**EclatEngAssignment Test Automation Framework**

**1. Overview**

EclatEngAssignment is a **Cucumber BDD-based** automation framework designed to test the **MyLoft application**. It uses **Selenium WebDriver**, **Extent Reports**, and **ExcelReader** for test execution, reporting, and data-driven testing.

**2. Project Structure**

EclatEngAssignment

│── src

│ ├── main

│ │ ├── java

│ │ │ ├── config

│ │ │ │ ├── BaseObject.java

│ │ │ │ ├── Setup.java

│ │ │ │ ├── propertiesUtils.java

│ │ │ ├── utils

│ │ │ │ ├── ExcelReader.java

│ │ │ │ ├── ExtentReport.java

│ ├── test

│ │ ├── java

│ │ │ ├── config

│ │ │ │ ├── PageObjectManager.java

│ │ │ ├── feature

│ │ │ ├── pages

│ │ │ │ ├── myLoft.java

│ │ │ ├── stepDefinition

│ │ │ │ ├── Hooks.java

│ │ │ │ ├── myLoftStepDefinition.java

│ │ │ ├── Testrunner

│ │ │ │ ├── TestRunnerMyLoft.java

│── src/main/resources

│── AutomationReports

│ ├── extent-reports.html

**3. Key Components**

**🔹 Configuration (config/)**

* **BaseObject.java** – Provides a base class for other components like clickElement, WaitTillElementVisibility
* **Setup.java** – Handles WebDriver initialization, properties loading, and quitting the driver.
  + getDriver(): Returns WebDriver instance.
  + initializeDriver(): Initializes WebDriver.
  + quitDriver(): Closes the driver after execution.
* **propertiesUtils.java** – Manages properties file interactions.

**🔹 Utilities (utils/)**

* **ExcelReader.java** – Reads test data from Excel.
  + getData(int row, int col): Fetches data from Excel.
* **ExtentReport.java** – Generates test execution reports.
  + setupReport(): Initializes Extent Reports.
  + logStep(String, boolean): Logs test steps.
  + flushReport(): Finalizes and writes reports.

**🔹 Test Execution (test/java/)**

* **PageObjectManager.java** – Manages page objects and provides a single access point.
* **Feature Files (feature/)** – Contains Cucumber feature files for BDD scenarios.
* **Page Classes (pages/)**
  + **myLoft.java** – Implements page-specific actions with **POM**
* **Step Definitions (stepDefinition/)**
  + **Hooks.java** – Manages setup and teardown for test scenarios.
  + **myLoftStepDefinition.java** – Implements Cucumber step definitions.
    - user\_able\_to\_click\_login(): Clicks login button.
    - user\_enters\_valid\_email\_and\_password(String, String): Logs in with credentials.
* **Test Runner (Testrunner/)**
  + **TestRunnerMyLoft.java** – Executes tests with Cucumber options.

**🔹 Reports (AutomationReports/)**

* **Extent-Reports.html** – Stores execution results with test logs and screenshots.

**4. How to Run the Tests**

**🔸 Pre-requisites**

* Java **8 or higher**
* Maven **installed**
* ChromeDriver **configured as am Automatically download the latest ChromeDriver and use it .**

**🔸 Running Tests via Maven**

Execute the following command in the terminal:

**🔸 Running Tests via IDE**

* Open **TestRunnerMyLoft.java**
* Click **Run by TestNg**

**🔸 Generating Reports**

* Reports are automatically generated in:

AutomationReports/extent-reports.html

* Open the file in a browser to view results.

**5. Technologies Used**

* **Java** – Programming Language
* **Selenium WebDriver** – UI Automation
* **Cucumber BDD** – Behavior-Driven Testing
* **Extent Reports** – Test Reporting
* **Apache POI** – Excel Data Handling

**📌 Notes**

* All test data should be placed inside **Excel sheets** under resources/.
* Reports should be checked after every test run for failures.