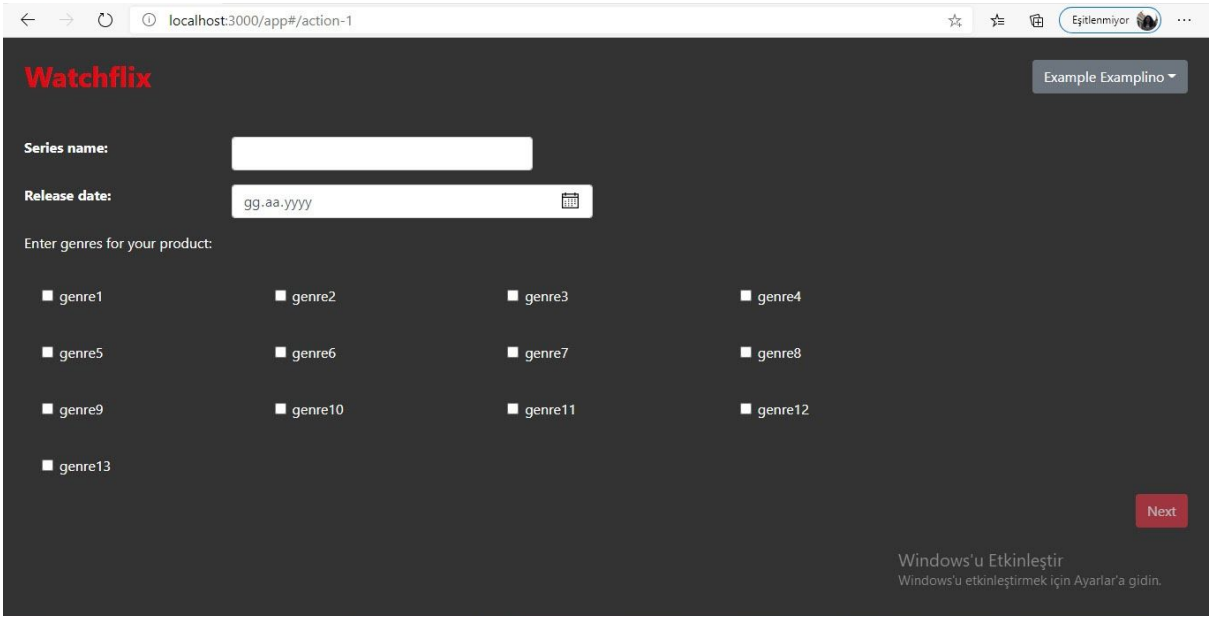

Figure 21. Series Details

If a user has chosen “Series” as media type, he/she is directed to this page. Here, users should fill forms including name, release date , genres. When the user clicks the finish button, the details will be saved and the user will be directed to the episode uploading page.

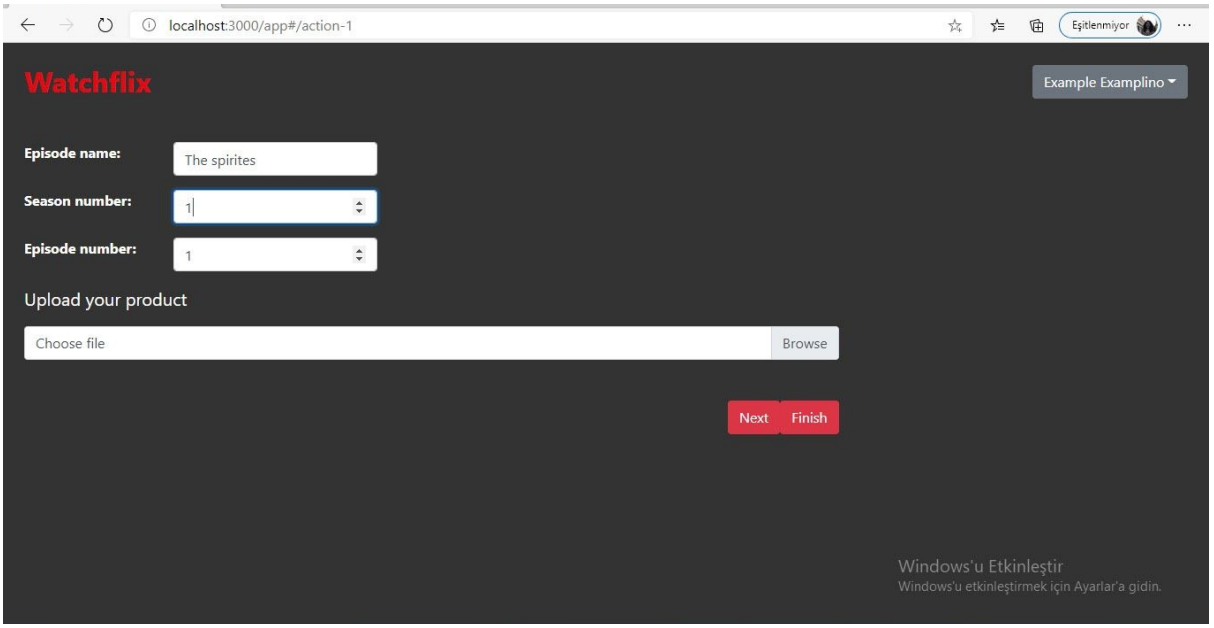

Figure 22. Series Details

If a user has chosen “Series” as media type, he/she is directed to this page. Here, users should fill forms including name, release date , genres. When the user clicks the finish button, the details will be saved and the user will be directed to the episode uploading page.

