**SEPM**

**Experiment 5**

**Aim:**

To Build the pipeline of jobs using Maven in Jenkins, create a pipeline script to Test and deploy an application over a server.

**Theory:**

**Apache Maven**

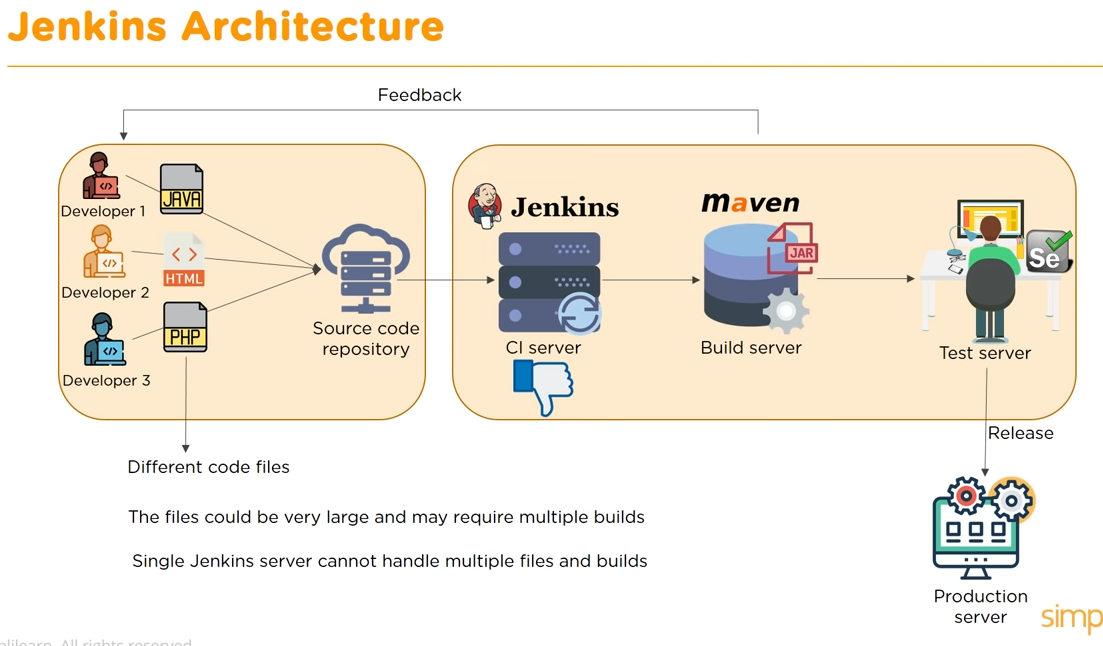
Maven is a build automation tool primarily used for Java projects. It provides a standard way to build, package, and deploy Java applications, as well as manage their dependencies.

Maven uses a project object model (POM) to manage the build process. The POM is an XML file that describes the project's configuration, dependencies, build profiles, and other details. Maven uses the POM to download dependencies, compile source code, create executable JAR files, run tests, and deploy the application to a server.

One of the key benefits of using Maven is its ability to manage dependencies automatically. Instead of manually downloading and installing each required library, Maven can fetch the dependencies from a central repository and add them to the project's classpath.

Another advantage of using Maven is its plugin architecture. Maven provides a large number of plugins for tasks such as unit testing, code analysis, and deployment. Developers can also create custom plugins to extend the functionality of Maven.

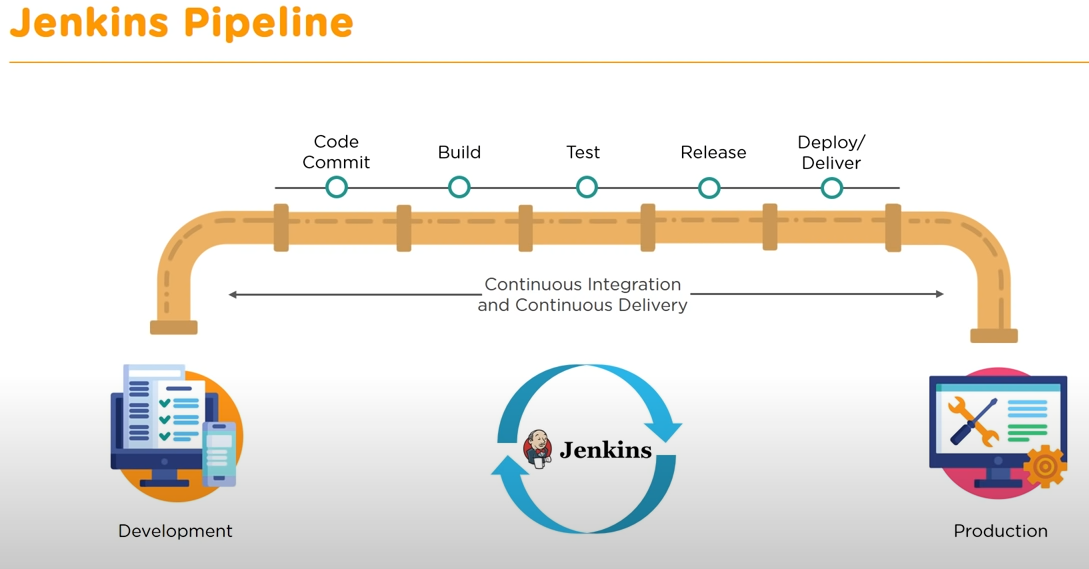
Maven is widely used in the Java community and is supported by popular integrated development environments (IDEs) such as Eclipse and IntelliJ IDEA. It is a valuable tool for managing complex Java projects and ensuring that builds are consistent and reproducible across different environments.

