**CS673 Software Engineering** 

**TV Bums - BUMTV**

**Software Test Document**

| **Team Member** | **Role(s)** | **Signature** | **Date** |
| --- | --- | --- | --- |
| Zhe Huang | QA | *Zhe Huang* | 09/25 |
| Mahim Choudhury | Securit/backup leader | *Mahim* | 10/7/23 |
| David Mulvihill | Configuration Management | *David Mulvihill* | 10/09/2023 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Revision history**

| **Version** | **Author** | **Date** | **Change** |
| --- | --- | --- | --- |
| 1.00 | Team | 10/09/2023 | Manual Testing Report  Automated Testing Report  Testing Metrics |
|  |  |  |  |

[Testing Summary](#_heading=h.gjdgxs)

[Manuel Tests Reports](#_heading=h.30j0zll)

[Automated Testing Reports](#_heading=h.1fob9te)

[Testing Metrics](#_heading=h.3znysh7)

[References](#_heading=h.2et92p0)

[Glossary](#_heading=h.tyjcwt)

# Testing Summary

We will test the following features of bumtv,

* User registration/email validation
  + - Test the form validation, registration success/failure, email verification success/failure
* User Login
  + - Test the user login success/failure
* Movie search
  + - Test the movie/tv show search capabilities
* Routing
  + - Test different routes for success/failure
* Add to watchlist
  + - Test add to watch list feature connected to a user
* DB connectivity
  + - Test many db connectivity,insertion, and update features

These are the methods that we plan to use for testing:

● **Unit Testing**: Unit testing will be used to test individual components/functions of the application.

● **Integration Testing**: This is to test the interaction between different components of the application.

● **System Testing**: To validate the application's functionality, performance, and reliability.

● **Acceptance Testing**: To ensure the system satisfies the requirements and is ready for delivery during final iteration.

● **Security Testing**: Verify the secured authentication, secure data transfer, scanning

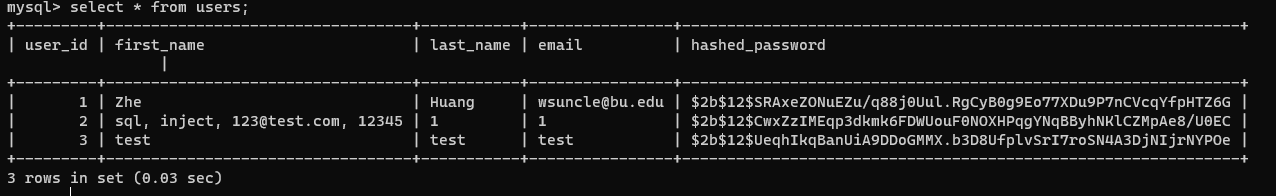
● **Performance Testing**: To ensure the system performs well under expected and peak load (optional)

The goal of testing is to check if the functionalities of the webapp that are mentioned in the requirements documents are met. We want to test the quality, security, functionalities of the BUMTV to make sure they all pass the expectation.

# Manual Testing Report

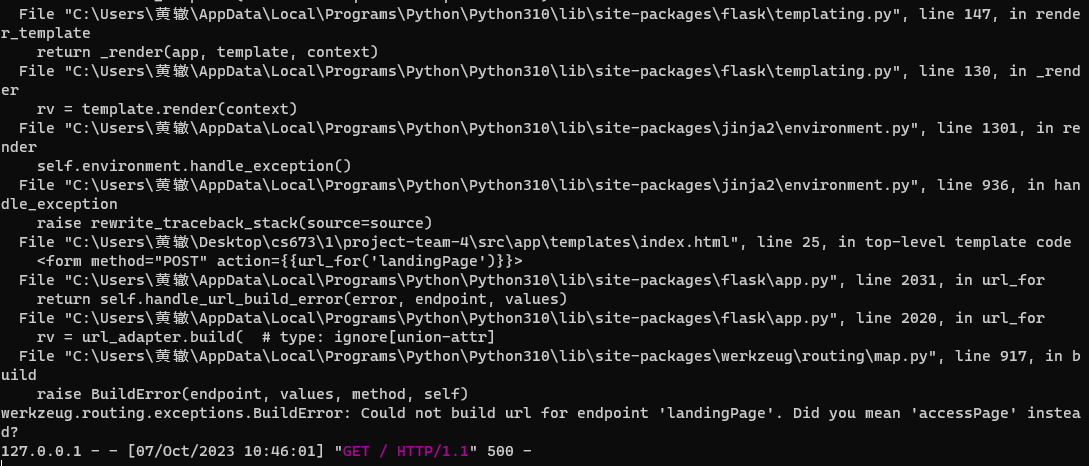
1.

