**Experiment 2: Working with Maven: Creating a Maven Project, Understanding the POM File, Dependency Management and Plugins**

**Step 1**: Open Your Terminal

**Step 2**: Use Maven Archetype to Generate a New Project

Maven comes with a set of archetypes that provide you with a standard project template. Use the following command to create a new Maven project:

mvn archetype:generate -DgroupId=com.example -DartifactId=MyMavenApp -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false

• **groupId**: Uniquely identifies your project’s group (like a package name).

• **artifactId**: The name of your project (the resulting artifact).

• **maven-archetype-quickstart**: A simple archetype that sets up a basic Java project with

a sample unit test.

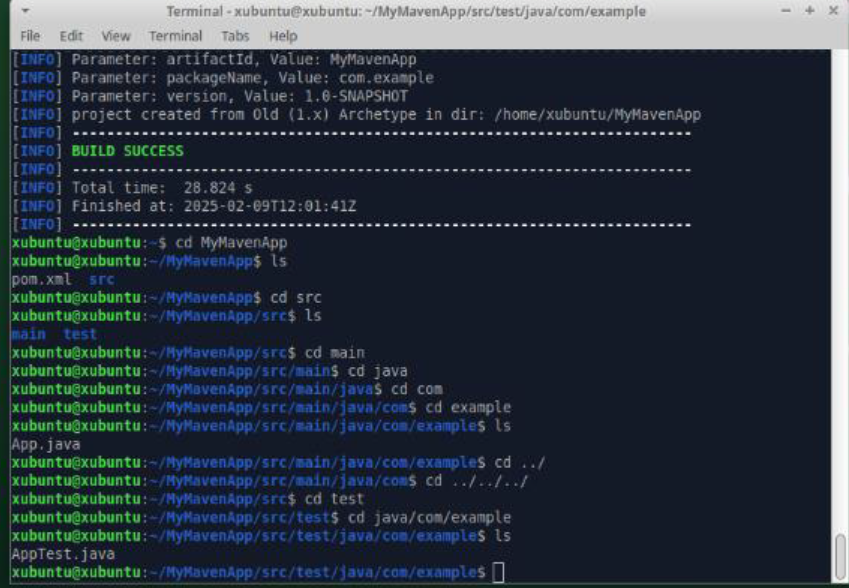
• **-DinteractiveMode=false**: Runs the command in non-interactive mode, using the

provided parameters.

**Step 3: Navigate to Your Project Directory**

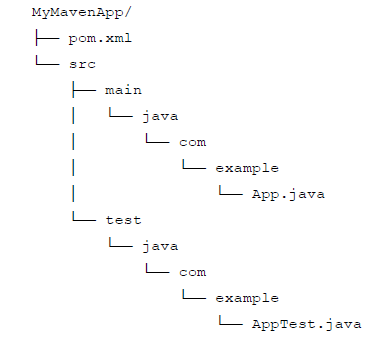
Once the command completes successfully, change your directory to the newly created project:

cd MyMavenApp



**3. Maven Project Layout and Components**

After generating the project, you will notice the following standard Maven directory structure:



**Explanation of Key Components**

• **pom.xml:**

The **Project Object Model (POM)** file is the core of any Maven project. It contains

configuration details such as project coordinates (groupId, artifactId, version),

dependencies, plugins, and build settings.

• **src/main/java:**

This directory holds the source code of your application. In our example, the package

structure com.example is created, and you have an App.java file.

• **src/test/java:**

This directory is for your test cases. The default example includes a basic test class,

AppTest.java.

