# Front-End Test Automation

## The "Movie Catalog" Web App

"**Movie Catalog**" is a web application designed to help users **manage their movie collections efficiently**. The platform provides functionalities for users to register, log in, edit their profiles, and manage their movie lists, including adding, editing, and deleting movies. The primary objective of this exam is to ensure the application's functionalities perform as expected through automated UI tests using Selenium IDE and Selenium WebDriver.

**Access the "Movie Catalog" Web App through its dedicated URL:**

[**https://d24hkho2ozf732.cloudfront.net/**](https://d24hkho2ozf732.cloudfront.net/)

## Application Functionality

**Key Features:**

* **Landing Page (Unregistered/Non-Logged Users)**
  + Contains a sidebar with the following options: "ABOUT US," "OUR SERVICES," "LOGIN," "REGISTER," and "COPYRIGHTS."
    - **Register**
      * Allows users to register by entering a first name, last name, username, email, password, and repeat password.
      * Upon successful registration, users are redirected to the login screen.
* **Log In**
* Enables registered users to log in using their email and password.
* **Home Page (Logged Users)**
* Provides access to additional functionalities for editing profile, "ADD MOVIE," "ALL MOVIES," "WATCHED MOVIES," "UNWATCHED MOVIES," and "LOGOUT."
* Displays a greeting message with the username.
* **Profile management**
  + Users can edit their profile details, including first name, last name, email, username, and profile   
    picture (via URL).
  + **Movie Creation**
* Logged users can create new movies with a title, description, poster URL, and YouTube trailer link.
* These movies are then listed on the "All Movies" page.
  + - These movies also appear on "Watched Movies"or "Unwatched Movies" pages, depending on whether the "Mark as Watched" checkbox is checked during the creation of the movie.
  + **Movie Management**
    - Includes options to edit and delete, mark as watched, and mark as unwatched.
    - View Details functionality is not yet implemented.

**Locators** or **selectors** are **intentionally tricky to find** in this exam. In real practice, QAs often have to deal with similar problems. It's important to insist that developers add clear IDs and other locators to make automated testing easier and more reliable.

## Selenium IDE

In this section, you will use Selenium IDE to automate and validate core user flows within the "Movie Catalog" application. This involves recording and running tests to ensure key functionalities are accessible and perform as expected for both unregistered and registered users.

### Preparation

Before beginning the automated tests, **manually register an account on the "Movie Catalog" application**. **Use this account for tasks involving logged-in user actions.**

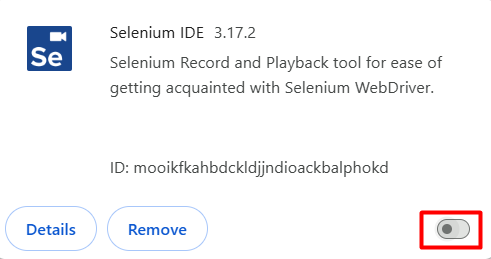
### Tests

**Selenium IDE & Google Chrome - Important Notice**

Selenium IDE is no longer officially supported in Google Chrome. The browser may disable it automatically due to security updates, and you will have to manually enable it each time from the extensions settings.

✅ **Recommended:** Use Mozilla Firefox, where Selenium IDE works reliably without restrictions.

✅ **If you must use it in Chrome:** Go to chrome://extensions/. Find Selenium IDE. Enable it manually (toggle switch).



⚠️ However, Chrome might disable it again after a restart! To avoid issues, we strongly recommend using Firefox instead.

#### 1.2.1. Landing Page Navigation Test (Non-Logged User)

* Use Selenium IDE to record a test that navigates to the Landing page as an unregistered or non-logged user.
* Verify that the page title is "Movie Catalog".
* Verify the presence of the ABOUT US, OUR SERVICES, LOGIN, REGISTER and COPYRIGHTS links in the Sidebar bar.