* Test case ID： iter1 01
* New or old: new
* Test items: Login sql injection
* Test priority: High
* Dependencies (to other test case/requirement if any):
* Preconditions: (if any)
* input data: firstname : sql, inject, 123@test.com,12345) #
* Test steps:1
* Postconditions:
* Expected output: block the sql injection
* Actual output: blocked



* Pass or Fail:pass
* Bug id/link: none
* Additional notes: mysql blocked the sql injection

2

* Test case ID： iter2 02
* New or old: new
* Test items: Open app
* Test priority: High
* input data: None
* Test steps: open the app.py
* Expected output: run successfully
* Pass or Fail:Fail
* Bug id/link:
* 
* Additional notes: landingpage.html change to the landing\_page.html

3.

* Test Case ID: Iter2 -03
* Test Name: Integration Test for User Registration
* New or Old: New
* Test Items: Integration between user registration components (Web-UI, database, authentication)
* Test Priority: Medium
* Dependencies: Database is ready
* Preconditions:

The application is running and accessible.

The database is empty or has been cleared of test data.

* Input Data:

First Name: Komuna

Last Name: Dimuna

Email: komuna.dimuna@gmail.com

Password: Password123

Confirm Password: Password123

* Test Steps:

Navigate to the registration page.

Enter the provided input data into the registration form.

Click the "Register" button.

* Postconditions:

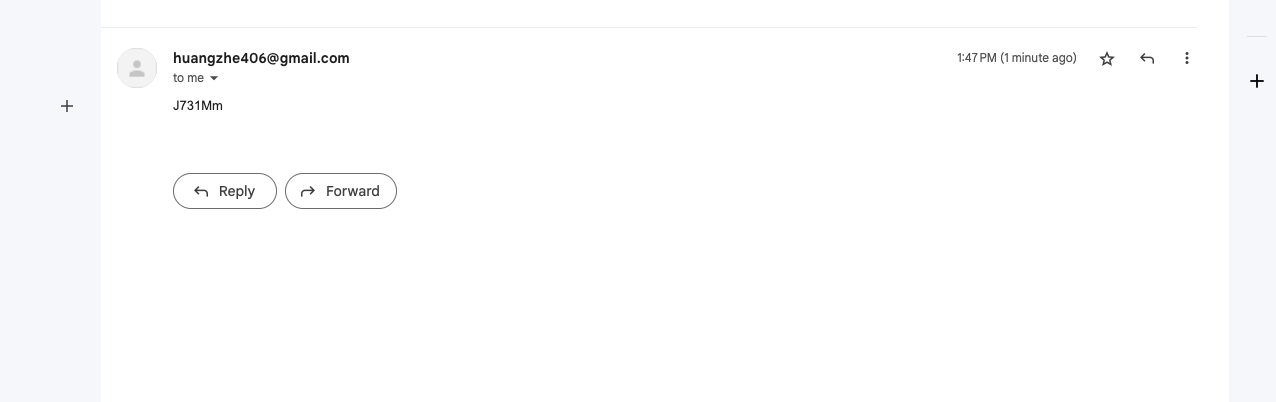
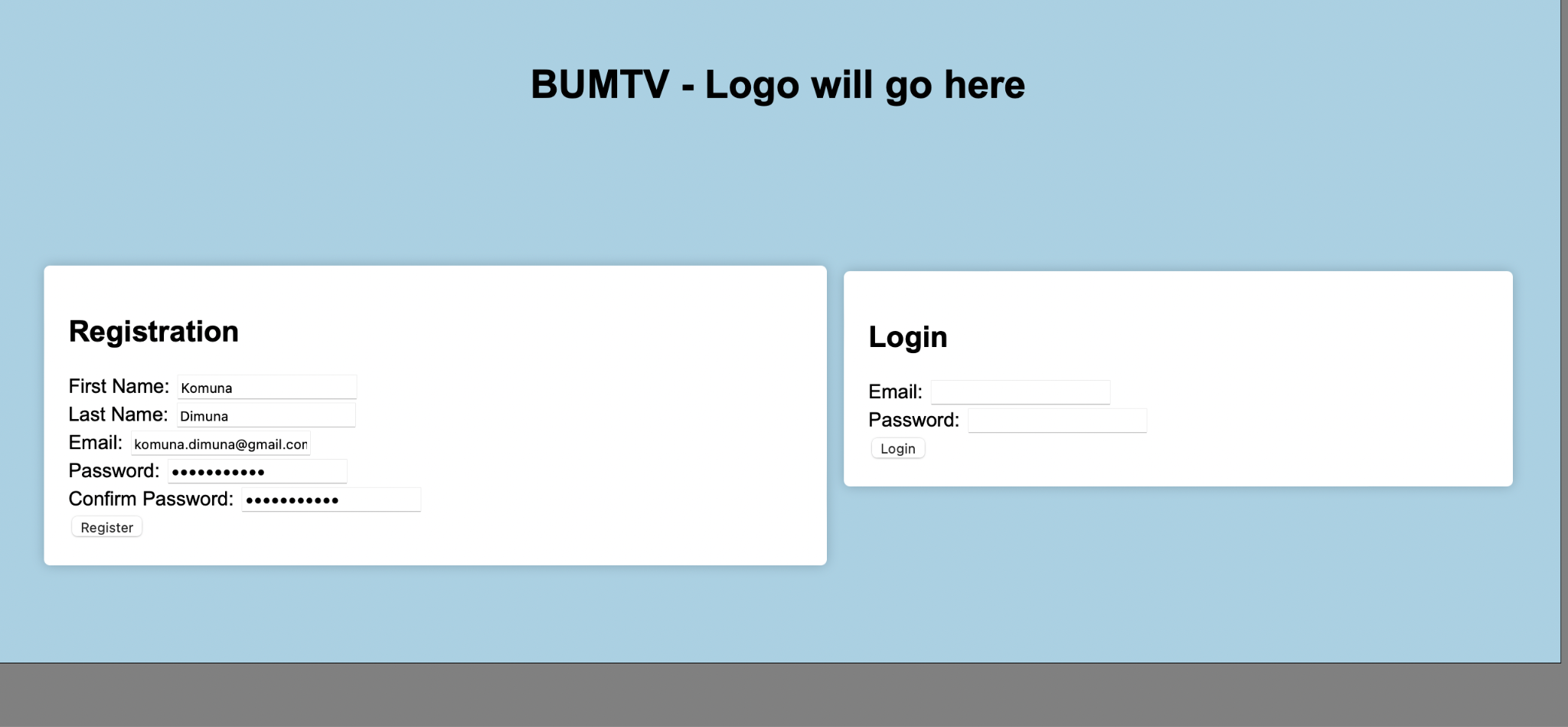
Completed registration process

