HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
NHẬP MÔN CÔNG NGHỆ PHẦN MỀM
INTRODUCTION TO SOFTWARE ENGINEERING

-----oOo-----

Sinh Viên: **Phạm Thị Ngọc Thảo**
Mã số sinh viên: **K24DTCN256**

I. THIẾT KẾ PHẦN MỀM (SOFTWARE DESIGN)

Yêu cầu bài toán:

- Xây dựng một website xem phim online cho phép người dùng:
  - Xem danh sách phim theo thể loại
  - Xem thông tin chi tiết của phim
  - Xem phim trực tuyến
  - Tìm kiếm phim theo tên, diễn viên, thể loại, tác giả, năm phát hành,…
- Người dùng có thể đăng ký/ đăng nhập vào hệ thống.
  - Thêm/loại bỏ phim vào/khỏi danh sách yêu thích

1. Xác định các lớp (Class)

1.1. Phương pháp Noun Phrase Approach

Các danh từ (Noun phrases)

- Phim (Movie)
- Thể loại (Genre)
- Xếp loại (Ranking)
- Người dùng (User)

Ta sẽ có các lớp (Class) sau:

- Movie
- Genre
- Ranking
- User

1.2. Use Case Driven Method

Use-Case chính:

- UC1: Quản lý danh sách phim
- UC2: Quản lý danh sách thể loại
- UC3: Chi tiết phim
- UC4: Tìm kiếm phim
- UC5: Đăng ký người dùng
- UC6: Đăng nhập hệ thống
- UC7: Quản lý phim yêu thích

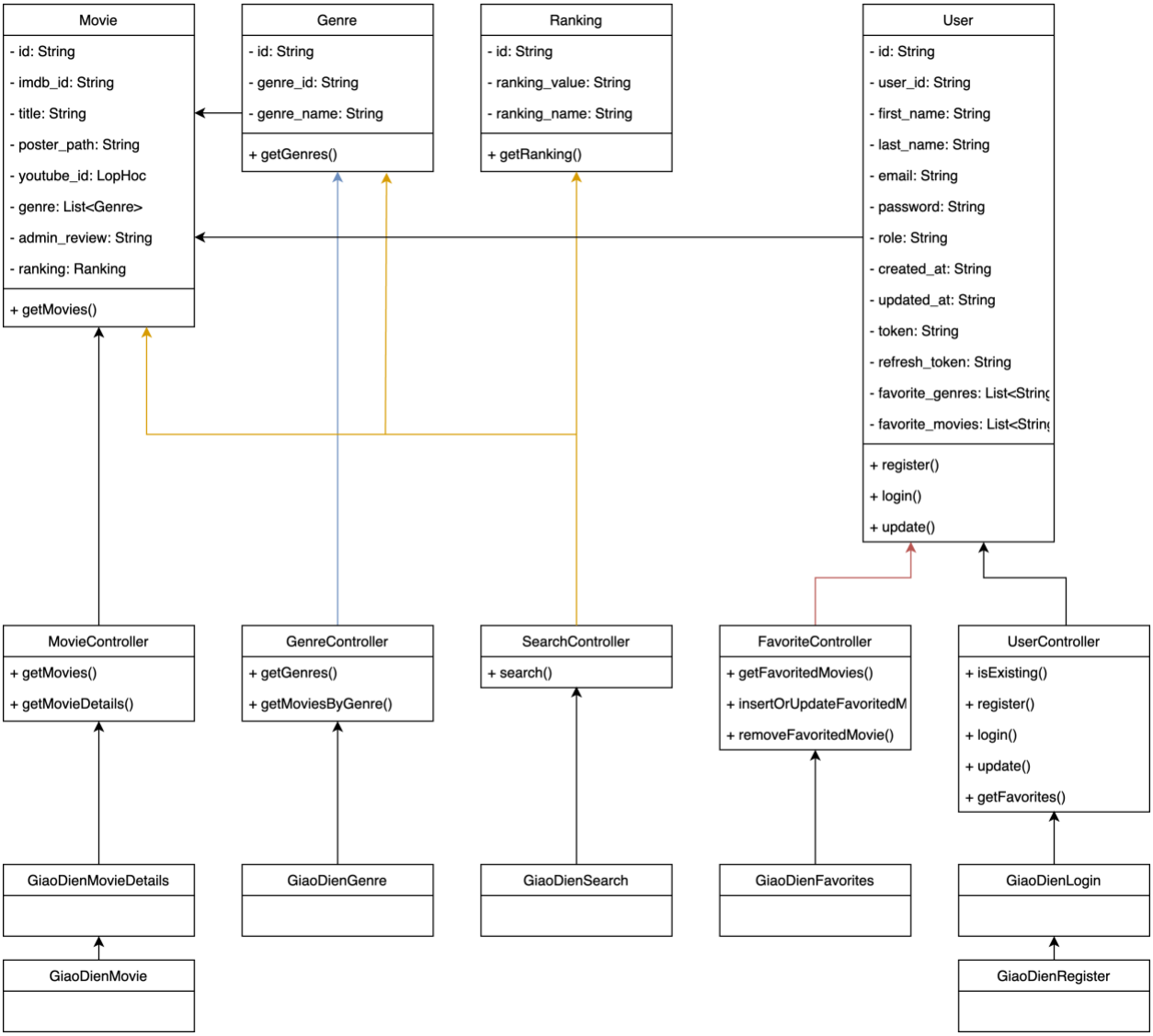
Các lớp (Class) liên quan được rút ra:

- UC1 “Quản lý danh sách phim” → Movie
- UC2 “Quản lý danh sách thể loại” → Genre, Ranking
- UC3 “Chi tiết phim” → Movie
- UC4 “Tìm kiếm phim” → Movie, Genre, Ranking
- UC5 “Đăng ký người dùng” → User
- UC6 “Đăng nhập hệ thống” → User
- UC7 “Quản lý phim yêu thích” → Movie

2. Phân loại lớp (Class)

Nhóm	Lớp	Mô tả
Entity class	Movie, Genre Ranking, User	Lưu trữ thông tin dữ liệu trong hệ thống
Boundary class	GiaoDienMovie, GiaoDienMovieDetails, GiaoDienGenre, GiaoDienFavorites GiaoDienSearch, GiaoDienRegister, GiaoDienLogin	Giao diện người dùng
Control class	MovieController, GenreController, SearchController, UserController, FavoriteController	Xử lý logic nghiệp vụ

3. Class Diagram



Bảng mô tả chức năng từng lớp:

Lớp	Vai trò	Chức năng
Movie	Entity	Quản lý thông tin phim
Genre	Entity	Quản lý thông tin thể loại
Ranking	Entity	Quản lý thông tin đánh giá
User	Entity	Quản lý thông tin người dùng
MovieController	Control	Xử lý thông tin phim
GenreController	Control	Xử lý thông tin thể loại
SearchController	Control	Xử lý thông tin đánh giá
FavoriteController	Control	Xử lý thông tin yêu thích phim
UserController	Control	Xử lý thông tin người dùng
GiaoDienMovie	Boundary	Giao diện danh sách phim
GiaoDienMovieDetails	Boundary	Giao diện chi tiết phim

GiaoDienGenre	Boundary	Giao diện thể loại phim
GiaoDienSearch	Boundary	Giao diện cho việc tìm kiếm
GiaoDienFavorites	Boundary	Giao diện quản lý phim được yêu thích
GiaoDienLogin	Boundary	Giao diện người dùng đăng nhập
GiaoDienRegister	Boundary	Giao diện đăng ký người dùng

II. Biểu đồ hoạt động Activity Diagram

1. Use Case 1: Quản lý danh sách phim

1.1. Precondition

N/A

1.2. Main Flow

- Người dùng mở ứng dụng.
- Người dùng chọn tab Home.
- Hệ thống hiển thị giao diện danh sách phim.

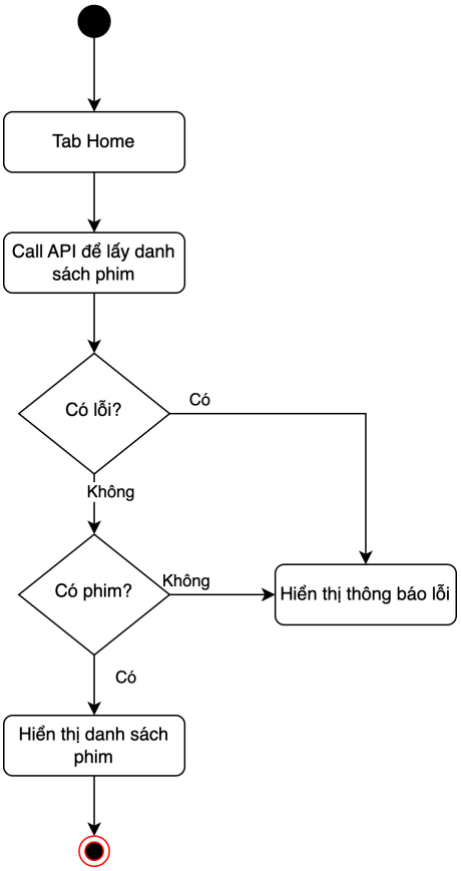
1.3. Sub-flows

N/A

1.4. Alternative Flows

- Nếu không có phim nào trong hệ thống, hiển thị thông báo “Không tìm thấy thông tin phim”.
- Nếu hệ thống có lỗi, hiển thị thông báo lỗi.

1.5. Activity Diagram



2. Use Case 2: Quản lý danh sách thể loại

2.1. Precondition

N/A

2.2. Main Flow

- Người dùng mở ứng dụng.
- Người dùng chọn tab Genre.
- Hệ thống hiển thị giao diện danh sách thể loại phim.

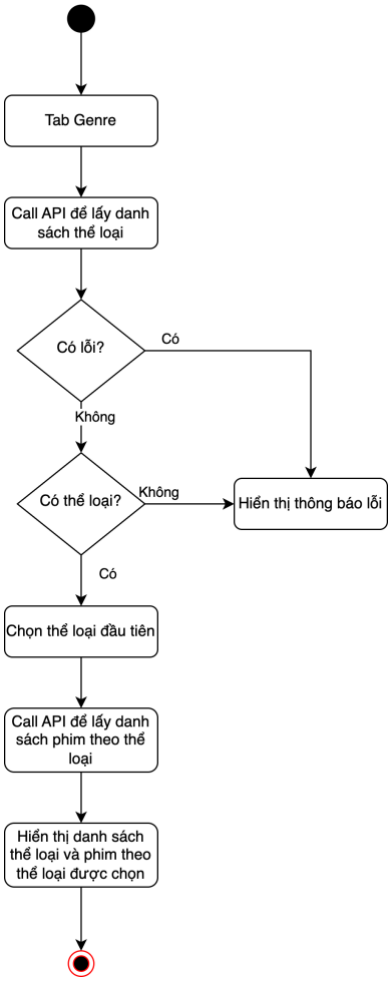
2.3. Sub-flows

N/A

2.4. Alternative Flows

- Nếu không có thể loại / phim nào trong thể loại được chọn trong hệ thống, hiển thị thông báo "Không tìm thấy thông tin phim".
- Nếu hệ thống có lỗi, hiển thị thông báo lỗi.

2.5. Activity Diagram



3. Use Case 3: Chi tiết phim

3.1. Precondition

Danh sách phim từ trang chủ / thể loại

3.2. Main Flow

- Người dùng click vào 1 phim trên danh sách.
- Hệ thống hiển thị giao diện chi tiết của phim.

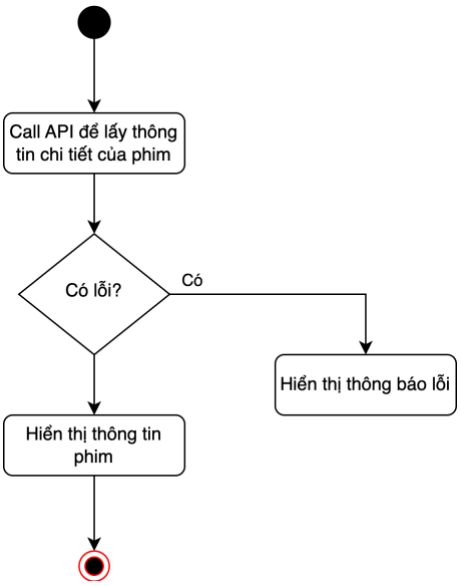
3.3. Sub-flows

N/A

3.4. Alternative Flows

- Nếu hệ thống có lỗi, hiển thị thông báo lỗi.

3.5. Activity Diagram



4. Use Case 4: Tìm kiếm phim

4.1. Precondition

N/A

4.2. Main Flow

- Người dùng mở ứng dụng.
- Người dụng chọn vào ô tìm kiếm, nhập thông tin và nhấn vào Tìm kiếm.
- Hệ thống hiển thị giao diện danh sách phim tìm được.

