**1. Allure Reports Test Automation**

Allure is a modern, flexible test reporting framework used to generate beautiful, detailed, and interactive test reports.  
It is especially popular in Selenium-based automation projects with Java, Maven, and TestNG.

It turns your automated test results into visual, browser-friendly dashboards that help QA teams and developers understand what happened during execution.

**2. Why to Use Allure Reports?**

Allure makes your test results more useful and readable:

* ✅ Displays test outcomes: passed, failed, skipped
* 📊 Includes charts and timelines
* 🔍 Shows individual test details with steps
* 📎 Supports screenshots and log attachments
* 🏷️ Groups failures by custom categories (like known bugs)

In short: **Allure adds real meaning and insight to your automated tests**.

**3. Allure Workflow**.

1. During test execution (e.g., using TestNG), Allure listens to test events.
2. It writes test data (name, result, steps) to files inside allure-results/.
3. You then run:
4. allure serve or allure generate

to convert those files into a **dynamic HTML report.**

**4. Key Features of Allure**

* **@Step annotations**: Used in your code to split test logic into visible steps
* **Attachments**: Add screenshots, logs, JSON, text files to failed tests
* **Categories**: Define types of failures (e.g., flaky tests, product bugs)
* **History view**: When used with CI, it tracks stability across builds
* **Retry support**: Highlights flaky/unstable tests automatically

**5. Allure with CI/CD (Jenkins, GitHub Actions)**

In DevOps workflows, Allure fits perfectly into your CI/CD pipeline:

* Jenkins can show Allure reports as a post-build action
* GitHub Actions can generate and upload reports to artifacts
* Teams get real-time test feedback via the browser

This makes Allure an essential tool for **automated quality visibility** in Agile projects.

**6. Summary**

Allure turns dry test output into rich, actionable reports.

It helps testers, developers, and managers:

* See failures clearly
* Understand test behaviour
* Track trends and test health over time

Whether you're working locally or in CI/CD, **Allure helps your tests speak clearly**.

**Working With Allure.**

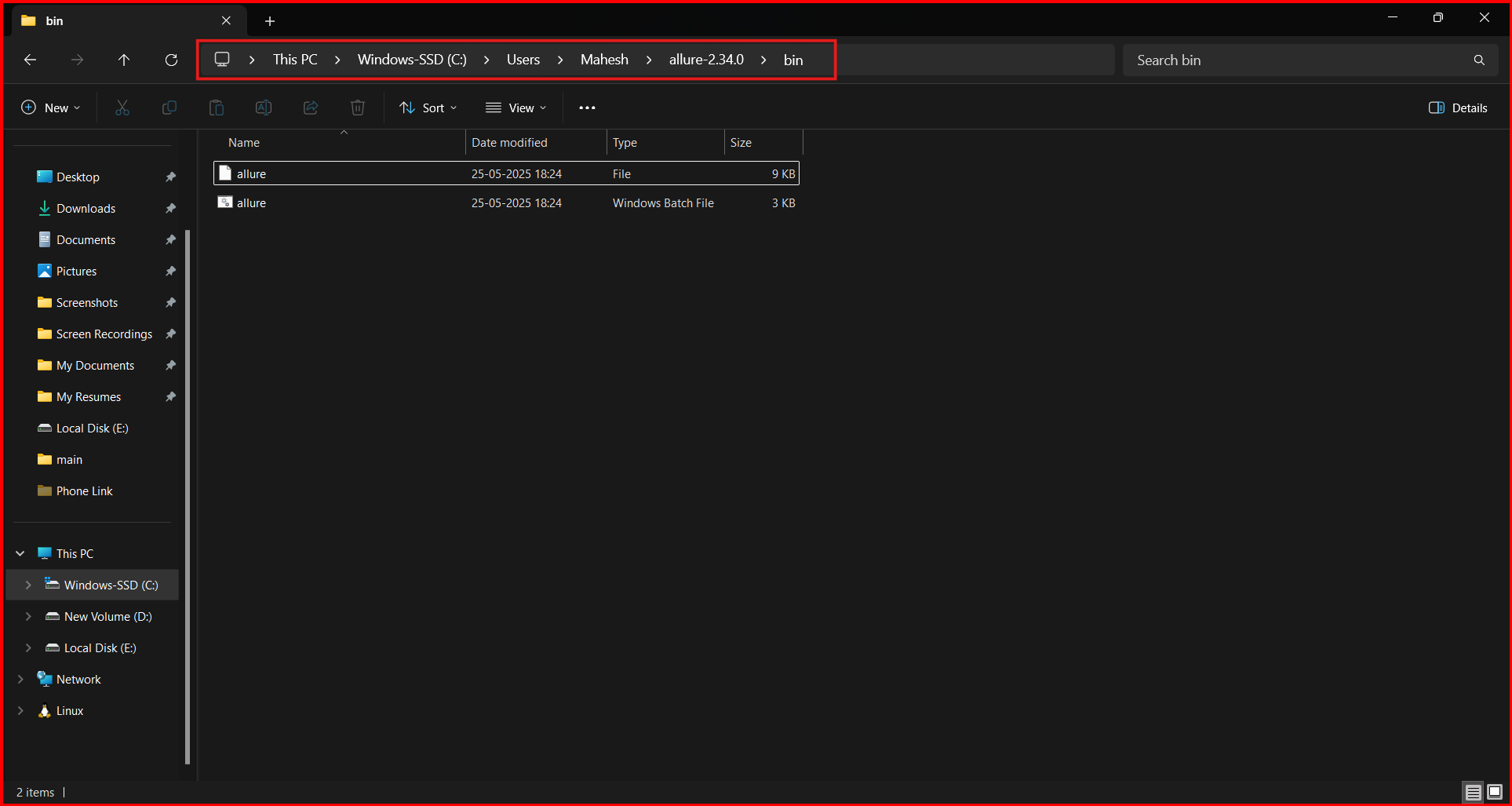
First, ensure the following dependencies are available in your project : java, Maven, TestNG, Allure, and Selenium Java.

