**Task 1**

**Introduction:**

In software development, applications often require **different database configurations** for **development, testing, and production** environments. Manually changing configurations is inefficient and error-prone. **Maven Profiles** allow us to switch between environments dynamically during the build process.

This document describes how to set up and use **Maven profiles** to manage different database configurations effectively.

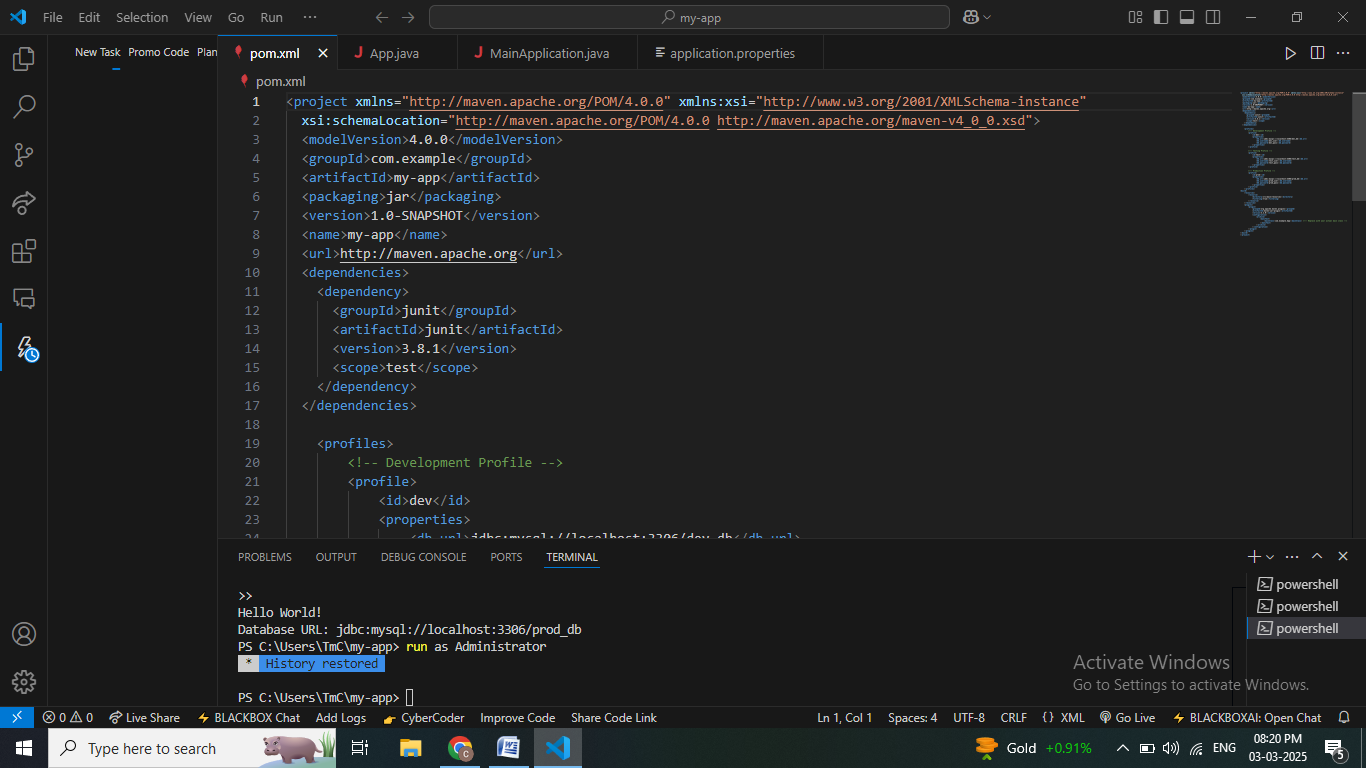
**Objective:**

The objective of this task is to:

* Configure environment-specific database settings using Maven Profiles in pom.xml.
* Use resource filtering to dynamically replace database properties in application.properties.
* Run the application with different profiles (dev, test, prod) and verify the configuration.