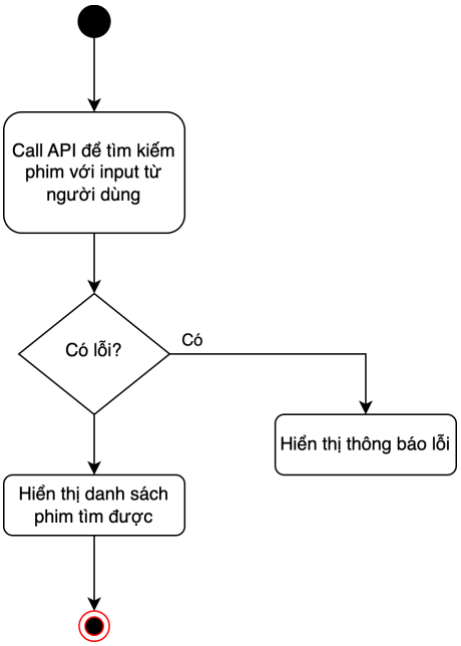
4.3. Sub-flows

N/A

4.4. Alternative Flows

- Nếu không tìm được phim trong hệ thống, hiển thị thông báo “Không tìm thấy thông tin phim”.
- Nếu hệ thống có lỗi, hiển thị thông báo lỗi.

4.5. Activity Diagram



5. Use Case 5: Đăng ký người dùng

5.1. Precondition

Thông tin người dùng chưa có trong hệ thống.

5.2. Main Flow

- Người dùng mở ứng dụng.
- Người dụng chọn vào chức năng Đăng ký.
- Hệ thống hiển thị giao diện để nhập thông tin đăng ký người dùng mới.

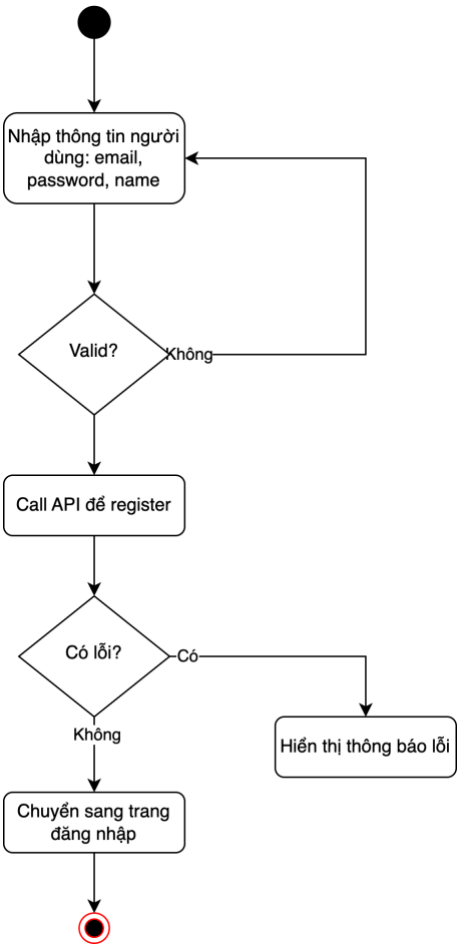
5.3. Sub-flows

N/A

5.4. Alternative Flows

- Thông tin email, password, name là bắt buộc.
- Nếu email đã tồn tại trong hệ thống, thông báo đến người dùng.
- Nếu đăng ký không thành công / hệ thống có lỗi, hiển thị thông báo lỗi.

5.5. Activity Diagram



6. Use Case 6: Đăng nhập hệ thống

6.1. Precondition

Người dùng đã đăng ký thông tin và chưa đăng nhập vào hệ thống.

6.2. Main Flow

- Người dùng mở ứng dụng.
- Người dụng chọn chức năng Đăng nhập.
- Hệ thống hiển thị giao diện để nhập thông tin đăng nhập.

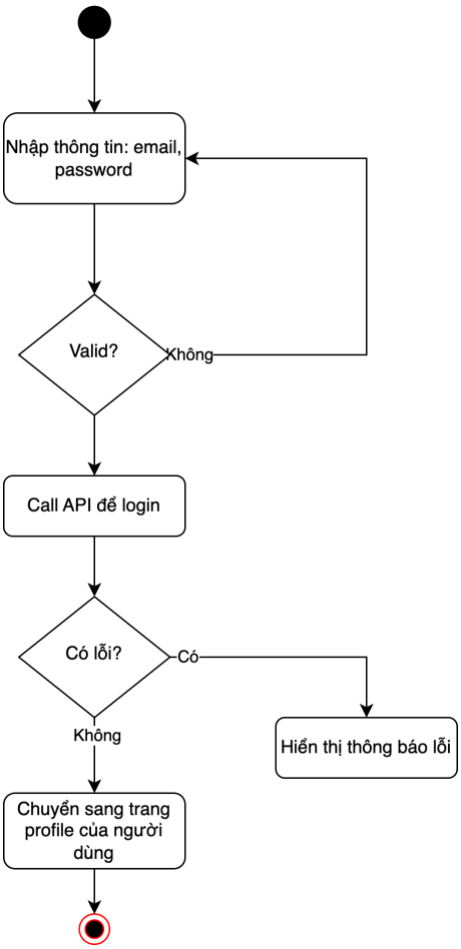
6.3. Sub-flows

N/A

6.4. Alternative Flows

- Thông tin email, password là bắt buộc.
- Nếu email chưa tồn tại trong hệ thống, thông báo đến người dùng.
- Nếu đăng nhập không thành công / hệ thống có lỗi, hiển thị thông báo lỗi.

6.5. Activity Diagram



7. Use Case 7: Quản lý phim yêu thích

7.1. Precondition

Người dùng đã đăng nhập vào hệ thống.

7.2. Main Flow

- Người dùng mở ứng dụng.
- Người dụng chọn tab Favorite.
- Hệ thống hiển thị giao diện danh sách phim mà người dùng đã lưu.

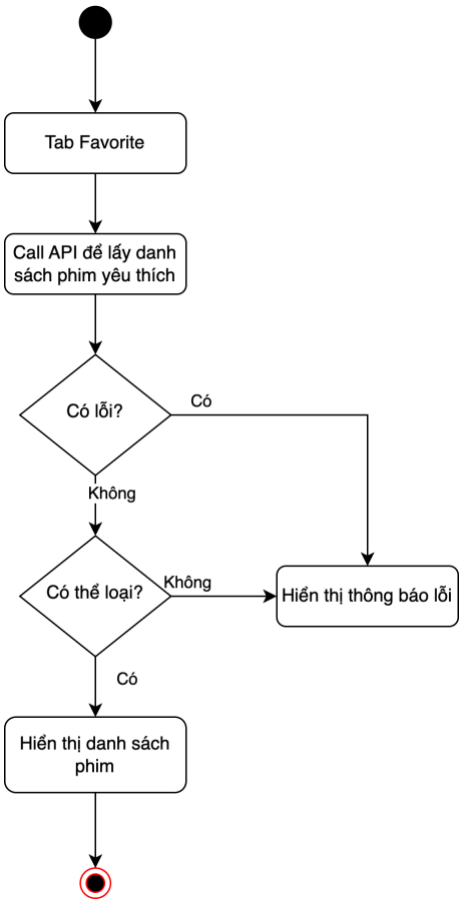
7.3. Sub-flows

N/A

7.4. Alternative Flows

- Nếu không có phim nào được lưu yêu thích trong hệ thống, hiển thị thông báo “Không tìm thấy thông tin phim”.
- Nếu hệ thống có lỗi, hiển thị thông báo lỗi.

7.5. Activity Diagram



III. Biểu đồ tương tác Sequence Diagram

1. UC1: Quản lý danh sách phim

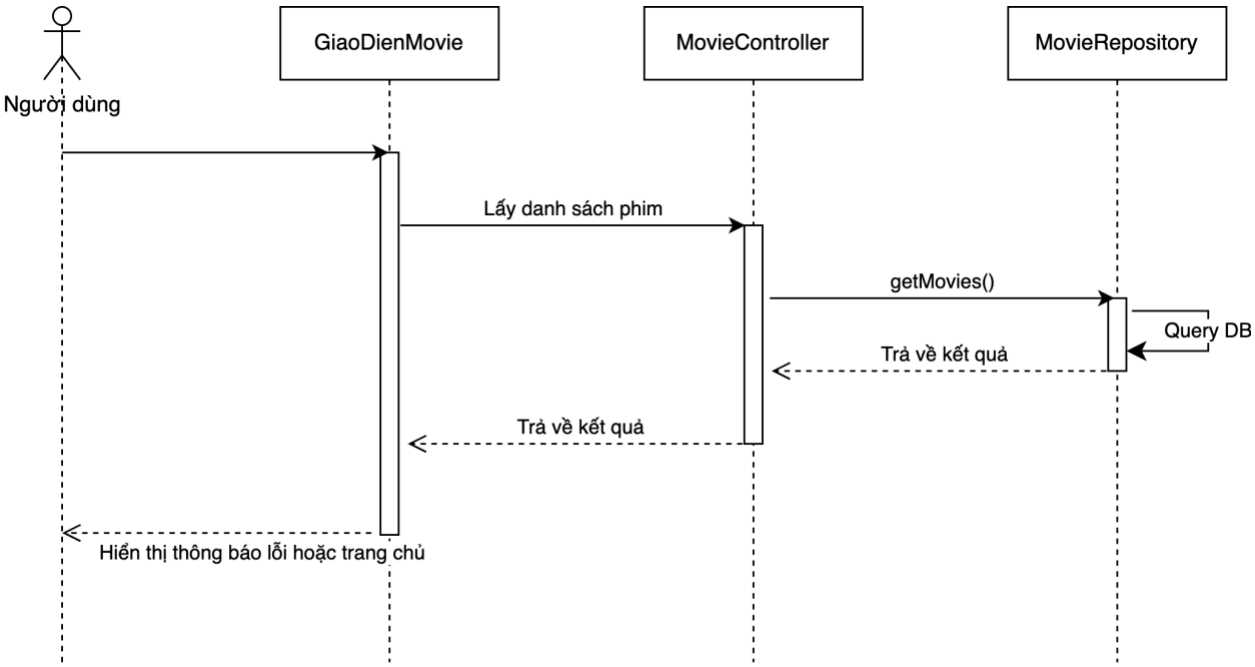
1.1. Đối tượng tham gia

Loại	Đối tượng	Chức năng
Actor	Người dùng	Người dùng thực hiện việc duyệt tab Home
Boundary	GiaoDienMovie	Giao diện sẽ hiển thị danh sách phim
Control	MovieController	Lớp điều khiển xử lý logic lấy và hiển thị danh sách phim
Entity	Movie, MovieRepository	Lớp chứa dữ liệu về phim

1.2. Mô tả luồng trình tự

- Người dùng mở ứng dụng ở trang Home, hoặc click vào tab Home.
- GiaoDienMovie gửi thông tin lấy danh sách phim đến MovieController.
- MovieController gọi phương thức getMovies() để lấy thông tin danh sách phim.
- MovieRepository trả về kết quả.
- MovieController gửi kết quả lại cho GiaoDienMovie.
- Nếu có kết quả → hiển thị thông tin. Nếu có lỗi → hiển thị thông báo lỗi.

1.3. Diagram



2. UC2: Quản lý danh sách thể loại

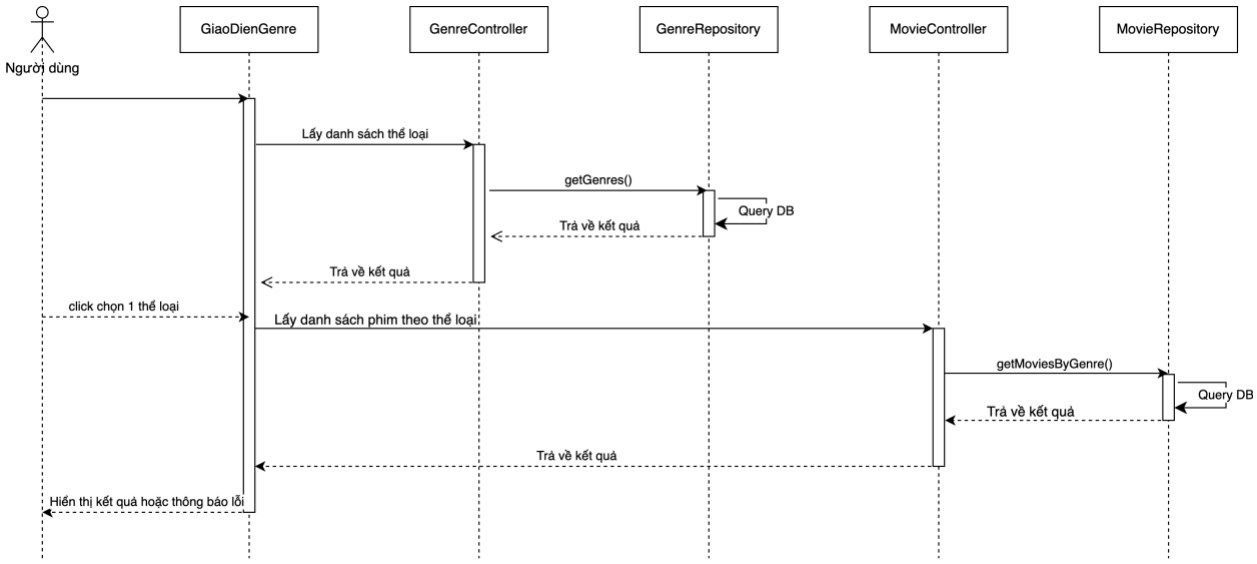
2.1. Đối tượng tham gia

Loại	Đối tượng	Chức năng
Actor	Người dùng	Người dùng thực hiện việc duyệt tab Genre
Boundary	GiaoDienGenre	Giao diện sẽ hiển thị danh sách thể loại và phim trong từng thể loại
Control	GenreController, MovieController	Lớp điều khiển xử lý logic lấy và hiển thị danh sách thể loại và phim trong thể loại
Entity	Genre, GenreRepository, Movie, MovieRepository	Lớp chứa dữ liệu về thể loại và phim

2.2. Mô tả luồng trình tự

- Người dùng click vào tab Genre.
- GiaoDienGenre gửi thông tin lấy danh sách thể loại đến GenreController.
- GenreController gọi phương thức getGenres() để lấy thông tin danh sách thể loại.
- GenreRepository trả về kết quả.
- GenreController gửi kết quả lại cho GiaoDienGenre.
- GiaoDienGenre gửi thông tin lấy danh sách phim cho thể loại đầu tiên / hoặc được chọn bởi người dùng đến MovieController.
- MovieController gọi phương thức getMovieByGenre() để lấy thông tin danh sách phim.
- MovieRepository trả về kết quả.
- MovieController gửi kết quả lại cho GiaoDienGenre.
- Nếu có kết quả → hiển thị thông tin. Nếu có lỗi → hiển thị thông báo lỗi.