Install allure zip file from here: <https://github.com/allure-framework/allure2/releases/tag/2.34.0>

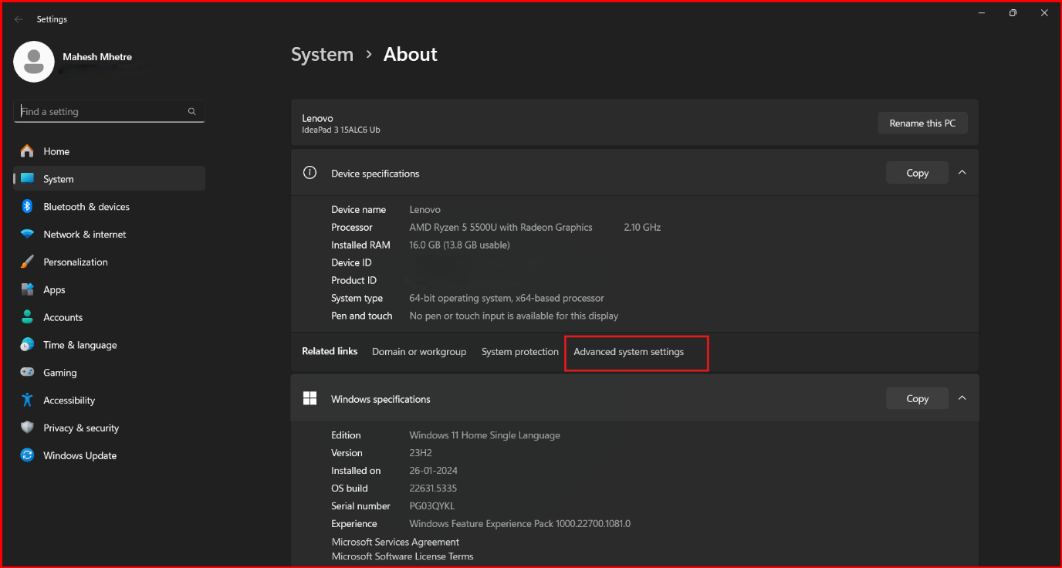
To install Allure, extract the ZIP and add it to your system’s environment variables:

Step 1: Extract the ZIP file and save it in a permanent location (not a temporary directory).

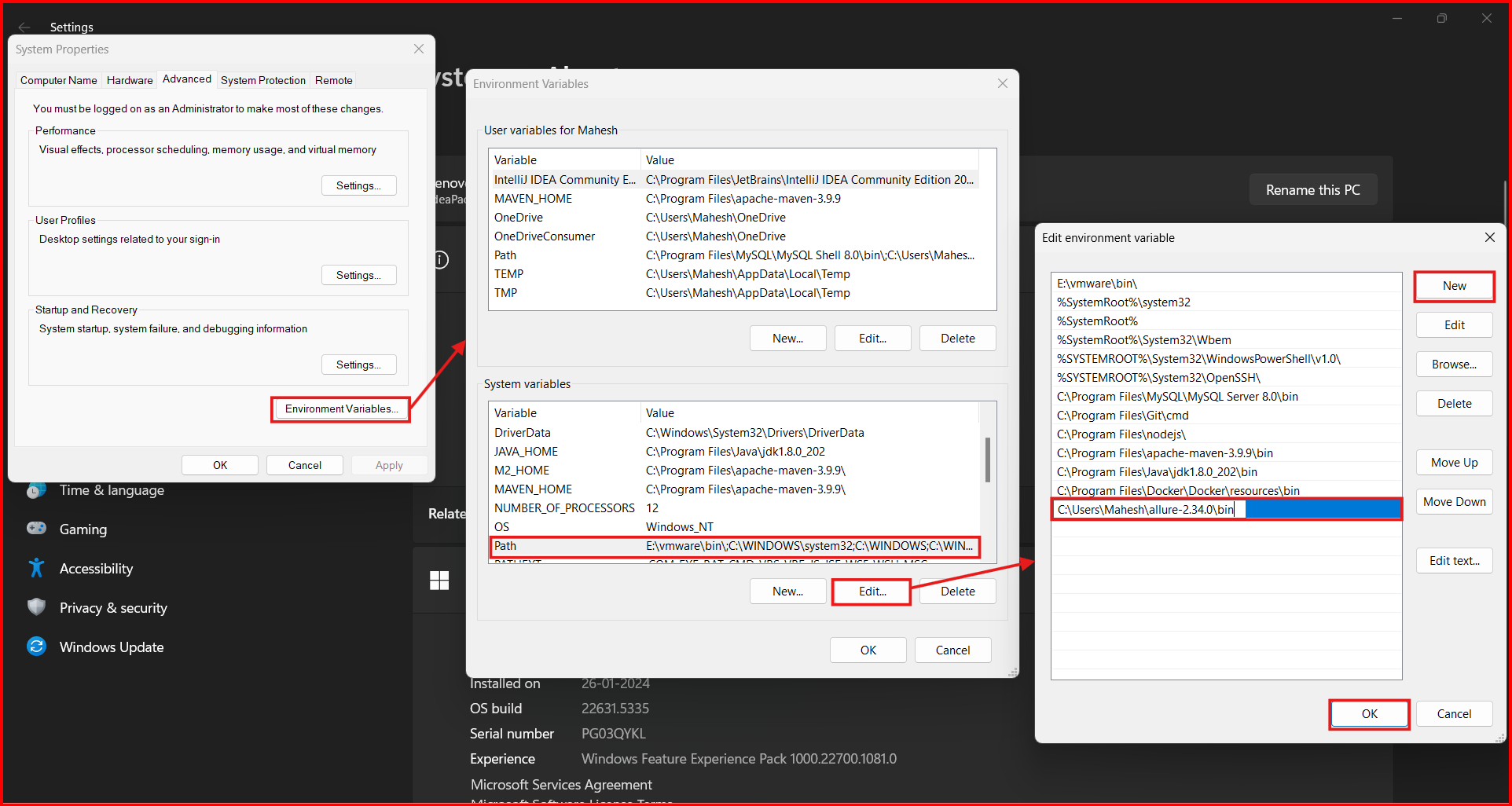
Step 2: Open the extracted folder and copy the path to the 'bin' directory.



