**Project Setup & Execution Report**

***Automation Framework for Incode Test Task by Djordje Mihailovic***

**Overview**

This report provides comprehensive instructions for setting up and executing the test automation framework developed for this test task. The framework is built using Java, Selenium WebDriver, and Cucumber with JUnit, following standard industry practices for web application testing.

**System Requirements**

**Required Software**

* **Java Development Kit (JDK) 21**
* **Maven 3.8+**
* **Chrome** (latest versions recommended)
* **Git** (optional, for version control)
* **IntelliJ IDEA** ( My IDE of choice for Java projects, other IDEs could be used )

**Installation & Setup ( for Intellij )**

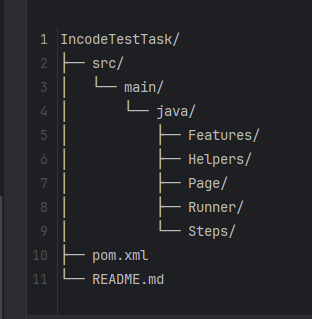
After the installation of required software, open your IDE and run the following commands:

1. **Open IntelliJ IDEA**
2. **Clone the repository:**
   * Select File > New > Project from Version Control (or click Get from VCS on welcome screen)
   * Paste the URL: ***https://github.com/DjordjeMihailovic/IncodeAssignment.git***
   * Choose a directory location on your computer
   * Click Clone
3. **Wait for the project to load:**
   * IntelliJ will index files and configure the project
   * If prompted about Maven/Gradle build files, choose "Trust Project"
4. **Verify project setup:**
   * Check if all dependencies are resolved
   * If you see red underlines in code or dependency errors, proceed to rebuilding with Maven by running in the command line:

*mvn clean install*

This will let the Maven build the project and handle all dependencies, after it is done you are all set on importing this project and ready to go!

**Project Structure Overview**

* + src/main/java: Contains all test code  
      
    
    - Features: Cucumber feature files with test scenarios
    - Helpers: Utility classes
    - Page: Page Object Model classes
    - Runner: Cucumber test runner configuration
    - Steps: Step definitions for Cucumber

**Configuration**

* + Config file is located in /data; it holds the information about browser and environment that is used when running tests locally

**Running Tests**

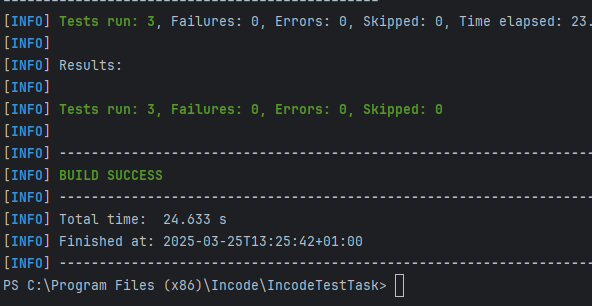
**Command Line Execution**

Now that all is set you can run tests in a few different ways;

Run the tests with the tag I created for this assignment with Maven from command line:

*mvn test "-Dcucumber.filter.tags=@AutomationAssignment"*

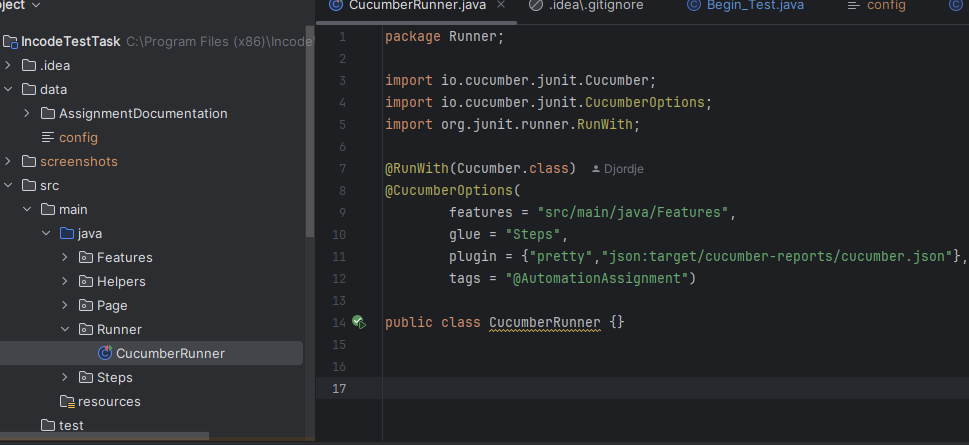
This tells Maven to execute tests that are tagged with @*AutomationAssignment*



**IDE Execution**

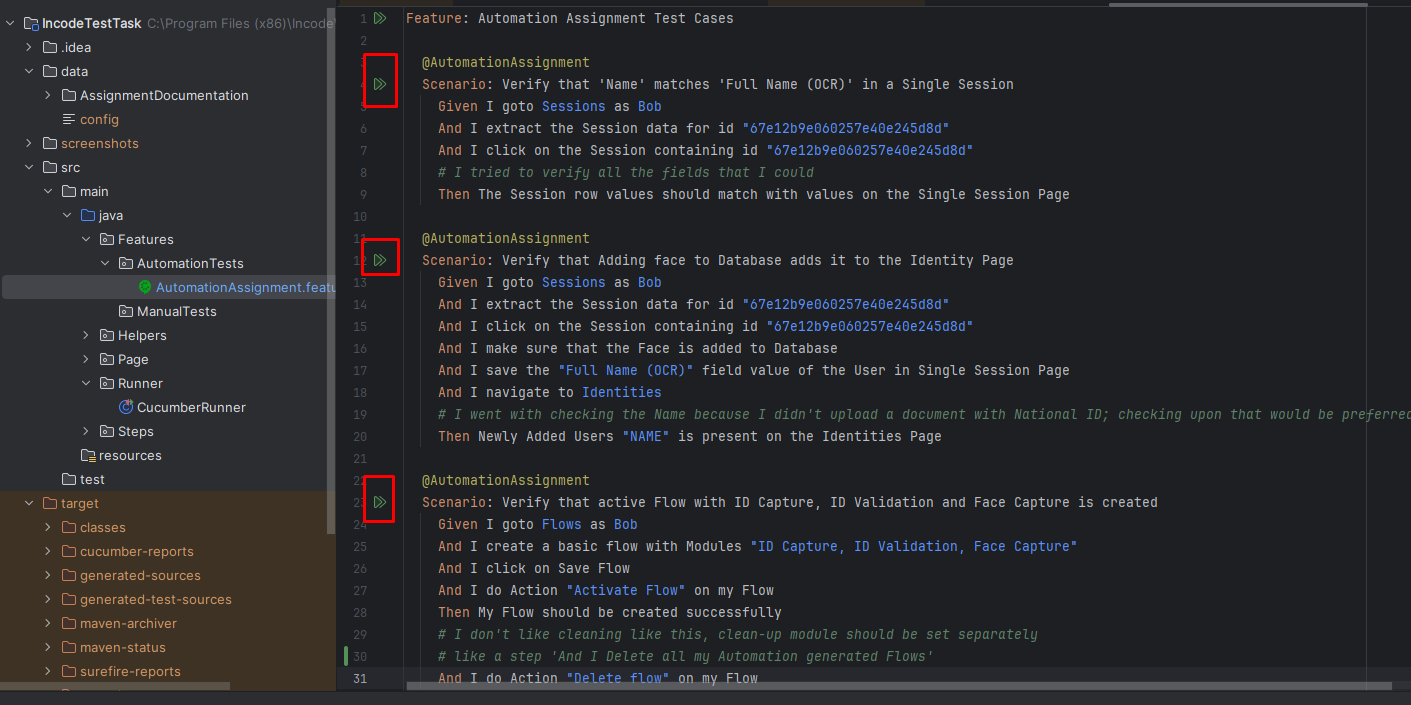
**via CucumberRunner**

1. Navigate to src/main/java/Runner/CucumberRunner.java
2. Right-click and select "Run CucumberRunner"
3. This will run all the tests set in accordance with this runner



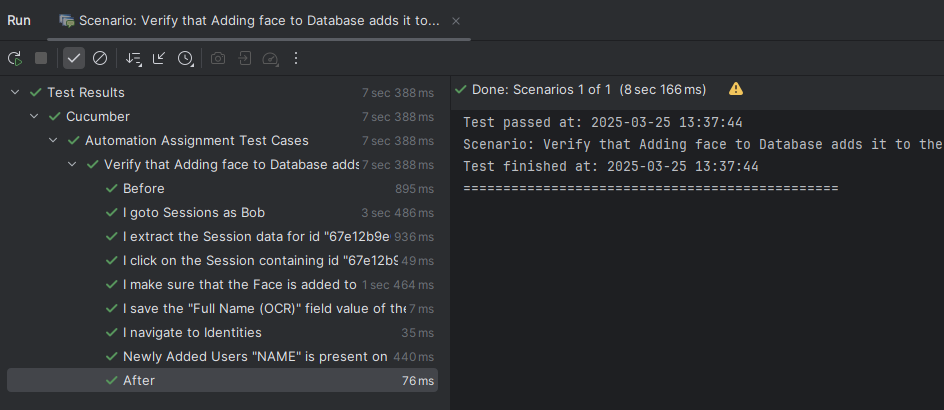
**Individually, from .feature file**

1. Navigate to src/main/java/Features/AutomationTests
2. Open a .feature file
3. Click on Play button to the left of the Scenario that you want to execute



**Example Test Execution**

When the test(s) begin to execute, in the Run window( usually on) the bottow of your IDE you will see a detailed overview of the steps and the test itself; If they passed or failed, time for execution, messages that are triggered by them etc;



**Reports & Logs**

* **Cucumber Reports**: Generated in target/cucumber-reports after test execution
* **Screenshots**: Captured automatically on test failure
* **Logs**: Standard output and error logs available during execution