2.3. Diagram



3. UC3: Chi tiết phim

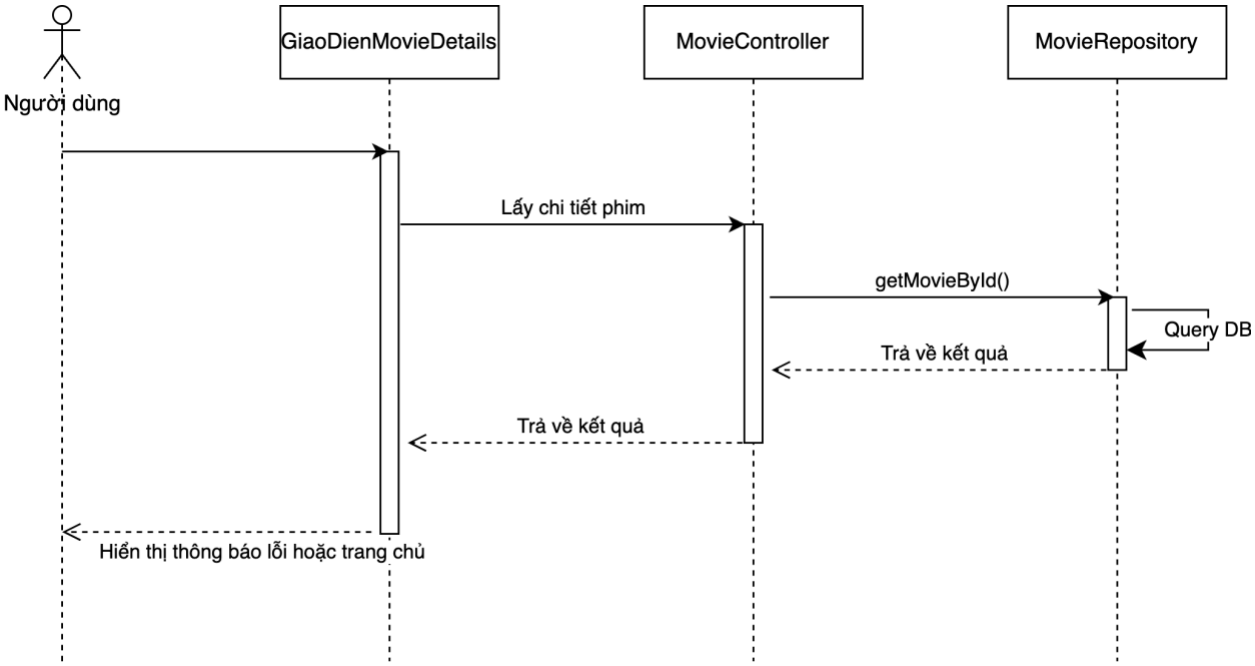
3.1. Đối tượng tham gia

Loại	Đối tượng	Chức năng
Actor	Người dùng	Người dùng thực hiện việc xem chi tiết phim
Boundary	GiaoDienMovieDetails	Giao diện sẽ hiển thị thông tin chi tiết phim
Control	MovieController	Lớp điều khiển xử lý logic lấy chi tiết phim
Entity	Movie, MovieRepository	Lớp chứa dữ liệu về phim

3.2. Mô tả luồng trình tự

- Người dùng click vào 1 phim ở danh sách phim.
- GiaoDienMovieDetails gửi thông tin lấy thông tin chi tiết phim đến MovieController.
- MovieController gọi phương thức getMovieById() để lấy thông tin chi tiết phim.
- MovieRepository trả về kết quả.
- MovieController gửi kết quả lại cho GiaoDienMovieDetails.
- Nếu có kết quả → hiển thị thông tin. Nếu có lỗi → hiển thị thông báo lỗi.

3.3. Diagram



4. UC4: Tìm kiếm phim

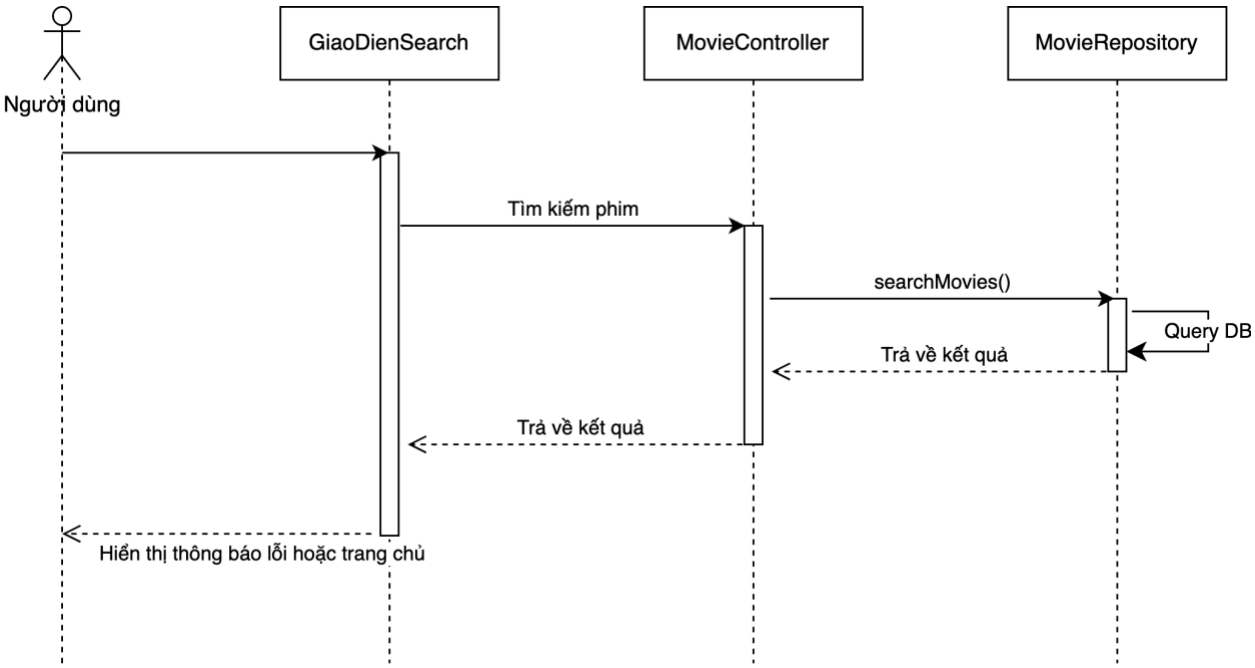
4.1. Đối tượng tham gia

Loại	Đối tượng	Chức năng
Actor	Người dùng	Người dùng thực hiện việc tìm kiếm phim
Boundary	GiaoDienSearch	Giao diện sẽ hiển thị thông tin chi tiết phim
Control	MovieController	Lớp điều khiển xử lý logic tìm kiếm phim
Entity	Movie, MovieRepository	Lớp chứa dữ liệu về phim

4.2. Mô tả luồng trình tự

- Người dùng click vào search để bắt đầu tìm kiếm phim.
- GiaoDienSearch gửi thông tin tìm kiếm phim đến MovieController.
- MovieController gọi phương thức searchMovies() để lấy tìm kiếm phim.
- MovieRepository trả về kết quả.
- MovieController gửi kết quả lại cho GiaoDienSearch.
- Nếu có kết quả → hiển thị thông tin. Nếu có lỗi → hiển thị thông báo lỗi.

4.3. Diagram



5. UC5: Đăng ký người dùng

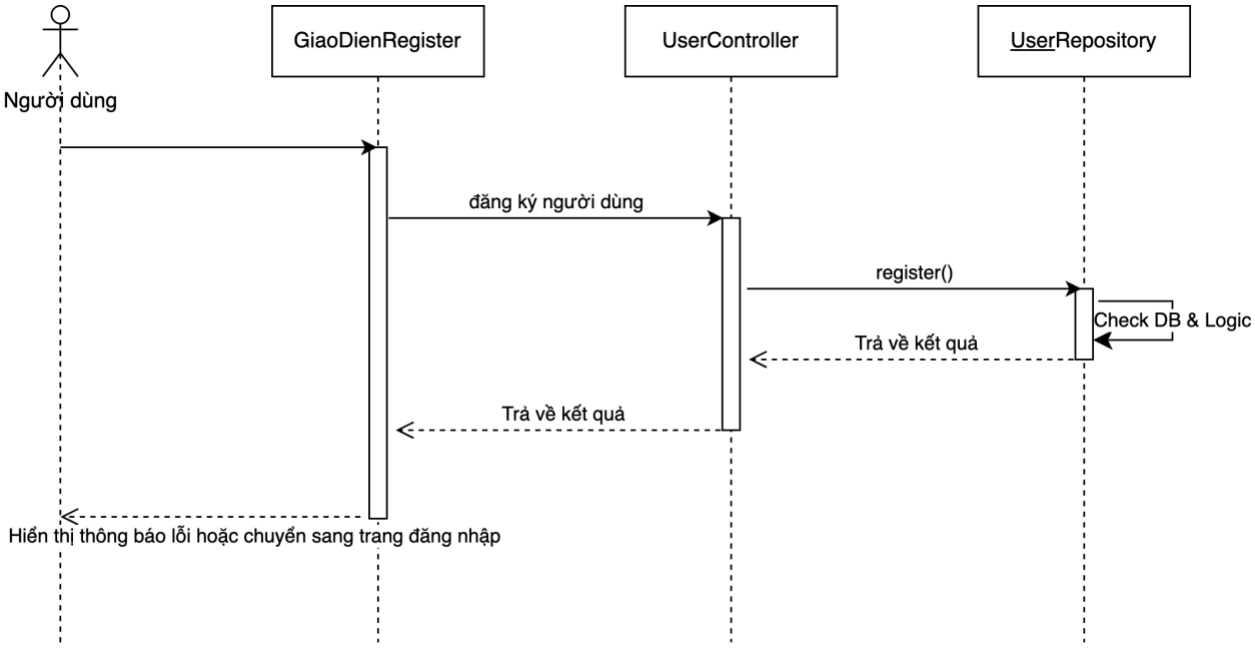
5.1. Đối tượng tham gia

Loại	Đối tượng	Chức năng
Actor	Người dùng	Người dùng thực hiện việc đăng ký thông tin
Boundary	GiaoDienRegister	Giao diện sẽ hiển thị thông tin đăng ký người dùng
Control	UserController	Lớp điều khiển xử lý logic liên quan người dùng
Entity	User, UserRepository	Lớp chứa dữ liệu về người dùng

5.2. Mô tả luồng trình tự

- Người dùng click vào Đăng ký để bắt đầu đăng ký.
- GiaoDienRegister gửi thông tin người dùng nhập vào đến UserController.
- UserController gọi phương thức register() để đăng ký.
- UserRepository kiểm tra thông tin người dùng, trả về kết quả.
- UserController gửi kết quả lại cho GiaoDienRegister.
- Nếu thành công → hiển thị thông tin. Nếu có lỗi → hiển thị thông báo lỗi.