****

A pipeline is an important concept in software development because it allows for the automation of the entire software delivery process, from development through testing and deployment. By breaking down the software delivery process into smaller, more manageable stages, a pipeline can help teams to increase efficiency, improve quality, and reduce errors.

Here are some reasons why a pipeline is important:

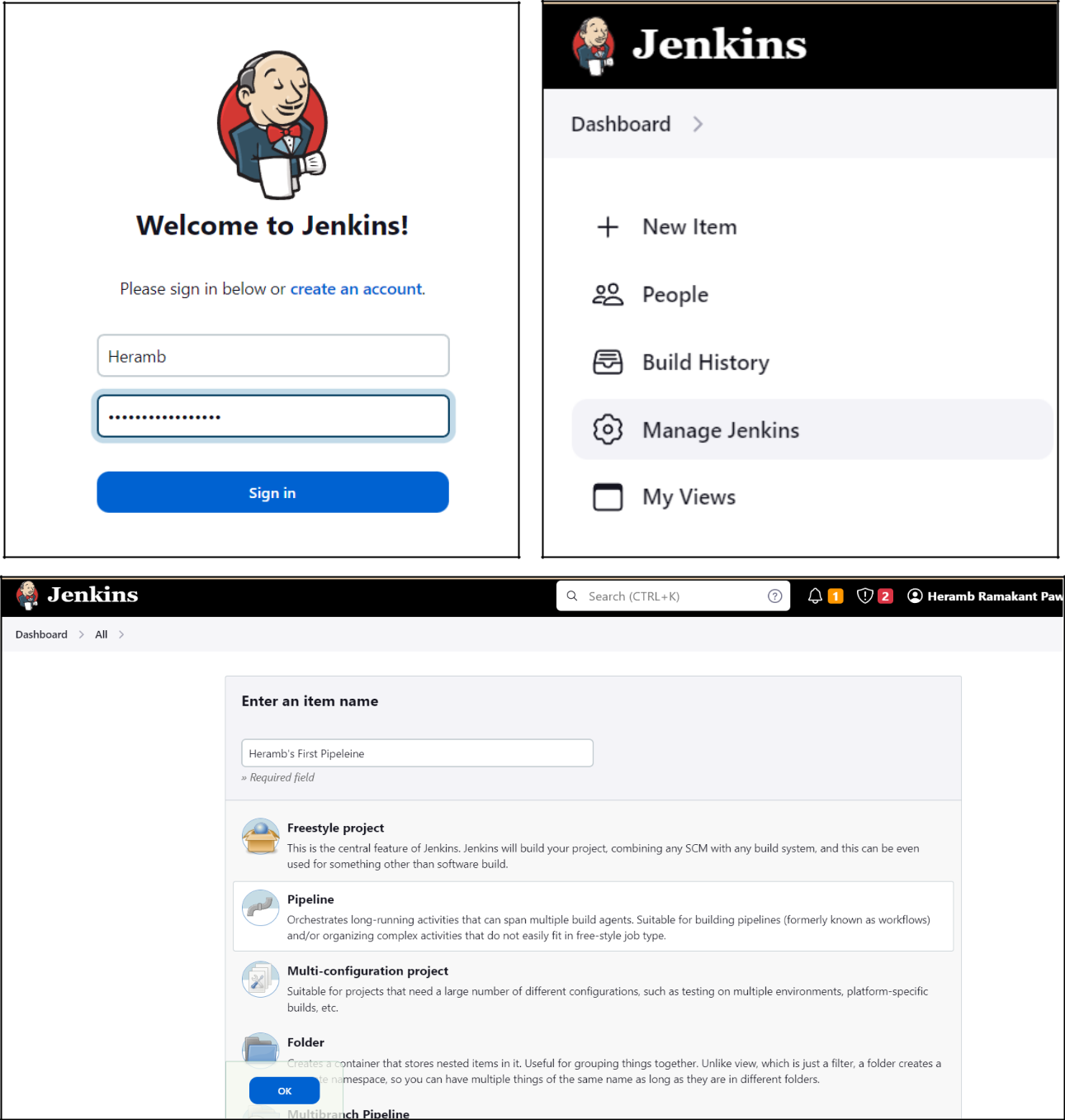
1. Automation: A pipeline automates the entire software delivery process, including building, testing, and deploying the application. This reduces the risk of human error, speeds up the delivery process, and makes it easier to maintain consistency across different environments.
2. Collaboration: A pipeline encourages collaboration between developers, testers, and operations teams by breaking down the software delivery process into smaller stages. This promotes better communication and ensures that everyone is working toward the same goal.
3. Quality assurance: A pipeline makes it easier to test and validate the application at every stage of the delivery process. This ensures that bugs and errors are caught early, before they can cause problems in production.
4. Continuous delivery: A pipeline enables continuous delivery, meaning that changes can be delivered to production quickly and frequently. This allows teams to respond more quickly to customer needs and feedback.
5. Scalability: A pipeline can be easily scaled to handle larger or more complex projects. This allows teams to work more efficiently and ensures that the delivery process can keep up with the demands of the project.

