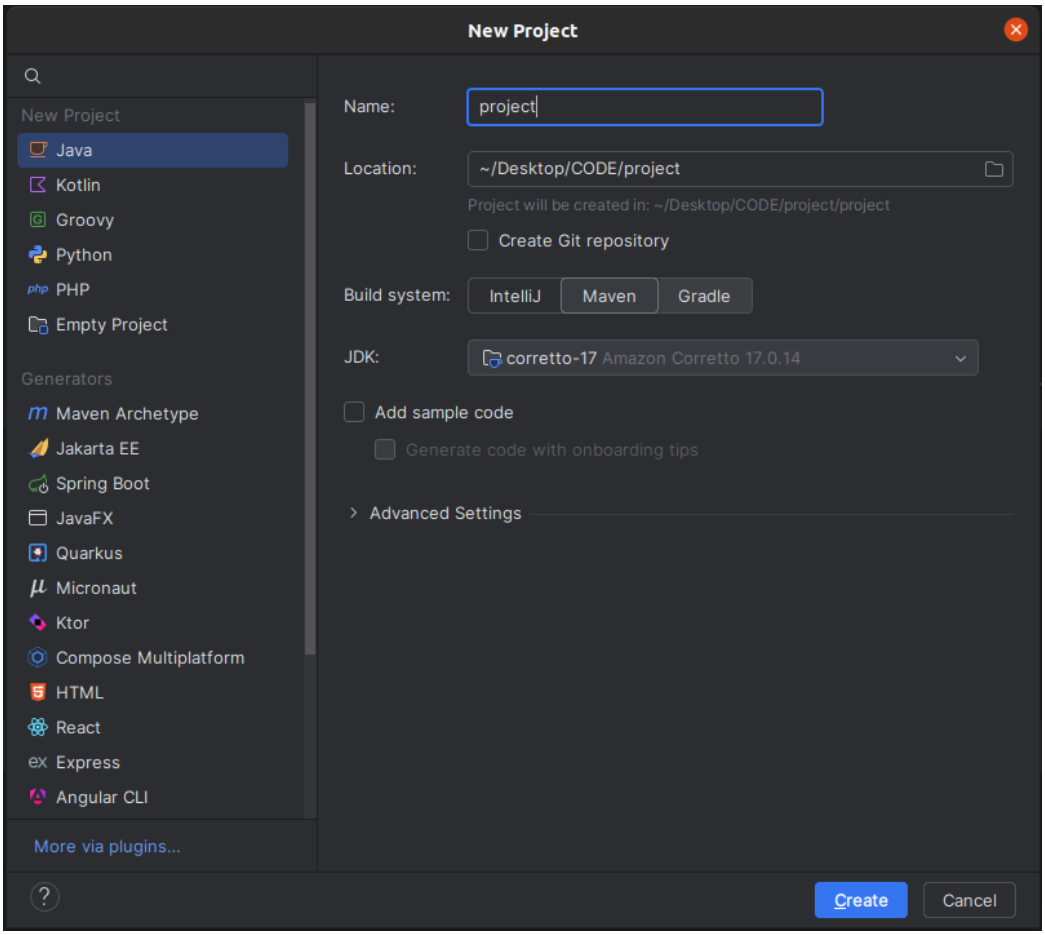
Tutorial: Setting Up a Maven Project (IntelliJ IDEA)

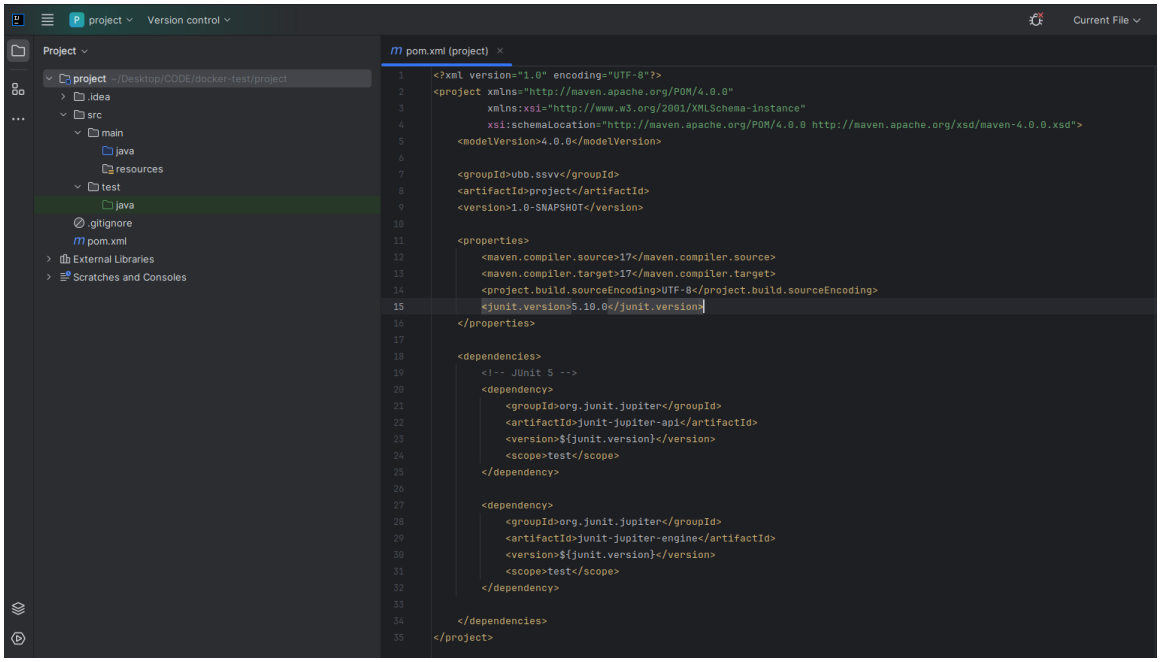
Step 1: Creating a New Maven Project



1. Open IntelliJ IDEA.
2. Navigate to File > New > Project.
3. In the 'New Project' window:
   - Select 'Java'.
   - Set 'Name' to: project.
   - Choose 'Maven' as the Build System (Maven is a project management tool that automates the build process).
   - Select JDK 17 (JDK is required for compiling Java code. You may need to install it from IntelliJ).
4. Click 'Next' and then 'Finish' to create the project.

⚠ Ensure JDK 17 is used for consistency in future labs.

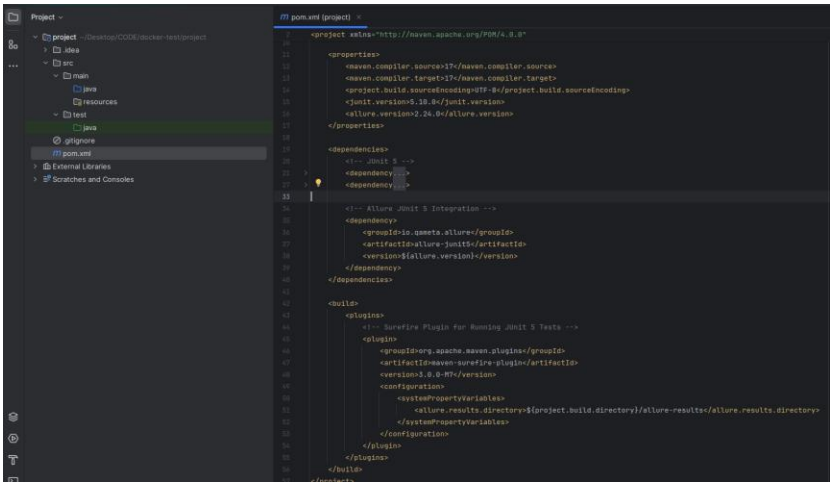
Step 2: Adding JUnit 5 Dependencies



JUnit 5 is a testing framework for Java applications.
To add JUnit 5 to the project:
1. Open the pom.xml file (Maven's configuration file).
2. Add the following dependencies:

```
<properties>
  <junit.version>5.10.0</junit.version>
</properties>

<dependencies>
  <!-- JUnit 5 -->
  <dependency>
    <groupId>org.junit.jupiter</groupId>
    <artifactId>junit-jupiter-api</artifactId>
    <version>${junit.version}</version>
    <scope>test</scope>
  </dependency>
  <dependency>
    <groupId>org.junit.jupiter</groupId>
    <artifactId>junit-jupiter-engine</artifactId>
    <version>${junit.version}</version>
    <scope>test</scope>
  </dependency>
</dependencies>
```

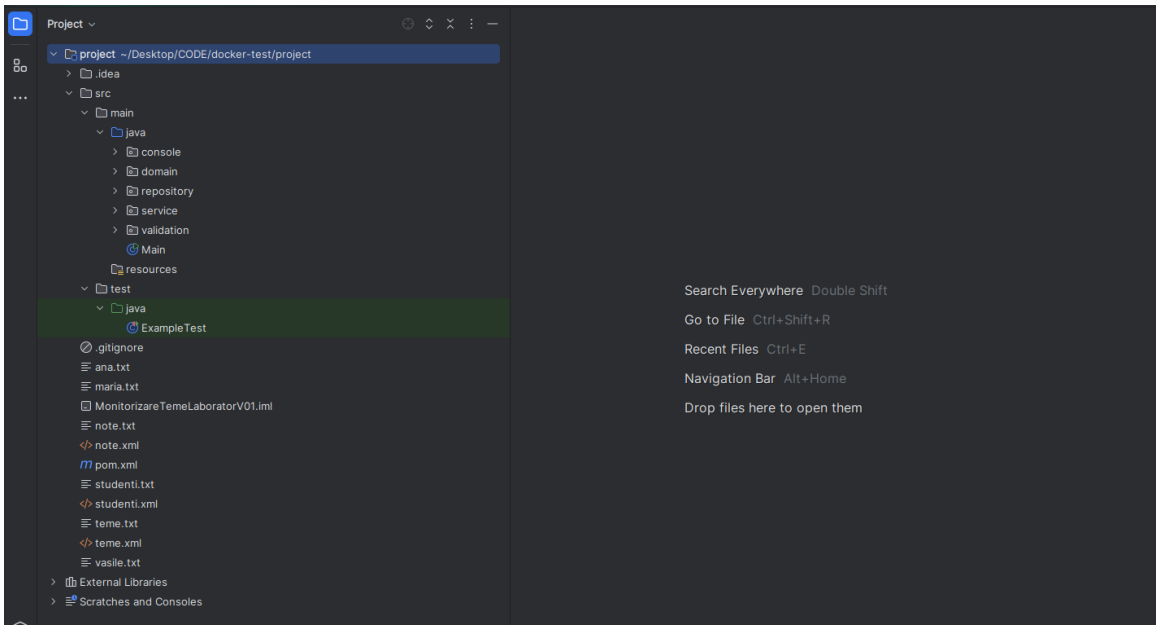
Step 3: Adding Allure Dependencies



Allure is a reporting tool for test execution results.
To integrate Allure with JUnit 5:
1. Open the pom.xml file.
2. Add the following configuration:

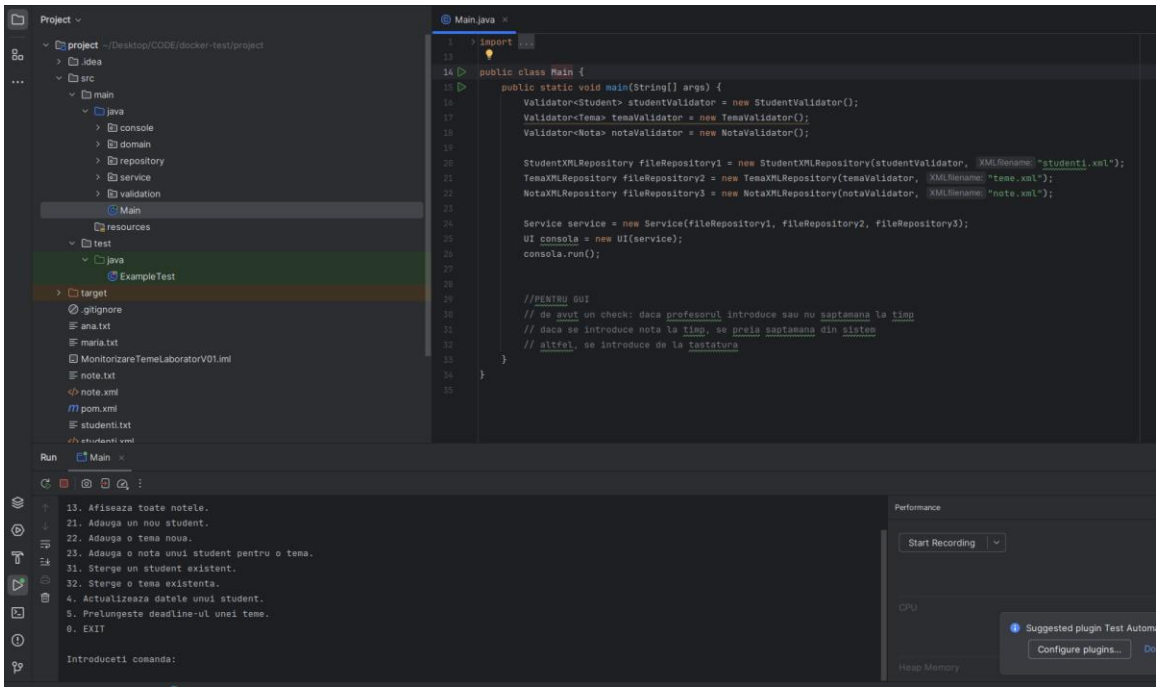
```
<properties>
  <allure.version>2.24.0</allure.version>
</properties>

<dependencies>
  <!-- Allure JUnit 5 Integration -->
  <dependency>
    <groupId>io.qameta.allure</groupId>
    <artifactId>allure-junit5</artifactId>
    <version>${allure.version}</version>
  </dependency>
</dependencies>

<build>
  <plugins>
    <!-- Surefire Plugin for Running JUnit 5 Tests -->
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-surefire-plugin</artifactId>
      <version>3.0.0-M7</version>
      <configuration>
        <systemPropertyVariables>
          <allure.results.directory>${project.build.directory}/allure-
results</allure.results.directory>
        </systemPropertyVariables>
      </configuration>
    </plugin>
  </plugins>
</build>
```

Step 4: Organizing the Maven Project Structure



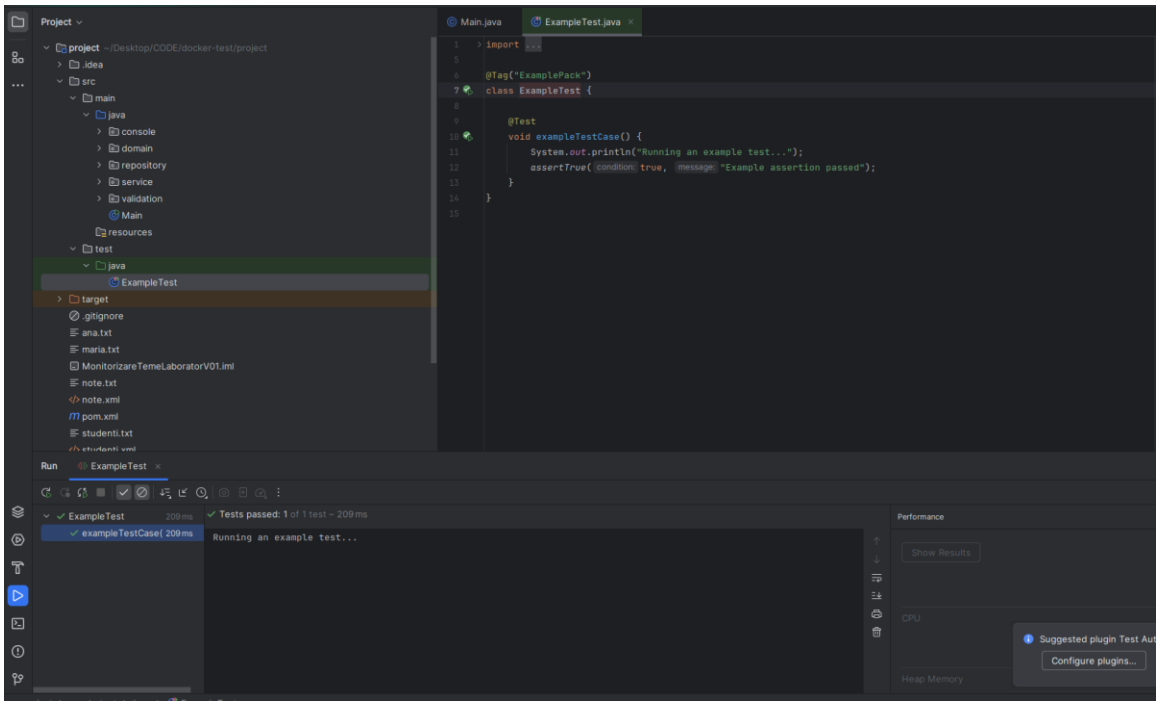
Maven follows a structured project layout:
- Place business logic classes under src/main/java.
- Place test classes under src/test/java.
- Store static files (e.g., XML configurations) in the root of the project.
- Include a .gitignore file in the project root.

Step 5: Running the Main Class



To validate that the project starts successfully:
1. Open src/main/java/Main.java.
2. Run the main method from IntelliJ to confirm the project setup.

Step 6: Running Example Test



To verify that JUnit 5 tests run correctly:
1. Open src/test/java/ExampleTest.java.
2. Execute t9he test from IntelliJ or using Maven with `mvn test`.