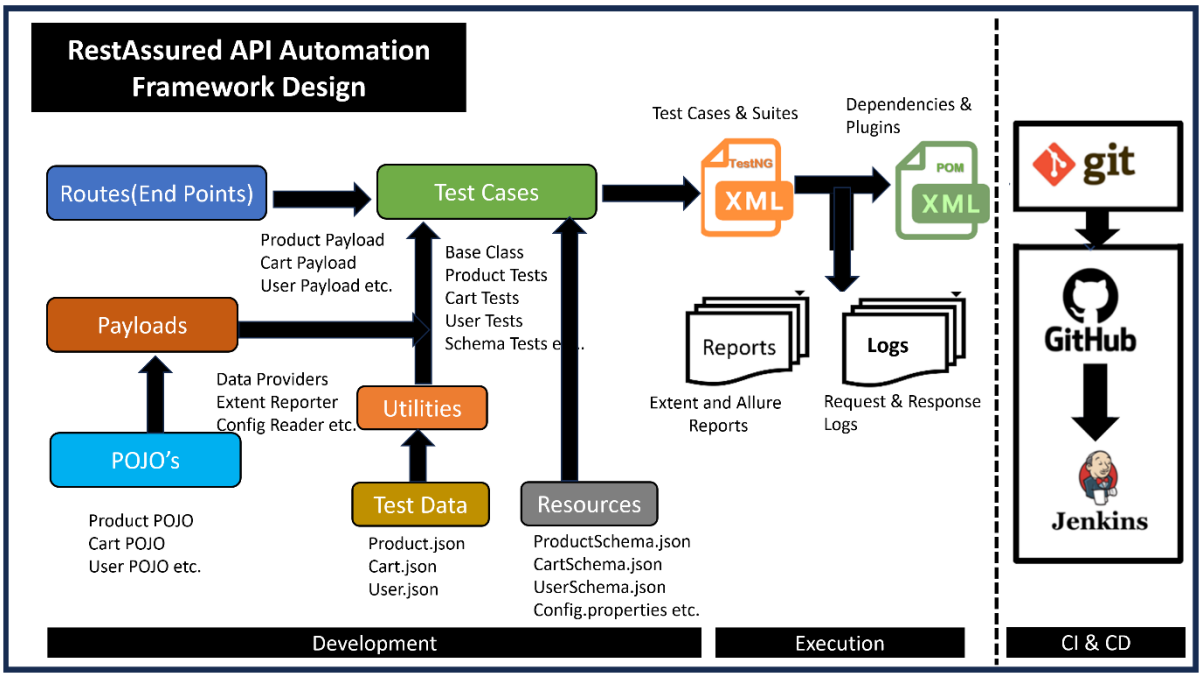
RestAssured API Automation Framework Design



Setting Up the Project

1. Initialize a Maven Project

- Create a new Maven project.

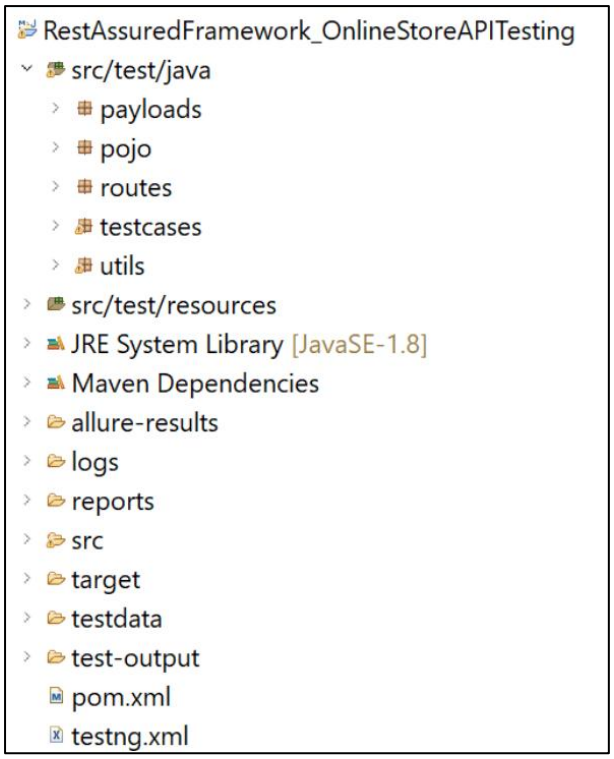
2. Add Required Dependencies in pom.xml

Include the following dependencies:

- rest-assured
- testng
- json-path
- xml-path
- json-schema-validator
- json
- gson
- scribejava-apis
- javafaker
- jackson-databind
- allure-testng
- extentreports

3. Set Up the Folder Structure

- Organize the project into appropriate packages.



Developing the Core Components

4. Define API Endpoints (Routes)

- Create a Routes class under the routes package to store all API endpoints.

5. Create POJO Classes

Define POJOs under the pojo package for request and response handling:

- Product
- Cart
- User
- Login

6. Implement Payload Methods

Develop a Payload.java class to generate payloads dynamically for:

- Product
- Cart
- User
- Login

7. Develop a Base Class

Create a BaseClass.java containing common and helper methods for all test cases.
To add a logging feature, update the setup() method in the Base Class with the following code:

```
RequestLoggingFilter requestLoggingFilter;
ResponseLoggingFilter responseLoggingFilter;

@BeforeClass
public void setup() throws FileNotFoundException {
    // Set base URI for all API calls
    RestAssured.baseURI = Routes.BASE_URL;

    // Setup filters for logging
    FileOutputStream fos = new FileOutputStream(".\\logs\\test_logging.txt");
    PrintStream log = new PrintStream(fos, true);
    requestLoggingFilter = new RequestLoggingFilter(log);
    responseLoggingFilter = new ResponseLoggingFilter(log);

    RestAssured.filters(requestLoggingFilter, responseLoggingFilter);
}
```

8. Maintain Config Details

- Store configuration details in a config.properties file.
- Write a utility to read the file.

Writing and Executing Tests

9. Implement API Test Cases

Develop test classes for different API functionalities:

- ProductTests
- CartTests
- UserTests
- LoginTests

10. Create Data-Driven Tests

- Implement TestNG Data Providers under the utils package.
- Store test data under the testdata folder:
    - Product.json
    - Cart.json
    - User.json
- Write data-driven test cases.

11. Generate Extent Reports

- Implement Extent Reports using an ExtentReporter utility class under utils.

- Configure the ExtentReporter (Listener class) in testng.xml.

12. Generate Allure Reports

- Install Maven & Allure Binaries and configure the system path:
    - https://maven.apache.org/download.cgi
    - https://allurereport.org/docs/install-for-windows/
- Verify installation:

```
mvn --version
allure --version
```

- Run tests using testng.xml.
- Generate Allure Reports:

```
allure serve <<Location of allure-results folder>>
```

13. Execute Failed Tests

- Re-run only failed test cases using testng-failed.xml.

14. Run Tests in Parallel

- Configure testng.xml for parallel execution by setting parallel="tests".

15. Execute Tests via Maven or Batch Script

- Run tests from the command prompt using Maven.
- Alternatively, execute tests using a run.bat file.

Continuous Integration and Version Control

16. Push Code to GitHub

- Commit and push the project to a GitHub repository.

17. Integrate Tests with Jenkins

- Configure and execute tests in Jenkins for continuous integration.