#### 1.2.2. Log In Test

* Using Selenium IDE, create a test that logs into the "Movie Catalog" application with your manually registered account.
* Input your registered email and password into the login form and submit it.
* Verify the welcome message reads "WELCOME, YOUR-USERNAME"
* Verify that the Sidebar includes links to "ADD MOVIE", "ALL MOVIES", "WATCHED MOVIES" and "UNWATCHED MOVIES".
* Log out and verify the subheading "Your personalized movie sanctuary!" is present on the Landing page, indicating the user is logged out.

#### 1.2.3. Edit Profile Test

* Write a Selenium IDE test that logs in with the user's credentials from the manually registered account and updates the profile information.
* Use the credentials to log in to the application.
* After logging in, navigate to the "" page.
* Click on the "Edit " button.
* Update the profile fields with new data:
  + **First Name:** Change to a new first name.
  + **Last Name:** Change to a new last name.
* Submit the changes. Verify that the updated information is correctly displayed:
  + **Names:** Assert that both the new first and the last name are changed and displayed together.
* Logout and verify redirection to home page by asserting the LOGIN link presence.

### Tests Organization

* Ensure all recorded tests are organized into a test suite.
* Give the test suite a clear and descriptive name that indicates its purpose.
* Name each test appropriately to reflect the specific task or functionality being tested.
* Arrange the tests in a logical sequence within the suite.
* Each test should start in a new browser window to ensure a fresh state.
* Maximize the browser window at the start of each test to ensure all elements are visible and to maintain consistency across different screen resolutions.

## Selenium WebDriver

In this section, you will set up and execute automated UI tests for the "Movie Catalog" application using Selenium WebDriver. The goal is to validate various functionalities of the application, ensuring it behaves as expected under different scenarios. You are free to use the Page Object Model (POM) for better test structure and maintainability, but using POM is not mandatory—you can write your tests without it if you prefer.

### Setup and Preparation

**Create a New NUnit Project**

**Install Necessary Packages**

* Selenium.WebDriver
* Selenium.Support
* FirefoxDriver (recommended) or ChromeDriver (still works)

**Include the relevant namespaces in your test classes by adding using directives**

* using OpenQA.Selenium;   
  Instantiating and managing WebDriver instances, interacting with web elements, and controlling browser actions.
* using OpenQA.Selenium.Chrome;  
  Launching Chrome, setting Chrome-specific options, and managing ChromeDriver.
* using OpenQA.Selenium.Interactions;  
  Performing complex user interactions that go beyond simple clicks or text entry.
* using OpenQA.Selenium.Support.UI;   
  Implementing explicit waits, which are essential for handling dynamic content and synchronizing tests.

**Create [OneTimeSetup] or [Setup] method**

* Configure Chrome options and set preferences.
* Initialize the WebDriver.
* Maximize the browser window and set implicit wait times.
* Perform an initial login to the application, ensuring the session is ready for the tests.

**Add [OneTimeTearDown] or [TearDown] method**

* Close the browser and end the WebDriver session.

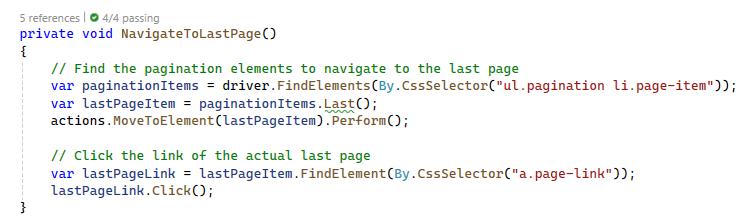
### Automated Tests

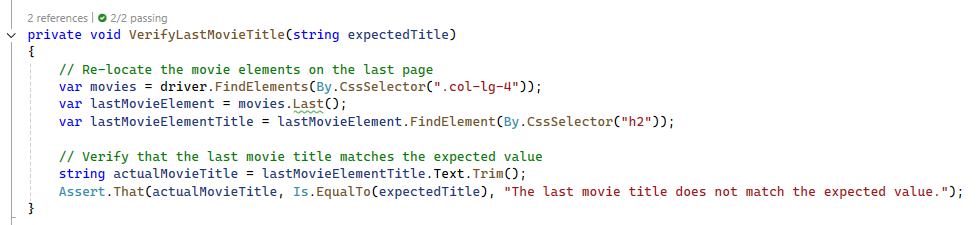
* **Tests should follow a logical sequence -** Use the [Order] attribute
* **Each test should start with a consistent state**
* **Name your tests descriptively**