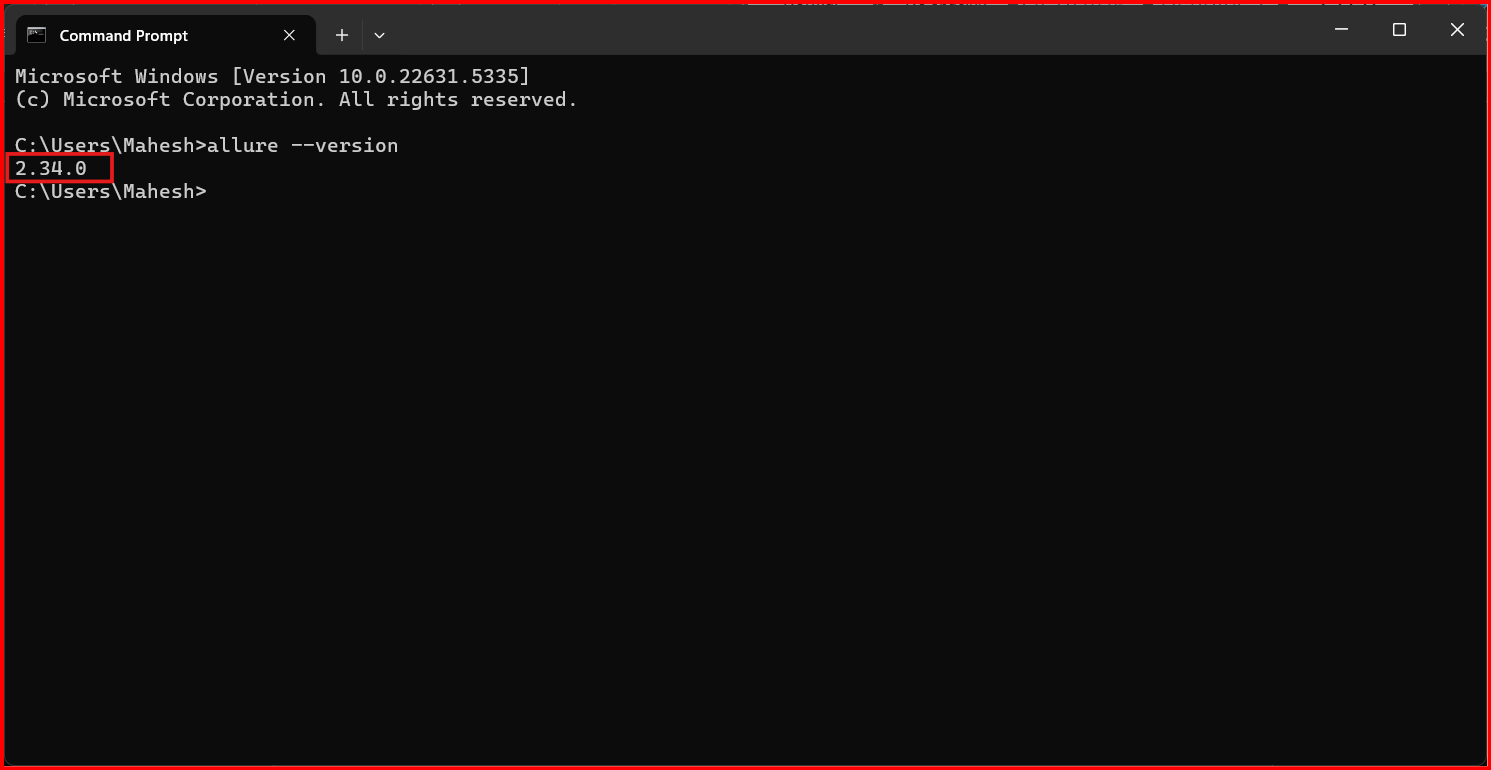
Step 3: Navigate to 'System Settings > About > Advanced System Settings'.



Step 4: Add the 'bin' path to your system’s environment variables under the PATH section.



Step 5: Open Command Prompt and type `allure --version` to confirm Allure is installed correctly.



If the version is displayed, Allure was successfully installed on your system.

Adding Allure Reporting to Your Maven Project

To enable Allure reporting, add the following dependencies to your Maven project's `pom.xml`:

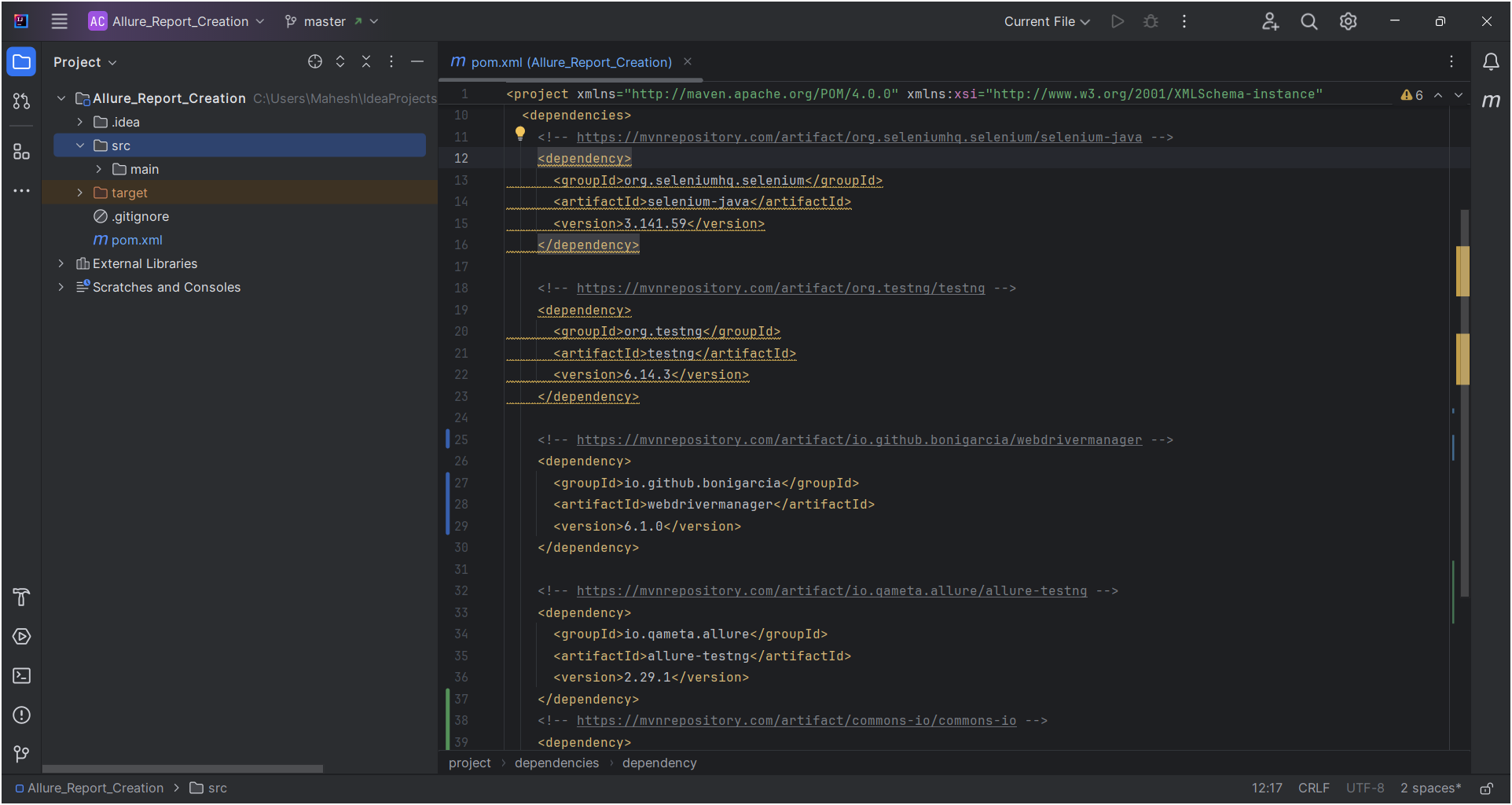
Note: Since we're using Allure with TestNG, you must include the TestNG dependency.

testng dependency : [Maven Repository: org.testng » testng](https://mvnrepository.com/artifact/org.testng/testng)

allure-testng dependency : [Maven Repository: io.qameta.allure » allure-testng](https://mvnrepository.com/artifact/io.qameta.allure/allure-testng)

WebDriver Manager : [Maven Repository: io.github.bonigarcia » webdrivermanager](https://mvnrepository.com/artifact/io.github.bonigarcia/webdrivermanager)

selenium-java dependency : [Maven Repository: org.seleniumhq.selenium » selenium-java](https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java)