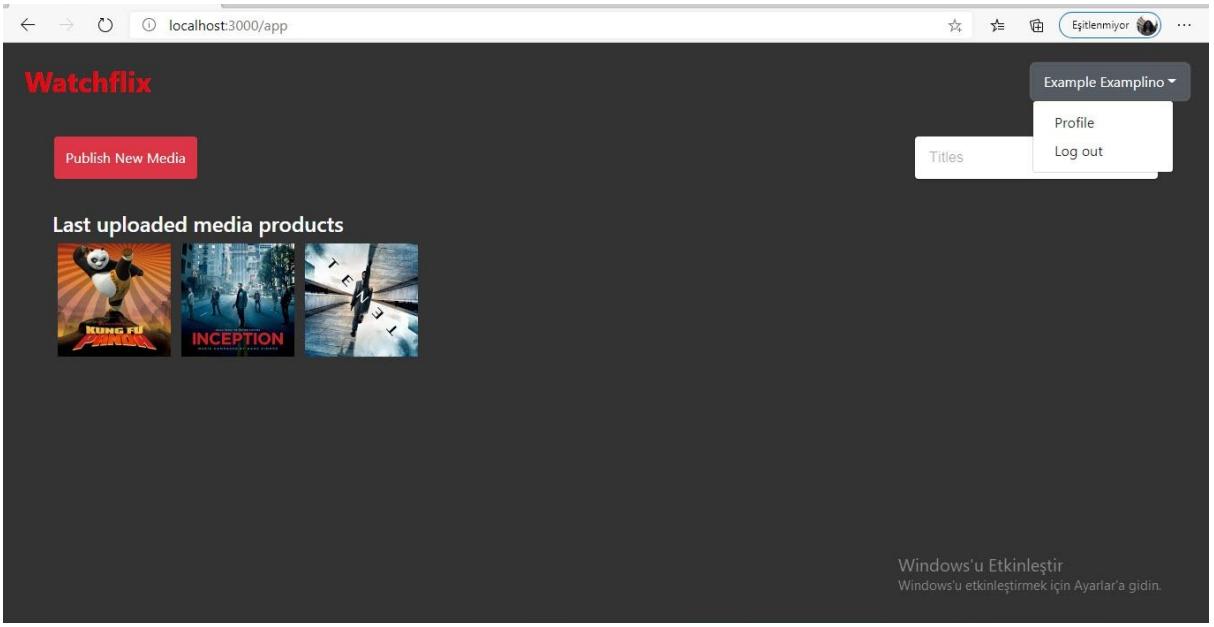

Figure 23. Dropdown menu on company user name press

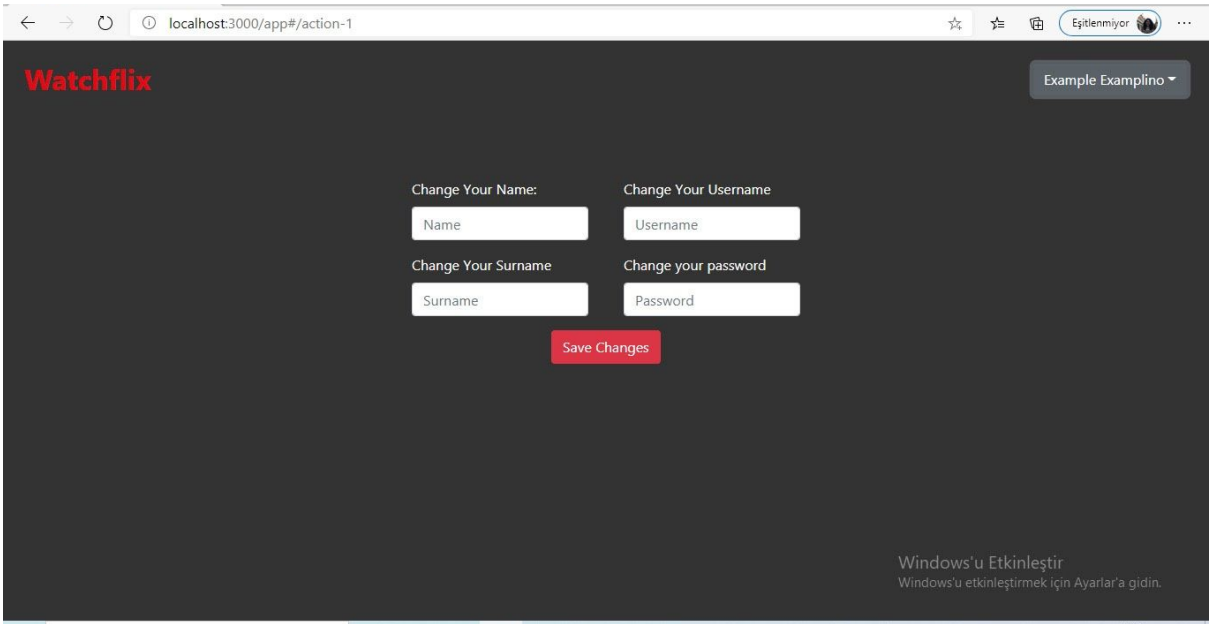

Figure 24. Company User Profile Settings

Same as for normal users. Company Users can change the above fields of their profiles. Username should be unique, otherwise an error is prompted under the “Save Changes” button.

7. Website

https://javidhaji-zada.github.io/Watchflix/