**IMPORTANT!** When testing the Movie Catalog application, it is very important to understand the following points regarding **pagination** and **movie listings**:

* The application **displays only three movies per page**. If you add more than three, they will be spread across multiple pages.
* If your test setup includes **only three movies**, your tests will pass without issues.
* However, if you **add a fourth movie**, it will appear on a new page. If your test does not handle pagination, it will likely fail.
* **What you should do:**
  + **Option 1:** To keep it simple, make sure you only have three movies while testing **– this approach is perfectly valid.**
  + **Option 2 (Bonus Points! 🎯):** Try implementing pagination handling in your tests to verify movies across multiple pages.

💡 **Hint:** Here are two methods to help you handle pagination.





These methods will help you:  
✅ **Navigate to the last page** of the movie catalog.  
✅ **Verify the title of the last movie** on that page.

#### Add Movie Without Title Test

* **Navigate to the Add Movie Page:** Direct the browser to the Add Movie page.
* **Enter Invalid Movie Title:** Leave the title field empty to simulate invalid input.
* **Submit the Form:** Locate the 'Add' button and click it.
* **Verify the Error Message:** Confirm that an appropriate error message is displayed indicating that the title field is required.

#### Add Movie Without Description Test

* **Generate a Random Movie Title:** Create a random movie title to use for testing.
* **Navigate to the Add Movie Page:** Direct the browser to the Add Movie page.
* **Enter Valid Title and Invalid Description:** Fill in the title field with the generated title and leave the description field empty.
* **Submit the Form:** Locate the 'Add' button and click it.
* **Verify the Error Message:** Confirm that an appropriate error message is displayed indicating that the description field is required.

#### 2.2.3. Add Movie with Random Title Test

* **Generate Random Movie Title and Description:** Create a random movie title and description for testing.
* **Navigate to the Add Movie Page:** Direct the browser to the Add Movie page.
* **Enter Movie Details:** Fill in the title and description fields with the generated data.
* **Submit the Form:** Locate the 'Add' button and click it.
* **Navigate to the Last Page of Movie Listings:** Find and click the link to the last page of the movie listings.
* **Verify the Last Added Movie Title:** Ensure that the title of the last added movie matches the generated title.

#### Edit Last Added Movie Test

* **Navigate to the Last Page of Movie Listings**: Find and click the link to the last page of the movie listings.
* **Edit the Last Movie:** 
  + Locate the last movie and click the 'Edit' button.
  + Change the movie title and save the changes.
* **Verify the Success Message:** Confirm that a success message is displayed indicating that the movie was edited successfully.

#### Mark Last Added Movie as Watched Test

* **Navigate to the Last Page of Movie Listings:** Find and click the link to the last page of the movie listings.
* **Mark the Last Movie as Watched**: Locate the last movie and click the 'Mark as Watched' button.
* **Navigate to the Watched Movies Page:** Direct the browser to the Watched Movies page.
* **Navigate to the Last Page of Watched Movies Listings:** Find and click the link to the last page of the watched movies listings.
* **Verify the Last Watched Movie Title:** Ensure that the title of the last watched movie matches the   
  expected value.

#### Delete Last Added Movie Test

* **Navigate to the Last Page of Movie Listings:** Find and click the link to the last page of the movie listings.
* **Delete the Last Movie:** Locate the last movie and click the 'Delete' button.
* **Verify the Success Message:** Confirm that a success message is displayed indicating that the movie was deleted successfully.