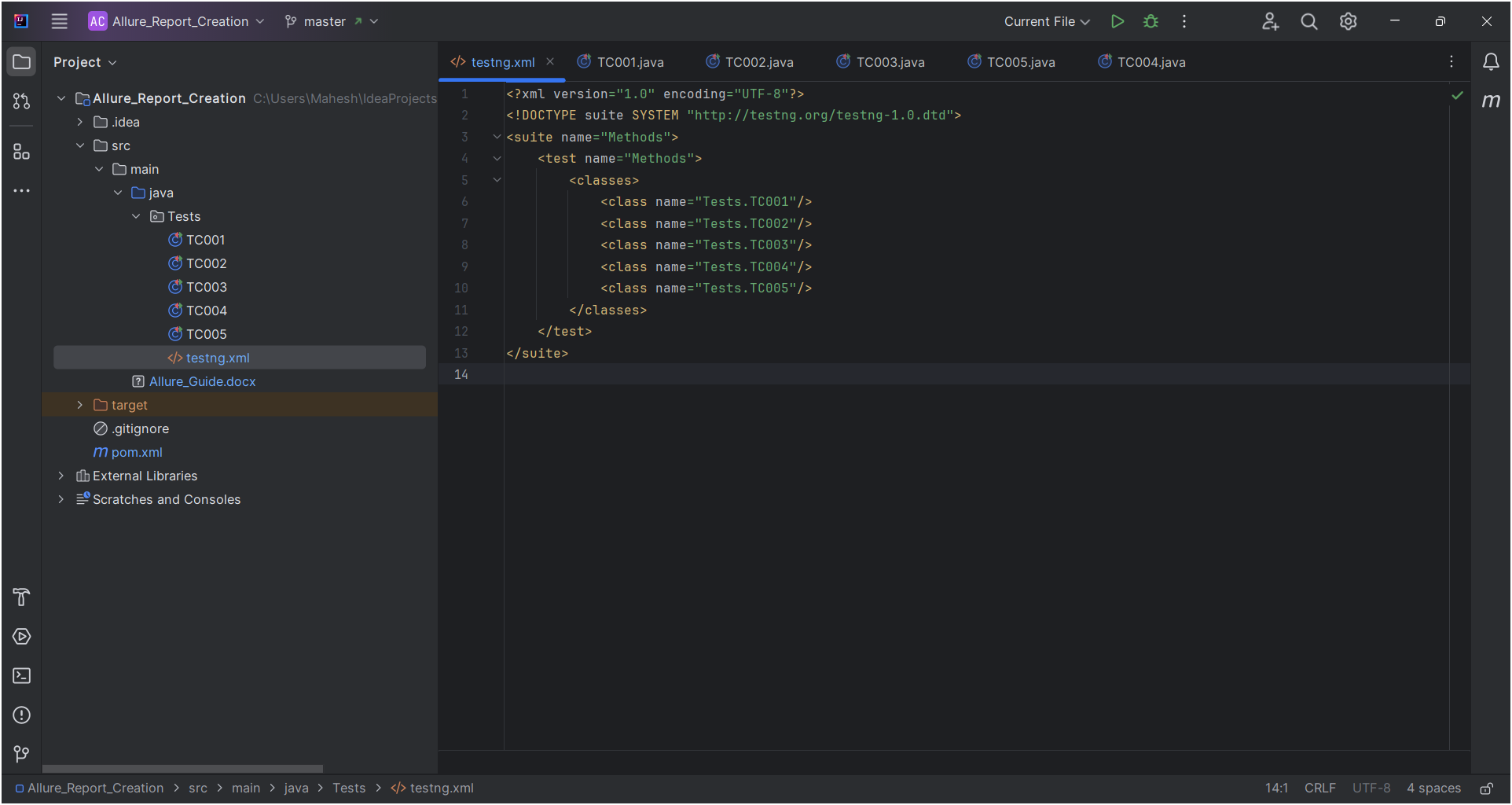
****

\*\*✅ Setup Complete\*\*

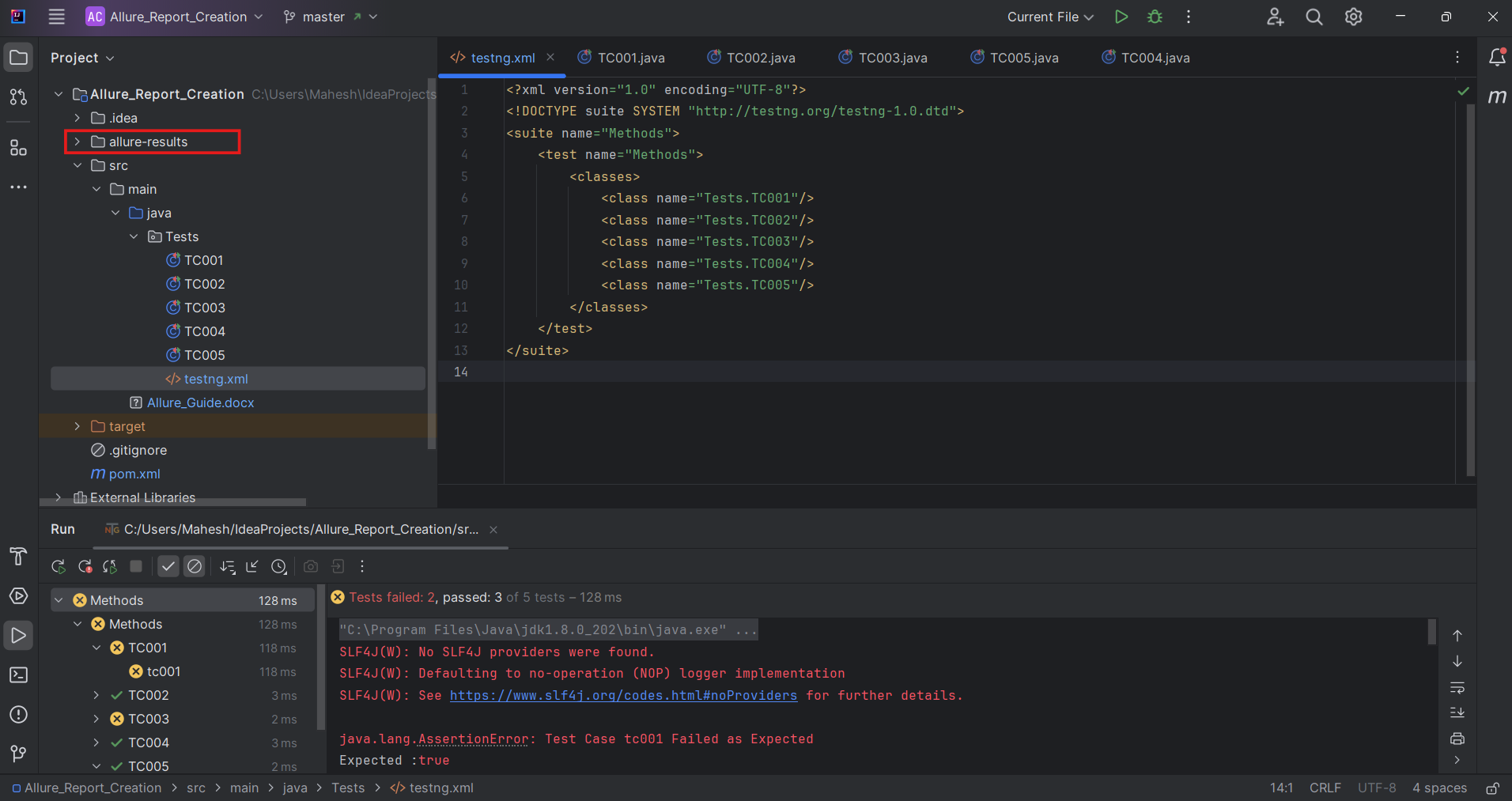
\*\*📊 Creating an Allure Report\*\*

Allure Report allows you to create html report of you executed TestNG Test Cases so can easily access you Test Execution Report into Attractive Graphical way , so lets begin with it follow the steps given below..

