Automation Framework Overview

The **Automation Testing Framework** for the ZenInbox is designed to facilitate efficient testing of web applications through automation. This framework leverages enhance maintainability and readability, ensuring a robust testing process.

Key Features

- **Test Automation for Web Application**:

  The framework is designed to automate the testing of the ZenInbox web application, focusing on the login functionality using different sets of credentials.

- **Data-Driven Testing**:

  A **DataProvider** is used to supply multiple sets of login credentials for the tests, making it easy to extend the tests with additional data without changing the core logic of the test.

- **Modular and Scalable**:

  The framework is designed in a modular way. New test cases and functionalities can be added by simply creating new methods or enhancing existing ones. This ensures that the framework is scalable for future needs.

- **Allure Reporting**:

  The framework is integrated with **Allure**, which provides visually rich and interactive test reports. These reports include step-by-step logs, status information (e.g., passed/failed), and detailed descriptions of each test case.

- **Cross-Browser Testing Support**:

  The framework supports Selenium WebDriver, which allows for the possibility of cross-browser testing by simply configuring different WebDriver settings (though only Chrome is set up in this implementation).

- **TestNG Integration**:

  The framework uses **TestNG** for test execution, providing built-in features like test grouping, parallel execution, and test reporting. The tests are grouped and managed via TestNG annotations.

Technologies Used:

- **Selenium WebDriver**: Version 4.33.0
- **Java**: Version 17
- **TestNG**: Version 7.10.2
- **Maven**: Version 3.6.3

-The document and framework designed by Pavan Nanaware.

2. Framework Architecture

1. Components:

- **Test Scripts**: Contains the actual test cases organized by features.
- **Reporting**: Utilizes **allure report** for detailed HTML reports.

2. Setup Instructions

Prerequisites:

1. **Java Development Kit (JDK) 11**
2. **Apache Maven**
3. **IDE**: Eclipse

Clone the Repository:

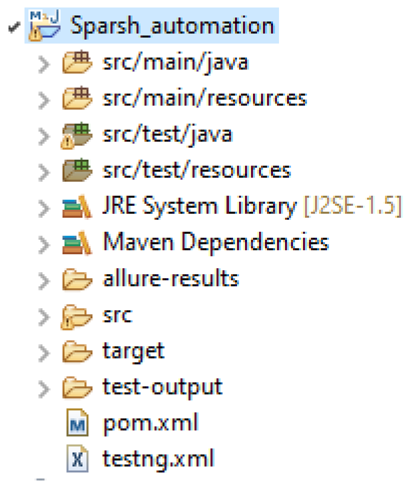
git clone Pavan7T/SparshTask: Automation Task

cd automation-framework

Build the Project:

- mvn compile or mvn verify
- mvn test

Project File Structure:



```
Sparsh_automation
  src/main/java
  src/main/resources
  src/test/java
  src/test/resources
  JRE System Library [J2SE-1.5]
  Maven Dependencies
  allure-results
  src
  target
  test-output
  pom.xml
  testng.xml
```

