**MovieList SRS**

Developed by Melnychuk, Saienko, Horin

**MovieList** – a web application designed for users who want to keep track of their past movie experiences and plan future ones.

[Try MovieList App](https://movielist-eqfngja8bddfcqh4.westeurope-01.azurewebsites.net/)

**User Roles and Use Case**

Guest – users who are not logged in have limited access to application functionality.

User – users who registered and logged in into account have functionality for convenient usage.

Admin – users who have special account with extended functionality for maintaining the platform.

Зображення, що містить знімок екрана, чорно-білий, чорний, мистецтво

Вміст, створений ШІ, може бути неправильним.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Role | | | UseCase | Description | Action | Tests |
| Admin | User | Guest | **Registration** | **Guest can create an account.** *The user enters their details (name, email, password) and clicks "Register". After successful registration, they gain access to their personal account.* | Register(RegisterViewModel model) |  |
| **Login** | **Guest can log in to the account.** *The user enters their email and password, then clicks "Login". After successful authentication, they are redirected to the homepage.* | Login(LoginViewModel model) |  |
| **View rating** | **Guest can view the movie ratings.** *The user opens a movie page and sees the rating displayed as a number.* | ViewRating() |  |
| **Search movies** | **Guest can search movies in rating.** *The user enters a movie infornation in the search bar and clicks "Search". They then see a list of movies matching the query.* | Search(string searchTerm) |  |
|  | **Logout** | **User can log out of their account.** *The user clicks the "Logout" button in their profile. After logging out, their session ends, and they are redirected to the login page.* | Logout() |  |
| **View Profile** | **User can see his profile information.** *User can go to tab Profile and see his email and username.* | Details() |  |
| **View Movie Information** | **User can see eacth movie full information.** *User can click on any movie and see its movie full information.* | Details(int id) |  |
| **Add to Seen It!** | **User can add movies to their personal watchedlist.** *The user opens a movie page and clicks the "Add to Seen It!" button. The movie is then added to their personal "Seen It!" list.* | AddSeenIt(int id) |  |
| **Add to To Watch!** | **User can add movies they want to watch to their personal list.** *The user opens a movie page and clicks the "Add to To Watch!" button. The movie is then added to their personal "To Watch!" list.* | AddToWatch(int movieId) |  |
| **View Seen It!** | **"User can view his Seen It! list.** *The user can go to tab Seen It! and see his own list of movies he marked as Seen It!".* | GetAllSeenIt() |  |
| **View To Watch!** | **"User can view his To Watch! list.** *The user can go to tab To Watch! and see his own list of movies he marked as To Watch!".* | GetAllToWatch() |  |
| **Remove from lists** | **User can remove movies from their personal list where movie is registered.**  *The user clicks on button Remove from list and movie is removed from Seen It! or To Watch! dependind where it listed at thst moment.* | RemoveFromLists(int id) |  |
| **Add movies** | **If the user can`t find the movie they want, they can add it themselves.**  *The user fills out a form with movie details (title, description, release year, etc.) and clicks "Add". The movie is then added to the database.* | Create(MovieViewModel movieViewModel) |  |
| **Rate movie** | **User can rate from 1 to 10 movies they have watched.  *The user selects a movie from their "Seen It!" list, assigns a rating from 1 to 10, and clicks "Submit". The rating is reflected in the movie’s overall score.*** | RateMovie(int movieId, int rating) |  |
| **Search throught lists** | **User can serch movies among movies in Seen It! or To Watcth! lists** *The user can enter information in serch bar and click button Serch and get result of his query.* | SearchInList(string title, string listType) |  |
| **Report false information** | **A user can report incorrect movie information.**  *The user finds a movie with incorrect information, clicks the "Report Error" button, fills out a form describing the issue, and clicks "Send". The report is sent to the administrators.* | Create(ReportViewModel reportViewModel) |  |
|  |  | **View All Users** | **Admin can view all registerd users.** *Admin can go to tab Users to see the list of registerd users.* | GetAll() |  |
| **Delete a user** | **Admin can delete users.** *The admin finds a user in the list, clicks the "Delete" button, and confirms the action. The user’s account is removed from the system.* | DeleteUser(string userId) |  |
| **View recent activity** | **Admin can view recently added movies.** *The admin opens the page with a list of recently added movies and views their details.* | GetActivity(DateTime? startDate, DateTime? endDate) |  |
| **Edit a movie** | **Admin can edit movie information.** *The admin finds a movie, clicks the "Edit" button, makes changes, and saves them. The updated information is displayed on the movie page.* | Update(int id, MovieViewModel movieViewModel) |  |
| **Remove movie** | **Admin can delete movies.** *The admin finds a movie, clicks the "Delete" button, and confirms the action. The movie is removed from the database.* | Delete(int id) |  |
| **Ban a user** | **Admin can block and unblock users.**  *The admin finds a user, clicks the "Block" or "Unblock" button. The user’s status is updated accordingly.* | Ban(string id) |  |
| **View reports** | **Admin can view user reports of incorrect movie** **information.**  *The admin opens the reports page, where they can view details of each report.* | GetAll() |  |
| **Delete report** | **Admin can delete reports of incorrect movie information.**  *The admin opens the reports page, where they can delete report.* | Delete(int id) |  |
| **Filtre report** | **Admin can filter reports by the creation date.**  *The admin opens the reports page, where they can filter report.* | Filter(DateTime? startDate, DateTime? endDate) |  |
| **Add Genre** | **Admin can add new Genre.** *The admin opens a Genre page where he can create new Genre.* | Create(GenreViewModel genreViewModel) |  |
| **View Genres** | **Admin can view exisiting Genres.** *The admin opens a Genre page where he can view all Genres.* | GetAll() |  |
| **Edit Genre** | **Admin can edit Genre.** *The admin opens a page where he can edit existing Genre.* | Update(int id, GenreViewModel genreViewModel) |  |
| **Delete Genre** | **Admin can delete existing Genre.** *The admin opens a Genre page where he can delete existing Genre.* | Delete(int id) |  |
| **Add Country** | **Admin can add new Country.** *The admin opens a Country page where he can create new Country.* | Create(CountryViewModel countryViewModel) |  |
| **View Countries** | **Admin can view exisiting Countries.** *The admin opens a Country page where he can view all Countries.* | GetAll() |  |
| **Edit Country** | **Admin can edit Country.** *The admin opens a page where he can edit existing Country.* | Update(int id) |  |
| **Delete Country** | **Admin can delete existing Country.** *The admin opens a Country page where he can delete existing Country.* | Delete(int id) |  |

Additional information about table:

* Each user role has access to the functionalities assigned to it, as well as those designated for the roles above it
* **Usecases in bold** have the highest priority in the development process