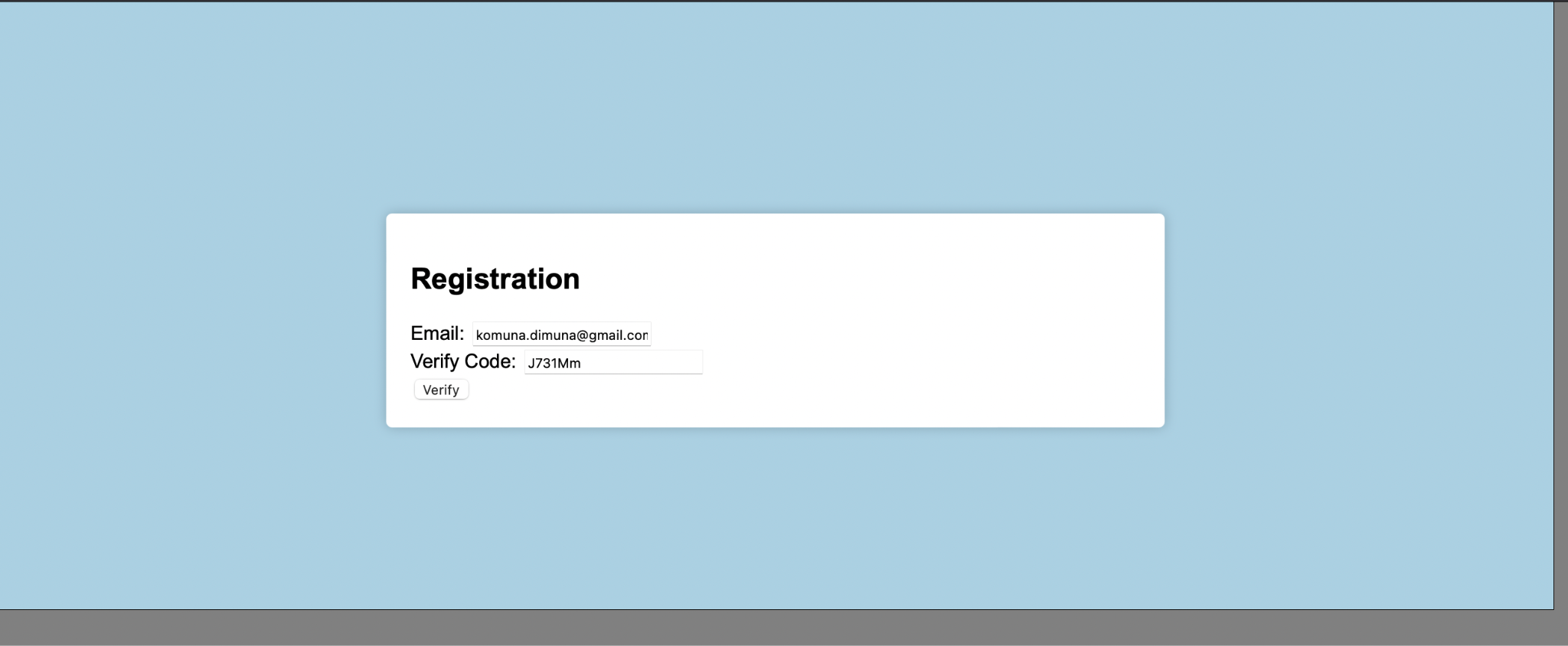
* Expected Output:

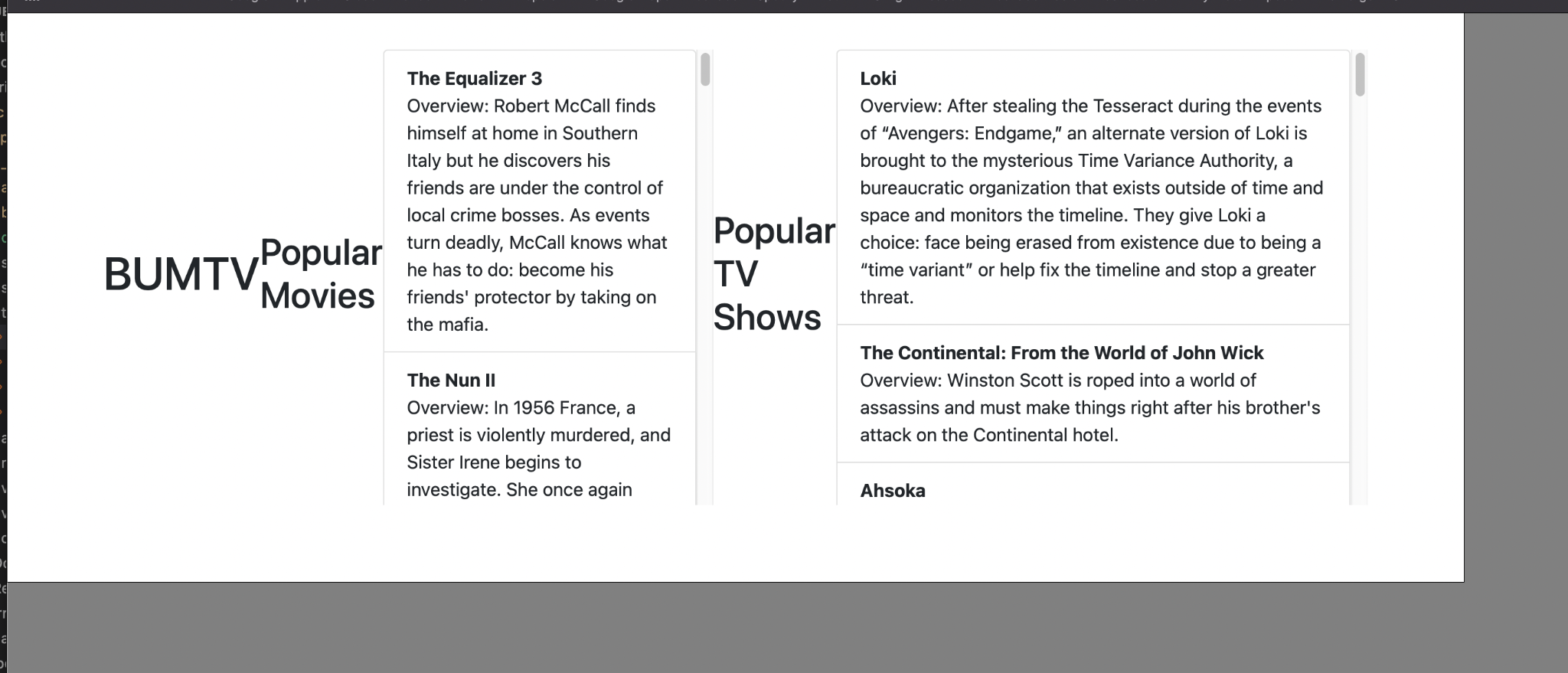
User is redirected to the verification page.

A verification email is sent to the provided email address.

* Actual Output:







User is redirected to the verification page.

A verification email is sent to the provided email address.

Once verified, they can see the landing page.

* Pass or Fail: Pass

4.

* Test Case ID: Iter2 -04
* Test Name: Wrong registration information test
* New or Old: New
* Test Items: Integration between user registration components (Web-UI, database, authentication)
* Test Priority: Medium
* Dependencies: Database is ready
* Preconditions:

The application is running and accessible.

The database is empty or has been cleared of test data.

* Input Data:

First Name: wew

Last Name: Dfada

Email: komuna.123

Password: 1233

Confirm Password: 123123123123

* Test Steps:

Navigate to the registration page.

Enter the provided input data into the registration form.

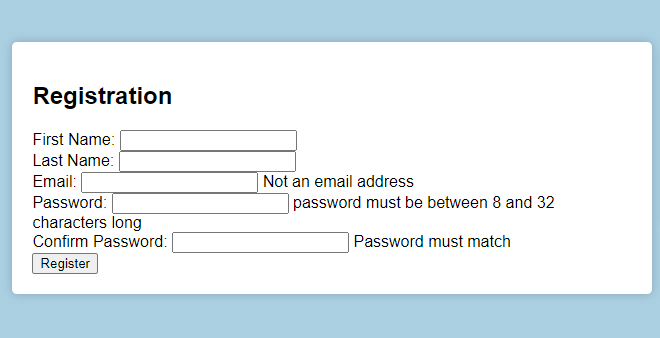
Click the "Register" button.