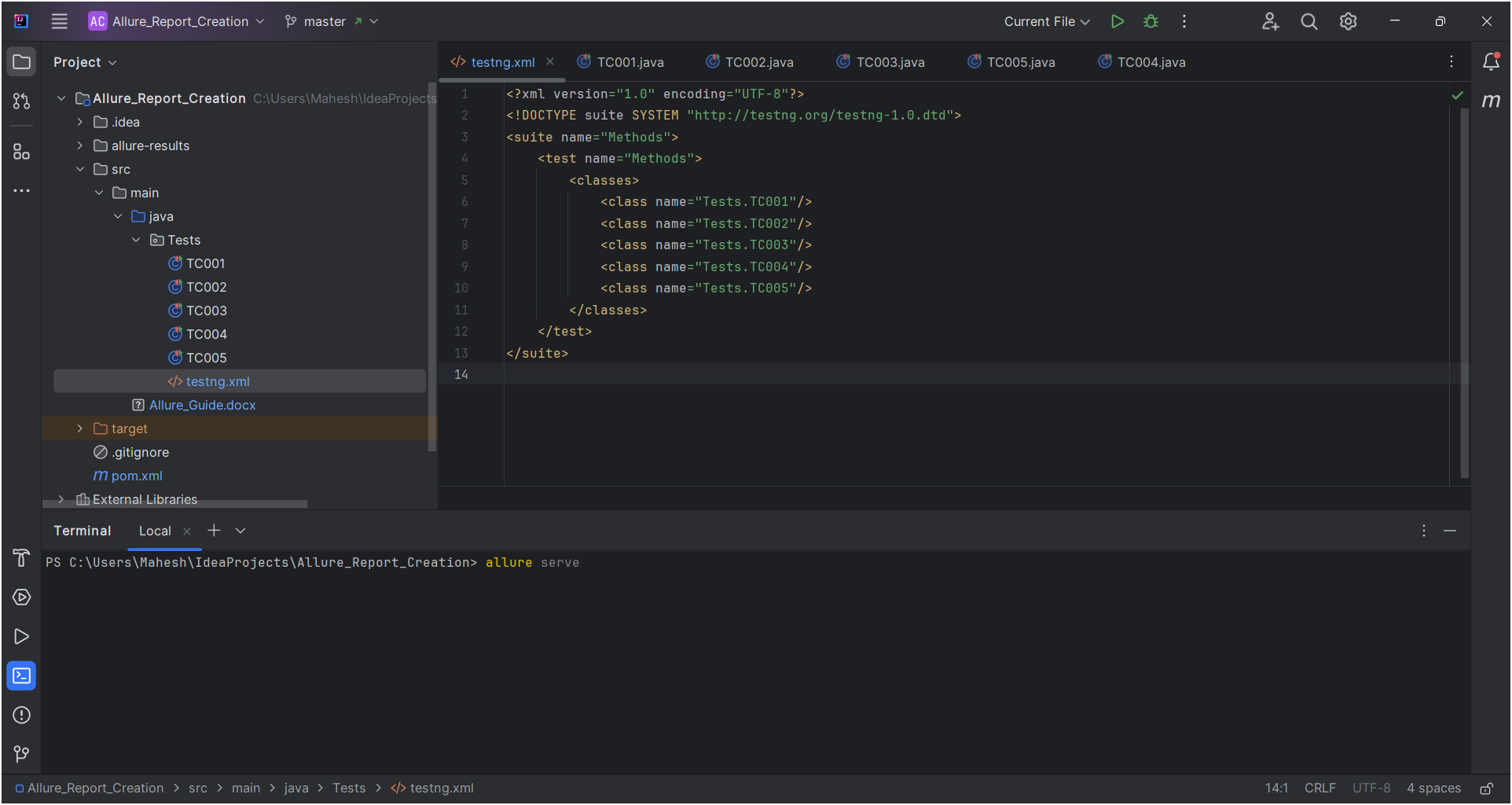
Step 1: Create your test cases and a suite files.



Step 2: Run the `testng.xml` suite file. After execution, a folder named `allure-results` will appear in your project directory.

