**Experiment 4: Practical Exercise: Build and Run a Java Application with Maven, Migrate the Same Application to Gradle**

**Part A: Build and Run a Java Application with Maven**

**1. Create the Maven Project**

1. Open your Terminal.

2. Generate the Maven Project using the Quickstart Archetype:

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

The mvn archetype:generate command initializes a new Maven project based on a specified archetype, providing a structured template for development. By defining parameters like groupId, artifactId, and archetypeArtifactId, you can customize the project's base configuration to suit your needs.

**( PASTE SCREENSHOT ,** Capture the terminal output showing the successful project generation.**)**

3. Change Directory into the Newly Created Project:

Command : cd my-app

The command cd my-app changes the current directory to my-app, allowing you to work within that project folder.

4. Update POM file by adding plugins to it

The pom.xml file configures a Maven project with a dependency on JUnit version 3.8.1, designated for testing purposes. It also includes the Maven Jar Plugin (version 3.2.0) to package the project into a JAR file, specifying com.example.App as the main class in the manifest.

5. Package the application :

Command : mvn package

**( PASTE SCREENSHOT, showing output of “Build success” )**

6. Run the maven project

Command : java -jar target/my-app-1.0-SNAPSHOT.jar

**( PASTE SCREENSHOT, showing “Hello World” )**

**Part B: Migrate the Application to Gradle**

**In this part, you will create a Gradle project that contains the same Java application code, then build and run it using Gradle.**

**1. Create a New Gradle Project**

1. Open a New Terminal Window or Navigate Back to Your Workspace.

2. Create a New Directory for the Gradle Project:

Command : mkdir HelloMavenGradle

3. Go to newly created directory

Command: cd HelloMavenGradle

The commands mkdir HelloMavenGradle and cd HelloMavenGradle first create a new directory named HelloMavenGradle and then navigate into it.

4. Initialize the Gradle Project Using the Java Application Type:

Command : gradle init --type java-application

The gradle init --type java-application command initializes a new Gradle project configured for a Java application. This setup includes the necessary directories and files, such as source folders and build scripts, to streamline the development process.

**Expected Output: You will see interactive prompts or a confirmation message stating that the project has been generated.**

**( PASTE SCREENSHOT, showing confirmation message )**

**2. Adjust the Gradle Project to Use the Same Code**

**1. Create a new directory in gradle root**

Command : mkdir -p src/main/java/com/example

The mkdir -p src/main/java/com/example command creates the specified directory structure, including any necessary parent directories that do not already exist. This is particularly useful in Java projects to establish the standard package hierarchy for organizing source code files

**( PASTE SCREENSHOT, showing new directory )**

2. **Copy app.java file from maven to gradle**

Note : File paths are subjective to individual systems

 Command : mv /home/student-devops/my-app/src/main/java/com/ /home/student-devops/hellomavengradle/src/main/java/com/example/

**3. Edit build.gradle file, update application part**

mainClassName = 'com.example.App'

**4. gradle build**

The gradle build command compiles the project's source code, runs tests, and packages the application into its distributable format, such as a JAR file

**5. gradle run**

The gradle run command executes the application by compiling the source code and launching the main class defined in the build

**( PASTE SCREENSHOT, showing “Hello world”)**