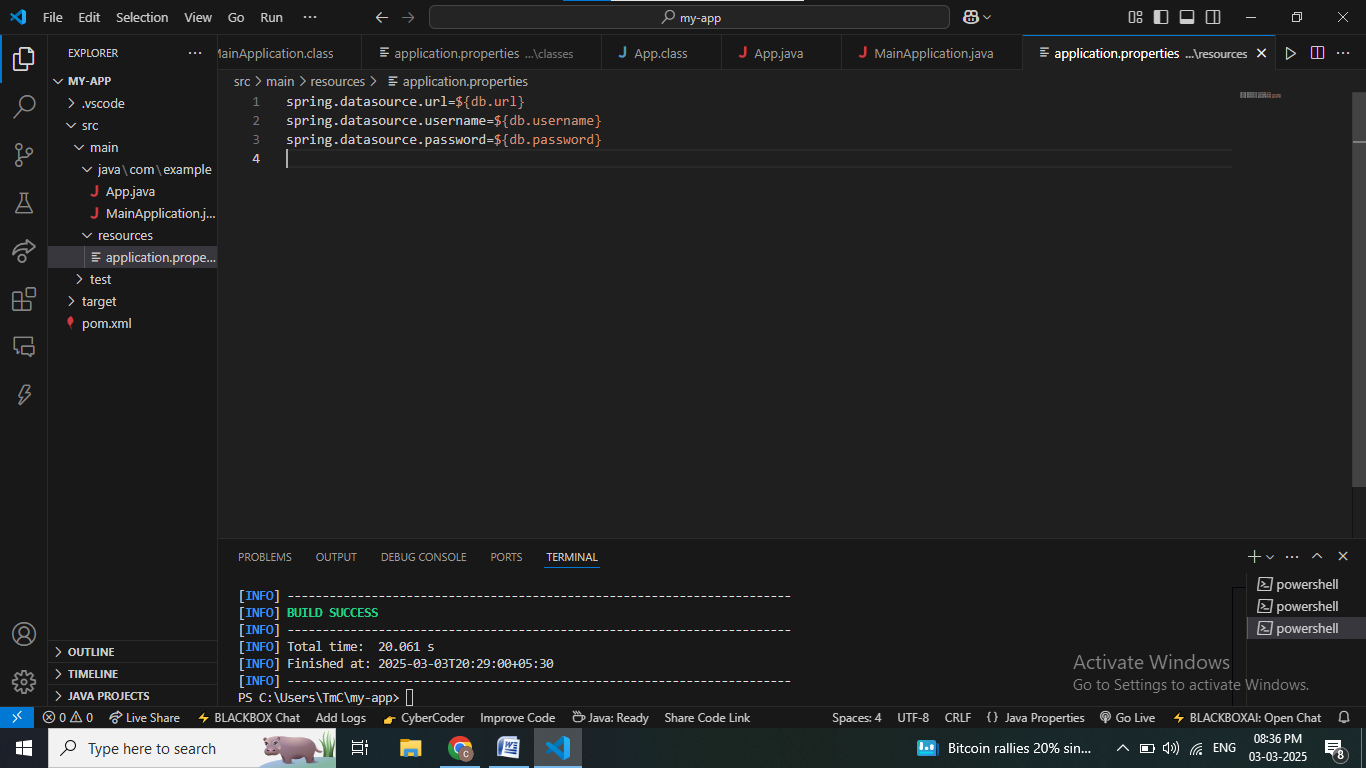
**Step 1: Define Maven Profiles in pom.xml**

* Created profiles for **development, testing, and production**.
* Each profile contains **database URL, username, and password** properties.