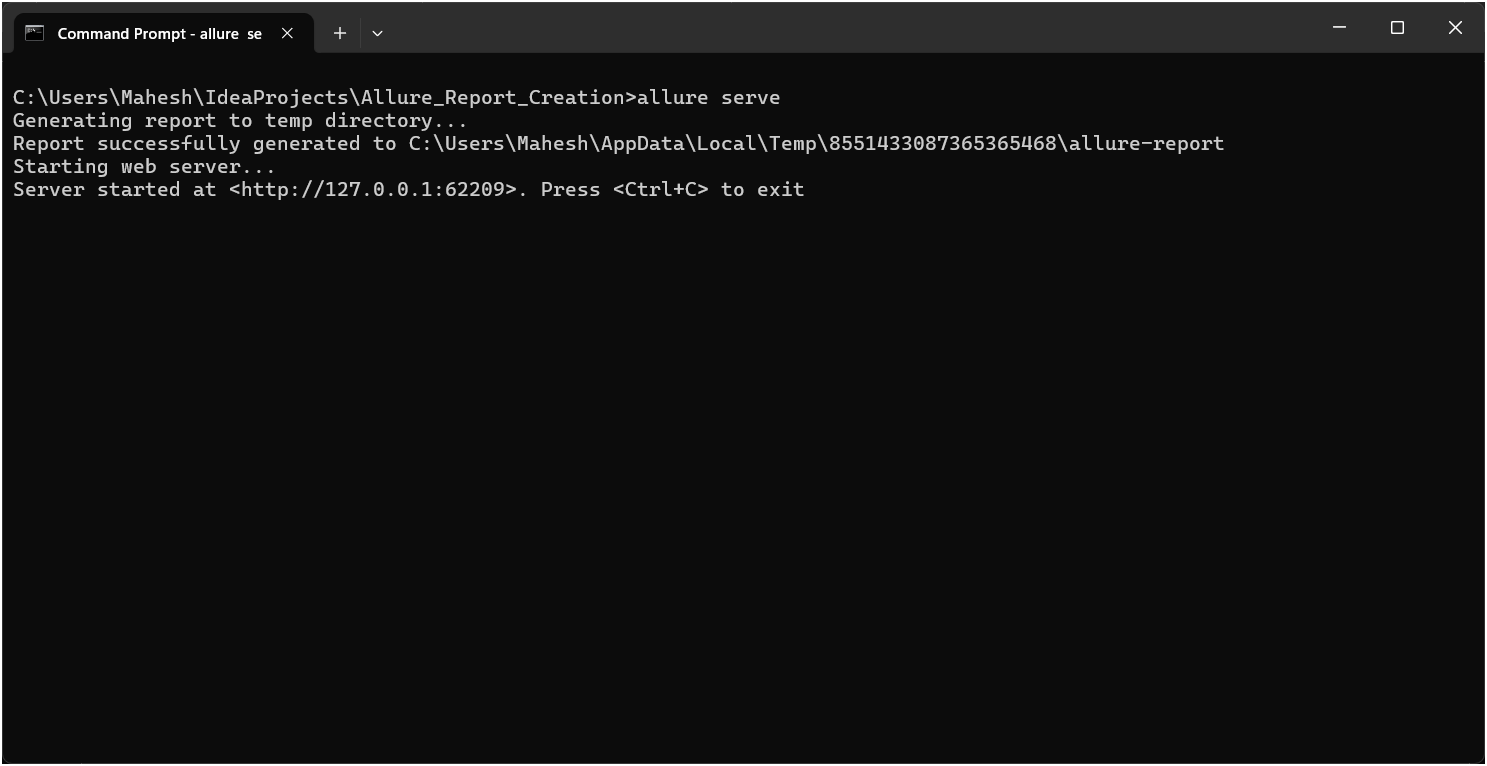


Step 3: Open the terminal in your project directory and run the command `allure serve`.

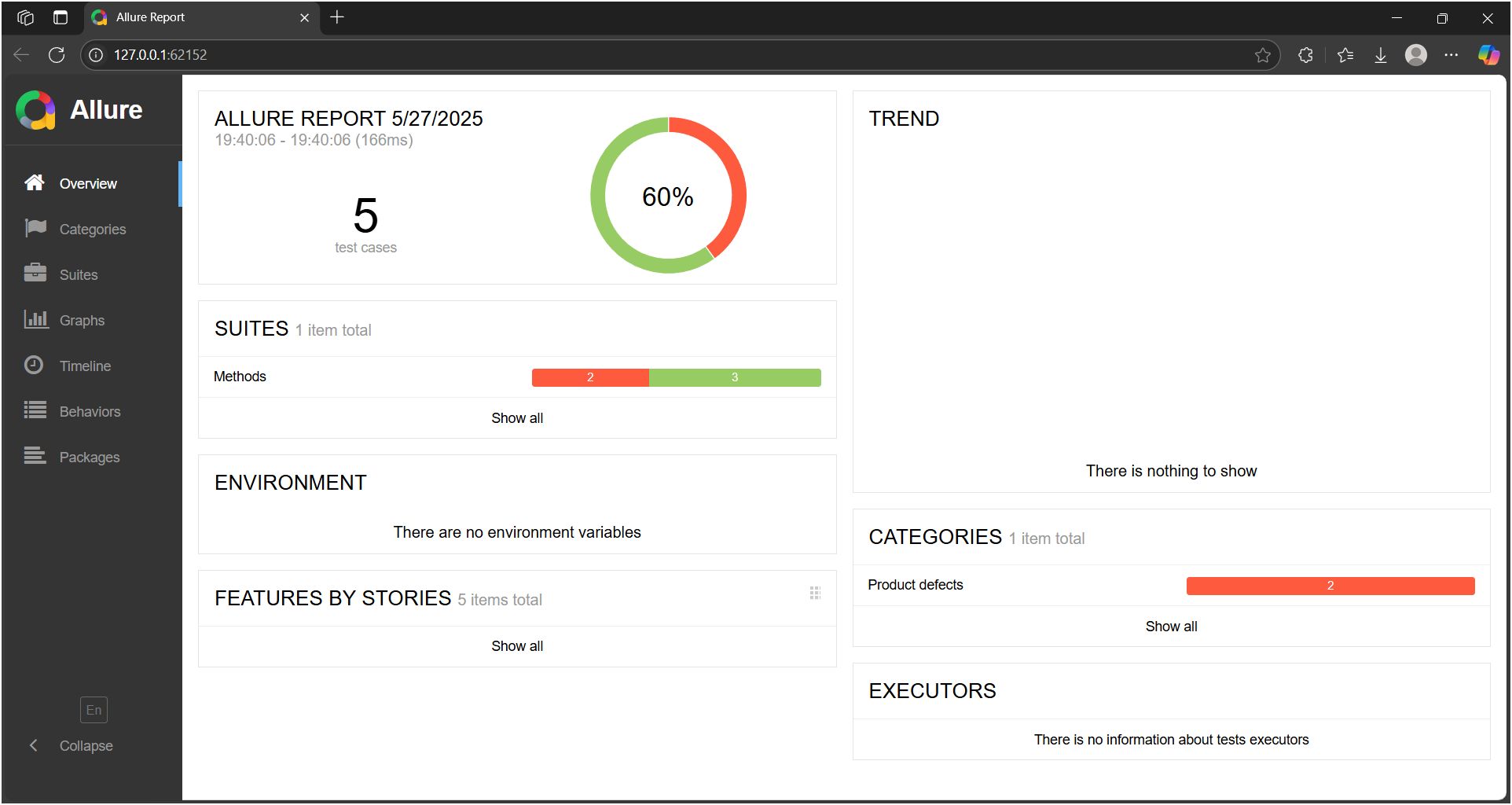


if you want can generate report from command prompt or power

* + open command prompt from your project folder.
  + Run command allure serve



🎉 Congratulations! Your Allure report will now be generated and viewable in your browser.



Note: This Project is user to clear attachment of allure only now We work on our main Project.

**\*\*\*\*\*\*\*\*\*\*\*\***

Allure Annotations:

* @Description("scenario where a user tries to register without filling in the required fields.")
* @Epic("User Registration")
* @Severity(SeverityLevel.*CRITICAL*)
* @Feature("Feature1: Registration") // this will shown in Behaviour Section
* @Story("Story2: Empty Required Fields") // this will shown in Behaviour Section

Adding Screen-Shots with Report:

First we have to get screen shot on error for that we use ITestResult class.