5.3. Diagram



6. UC6: Người dùng đăng nhập

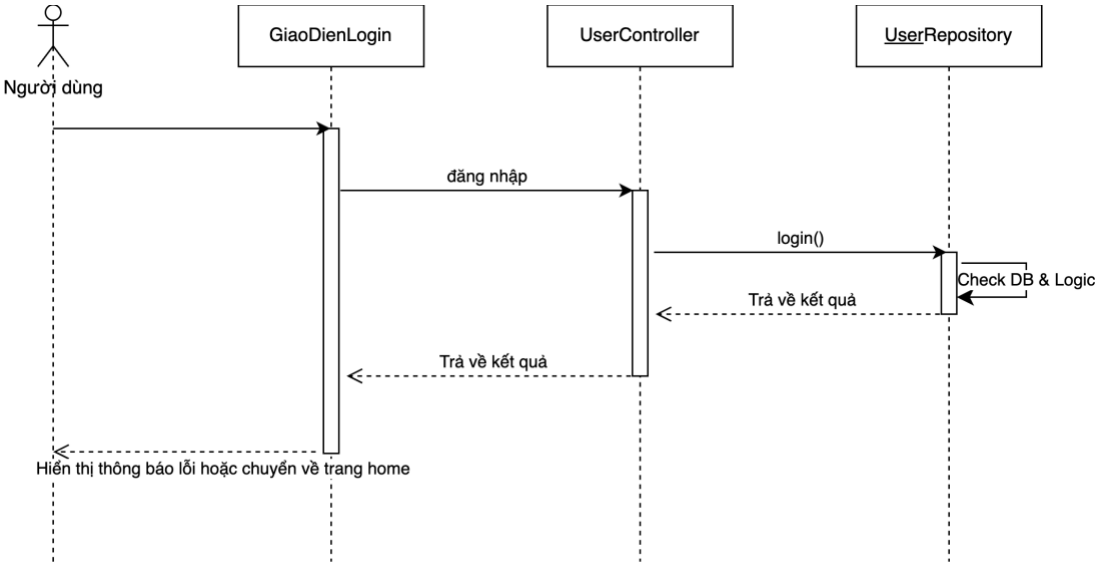
6.1. Đối tượng tham gia

Loại	Đối tượng	Chức năng
Actor	Người dùng	Người dùng thực hiện việc đăng nhập
Boundary	GiaoDienLogin	Giao diện sẽ hiển thị thông tin người dùng đăng nhập
Control	UserController	Lớp điều khiển xử lý logic liên quan người dùng
Entity	User, UserRepository	Lớp chứa dữ liệu về người dùng

6.2. Mô tả luồng trình tự

- Người dùng click vào Đăng nhập để bắt đầu đăng nhập.
- GiaoDienLogin gửi thông tin người dùng nhập vào đến UserController.
- UserController gọi phương thức login() để đăng nhập.
- UserRepository kiểm tra thông tin người dùng nhập, trả về kết quả.
- UserController gửi kết quả lại cho GiaoDienLogin.
- Nếu thành công → hiển thị thông tin và chuyển về trang chủ. Nếu có lỗi → hiển thị thông báo lỗi.

6.3. Diagram



7. UC7: Quản lý phim yêu thích

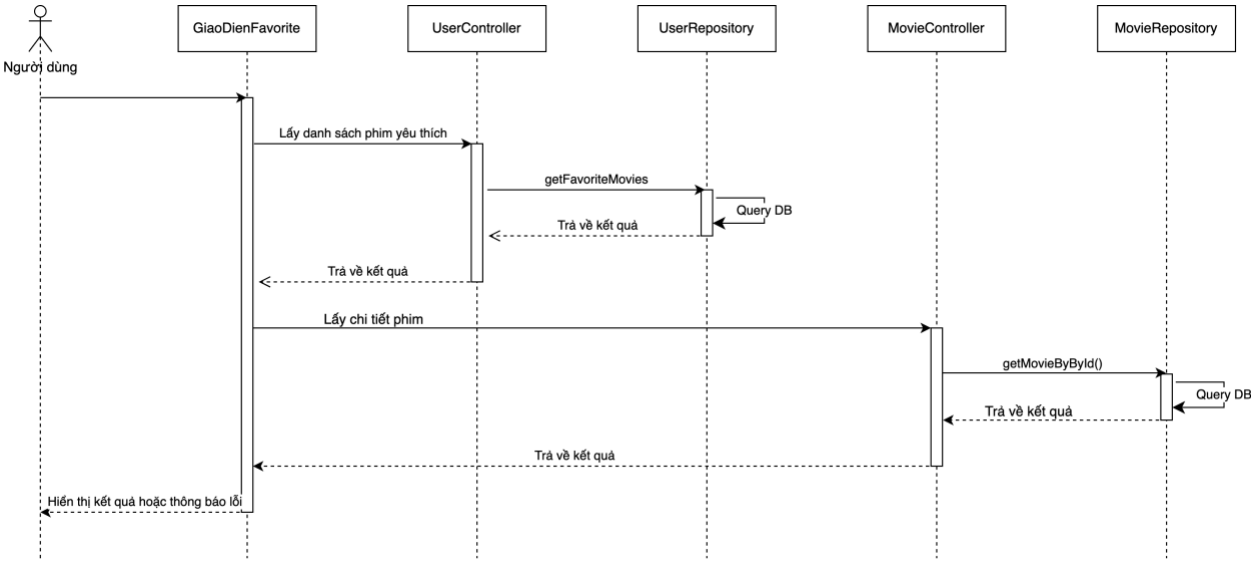
7.1. Đối tượng tham gia

Loại	Đối tượng	Chức năng
Actor	Người dùng	Người dùng đã thực hiện việc đăng nhập, sau đó click vào tab Favorite
Boundary	GiaoDienFavorites	Giao diện sẽ hiển thị danh sách phim đã được lưu là yêu thích
Control	UserController	Lớp điều khiển xử lý logic liên quan phim yêu thích của người dùng
Entity	User, UserRepository, Movie, MovieRepository	Lớp chứa dữ liệu về người dùng, phim

7.2. Mô tả luồng trình tự

- Người dùng click vào tab Favorites để xem danh sách phim đã yêu thích.
- GiaoDienFavorites gửi thông tin người dùng đã đăng nhập đến UserController.
- UserController gọi phương thức getFavoriteMovies().
- UserRepository kiểm tra thông tin người dùng nhập, trả về kết quả.
- UserController trả kết quả về GiaoDienFavorites.
- GiaoDienFavorites gửi thông tin danh sách phim đến MovieController.
- MovieController gọi phương thức getMovieById().
- MovieRepository kiểm tra thông tin phim, trả về kết quả.
- MovieController trả kết quả về GiaoDienFavorites.
- Nếu có kết quả → hiển thị thông tin. Nếu có lỗi → hiển thị thông báo lỗi.

7.3. Diagram



IV. Biểu đồ tương tác Collaboration Diagram

1. UC1: Quản lý danh sách phim

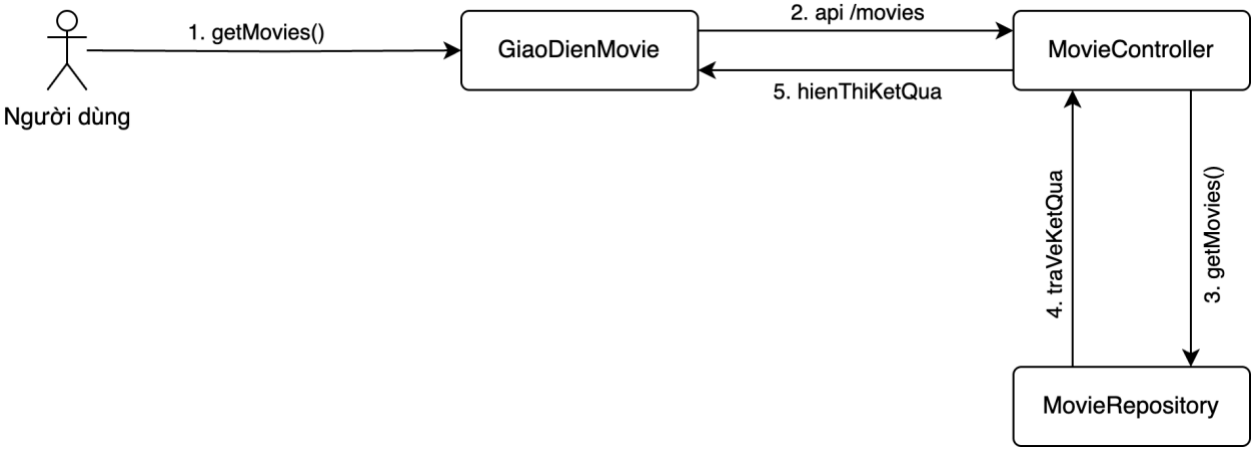
1.1. Đối tượng tham gia

- :Người dùng
- :GiaoDienMovie
- :MovieController
- :MovieRepository, :Movie

1.2. Thông điệp (đánh số thứ tự)

- Người dùng → GiaoDienMovie: 1. getMovies()
- GiaoDienMovie → MovieController: 2. api /movies
- MovieController → MovieRepository: 3. getMovies()
- MovieRepository → MovieController: 4. traVeKetQua()
- MovieController → GiaoDienMovie: 5. hienThiKetQua ()

1.3. Diagram



1.4. Ghi chú các thông điệp chính

STT	Thông điệp	Nguồn → Đích	Ý nghĩa
1	getMovies ()	Người dùng → GiaoDienMovie	Người dùng vào trang Home hoặc click vào tab Home để bắt đầu quy trình
2	Api /movies	GiaoDienMovie → MovieController	Giao diện call api sang lớp điều khiển để xử lý
3	getMovies()	MovieController → MovieRepository	Truy xuất thông tin dữ liệu phim trong hệ thống
4	traVeKetQua()	MovieRepository → MovieController	Trả kết quả

5	hienThiKetQua()	MovieController → GiaoDienMovie	Hiển thị kết quả (hiển thị lỗi hoặc mở trang chủ với danh sách phim)

2. UC2: Quản lý danh sách thể loại

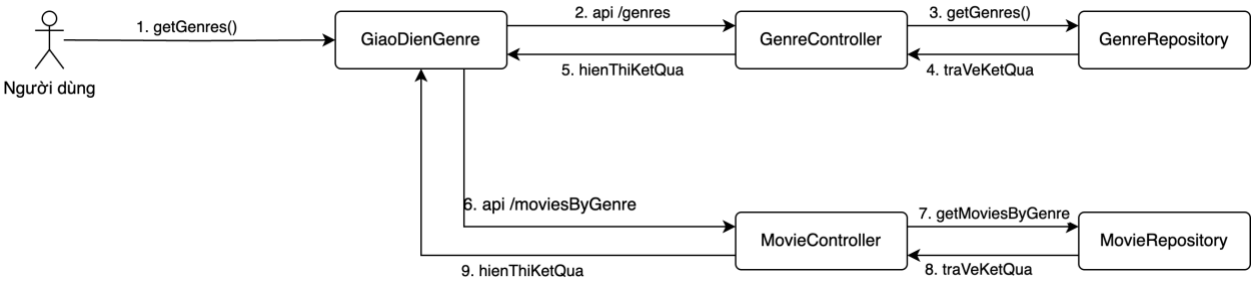
2.1. Đối tượng tham gia

- :Người dùng
- :GiaoDienGenre
- :GenreController, :MovieController
- :GenreRepository, :Genre, :MovieRepository, :Movie

2.2. Thông điệp (đánh số thứ tự)

- Người dùng → GiaoDienGenre: 1. getGenres()
- GiaoDienGenre → GenreController: 2. api /genres
- GenreController → GenreRepository: 3. getGenres()
- GenreRepository → GenreController: 4. traVeKetQua()
- GenreController → GiaoDienGenre: 5. hienThiKetQua ()
- GiaoDienGenre → MovieController: 6. api /moviesByGenre()
- MovieController → MovieRepository: 7. getMoviesByGenre ()
- MovieRepository → MovieController: 8. traVeKetQua ()
- MovieController → GiaoDienGenre: 9. hienThiKetQua ()

2.3. Diagram



2.4. Ghi chú các thông điệp chính

STT	Thông điệp	Nguồn → Đích	Ý nghĩa
1	getGenres ()	Người dùng → GiaoDienGenre	Người dùng click vào tab Genre để bắt đầu quy trình
2	Api /genres	GiaoDienGenre → GenreController	Giao diện call api sang lớp điều khiển để xử lý
3	getGenres()	GenreController → GenreRepository	Truy xuất thông tin dữ liệu thể loại trong hệ thống

4	traVeKetQua()	GenreRepository → GenreController	Trả kết quả
5	hienThiKetQua()	GenreController → GiaoDienGenre	Hiển thị kết quả / hiển thị lỗi
6	Api /getMoviesByGenre	GiaoDienGenre → MovieController	Giao diện call api sang lớp điều khiển để xử lý
7	getMoviesByGenre()	MovieController → MovieRepository	Truy xuất thông tin phim trong hệ thống
8	traVeKetQua()	MovieRepository → MovieController	Trả kết quả
9	hienThiKetQua()	MovieController → GiaoDienGenre	Hiển thị kết quả (hiển thị lỗi hoặc tab Gen với danh sách phim)