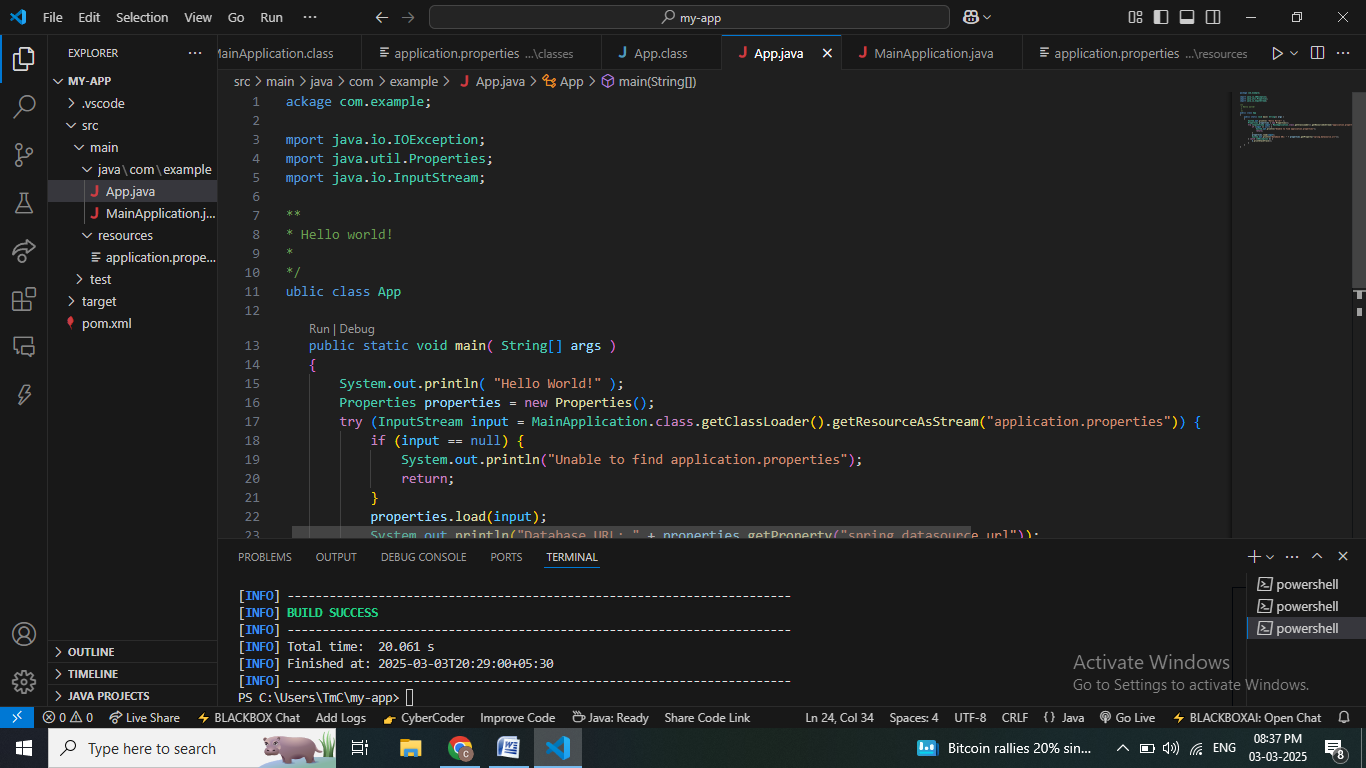
**Step 2: Create application.properties with Placeholders**