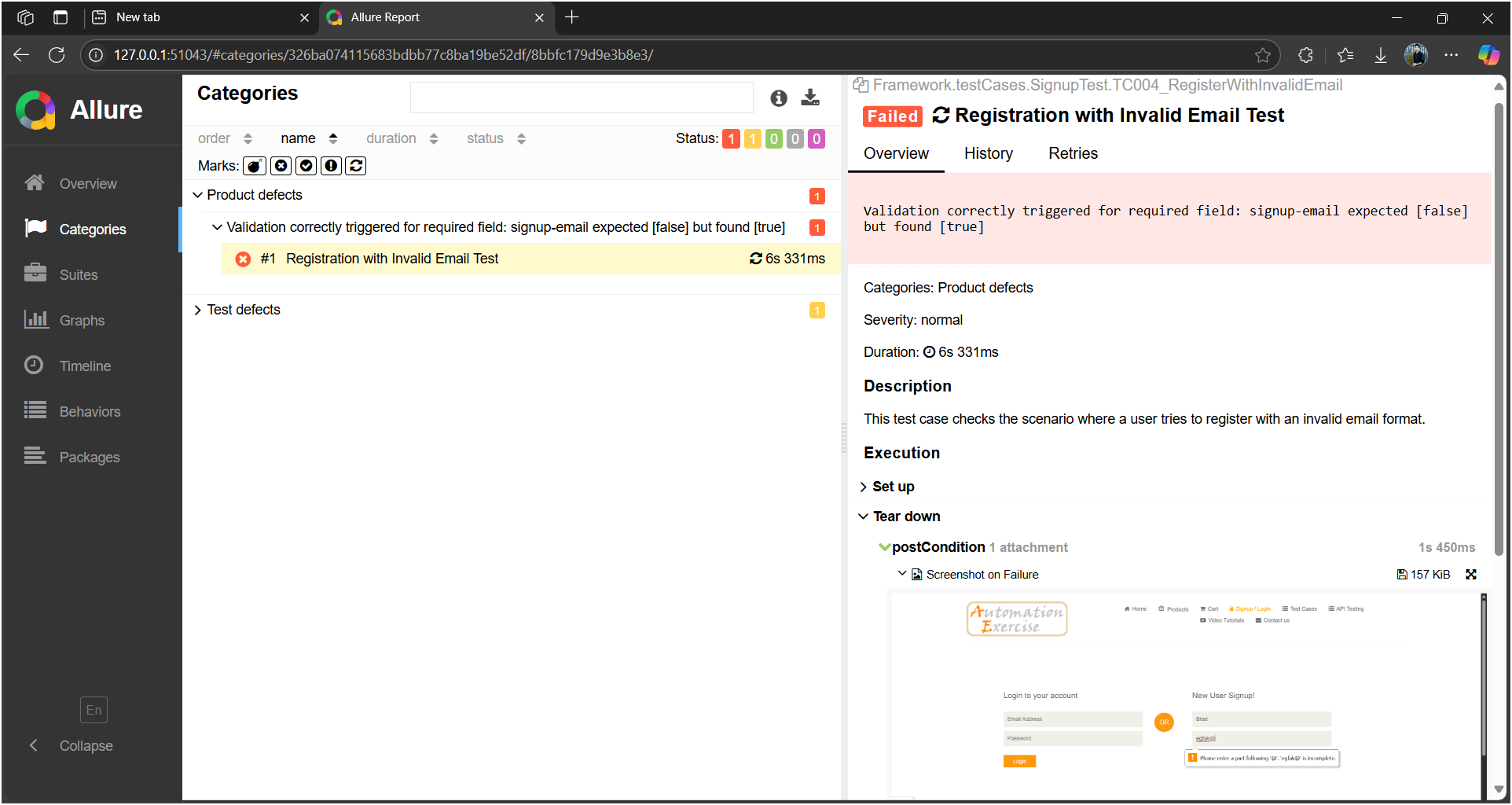
Program to get Screen shot on test failure:

|  |
| --- |
| import org.apache.commons.io.FileUtils;  import org.openqa.selenium.TakesScreenshot;  import org.openqa.selenium.OutputType;  import org.openqa.selenium.WebDriver;  import org.testng.ITestResult;  // This method captures a screenshot when a test fails.  public static void captureOnFailure(WebDriver driver, ITestResult result) throws Exception {  if (result.getStatus() == ITestResult.*FAILURE*) {  Thread.*sleep*(1000);  File screenshot = ((TakesScreenshot) driver).getScreenshotAs(OutputType.*FILE*);  try {  FileUtils.*copyFile*(screenshot, new File("screenshots/" + result.getName() + ".png"));  System.*out*.println("Screenshot saved for failed test: " + result.getName());  } catch (IOException e) {  e.printStackTrace();  }  } } |

Program to Attach our Screen shot to allure Report we have to change in the above program:

|  |
| --- |
| import org.apache.commons.io.FileUtils;  import org.openqa.selenium.TakesScreenshot;  import org.openqa.selenium.OutputType;  import org.openqa.selenium.WebDriver;  import org.testng.ITestResult;  import io.qameta.allure.Attachment;  // This method captures a screenshot when a test fails and attaches it to the Allure report. public static void captureOnFailure(WebDriver driver, ITestResult result) throws Exception {  if (result.getStatus() == ITestResult.*FAILURE*) {  Thread.*sleep*(1000); // Wait for a second to ensure the page is fully loaded before capturing the screenshot.  File screenshot = ((TakesScreenshot) driver).getScreenshotAs(OutputType.*FILE*); // Capture screenshot on failure  try {  FileUtils.*copyFile*(screenshot, new File("screenshots/" + result.getName() + ".png")); // Save the screenshot with the test name  *log*().info("Screenshot saved for failed test: " + result.getName());  Allure.*addAttachment*("Screenshot on Failure", FileUtils.*openInputStream*(screenshot)); // Attach the screenshot to Allure report  } catch (IOException e) {  e.printStackTrace();  }  } } |

Now we are done with Screen Shot Attachment now run your Suite file and run the Command : allure serve.



You can see our Screenshot is Successfully Attached to Your Failed Test Case..

Allure report History: