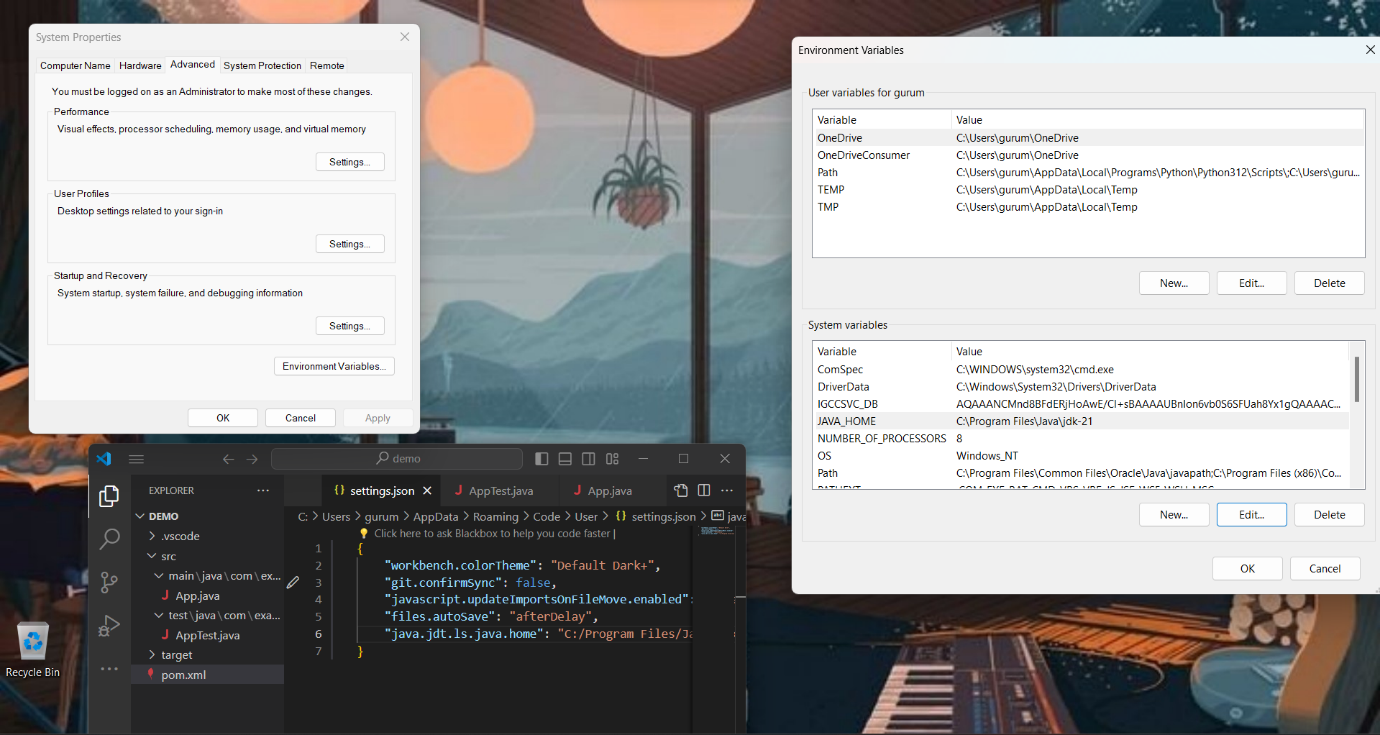
***Day – 1 Configuration for setting up the environment.***