**Location:** src/main/resources/application.properties

****

**Step 3: Implement Java Code to Read Properties**

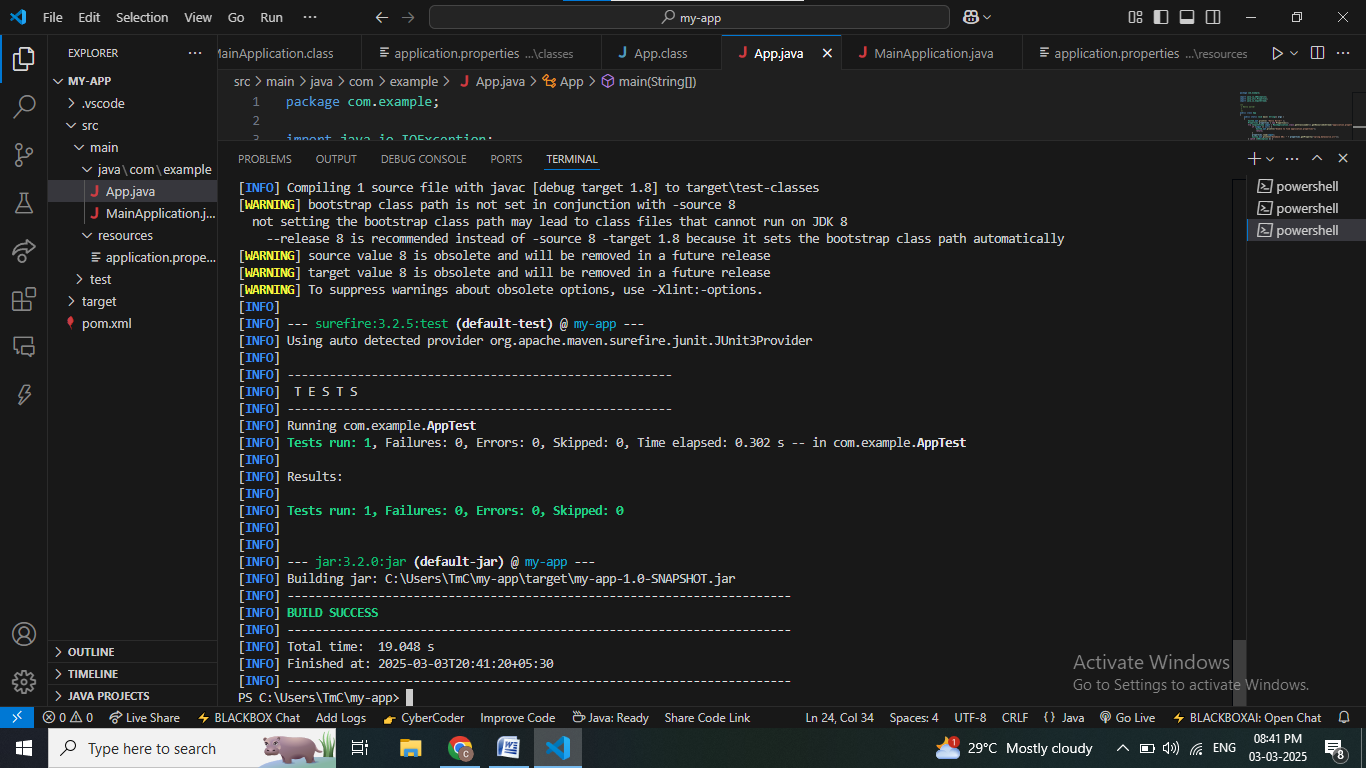
* It is used to print the hello world program
* And it used to print the database location