* Postconditions:

Fail to registrate, print error message

* Expected Output:

Print error message

* Actual Output:
* 
* Pass or Fail: Pass

5.

* Test Case ID: Iter2 -05
* Test Name: Used email test
* New or Old: New
* Test Items: Integration between user registration components (Web-UI, database, authentication)
* Test Priority: Medium
* Dependencies: Database is ready
* Preconditions:

The application is running and accessible.

The database is empty or has been cleared of test data.

* Input Data:

First Name: Zhe

Last Name: Huang

Email: wsuncle@bu.edu

Password: 123123123

Confirm Password: 123123123

* Test Steps:

Navigate to the registration page.

Enter the provided input data into the registration form.

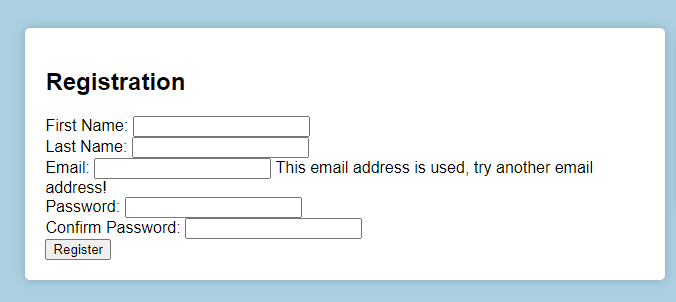
Click the "Register" button.

* Postconditions:

Fail to registrate, print error message

* Expected Output:

Print error message

* Actual Output:
* 
* Pass or Fail: Pass

6.

* Test Case ID: Iter2 -06
* Test Name: login test--no register email
* New or Old: New
* Test Items: Integration between user registration components (Web-UI, database, authentication)
* Test Priority: Medium
* Dependencies: Database is ready
* Preconditions:

The application is running and accessible.

The database is empty or has been cleared of test data.

* Input Data:

Email: 222@bu.edu

Password: 123123123

* Test Steps:

Navigate to the login page.

Enter the provided input data into the login form.

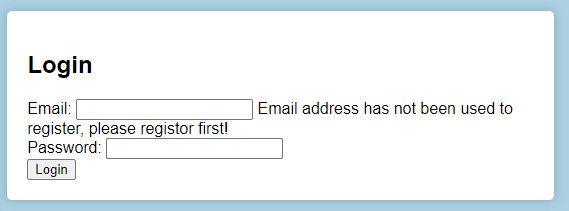
Click the "login" button.

* Postconditions:

Fail to registrate, print error message

* Expected Output:

Print error message

* Actual Output:
* 
* Pass or Fail: Pass

7.

* Test Case ID: Iter2 -07
* Test Name: login test--no verified email
* New or Old: New
* Test Items: Integration between user registration components (Web-UI, database, authentication)
* Test Priority: Medium
* Dependencies: Database is ready
* Preconditions:

The application is running and accessible.

The database is empty or has been cleared of test data.

* Input Data:

Email: tstorm1538@gmail.com

Password: 123123123

* Test Steps:

Navigate to the login page.

Enter the provided input data into the login form.

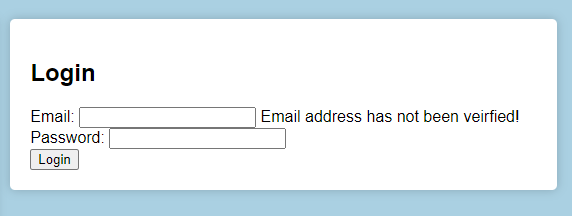
Click the "login" button.

* Postconditions:

Fail to registrate, print error message

* Expected Output:

Print error message

* Actual Output:
* 
* Pass or Fail: Pass

8.

* Test Case ID: Iter2 -08
* Test Name: login test--incorrect password
* New or Old: New
* Test Items: Integration between user registration components (Web-UI, database, authentication)
* Test Priority: Medium
* Dependencies: Database is ready
* Preconditions:

The application is running and accessible.

The database is empty or has been cleared of test data.

* Input Data:

Email: wsuncle@bu.edu

Password: 222222222

* Test Steps:

Navigate to the login page.

Enter the provided input data into the login form.

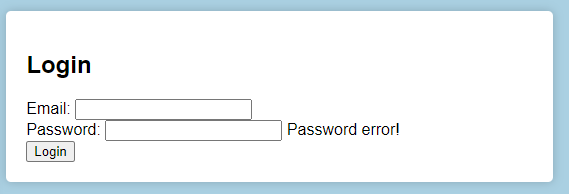
Click the "login" button.

* Postconditions:

Fail to registrate, print error message

* Expected Output:

Print error message

* Actual Output:
* 
* Pass or Fail: Pass

9.