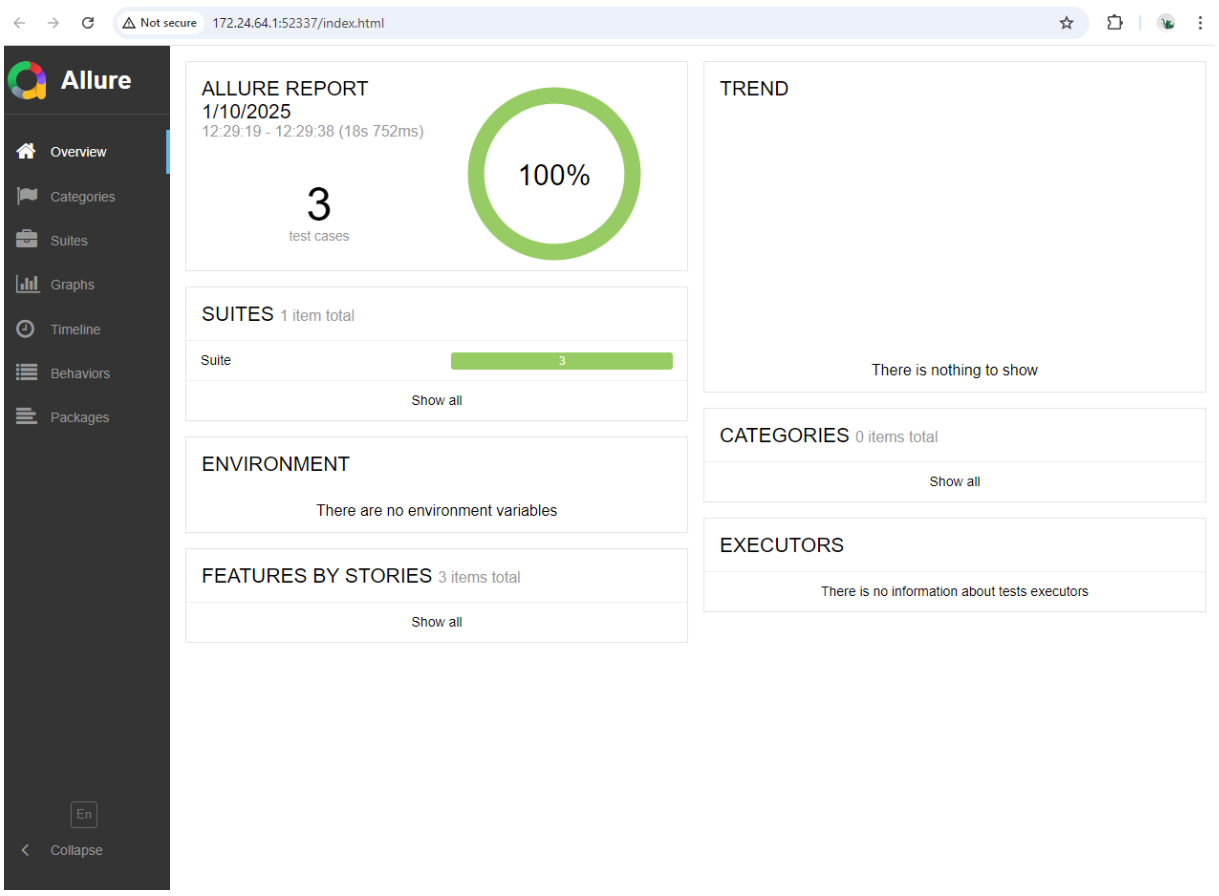
-The document and framework designed by Pavan Nanaware.

3. Reporting

We use **Allure Reports** for generating reports. Reports are generated in the allure report's directory after test execution. To use allure reports we need to install allure windows application. And the use following commands to view reports

Viewing Reports:

- Navigate to automation-framework/allure-reports in directory.
- Use command: allure serve D:\JS\Task\sparsh\allure-results
- These reports will be integrated with bug tracing tools like JIRA TFS or devOps.



ALLURE REPORT
1/10/2025
12:29:19 - 12:29:38 (18s 752ms)
3
test cases
100%

TREND
There is nothing to show

SUITES 1 item total
Suite 3
Show all

CATEGORIES 0 items total
Show all

ENVIRONMENT
There are no environment variables

EXECUTORS
There is no information about tests executors

FEATURES BY STORIES 3 items total
Show all

-The document and framework designed by Pavan Nanaware.

4. Future Enhancements

- **CI/CD Integration**: We will Setup Jenkins for automated test execution on each commit.
- **Use of Page Object design pattern**
- **Configuration data files and Read Utility implementation**
- **External data for test cases and their read utilities**

Conclusion:

Due to time constraints and office work, I have created a **simple but functional** test automation framework for ZenInbox, focusing on automating the login functionality. The framework is structured in a way that ensures:

- **Maintainability**: The code is clean, modular, and well-documented, making it easy to extend and maintain.
- **Scalability**: New test cases, data sets, and scenarios can be added effortlessly, providing scalability for future testing needs.
- **Flexibility**: The DataProvider design allows for testing with different sets of input data, reducing the need to write duplicate test cases.
- **Comprehensive Reporting**: Integration with **Allure** provides rich, interactive test reports that detail each test step, the results, and any failed tests, making it easier to track issues.
- **Easy Jenkins Integration**: The framework is built in a way that allows for easy integration into a **Jenkins CI/CD pipeline**. This will automate the execution of tests whenever there is a new code push or at scheduled intervals.

-

-The document and framework designed by Pavan Nanaware.