3. UC3: Chi tiết phim

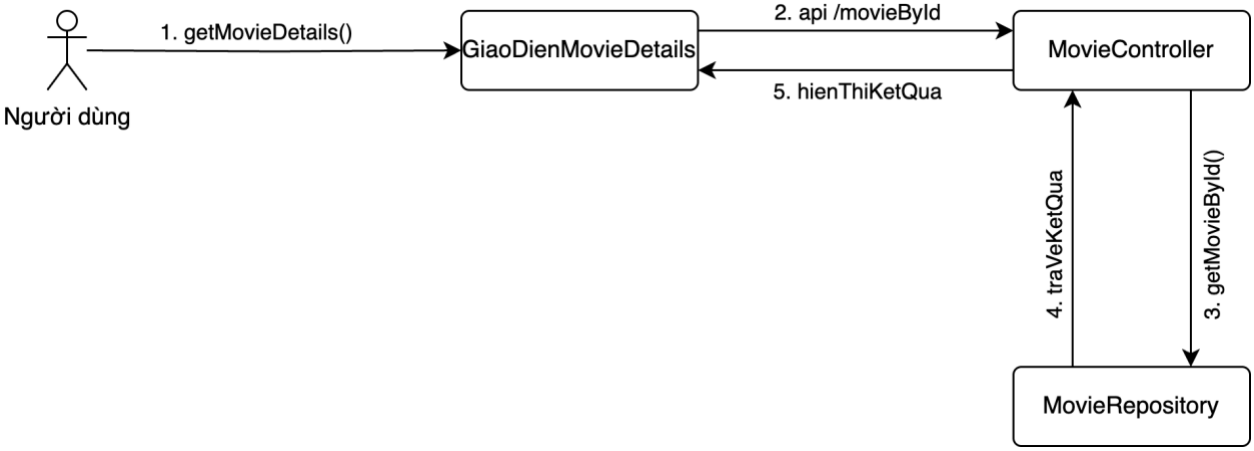
3.1. Đối tượng tham gia

- :Người dùng
- :GiaoDienMovieDetails
- :MovieController
- :MovieRepository, :Movie

3.2. Thông điệp (đánh số thứ tự)

- Người dùng → GiaoDienMovieDetails: 1. getMovieDetails()
- GiaoDienMovieDetails → MovieController: 2. api /movieById
- MovieController → MovieRepository: 3. getMovieById()
- MovieRepository → MovieController: 4. traVeKetQua()
- MovieController → GiaoDienMovieDetails: 5. hienThiKetQua ()

3.3. Diagram



3.4. Ghi chú các thông điệp chính

STT	Thông điệp	Nguồn → Đích	Ý nghĩa
1	getMovieDetails()	Người dùng → GiaoDienMovieDetails	Người dùng click vào 1 phim để bắt đầu quy trình
2	Api /movieById	GiaoDienMovieDetails → MovieController	Giao diện call api sang lớp điều khiển để xử lý
3	getMovieById()	MovieController → MovieRepository	Truy xuất thông tin dữ liệu phim trong hệ thống
4	traVeKetQua()	MovieRepository → MovieController	Trả kết quả
5	hienThiKetQua()	MovieController → GiaoDienMovieDetails	Hiển thị kết quả (hiển thị lỗi hoặc mở trang chi tiết của phim)

4. UC4: Tìm kiếm phim

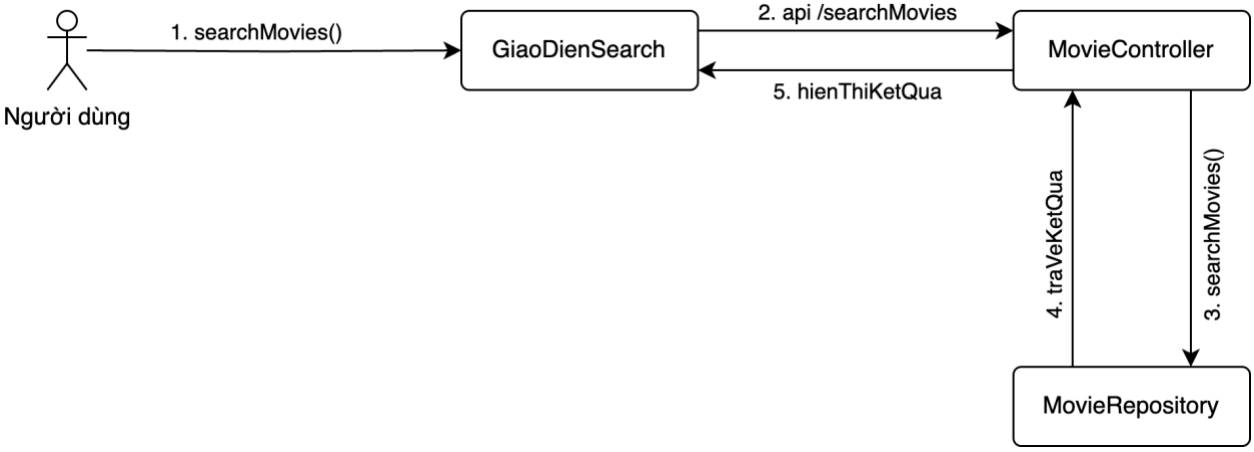
4.1. Đối tượng tham gia

- :Người dùng
- :GiaoDienSearch
- :SearchController
- :MovieRepository, :Movie

4.2. Thông điệp (đánh số thứ tự)

- Người dùng → GiaoDienSearch: 1. searchMovies()
- GiaoDienSearch → SearchController: 2. api /searchMovies
- SearchController → MovieRepository: 3. searchMovies()
- MovieRepository → SearchController: 4. traVeKetQua()
- SearchController → GiaoDienSearch: 5. hienThiKetQua ()

4.3. Diagram



4.4. Ghi chú các thông điệp chính

STT	Thông điệp	Nguồn → Đích	Ý nghĩa
1	searchMovies()	Người dùng → GiaoDienSearch	Người dùng click vào search để bắt đầu quy trình
2	Api /searchMovies	GiaoDienSearch → MovieController	Giao diện call api sang lớp điều khiển để xử lý
3	searchMovies()	MovieController → MovieRepository	Truy xuất thông tin dữ liệu phim trong hệ thống
4	traVeKetQua()	MovieRepository → MovieController	Trả kết quả
5	hienThiKetQua()	MovieController → GiaoDienSearch	Hiển thị kết quả (hiển thị lỗi hoặc danh sách phim tìm được)

5. UC5: Đăng ký người dùng

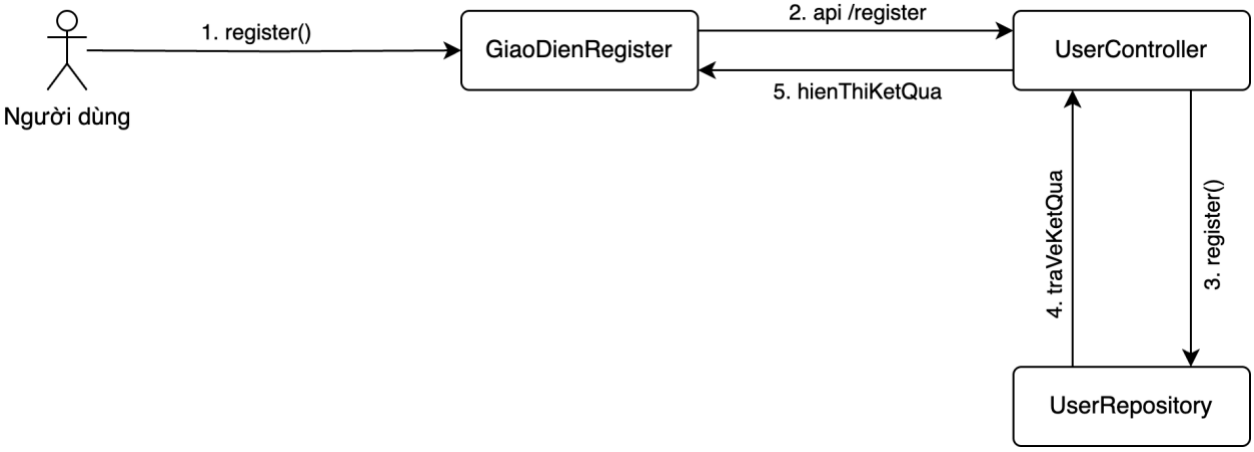
5.1. Đối tượng tham gia

- :Người dùng
- :GiaoDienRegister
- :UserController
- :UserRepository, :User

5.2. Thông điệp (đánh số thứ tự)

- Người dùng → GiaoDienRegister: 1. register()
- GiaoDienRegister → UserController: 2. api /register
- UserController → UserRepository: 3. register()
- UserRepository → UserController: 4. traVeKetQua()
- UserController → GiaoDienRegister: 5. hienThiKetQua ()

5.3. Diagram



5.4. Ghi chú các thông điệp chính

STT	Thông điệp	Nguồn → Đích	Ý nghĩa
1	register()	Người dùng → GiaoDienRegister	Người dùng click vào Đăng ký để bắt đầu quy trình
2	Api /register	GiaoDienRegister → UserController	Giao diện call api sang lớp điều khiển để xử lý
3	register()	UserController → UserRepository	Truy xuất thông tin dữ liệu người dùng trong hệ thống
4	traVeKetQua()	UserRepository → UserController	Trả kết quả
5	hienThiKetQua()	UserController → GiaoDienRegister	Hiển thị kết quả (hiển thị lỗi hoặc đăng ký thành công và chuyển sang trang Đăng nhập)

6. UC6: Người dùng đăng nhập

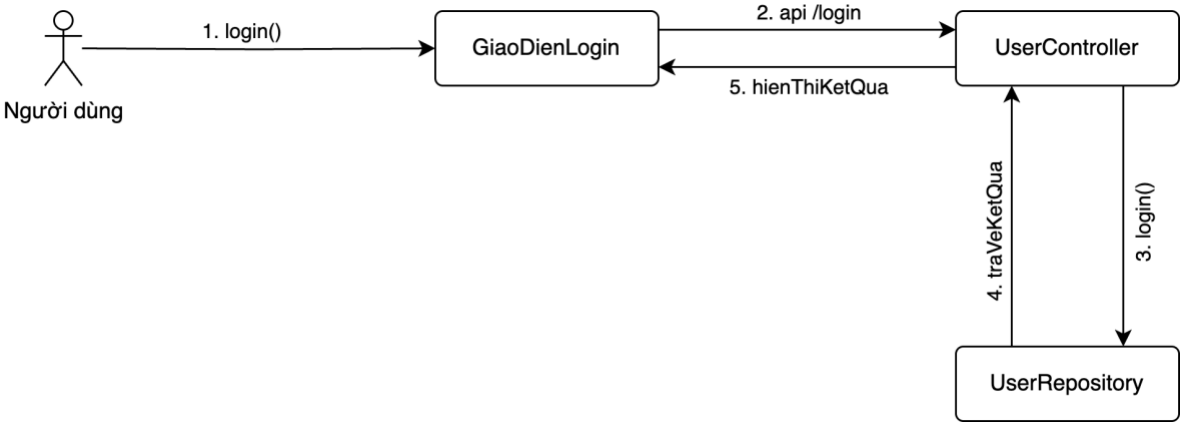
6.1. Đối tượng tham gia

- :Người dùng
- :GiaoDienLogin
- :UserController
- :UserRepository, :User

6.2. Thông điệp (đánh số thứ tự)

- Người dùng → GiaoDienLogin: 1. login()
- GiaoDienLogin → UserController: 2. api /login
- UserController → UserRepository: 3. login()
- UserRepository → UserController: 4. traVeKetQua()
- UserController → GiaoDienLogin: 5. hienThiKetQua ()

6.3. Diagram



6.4. Ghi chú các thông điệp chính

STT	Thông điệp	Nguồn → Đích	Ý nghĩa
1	login()	Người dùng → GiaoDienLogin	Người dùng click vào Đăng nhập để bắt đầu quy trình
2	Api /login	GiaoDienLogin → UserController	Giao diện call api sang lớp điều khiển để xử lý
3	login()	UserController → UserRepository	Truy xuất thông tin dữ liệu người dùng trong hệ thống
4	traVeKetQua()	UserRepository → UserController	Trả kết quả
5	hienThiKetQua()	UserController → GiaoDienLogin	Hiển thị kết quả (hiển thị lỗi hoặc đăng nhập thành công và chuyển sang trang chủ)

7. UC7: Quản lý phim yêu thích

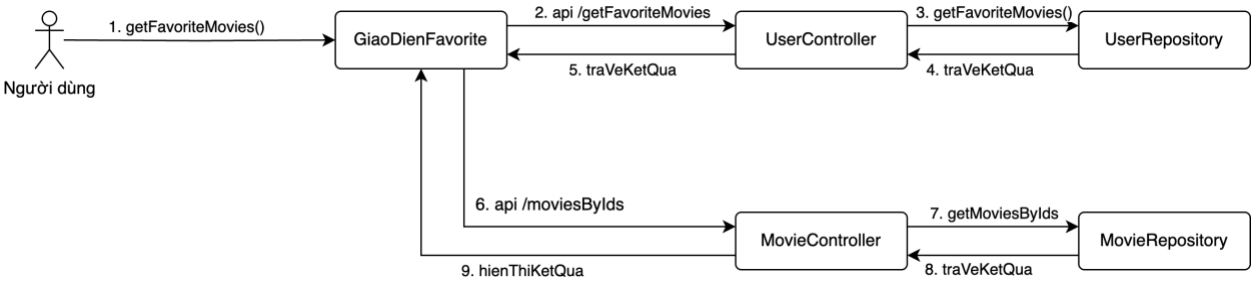
7.1. Đối tượng tham gia

- :Người dùng
- :GiaoDienFavorite
- :UserController, :MovieController
- :UserRepository, :User
- :MovieRepository, :Movie

7.2. Thông điệp (đánh số thứ tự)

- Người dùng → GiaoDienFavorite: 1. getFavoriteMovies()
- GiaoDienFavorite → UserController: 2. api /getFavoriteMovies
- UserController → UserRepository: 3. getFavoriteMovies()
- UserRepository → GiaoDienFavorite: 4. traVeKetQua()
- GiaoDienFavorite → MovieController: 6. api /getMoviesByIds()
- MovieController → MovieRepository: 7. getMoviesByIds()
- MovieRepository → MovieController: 8. traVeKetQua()
- MovieController → GiaoDienFavorite: 9. hienThiKetQua()