* Test Case ID: Iter2 -09
* Test Name: login test--success
* New or Old: New
* Test Items: Integration between user registration components (Web-UI, database, authentication)
* Test Priority: Medium
* Dependencies: Database is ready
* Preconditions:

The application is running and accessible.

The database is empty or has been cleared of test data.

* Input Data:

Email: wsuncle@bu.edu

Password: 123123123

* Test Steps:

Navigate to the login page.

Enter the provided input data into the login form.

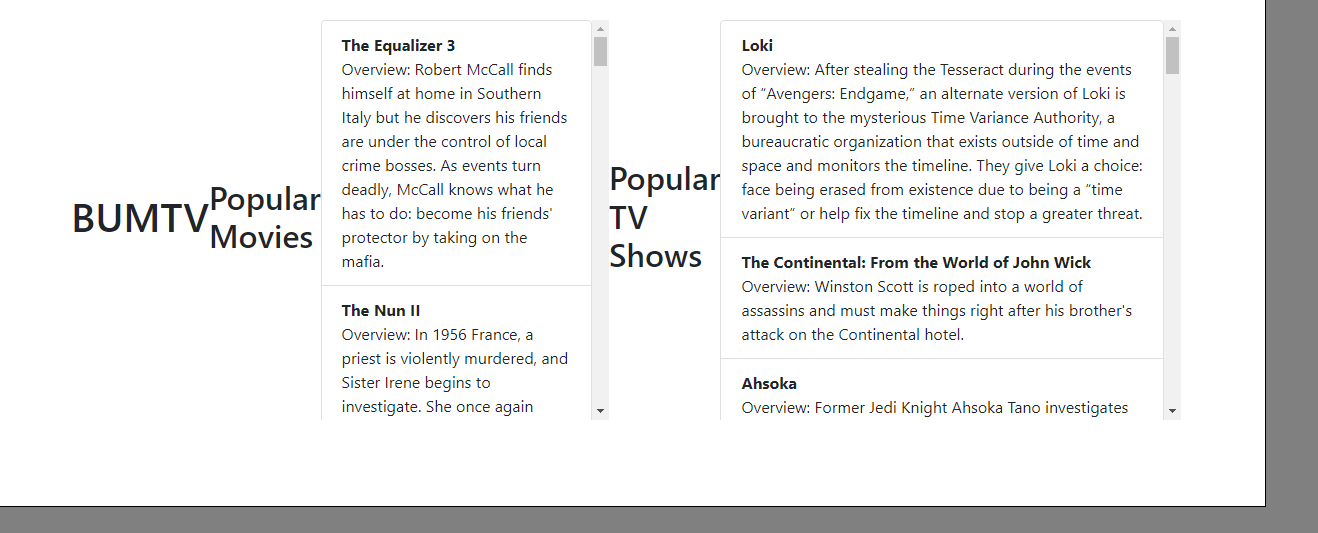
Click the "login" button.

* Postconditions:

Fail to registrate, print error message

* Expected Output:

Print error message

* Actual Output:
* 
* Pass or Fail: Pass

10.

* Test Case ID: Iter3 -10
* Test Name: search function test
* New or Old: New
* Test Items: Integration between user registration components (Web-UI, database, authentication)
* Test Priority: Medium
* Dependencies: Database is ready
* Preconditions:

The application is running and accessible.

The database is empty or has been cleared of test data.

* Input Data:

Harry potter

* Test Steps:

Navigate to search page

Enter the provided input data into the search form.

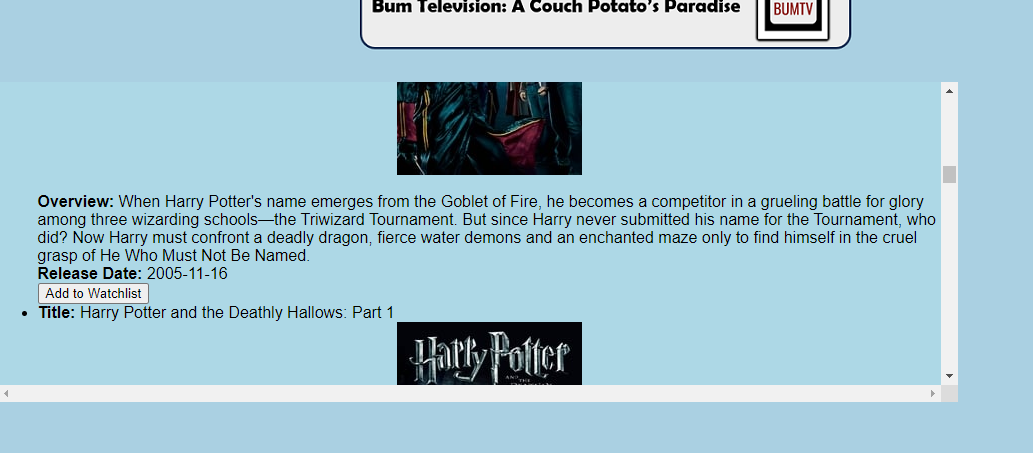
Click the "search" button.

