Automation Framework - Detailed Explanation

1️⃣ Project Structure (Maven-based Framework)

📌 Folder Structure:

```
AutomationFramework/
├── src/main/java/
│   ├── pages/              # Page Object Model (POM) classes
│   ├── utils/              # Utility functions (e.g., WebDriver setup)
│
├── src/test/java/
│   ├── testcases/          # Test classes (TestNG/Cucumber tests)
│   ├── stepdefinitions/    # Cucumber step definitions
│   ├── testrunners/        # Cucumber test runners
│
├── src/test/resources/
│   ├── features/           # Cucumber feature files
│   ├── testng.xml          # TestNG configuration file
│
├── pom.xml                 # Maven dependencies
├── extent-config.xml       # Extent Report Configuration
├── README.md
```

2️⃣ **Page Object Model (POM)**

📌 **POM helps in maintaining test scripts by separating locators from test cases.**

✅ **Example:** LoginPage.java

```
package pages;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class LoginPage {
    WebDriver driver;

    @FindBy(id="username")
    WebElement username;

    @FindBy(id="password")
    WebElement password;

    @FindBy(id="login")
    WebElement loginButton;

    public LoginPage(WebDriver driver) {
        this.driver = driver;
        PageFactory.initElements(driver, this);
    }

    public void login(String user, String pass) {
        username.sendKeys(user);
        password.sendKeys(pass);
        loginButton.click();
    }
}
```

3️⃣ TestNG for Test Execution

📌 **TestNG is used to execute test cases in parallel, generate reports, and handle test dependencies.**

✅ **Example:** LoginTest.java

```
package testcases;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.Assert;
import org.testng.annotations.*;

import pages.LoginPage;

public class LoginTest {
    WebDriver driver;
    LoginPage loginPage;

    @BeforeClass
    public void setup() {
        driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://example.com");
        loginPage = new LoginPage(driver);
    }

    @Test
    public void testLogin() {
        loginPage.login("admin", "password123");
        Assert.assertEquals(driver.getTitle(), "Dashboard");
    }

    @AfterClass
    public void teardown() {
        driver.quit();
    }
}
```

4️⃣ **Cucumber for BDD**

📌 **Cucumber is used to write test scenarios in Gherkin syntax.**

✅ **Feature File (Login.feature)**

```
Feature: Login Functionality
  Scenario: User logs in with valid credentials
    Given User is on the login page
    When User enters "admin" and "password123"
    And Clicks on the login button
    Then User should be redirected to the dashboard
```

✅ **Step Definitions (LoginSteps.java)**

```
package stepdefinitions;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import pages.LoginPage;
import io.cucumber.java.en.*;

public class LoginSteps {
    WebDriver driver;
    LoginPage loginPage;

    @Given("User is on the login page")
    public void user_is_on_login_page() {
        driver = new ChromeDriver();
        driver.get("https://example.com");
        loginPage = new LoginPage(driver);
    }

    @When("User enters {string} and {string}")
    public void user_enters_username_and_password(String user, String pass) {
        loginPage.login(user, pass);
    }
```

```
    @Then("User should be redirected to the dashboard")
    public void user_should_be_redirected() {
        System.out.println("Dashboard Loaded");
        driver.quit();
    }
}
```

✅ **Runner File (TestRunner.java)**

```
package testrunners;
import io.cucumber.testng.AbstractTestNGCucumberTests;
import io.cucumber.testng.CucumberOptions;

@CucumberOptions(
    features = "src/test/resources/features",
    glue = "stepdefinitions",
    plugin = {"pretty", "html:target/cucumber-reports.html"}
)
public class TestRunner extends AbstractTestNGCucumberTests { }
```

5️⃣ Maven (Dependency Management - pom.xml)

📌 Maven helps manage project dependencies and build automation.

✅ pom.xml

```
<dependencies>
    <!-- Selenium -->
    <dependency>
        <groupId>org.seleniumhq.selenium</groupId>
        <artifactId>selenium-java</artifactId>
        <version>4.10.0</version>
    </dependency>

    <!-- TestNG -->
    <dependency>
        <groupId>org.testng</groupId>
        <artifactId>testng</artifactId>
        <version>7.4.0</version>
    </dependency>

    <!-- Cucumber -->
    <dependency>
        <groupId>io.cucumber</groupId>
        <artifactId>cucumber-java</artifactId>
        <version>7.2.0</version>
    </dependency>

    <dependency>
        <groupId>io.cucumber</groupId>
        <artifactId>cucumber-testng</artifactId>
        <version>7.2.0</version>
    </dependency>
</dependencies>
```

6️⃣ Reporting (TestNG, Extent Reports, Cucumber Reports)

📌 TestNG generates default reports, but for better visualization, we use Extent Reports.

✅ Example: Adding Extent Report (TestListener.java)

```
package utils;
import com.aventstack.extentreports.*;
import com.aventstack.extentreports.reporter.ExtentHtmlReporter;
import org.testng.*;

public class TestListener implements ITestListener {
    private static ExtentReports extent = new ExtentReports();
    private static ExtentHtmlReporter htmlReporter = new
ExtentHtmlReporter("test-output/ExtentReport.html");
    private static ExtentTest test;

    static {
        extent.attachReporter(htmlReporter);
    }

    @Override
    public void onTestStart(ITestResult result) {
        test = extent.createTest(result.getMethod().getMethodName());
    }

    @Override
    public void onTestSuccess(ITestResult result) {
        test.log(Status.PASS, "Test Passed");
    }

    @Override
    public void onTestFailure(ITestResult result) {
        test.log(Status.FAIL, "Test Failed: " + result.getThrowable());
    }

    @Override
```

```
    public void onFinish(ITestContext context) {
        extent.flush();
    }
}
```

7️⃣ testng.xml - Running Tests

📌 TestNG allows test execution control.

✅ Example:

```
<suite name="Test Suite">
    <test name="Login Tests">
        <classes>
            <class name="testcases.LoginTest"/>
        </classes>
    </test>
</suite>
```

Happy Learning! 🌸