***Report :***

Setting Up Environment for Java Maven Project

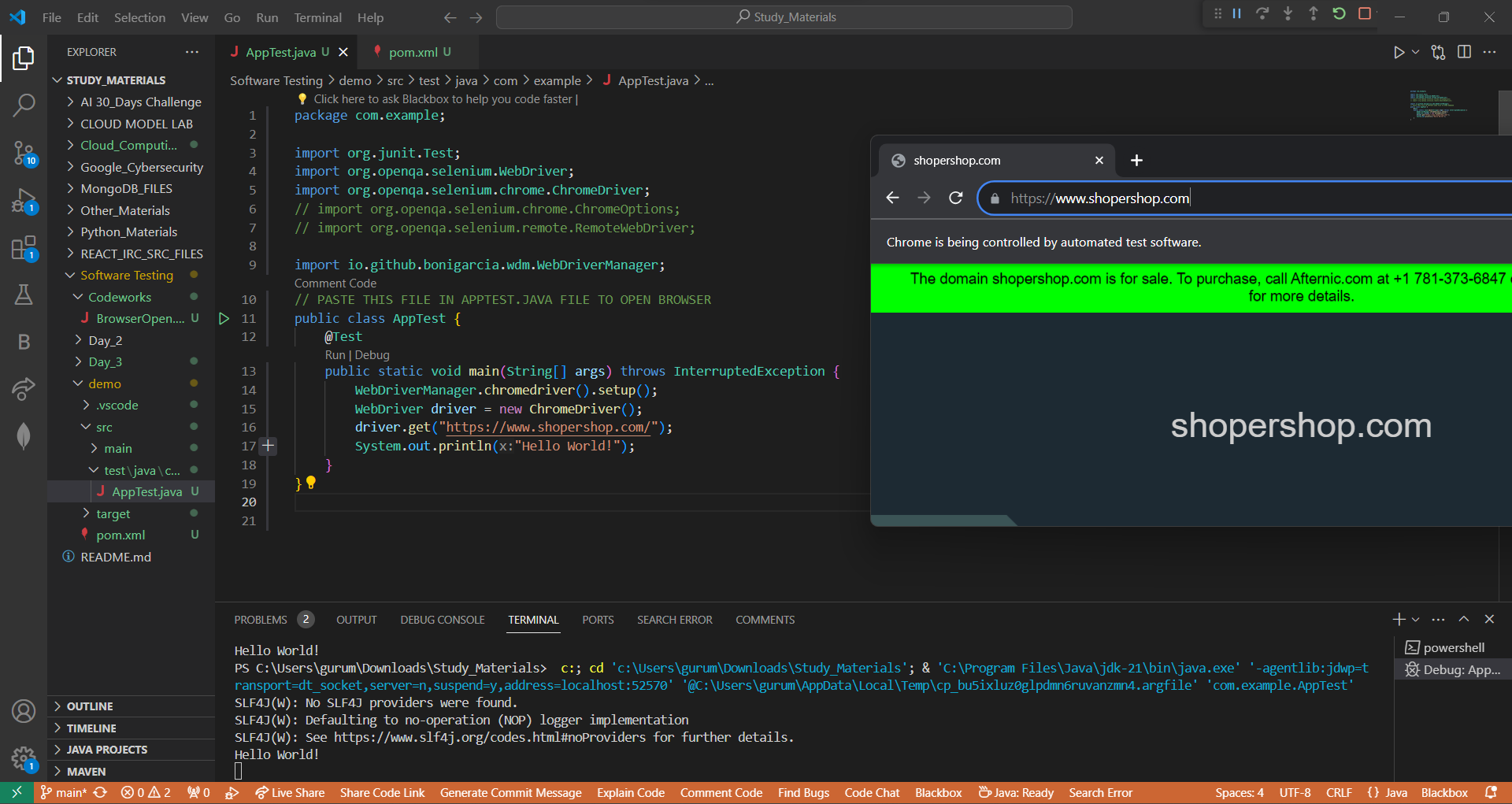
***Steps:***

* Install JDK: Download and install the latest JDK, set JAVA\_HOME.
* Install Maven: Download, extract, add to PATH, verify with mvn -version.
* Configure Maven: Optionally modify settings.xml for custom configurations.
* Create Project: Use Maven archetype to generate project structure.
* Import into IDE: Optional step to import project into IDE for development.
* Build & Run: Navigate to project directory, execute mvn clean install, run application.

***References:***

* Maven Documentation: maven.apache.org/guides/index.html
* Oracle JDK Downloads: oracle.com/java/technologies/javase-jdk11-downloads.html
* AdoptOpenJDK: adoptopenjdk.net/
* Apache Maven: maven.apache.org/

