**Lab Test 5 EECS 2311: Take Home Assignment**

User Stories:

* Login and register
* SQLite database and MySQL database
* Page for individual movies
* Discovery page with movies
* and genre filter options
* Search function

What you’re working on:

* Login and register and the .dat file: Suneel
* SQLite database and MySQL database: Rythem
* Page for individual movies: Sarah
* Genre filter options: Gjergj
* Search function: Andy

**Activity 1: End-to-End Testing**

User Story: Page for Individual Movies

The individual movie page should allow users to view detailed information about a selected movie, including its title, release date, and other relevant details. The user should also be able to navigate back to the previous screen. Users must be logged in to access movie information.

End-to-End Manual Test Cases

**Test Case 1: User Login Required for Movie Access**

Steps:

1. Launch the application.
2. Try to navigate to a movie details page without logging in.
3. Observe if the system restricts access.
4. If access is denied, attempt to log in with valid credentials.
5. Try accessing the movie details page again.

Expected Result:

* The user should be redirected to the login page if they are not logged in.
* After successful login, the user should be able to access the movie page.

Edge Cases:

* Attempt logging in with invalid credentials (incorrect password, non-existent user).
* Refresh the page after logging in—does the session persist?
* Log in, then let the session expire and try accessing the movie page again.
* Try navigating directly via URL instead of using UI navigation.

**Test Case 2: Movie Page Loads Successfully**

Steps:

1. Log in to the application.
2. Navigate to the discovery page.
3. Click on a movie to open its detail page.
4. Observe how long it takes for the page to load.
5. Check if all movie details (title, release date, description) are displayed.

Expected Result:

* The movie page should load successfully.
* The movie title, release date, and other details should be visible.
* No crashes or missing elements should occur.

Edge Cases:

* Try opening a movie with missing details (e.g., no title, no release date).
* Simulate a slow API response and check if a loading indicator appears.
* Open the movie page in a new tab and check if it loads correctly.

**Test Case 3: Back Button Functionality**

Steps:

1. Open the movie page.
2. Click the "Back" button in the UI.
3. Observe whether the navigation returns to the previous screen.

Expected Result:

* The page should navigate back to the discovery page without issues.

Edge Cases:

* Press the "Back" button multiple times quickly—does the navigation break?
* Use the browser's back button instead of the UI button—does the app behave correctly?
* Open the movie page in a new tab and press "Back"—what happens?
* Test on different screen sizes and devices (mobile, tablet).

**Test Case 4: UI Validation and Responsiveness**

Steps:

1. Open a movie page with different movie entries.
2. Observe the font, colors, and alignment.
3. Resize the browser window and test on different screen sizes.
4. Check for accessibility compliance (color contrast, font sizes).

Expected Result:

* The UI should be readable and visually appealing.
* Titles should be in a larger font.
* Contrast should be sufficient for readability.
* The layout should not break on mobile screens.

Edge Cases:

* Test on ultra-wide monitors and small screens.
* Verify behavior when long movie titles are displayed.
* Test screen reader compatibility.

**Activity 2: Code Review and Bug Reporting**

Bug Reports

**Bug 1: Missing Login Check for Movie Page Access**

Description: Users should not be able to access movie details without logging in.

Steps to Reproduce:

1. Open the application without logging in.
2. Try accessing a movie details page via direct URL or discovery page.
3. Observe whether access is granted.

Expected Result:

* The system should redirect the user to the login page before granting access.

Suggested Fix:

* Implement an authentication check in the movie page controller.
* Redirect unauthorized users to the login screen.
* Ensure API requests for movie details are protected by authentication middleware.

**Bug 2: Missing Error Handling for Null Movie Data**

Description: The application may crash when a movie object has missing fields (e.g., no title or release date).

Steps to Reproduce:

1. Open the movie details page for a movie with missing data.
2. Observe if the UI fails to render properly or crashes.

Expected Result:

* The page should gracefully handle missing data by displaying placeholder text.

Suggested Fix:

* Implement null checks before displaying data.
* Provide default placeholders like "Title Unavailable" and "Release Date Unknown."

**Bug 3: Hardcoded UI Styles**

Description: Colors and fonts are hardcoded in MoviePage.java, which could cause UI inconsistencies.

Steps to Reproduce:

1. Open the movie details page.
2. Compare the UI with other pages.

Expected Result:

* A centralized theme manager should be used to maintain UI consistency.

Suggested Fix:

* Move styles to a dedicated theme manager or CSS file.
* Avoid directly specifying colors and fonts in code.

**Code Smells and Design Issues**

**Code Smell: Long Method in MoviePage Constructor**

* Issue: The MoviePage constructor handles multiple UI setup tasks in a single method.
* Solution: Extract UI initialization logic into separate methods:
  + setupHeader()
  + setupMovieDetails()
  + setupButtons()

**Code Smell: Duplicate UI Styling Code**

* Issue: UI styles (fonts, colors) are repeated in multiple places.
* Solution: Create a helper function to apply consistent styles to components.

**Code Smell: Feature Envy**

* Issue: Some UI components directly manipulate movie data instead of delegating this to the model layer.
* Solution: Introduce a MoviePageViewModel to separate business logic from UI code.

**Code Smell: Inappropriate Intimacy**

* Issue: MoviePage directly accesses properties of Movie without proper encapsulation.
* Solution: Implement getter methods with validation to ensure safe access to data.