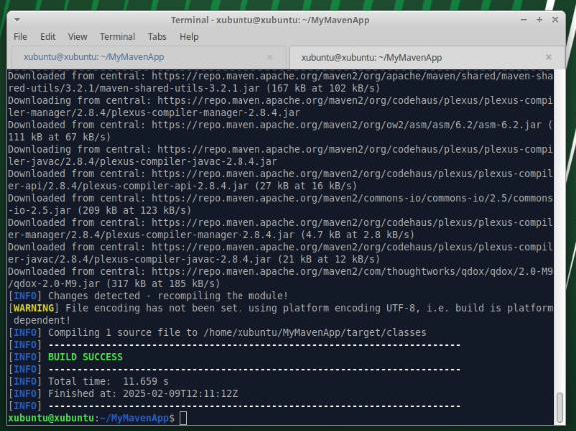
**4. Understanding the POM File (pom.xml)**

<project xmlns="[http://maven.apache.org/POM/4.0.0](http://maven.apache.org/POM/4.0.0" \t "_blank)"  xmlns:xsi="[http://www.w3.org/2001/XMLSchema-instance](http://www.w3.org/2001/XMLSchema-instance" \t "_blank)" xsi:schemaLocation="[http://maven.apache.org/POM/4.0.0](http://maven.apache.org/POM/4.0.0" \t "_blank)  [http://maven.apache.org/xsd/maven-4.0.0.xsd](http://maven.apache.org/xsd/maven-4.0.0.xsd" \t "_blank)"> <modelVersion>4.0.0</modelVersion>    
 <groupId>com.example</groupId>  
 <artifactId>MyMavenApp</artifactId>  
 <version>1.0-SNAPSHOT</version>   
 <properties>   
 <maven.compiler.source>11</maven.compiler.source> <maven.compiler.target>11</maven.compiler.target>  
 </properties>  
 <dependencies>  
 <dependency>  
 <groupId>junit</groupId>  
 <artifactId>junit</artifactId>  
 <version>4.13.2</version>  
 <scope>test</scope>  
</dependency>  
</dependencies>  
<build>  
<plugins>  
<plugin>  
 <groupId>org.apache.maven.plugins</groupId>  
 <artifactId>maven-compiler-plugin</artifactId>  
 <version>3.8.1</version>  
 <configuration>  
 <source>11</source> <target>11</target> </configuration>  
</plugin>  
<plugin>  
 <groupId>org.apache.maven.plugins</groupId>  
 <artifactId>maven-surefire-plugin</artifactId>  
 <version>2.22.2</version>  
</plugin>  
</plugins>  
</build>  
</project>

**Common Maven Commands**

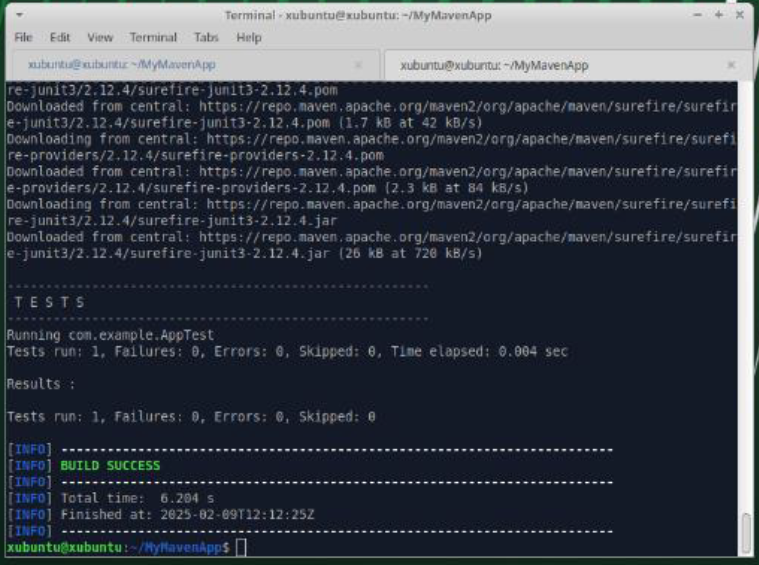
• **Compile the Project:**

mvn compile



• **Run Unit Tests:**

mvn test



• **Package the Application:**

mvn package

This command compiles, tests, and packages your code into a JAR file located in the

target directory.

• **Clean the Project:**

mvn clean

This removes any files generated by previous builds.