***Day – 2 Opening Shopershop website automatically***



***Report :***

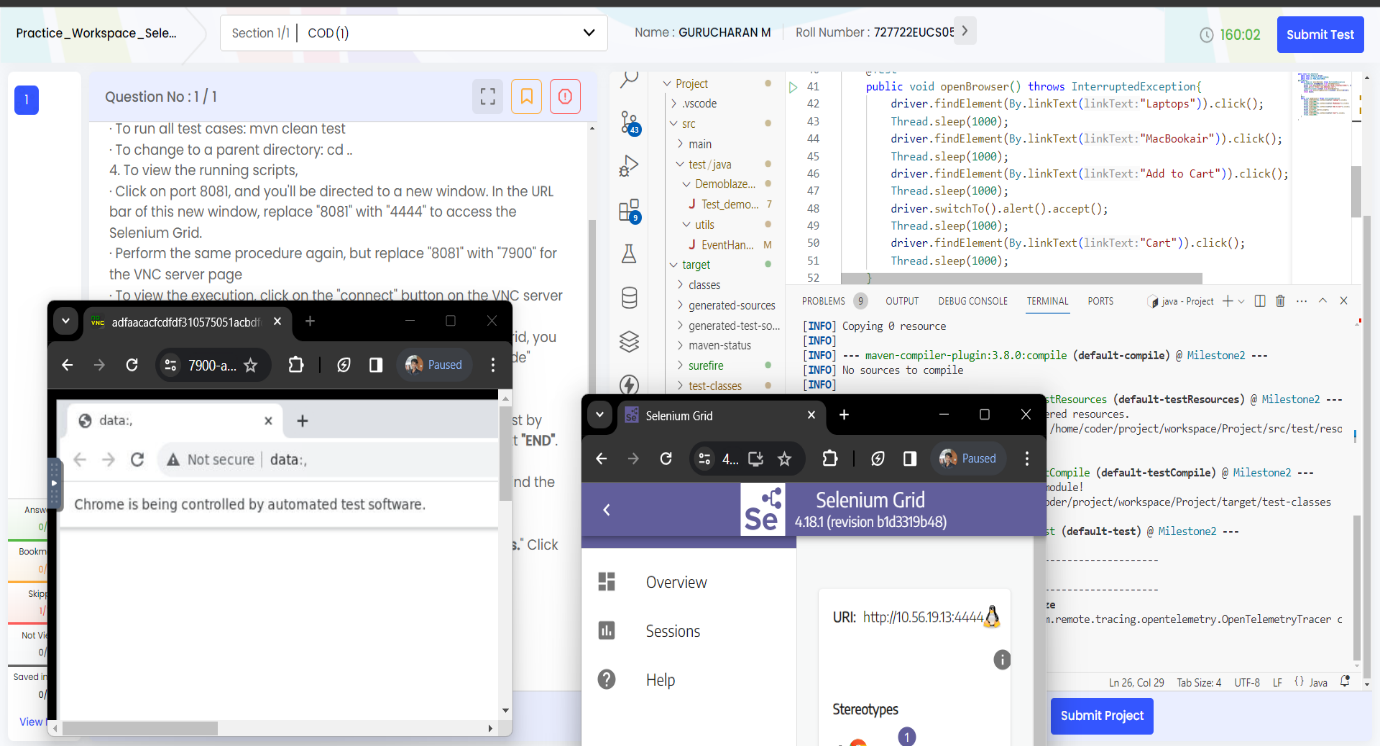
***Steps:***

* Install Selenium: Use pip install selenium to install the Selenium package.
* Download WebDriver: Obtain the WebDriver executable for your preferred browser (e.g., Chrome, Firefox).
* Set WebDriver Path: Specify the path to the WebDriver executable in the script.
* Write Python Script: Use Selenium to write a Python script to open the Shopershop website automatically.
* Execute Script: Run the Python script to open the website in a browser window.

***References:***

* Selenium Documentation: https://www.selenium.dev/documentation/en/
* Shopershop Website: [Provide website URL here]
* WebDriver Downloads:
* Chrome WebDriver: https://sites.google.com/a/chromium.org/chromedriver/downloads
* Firefox WebDriver: <https://github.com/mozilla/geckodriver/releases>

***Day – 3 Using Web Locators in websites***

******

***Report :***

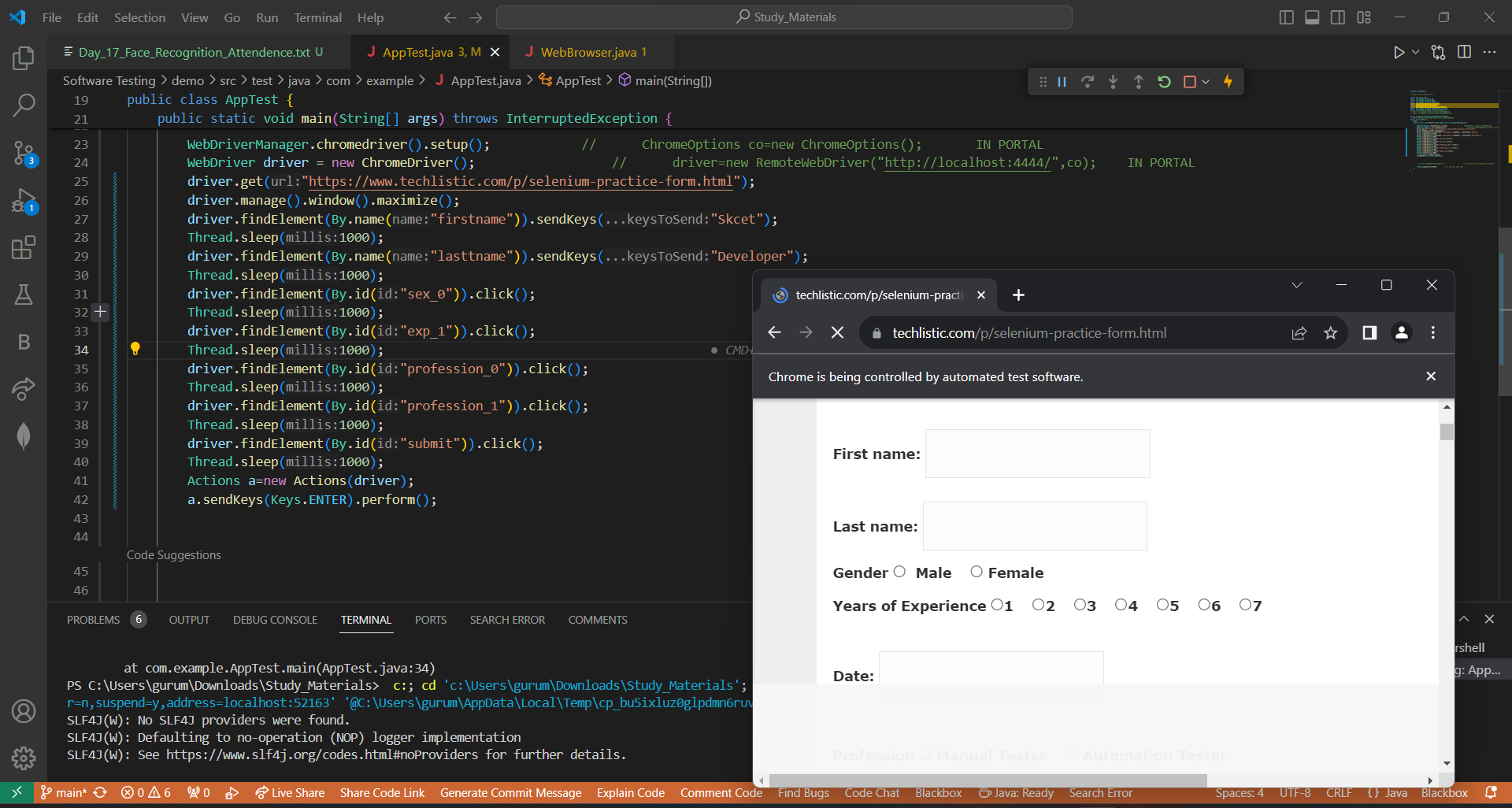
***Steps:***

* Inspect Element: Use browser developer tools (usually accessed via right-click > Inspect) to inspect the HTML structure of the web page.
* Identify Unique Attributes: Look for attributes like id, class, name, etc., that can uniquely identify the element.
* Choose Appropriate Locator: Based on the uniqueness and stability of attributes, select the most suitable locator type.
* Implement Locator in Code: Use the chosen locator type to interact with the element in your automation scripts or web development code.
* Verify and Debug: Test the functionality to ensure the locator works as expected. If not, refine the locator or choose an alternative.

***References:***

* W3Schools. "HTML DOM Attributes." Link
* MDN Web Docs. "Locating DOM elements using selectors." Link
* Selenium Documentation. "Locating Elements." Link

***Day – 4 Using Web Elements to interact with Forms***



***Report :***

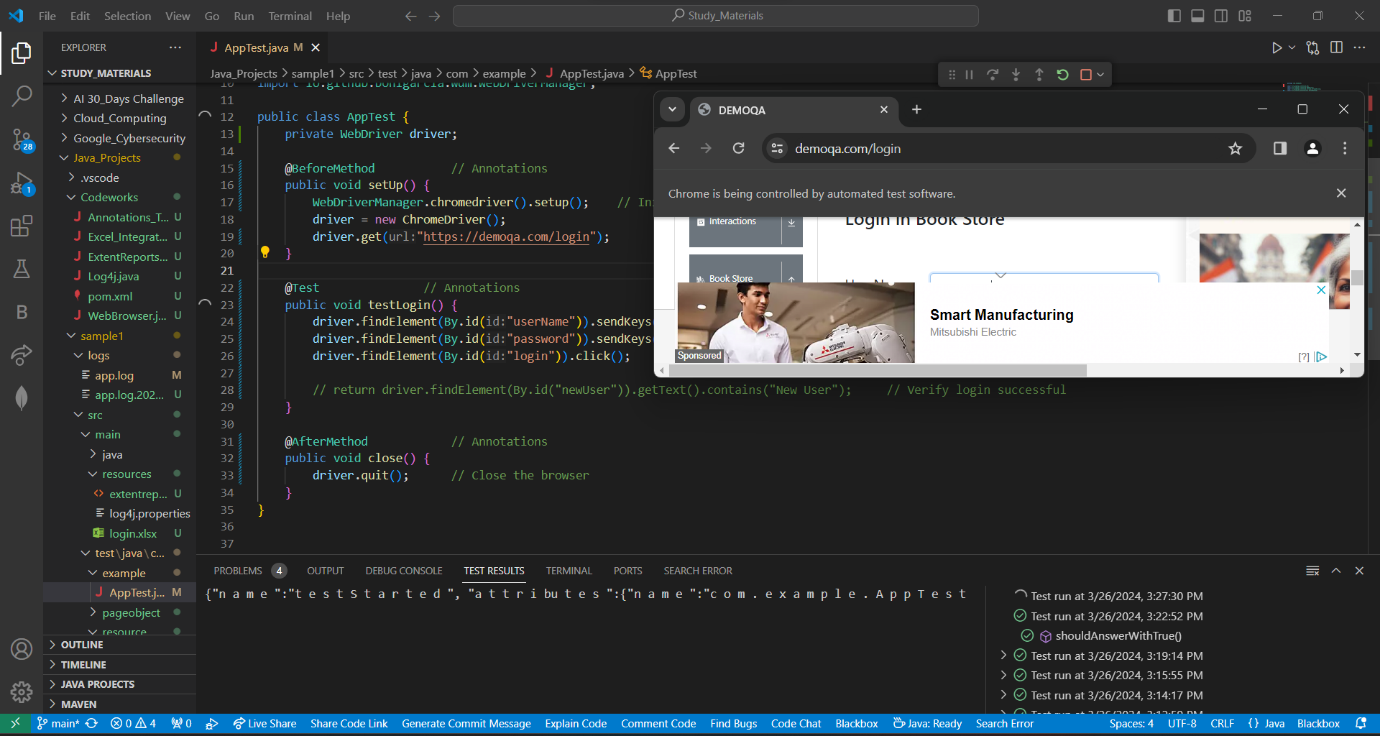
***Steps :***

* Set up your project by configuring Selenium WebDriver, TestNG, and Extent Reports dependencies.
* Write test cases where you interact with web elements using WebDriver and incorporate assertions for validation.
* Implement an Extent Report listener to generate HTML reports, capturing relevant information such as test steps and screenshots.
* Execute your tests using TestNG to run the defined test cases, triggering automatic generation of HTML reports by Extent Reports.
* Review the generated HTML reports after test execution to analyze comprehensive test results, including detailed step-by-step information.

***References :***

* Selenium WebDriver Documentation: https://www.selenium.dev/documentation/en/webdriver/
* TestNG Documentation: https://testng.org/doc/documentation-main.html
* Extent Reports Documentation: <https://extentreports.com/docs/versions/4/java/>

***Day – 5 Using Test Annotations***

******

***Report :***

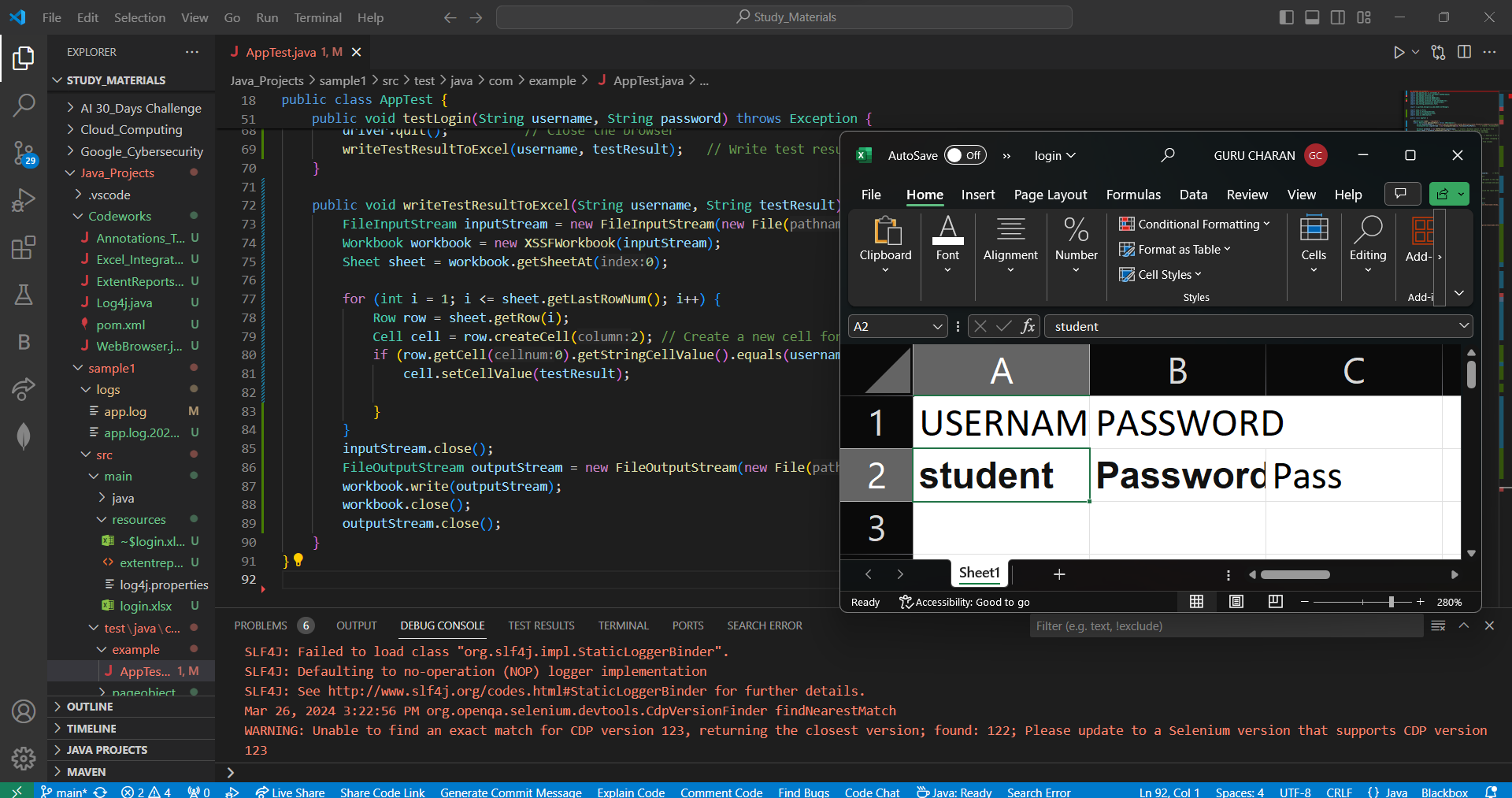
***Steps :***

* Introduction to test annotations in Java Selenium, referencing official Selenium documentation.
* Explanation and implementation of @Before annotation for test setup, referring to Selenium documentation and Java programming guides.
* Implementation of @After annotation for cleanup activities post-test execution, with guidance from Selenium and JUnit documentation.
* Usage and examples of @Test annotation for actual test cases in Java Selenium, citing examples from Selenium WebDriver and JUnit official resources.
* Additional insights and best practices from experienced Java Selenium developers, sourced from forums like Stack Overflow and blogs authored by Selenium experts.

***References :***

* Selenium Documentation: https://www.selenium.dev/documentation/en/
* JUnit Documentation: https://junit.org/junit5/docs/current/user-guide/
* Java Documentation: https://docs.oracle.com/en/java/

***Day – 6 Using Excel Integration***

******

***Report :***

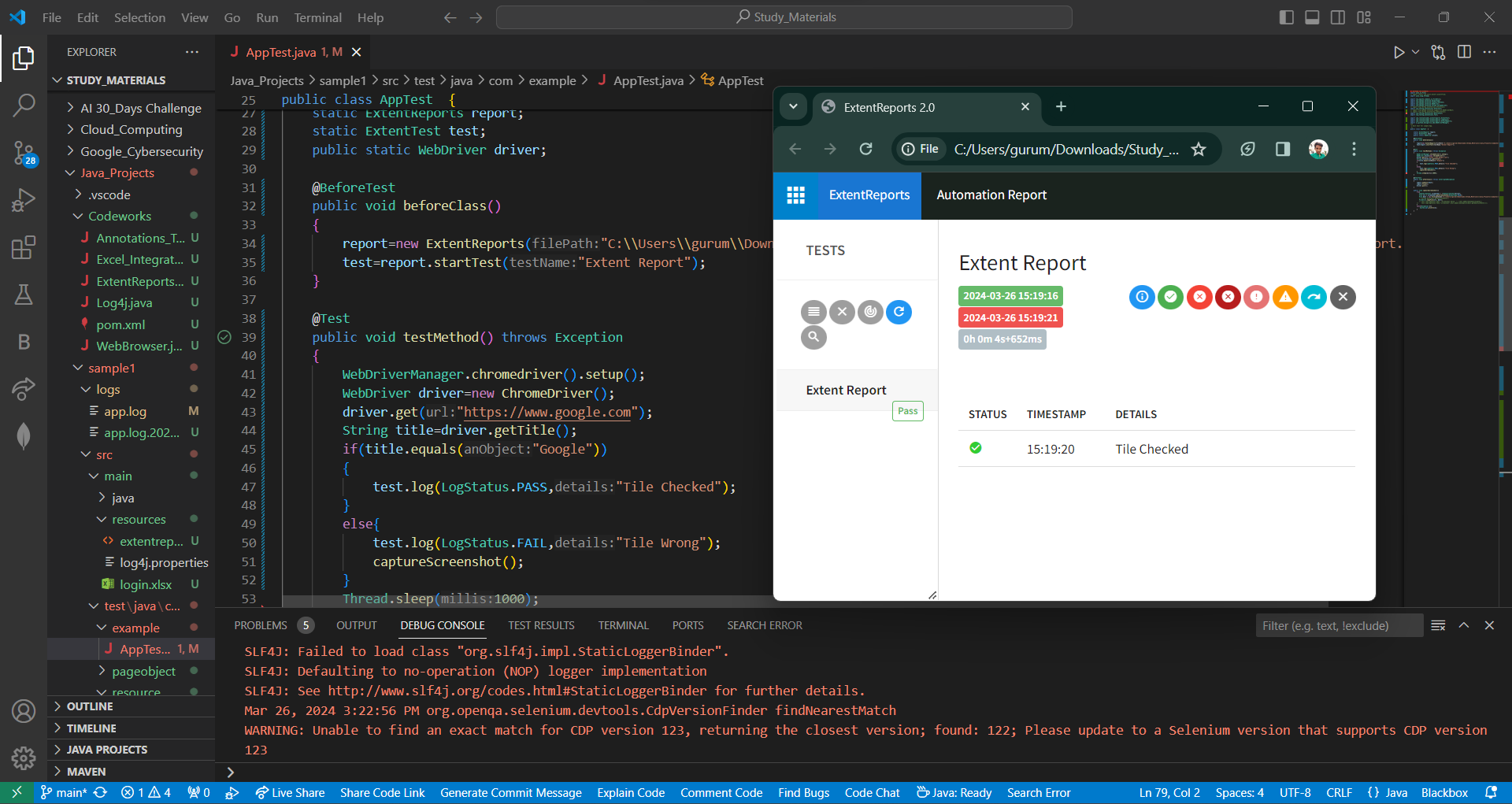
***Steps :***

* Introduction to the necessity of integrating Excel with Selenium for data-driven testing.
* Setup and configuration steps for Apache POI library, a commonly used tool for Excel manipulation in Java.
* Reading data from Excel sheets using Apache POI, encompassing opening files, accessing specific sheets, and retrieving data.
* Writing data to Excel sheets utilizing Apache POI functionalities, covering aspects such as creating new sheets, modifying existing ones, and appending data.
* Integration of Excel data into Selenium tests, demonstrating methods for parameterized testing and data-driventesting.

***References :***

* Apache POI OfficialDocumentation: https://poi.apache.org/
* Selenium Official Documentation: https://www.selenium.dev/documentation/en/
* Stack Overflow: [https://stackoverflow***.***com/](https://stackoverflow.com/)

***Day – 7 Using Extent Reports***

******

***Reports :***

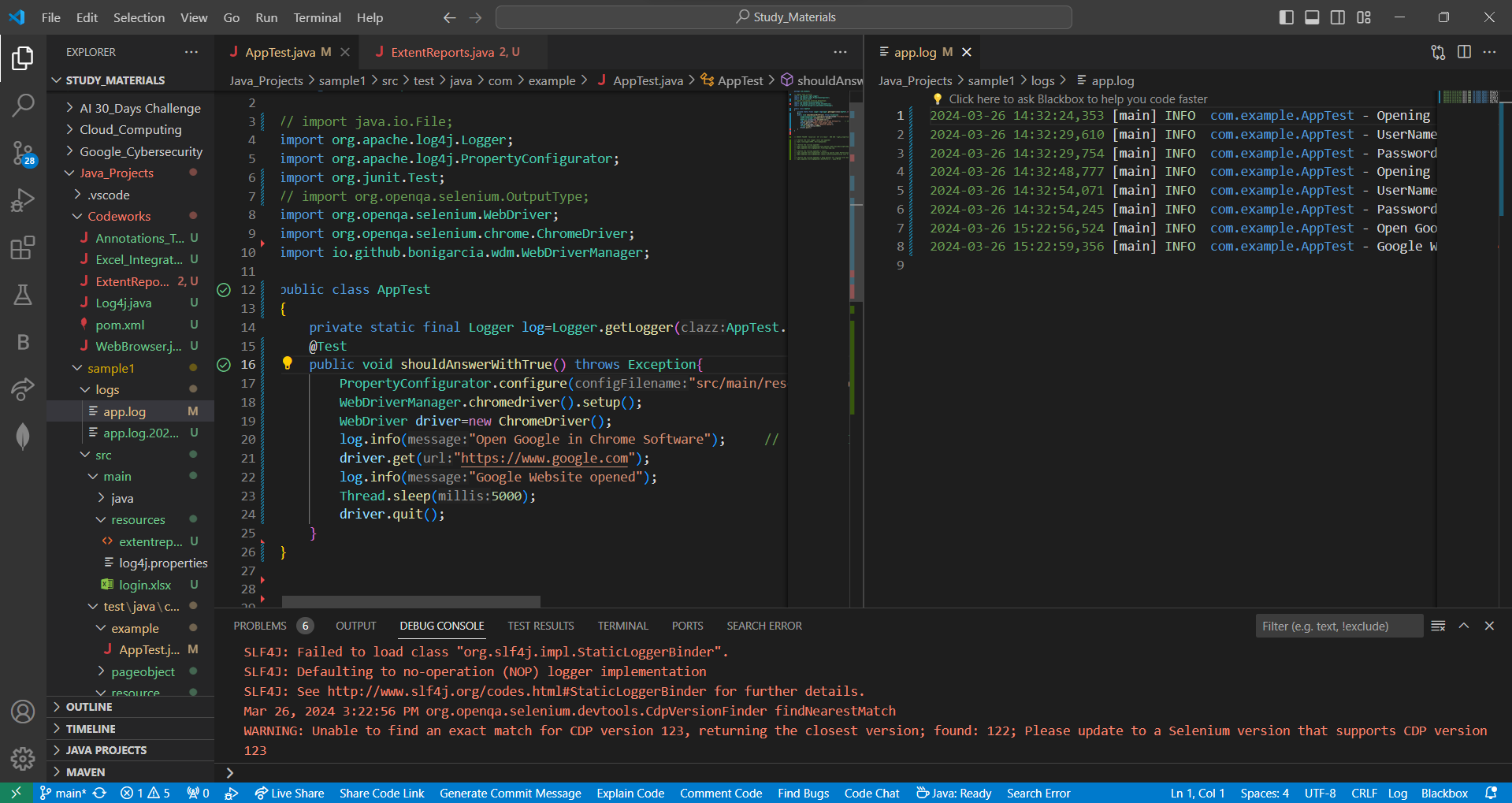
***Steps:***

* Introduction to Extent Reports and its significance in Selenium test reporting.
* Setup and configuration steps for integrating Extent Reports into Selenium projects.
* Generating test reports using Extent Reports, including creating test instances, logging test steps, and capturing screenshots.
* Customizing test reports with Extent Reports, such as adding metadata, categorizing tests, and including additional information.
* Integration of Extent Reports with Selenium test automation frameworks and best practices for effective reporting.

***References :***

* ExtentReports Documentation: https://extentreports.com/docs/versions/4/java/
* Selenium Documentation: https://www.selenium.dev/documentation/en/
* Stack Overflow: <https://stackoverflow.com/>

Day – 8 Using Log4J



***Reports :***

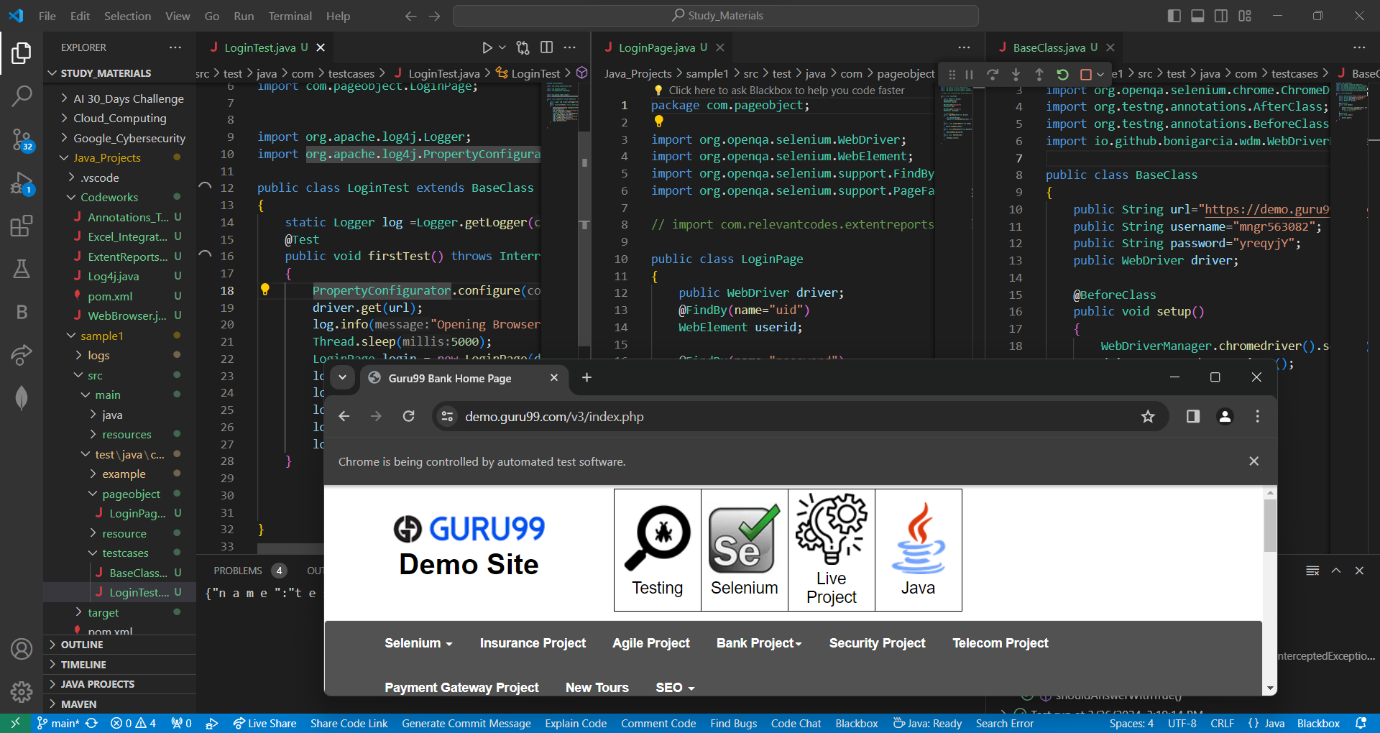
***Steps :***

* Incorporate Log4j dependency into your project's configuration, specifying the version required.
* Develop a Log4j configuration file (e.g., log4j2.xml), defining logging levels, appenders, and formatting preferences to tailor logging behavior.
* Initialize Log4j within your test suite or class, usually at the outset, setting the system property "log4j.configurationFile" .
* Employ Log4j's logging methods (e.g., debug, info, warn, error, fatal) strategically .
* Execute your Selenium tests as per standard procedure; Log4j seamlessly manages logging, adhering to the specifications delineated in your Log4j configuration file.

***References :***

* Log4j Official Website: https://logging.apache.org/log4j/2.x/
* Log4j Configuration Documentation: https://logging.apache.org/log4j/2.x/manual/configuration.html
* Log4j API Documentation: <https://logging.apache.org/log4j/2.x/log4j-api/apidocs/index.html>

***Day – 9 Using Hybrid Frameworks***

******

***Reports :***

***Steps :***

* Define framework components: Identify test scripts, reusable functions, configuration files, and test data management.
* Setup project and dependencies: Create a Java project in your IDE, manage dependencies like Selenium WebDriver and Apache POI.
* Organize framework structure: Structure the framework into packages for better organization and maintenance.
* Develop test scripts: Write test scripts using keywords and data-driven techniques for flexibility.
* Implement reporting and logging: Set up reporting mechanisms for analysis and debugging, incorporate logging for capturing debug information.

***References :***

* Selenium Official Documentation: SeleniumHQ
* Apache POI Documentation: Apache POI
* Log4j Documentation: Apache Log4j 2 Documentation