7.3. Diagram



7.4. Ghi chú các thông điệp chính

STT	Thông điệp	Nguồn → Đích	Ý nghĩa
1	getFavoriteMovies ()	Người dùng → GiaoDienFavorite	Người dùng click vào tab Favorite để bắt đầu quy trình
2	Api /getFavoriteMovies	GiaoDienFavorite → UserController	Giao diện call api sang lớp điều khiển để xử lý
3	getFavoriteMovies()	UserController → UserRepository	Truy xuất thông tin dữ liệu người dùng trong hệ thống
4	traVeKetQua()	UserRepository → UserController	Trả kết quả
5	traVeKetQua()	UserController → GiaoDienFavorite	Trả kết quả
6	Api getMoviesByIds()	GiaoDienFavorite → MovieController	Giao diện call api sang lớp điều khiển để xử lý
7	getMoviesByIds	MovieController → MovieRepository	Truy xuất thông tin dữ liệu phim trong hệ thống
8	traVeKetQua()	MovieRepository → MovieController	Trả kết quả
5	hienThiKetQua()	MovieController → GiaoDienFavorite	Hiển thị kết quả (hiển thị lỗi hoặc danh sách phim yêu thích)

V. Biểu đồ thành phần và triển khai (Component & Deployment)

1. Component Diagram

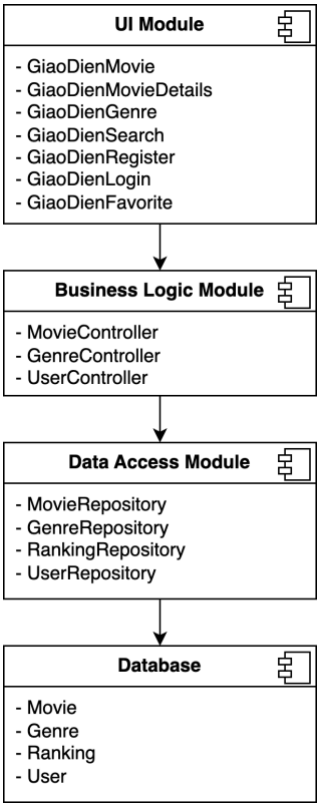
1.1. Đối tượng tham gia

Tên thành phần	Vai trò	Ghi chú
UI Module (Giao diện người dùng)	Cung cấp giao diện cho người dùng: đăng ký, đăng nhập, xem thông tin phim, thể loại, chi tiếp phim, phim yêu thích, tìm kiếm	Boundary Layer
Business Logic Module	Xử lý logic nghiệp vụ: xác thực người dùng, quản lý phim	Control Layer
Data Access Module	Kết nối và truy xuất dữ liệu trong cơ sở dữ liệu (CRUD)	Entity Layer
Database (Cơ sở dữ liệu)	Lưu trữ thông tin phim, thể loại, đánh giá, tài khoản	Physical Storage

1.2. Quan hệ phụ thuộc

- UI Module phụ thuộc vào Business Logic Module
- Business Logic Module phụ thuộc vào Data Access Module
- Data Access Module kết nối đến Database

1.3. Diagram



2. Deployment Diagram

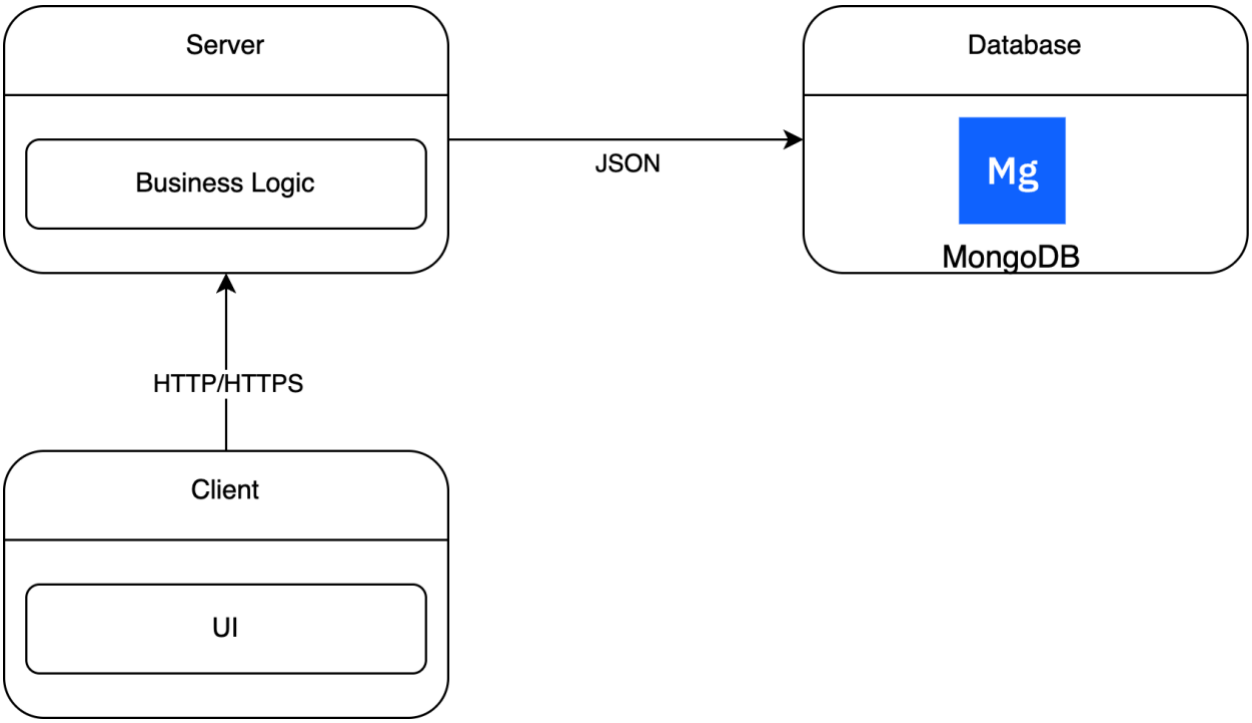
2.1. Đối tượng tham gia

Node	Vai trò	Thành phần triển khai
Client (Máy người dùng)	Giao diện ứng dụng desktop/web/mobile	UI Module
Application Server	Chạy nghiệp vụ chính của hệ thống	Business Logic + Data Access Modules
Database Server	Lưu trữ cơ sở dữ liệu	Database (MongoDB)

2.2. Kết nối vật lý

- Client ↔ Application Server: giao tiếp qua HTTP/HTTPS.
- Application Server ↔ Database Server: MongoDB JSON

2.3. Diagram



2.4. Mô tả cấu trúc triển khai

Hệ thống Xem phim online được triển khai theo mô hình 3 tầng (3-tier architecture) gồm:

2.4.1 Tầng Giao diện - Presentation Layer – UI Module

- Cài đặt trên máy người dùng hoặc trình duyệt web.
- Cung cấp giao diện trực quan để đăng ký, đăng nhập, và xem thông tin phim, thể loại, tìm kiếm.
- Giao tiếp với máy chủ qua giao thức HTTP/HTTPS.

2.4.2. Tầng Nghiệp vụ - Business Logic Layer

- Được triển khai trên Application Server.
- Chịu trách nhiệm xử lý yêu cầu từ UI, xác thực người dùng, quản lý phim/ thể loại.
- Gọi đến tầng truy cập dữ liệu để đọc/ghi thông tin.

2.4.3. Tầng Dữ liệu - Data Layer

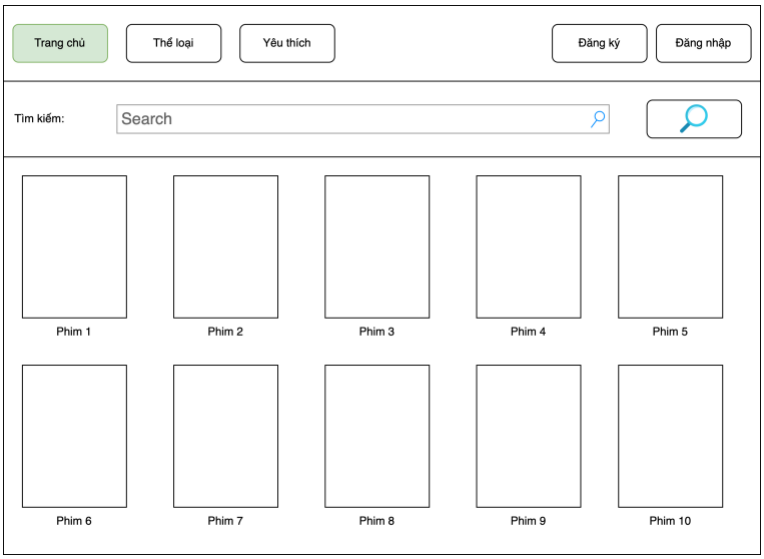
- Chạy trên Database Server, lưu trữ thông tin phim, thể loại, đánh giá và tài khoản.
- Giao tiếp với tầng nghiệp vụ thông qua API MongoDB framework.

B

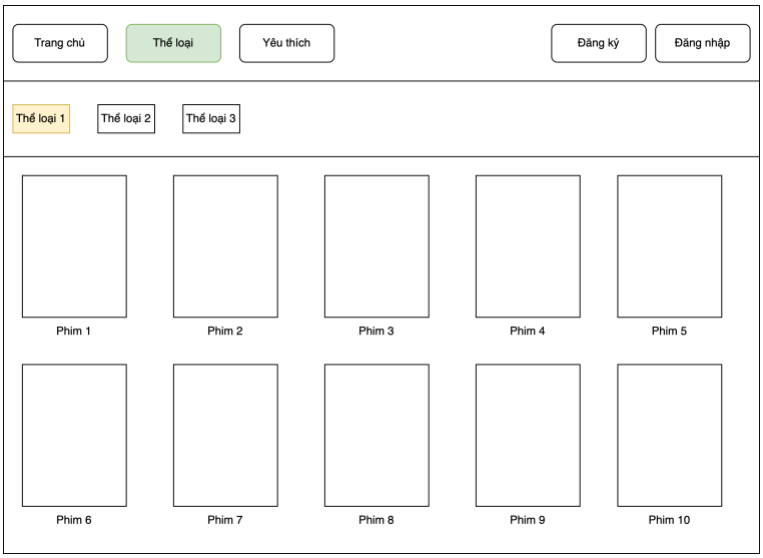
VI. Thiết kế giao diện & cơ sở dữ liệu

1. Thiết kế giao diện người dùng (UI)

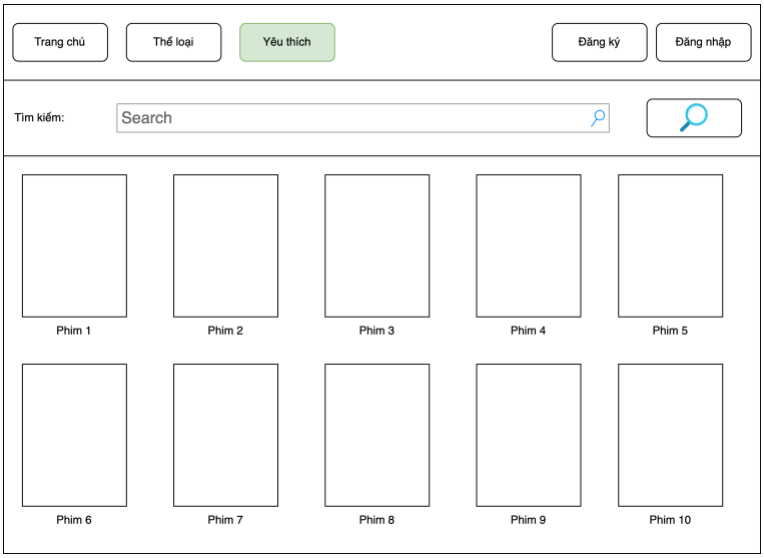
1.1. Màn hình chính (Trang chủ)



1.2. Màn hình Thể loại



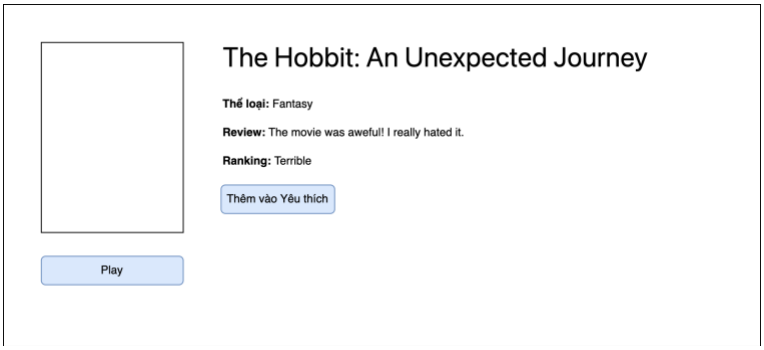
1.3. Màn hình Phim yêu thích



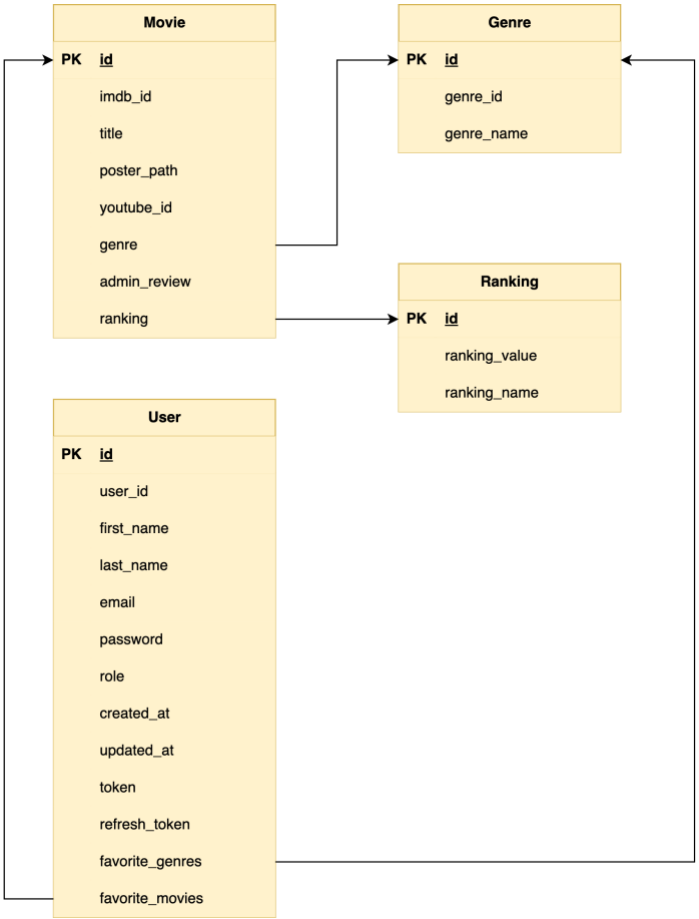
1.4. Màn hình Đăng ký người dùng

1.5. Màn hình Người dùng đăng nhập

1.5. Màn hình Chi tiết phim



2. Thiết kế sơ đồ cơ sở dữ liệu



VII. Kiến trúc phần mềm

1. Backend

1.1. Overview

- Spring Boot REST API kết nối dữ liệu MongoDB
- MongoDB uri được setup trong *src/main/resources/application.properties*
    - mongodb://localhost:27017/mystreamingdb

1.2. Cấu hình yêu cầu

- Java 17
- Maven 3+
- MongoDB chạy ở port 27017

1.3. Database setup (seed data)

Seed data (json) bên dưới thư mục json-files/. Import vào database mystreamingdb dùng command line:

```
cd json-files

mongoimport --db mystreamingdb --collection movies   --file movies.json   --jsonArray

mongoimport --db mystreamingdb --collection rankings --file rankings.json --jsonArray

mongoimport --db mystreamingdb --collection genres   --file genres.json   --jsonArray

mongoimport --db mystreamingdb --collection users    --file users.json    --jsonArray
```

1.4. Build

```
mvn clean package
```

Output file sẽ nằm trong thư mục target, ví dụ: target/streaming-api-0.0.1-SNAPSHOT.jar

1.5. Chạy local

```
mvn spring-boot:run
```

Hoặc chạy file jar trong thư mục output:

```
java -jar target/streaming-api-0.0.1-SNAPSHOT.jar
```

1.6. Deployment

Ứng dụng này có thể được triển khai dưới dạng file jar tiêu chuẩn của Spring Boot. Các cách triển khai phổ biến:
- Chạy file jar dưới hệ thống systemd trên máy ảo
- Đóng gói thành container và chạy với Docker/Kubernetes

2. Front-end

2.1. Cấu hình yêu cầu

- NodeJS LTS 18+
- npm

2.2 Hướng dẫn chạy ứng dụng

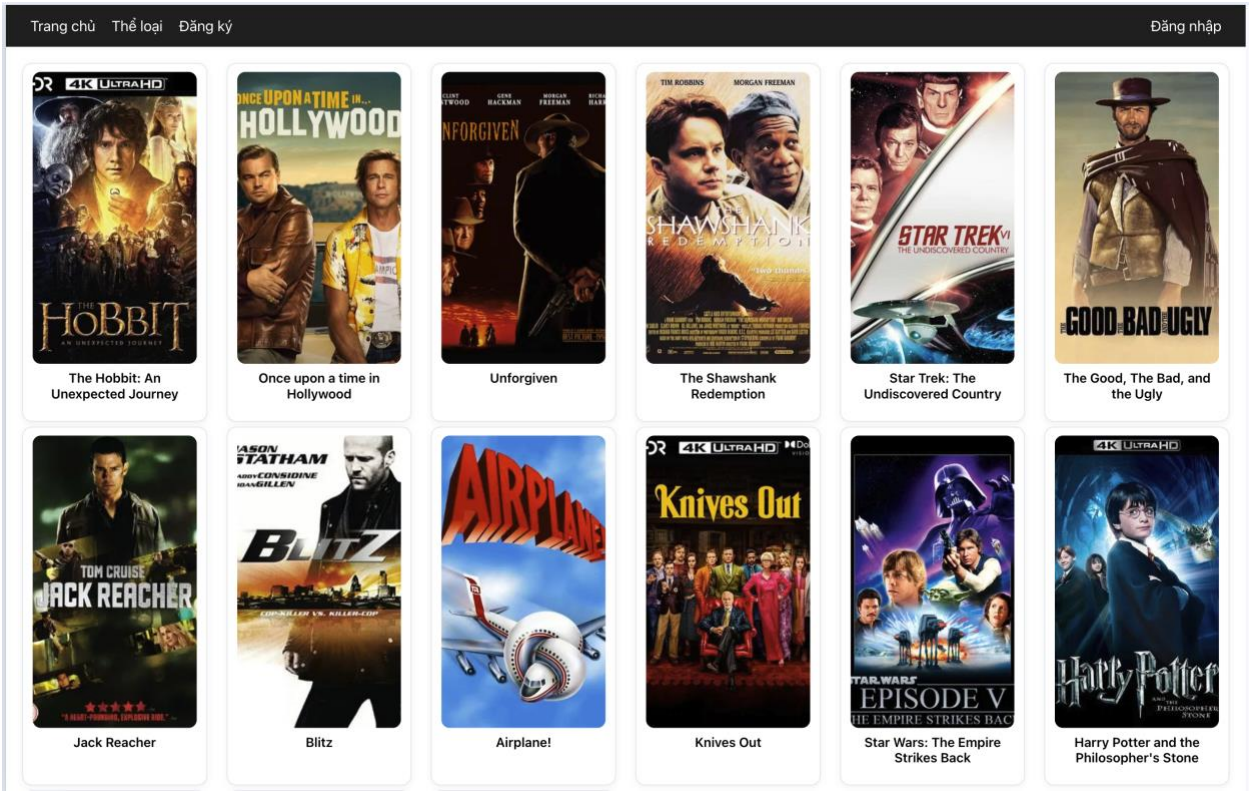
Hướng dẫn chạy (Visual Studio Code hoặc Terminal):

1. Mở thư mục **fronend**/
2. Run server backend API ở bên trên.
3. Run front-end:

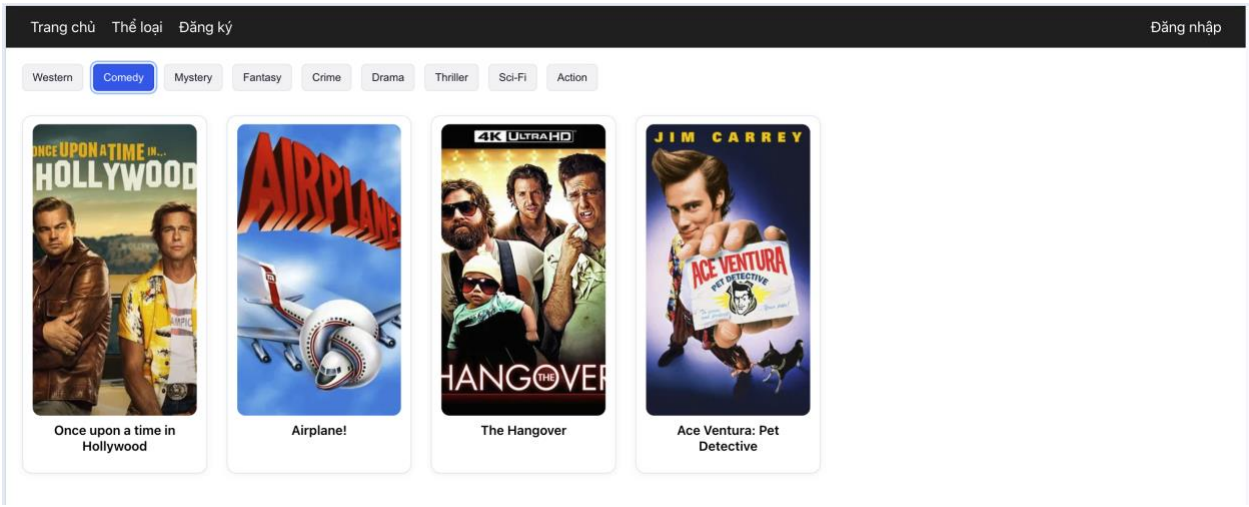
```
npm install

npm start
```

3. Giao diện chính

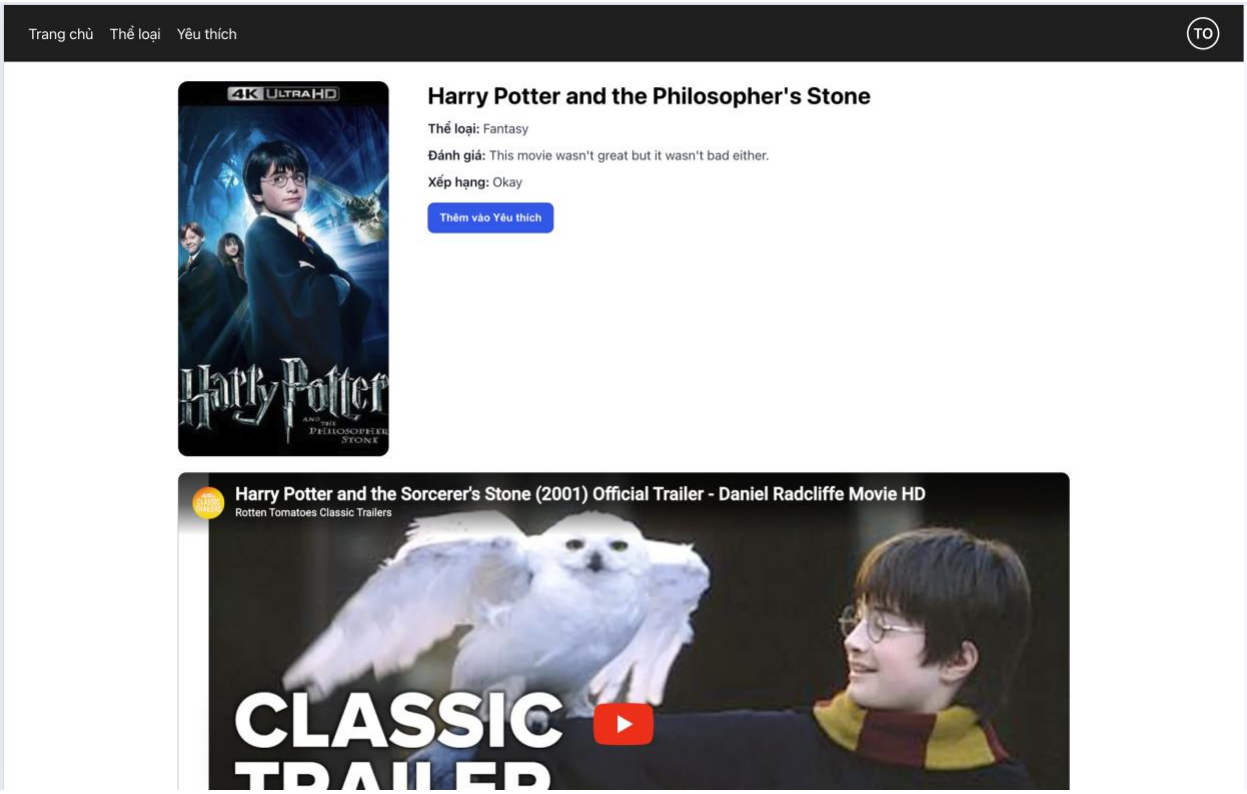
3.1. Trang chủ



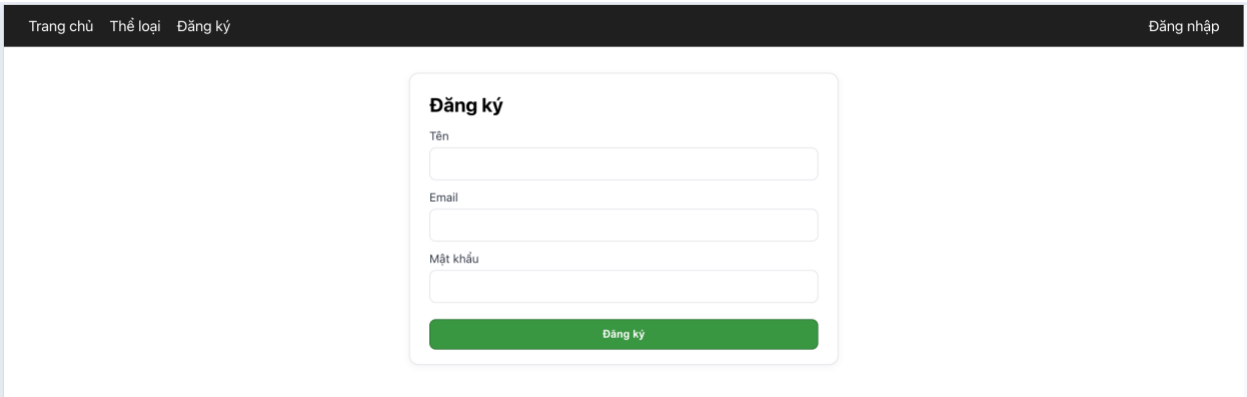
3.2. Thể loại



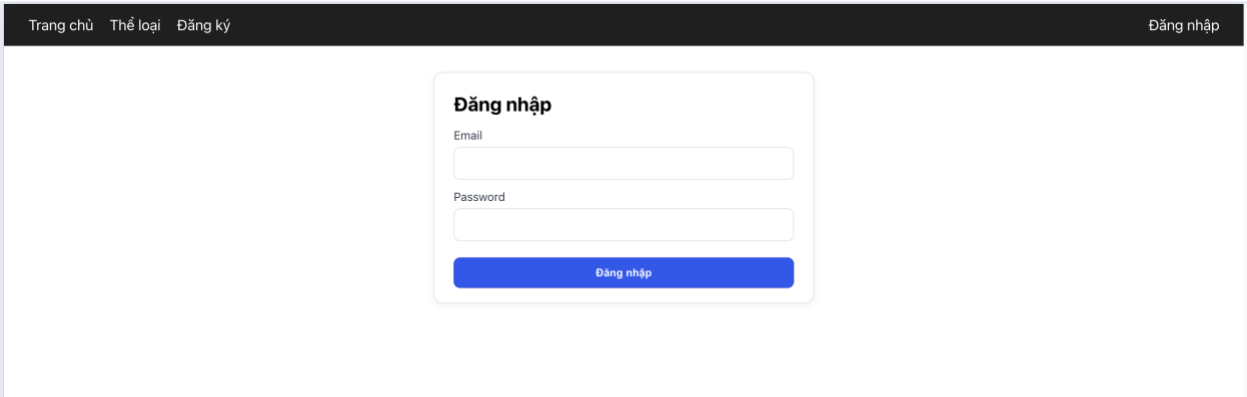
3.3. Chi tiết phim



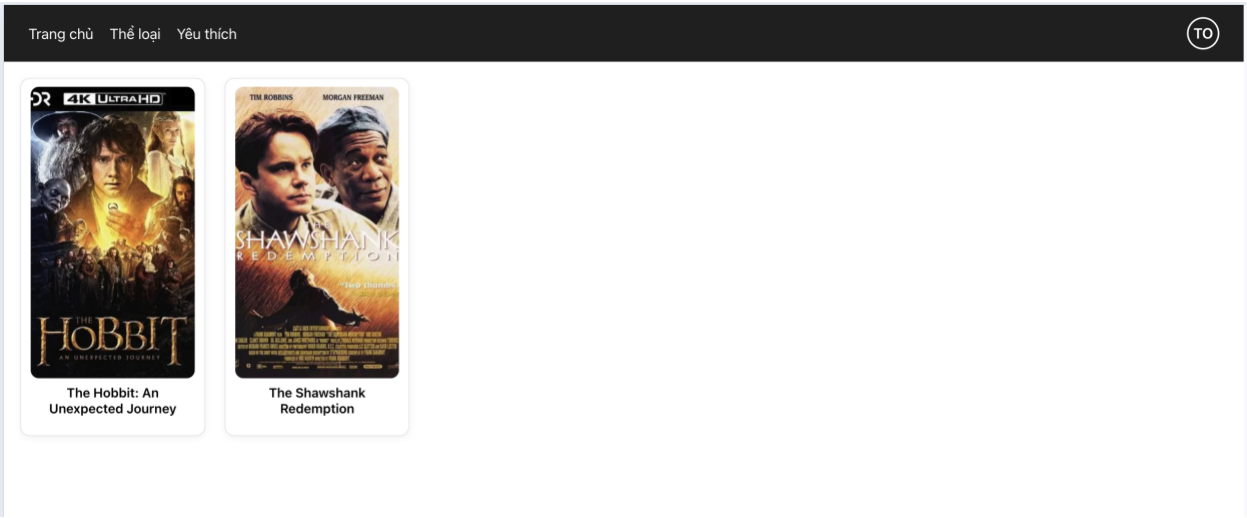
3.4. Đăng ký người dùng



3.5. Người dùng đăng nhập



3.6. Yêu thích của người dùng



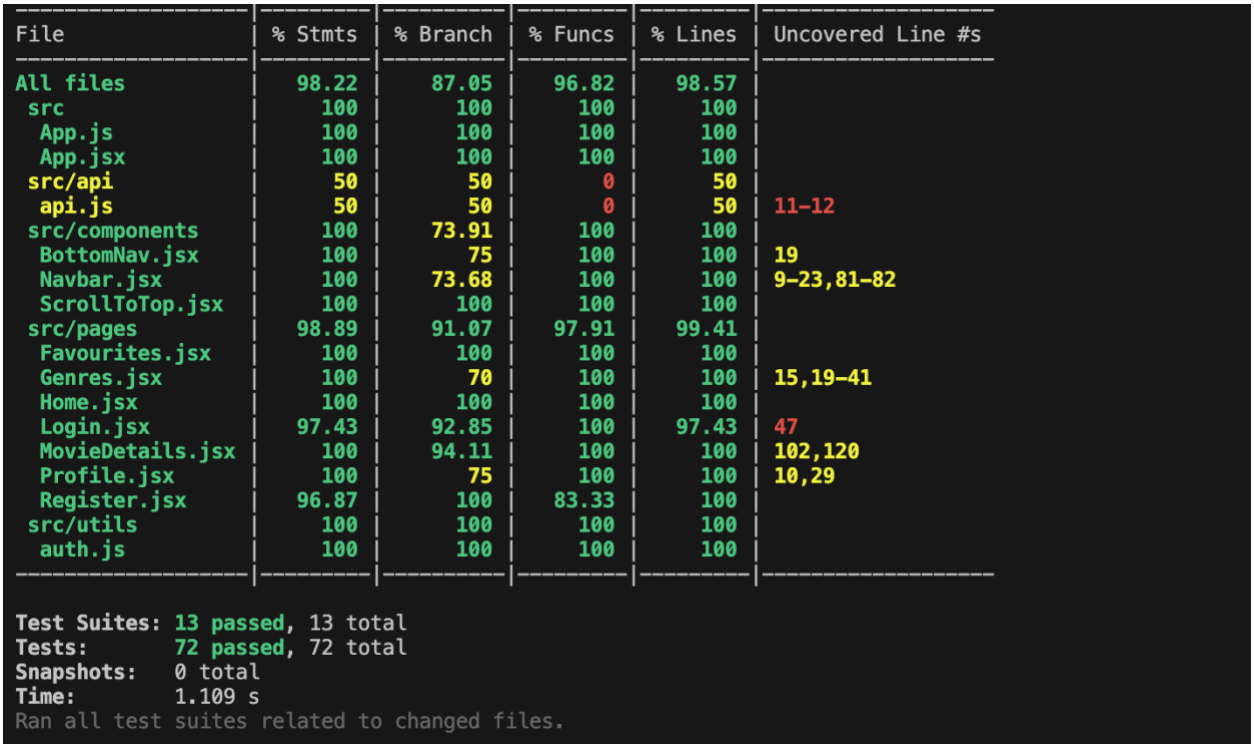
VIII. Kiểm thử đơn vị và kiểm thử tải

1. Tạo Unit Test

1.1. Chạy Unit Test

```
# Install dependencies
cd frontend
npm install

# Run all tests
npm test

# Run tests in watch mode
npm run test:watch

# Run with coverage report
npm test -- --coverage
```

1.2. Test Coverage



File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Line #s
All files	98.22	87.05	96.82	98.57	
src	100	100	100	100	
App.js	100	100	100	100	
App.jsx	100	100	100	100	
src/api	50	50	0	50	
api.js	50	50	0	50	11-12
src/components	100	73.91	100	100	
BottomNav.jsx	100	75	100	100	19
Navbar.jsx	100	73.68	100	100	9-23,81-82
ScrollToTop.jsx	100	100	100	100	
src/pages	98.89	91.07	97.91	99.41	
Favourites.jsx	100	100	100	100	
Genres.jsx	100	70	100	100	15,19-41
Home.jsx	100	100	100	100	
Login.jsx	97.43	92.85	100	97.43	47
MovieDetails.jsx	100	94.11	100	100	102,120
Profile.jsx	100	75	100	100	10,29
Register.jsx	96.87	100	83.33	100	
src/utils	100	100	100	100	
auth.js	100	100	100	100	

```
Test Suites: 13 passed, 13 total
Tests:       72 passed, 72 total
Snapshots:   0 total
Time:        1.109 s
Ran all test suites related to changed files.
```

1.3. Test Report

/backend/target/site/jacoco/index.html

StreamingAPI

StreamingAPI

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
com.streaming.api.models		80%		40%	116	194	0	31	0	74	0	4
com.streaming.api.controllers		100%		100%	0	30	0	87	0	19	0	5
com.streaming.api		100%		83%	1	8	0	13	0	5	0	2
com.streaming.api.dto		100%		n/a	0	1	0	1	0	1	0	1
Total	239 of 1,629	85%	144 of 268	46%	117	233	0	132	0	99	0	12

StreamingAPI > com.streaming.api.controllers

com.streaming.api.controllers

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
AuthController		100%		100%	0	6	0	44	0	3	0	1
UserController		100%		100%	0	13	0	28	0	7	0	1
MovieController		100%		100%	0	7	0	7	0	5	0	1
GenreController		100%		n/a	0	2	0	4	0	2	0	1
RankingController		100%		n/a	0	2	0	4	0	2	0	1
Total	0 of 326	100%	0 of 22	100%	0	30	0	87	0	19	0	5

StreamingAPI > com.streaming.api

com.streaming.api

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
ApiLoggingFilter		100%		83%	1	6	0	10	0	3	0	1
StreamingApiApplication		100%		n/a	0	2	0	3	0	2	0	1
Total	0 of 74	100%	1 of 6	83%	1	8	0	13	0	5	0	2

2. Load Test

```
→ load-tests git:(master) ✗ k6 run k6-load-test.js

      /\      Grafana   /‾‾/
   /\  /  \     |\  __   / /
  /  \/    \    | |/ /  /   ‾‾\
 /          \   |  (  |  (‾)  |
/ __________ \  |_|\_\  \_____/

    execution: local
       script: k6-load-test.js
       output: -

    scenarios: (100.00%) 1 scenario, 100 max VUs, 7m0s max duration (incl. graceful stop):
            * default: Up to 100 looping VUs for 6m30s over 6 stages (gracefulRampDown: 30s, gracefulStop: 30s)
```

```
INFO[0000] Starting load test...                         source=console
INFO[0000] Base URL: http://localhost:8080/api           source=console
INFO[0396] Load test completed!                          source=console


 ▉ THRESHOLDS

   errors
   ✓ 'rate<0.01' rate=0.00%

   http_req_duration
   ✓ 'p(95)<500' p(95)=98.08ms
   ✓ 'p(99)<1000' p(99)=117.43ms

   http_req_failed
   ✓ 'rate<0.01' rate=0.00%


 ▉ TOTAL RESULTS

   checks_total.......: 23513   59.340022/s
   checks_succeeded...: 100.00% 23513 out of 23513
   checks_failed......: 0.00%   0 out of 23513

   ✓ GET /movies status is 200
   ✓ GET /movies response time < 500ms
   ✓ Search movies status is 200
   ✓ Get by genre status is 200
   ✓ Register status is 200 or 400
   ✓ GET /users status is 200
   ✓ POST /rankings completed

   CUSTOM
   errors.....................: 0.00%  0 out of 0

   HTTP
```

```
    http_req_duration.....: avg=24.77ms min=317µs med=1.84ms max=181.58ms p(90)=86.16ms
p(95)=98.08ms
      { expected_response:true }...: avg=24.77ms min=317µs med=1.84ms max=181.58ms p(90)=86.16ms
p(95)=98.08ms
    http_req_failed.....: 0.00%  0 out of 20154
    http_reqs.....: 20154  50.862876/s

    EXECUTION
    iteration_duration.....: avg=7.15s   min=7.08s med=7.14s  max=7.31s    p(90)=7.2s    p(95)=7.22s
    iterations.....: 3359   8.477146/s
    vus.....: 1      min=1          max=100
    vus_max.....: 100    min=100        max=100

    NETWORK
    data_received.....: 2.1 GB 5.4 MB/s
    data_sent.....: 2.6 MB 6.5 kB/s




running (6m36.2s), 000/100 VUs, 3359 complete and 0 interrupted iterations
default ✓ [======================================] 000/100 VUs  6m30s
```