# **Lab Exercise 7-** **Refactoring a Java Program with GitHub Copilot and Automation using CI Feature of GitHub Actions**

**Objective:**

Learn how to:

1. **Create a Java Maven project** using **GitHub Copilot Chat**.
2. **Push the project to GitHub** using **Git commands**.
3. **Set up GitHub Actions** to **build** the project automatically.

**Prerequisites:**

* **GitHub Copilot Chat** enabled in **VS Code**.
* **Java (JDK 17 or later)** installed.
* **Maven** installed.
* **GitHub repository created** for the project.

**Step 1: Create a Java Maven Project**

**1.1 Generate Maven Project using Copilot Chat**

Ask Copilot:

/generate a Java Maven project with a basic main class

Expected Output: Copilot generates a **Maven project structure** with:

A computer screen shot of a program code

AI-generated content may be incorrect.

**1.2 Define the Main Class (App.java)**

Ask Copilot:

Add a main method to src/main/java/com/example/App.java that prints "Hello, GitHub Actions!"

Copilot modifies **App.java**:

package com.example;

public class App {

public static void main(String[] args) {

System.out.println("Hello, GitHub Actions!");

}

}

**Step 2: Build and Test the Project Locally**

**2.1 Run Maven Build**

Run the command:

mvn clean package

Expected Output: target/ directory is created with a compiled .jar file.

**2.2 Run the Application**

Run:

java -cp target/my-maven-project-1.0-SNAPSHOT.jar com.example.App

Expected Output:

Hello, GitHub Actions!

**Step 3: Push the Project to GitHub**

**3.1 Initialize Git and Commit Changes**

Run the following commands:

git init

git add .

git commit -m "Initial commit: Java Maven project"

**3.2 Link and Push to GitHub**

1. Store the repository URL in a chat variable:

#repo\_url = "https://github.com/yourusername/my-maven-project.git"

1. Push the project using Copilot:

/run git commands to add remote #repo\_url and push the code

Copilot will suggest:

git remote add origin https://github.com/yourusername/my-maven-project.git

git branch -M main

git push -u origin main

**Step 4: Set Up GitHub Actions for CI/CD**

**4.1 Create .github/workflows/maven.yml**

Ask Copilot:

Create a GitHub Actions workflow file for Maven build

Copilot generates .github/workflows/maven.yml:

name: Java Maven Build

on:

push:

branches:

- main

pull\_request:

branches:

- main

jobs:

build:

runs-on: ubuntu-latest

steps:

- name: Checkout Repository

uses: actions/checkout@v4

- name: Set Up JDK 17

uses: actions/setup-java@v3

with:

distribution: 'temurin'

java-version: '17'

cache: maven

- name: Build with Maven

run: mvn clean package

**Step 5: Push Workflow File and Verify**

1. Add workflow to Git:

git add .github/workflows/maven.yml

git commit -m "Added GitHub Actions workflow for Maven build"

git push origin master

1. Go to **GitHub → Actions** and check the build status.

**Conclusion**

* **Generated a Java Maven project** with GitHub Copilot Chat.
* **Pushed the project to GitHub** using #repo\_url.
* **Configured GitHub Actions** to **automate Maven builds**.

By following this lab, you will have a **CI/CD pipeline** for a **Java-based Maven project** with **GitHub Actions**!