****

**Step 4: Build the Application with Different Profiles**

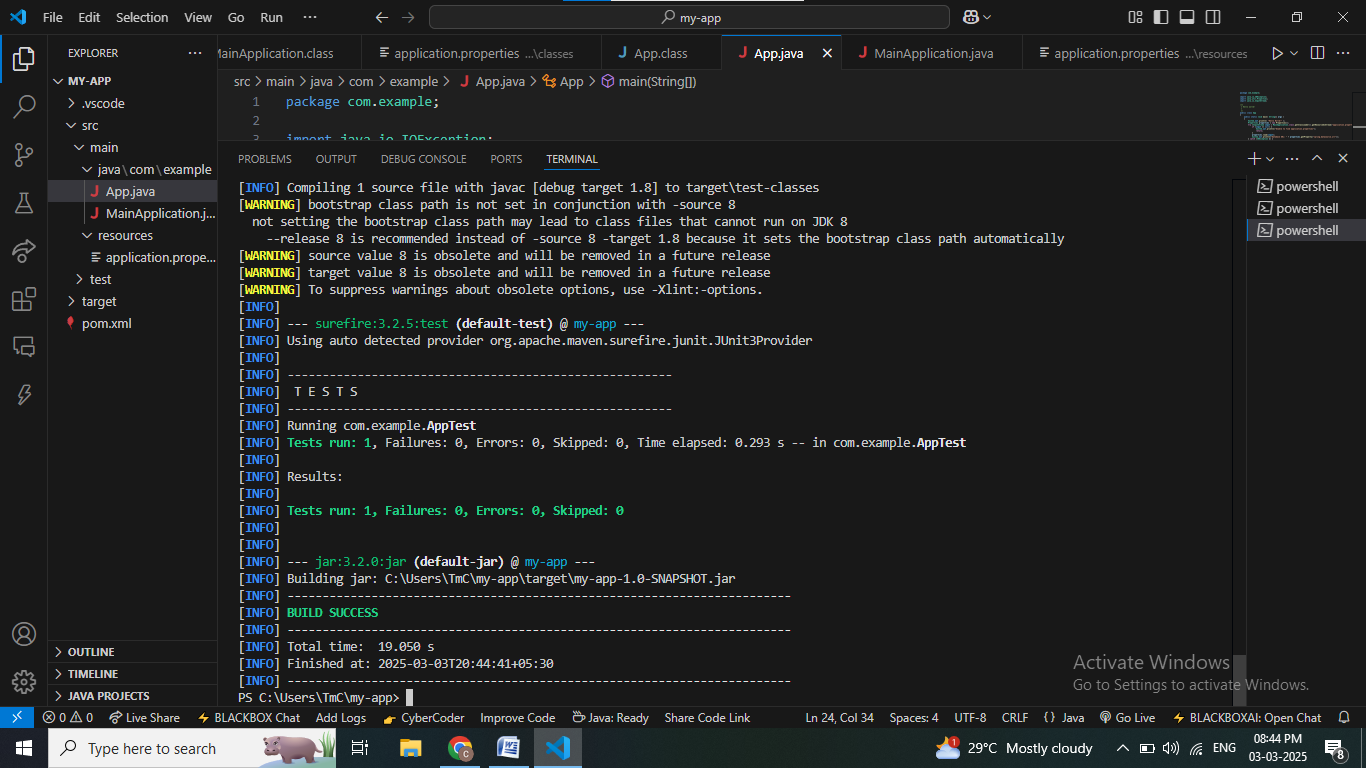
**Development Profile:**

mvn clean package –Pdev



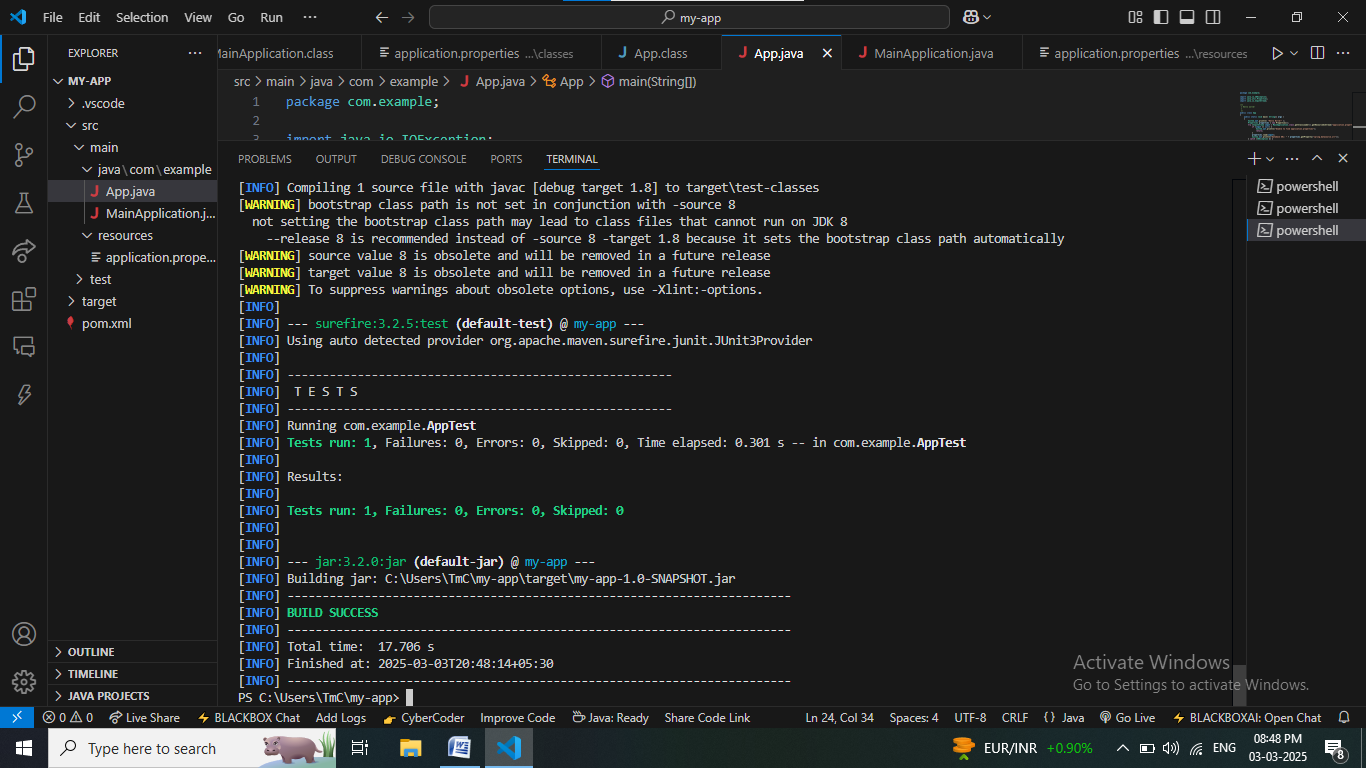
**Testing Profile:**

mvn clean package –Ptest



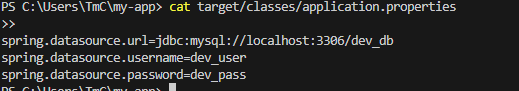
**Production Profile:**

mvn clean package –Pprod

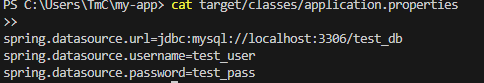


**Step 5: Verify Processed application.properties**

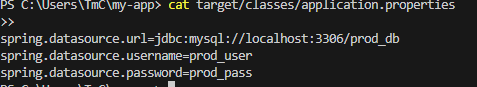
**Development:**

****

**Testing:**

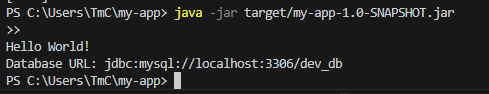
****

**Production:**

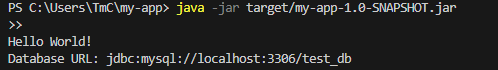
****

**Step 6: Run the JAR File and Validate Output**

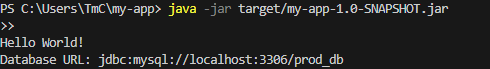
**Development:**

****

**Testing:**

****

**Production:**

****

**CONCLUSION:**

* Successfully implemented **Maven Profiles** to handle **different database configurations**.
* Verified that **application.properties was dynamically updated** during the build.
* Ensured that the application correctly **prints the configured database URL** when run.