* Postconditions:

Return the search results

* Expected Output:

Return the search results

* Actual Output:
* 
* Pass or Fail: Pass

11.

* Test Case ID: Iter3 -11
* Test Name: search function test
* New or Old: New
* Test Items: Integration between user registration components (Web-UI, database, authentication)
* Test Priority: Medium
* Dependencies: Database is ready
* Preconditions:

The application is running and accessible.

The database is empty or has been cleared of test data.

* Input Data:

click the ‘add to watchlist’ button of “**Loki**”

* Test Steps:

Navigate to landing page

click the ‘add to watchlist’ button of “**Loki**”

* Postconditions:

Add the movie to the watchlist

* Expected Output:

Add the movie to the watchlist

* Actual Output:



* Pass or Fail: Pass

12.

* Test Case ID: Iter3 -11
* Test Name: search function test
* New or Old: New
* Test Items: Integration between user registration components (Web-UI, database, authentication)
* Test Priority: Medium
* Dependencies: Database is ready
* Preconditions:

The application is running and accessible.

The database is empty or has been cleared of test data.

* Input Data:

click the ‘Remove from watchlist’ button of “**Loki**”

* Test Steps:

Navigate to landing page

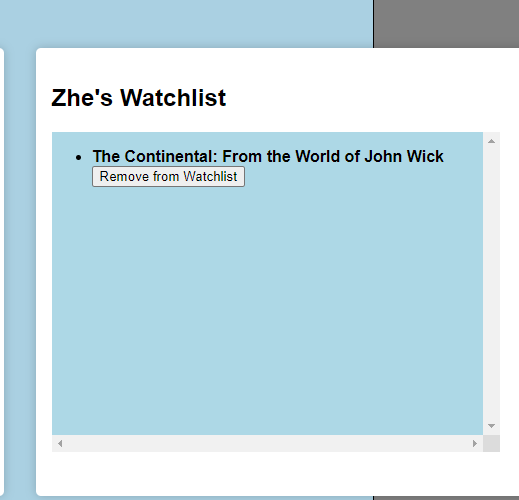
click the ‘Remove from watchlist’ button of “**Loki**”

* Postconditions:

Add the movie to the watchlist

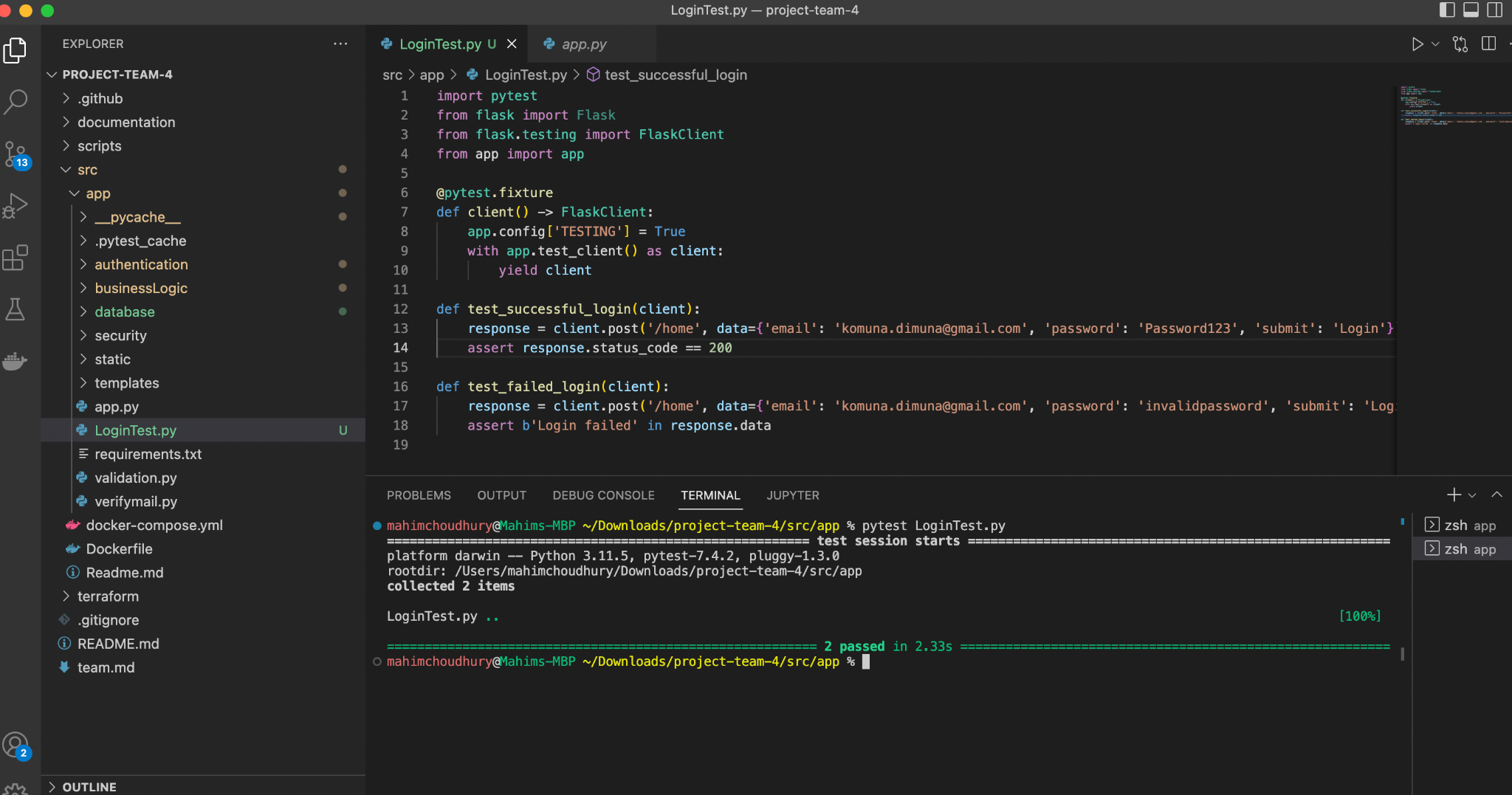
* Expected Output:

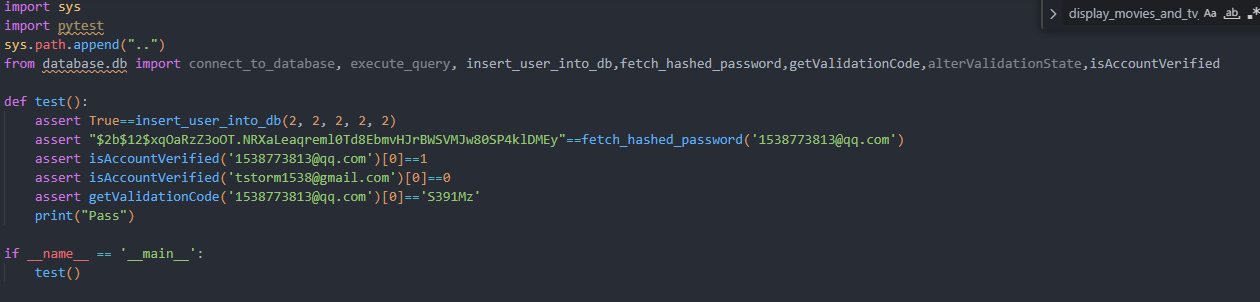
Add the movie to the watchlist

* Actual Output:
* 
* Pass or Fail: Pass

# Automated Testing Report

Describe briefly the automated testing you have done, including where the test code resides in your code repository, what test frameworks are used, and the screen shots or generated testing report.

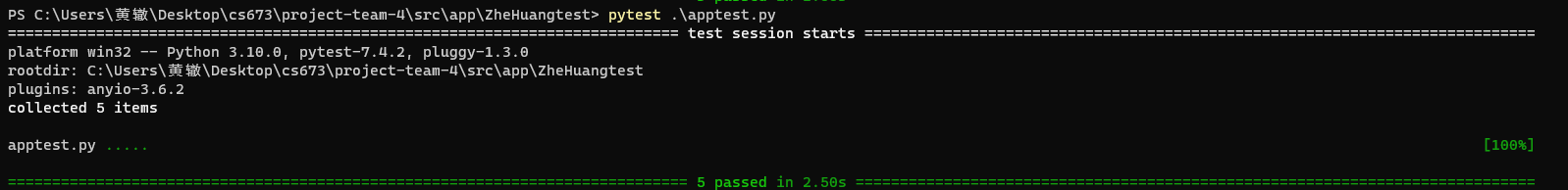
1. Used pytest to test the User Login(User that was created during the test case iter2-03):  
   
2. Use pytest to test database function





1. Pytest the login and verify





# Testing Metrics

In this section, you shall report any metrics used for the evaluation, e.g. # of test cases, test coverage, defects rate, etc.

## 

| Metric Name | Description |
| --- | --- |
| Code Coverage | How much of the code can be tested to ensure that bugs do not persist within the production of the software |
|  | 100% |
| DB Tables | How many databases and what databases is used in this program |
|  | db:users |
| Test Pass Rate | Test Pass Rate (TPR) is a metric used in software testing to measure the effectiveness of a testing process. It represents the percentage of test cases that pass successfully compared to the total number of test cases executed. In other words, it indicates the ratio of passed test cases to the total number of test cases attempted. |
|  | 100% |
| API Connectivity | API Connectivity, often simply referred to as API integration or API communication, is the process of enabling different software applications or systems to exchange data and interact with each other using Application Programming Interfaces (APIs). |
|  | All api are tested |

# References

# Glossary