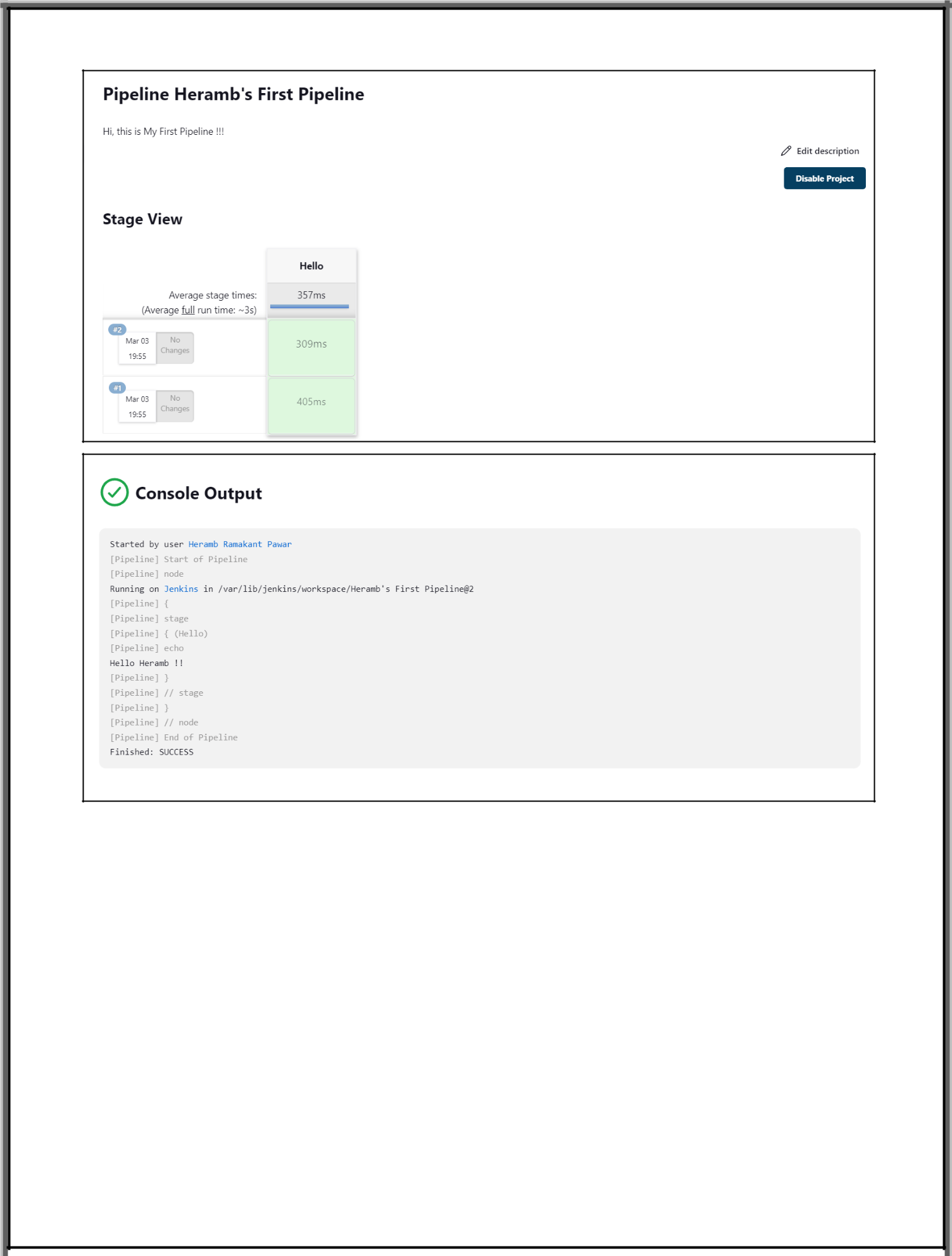
****

**Implementation:**

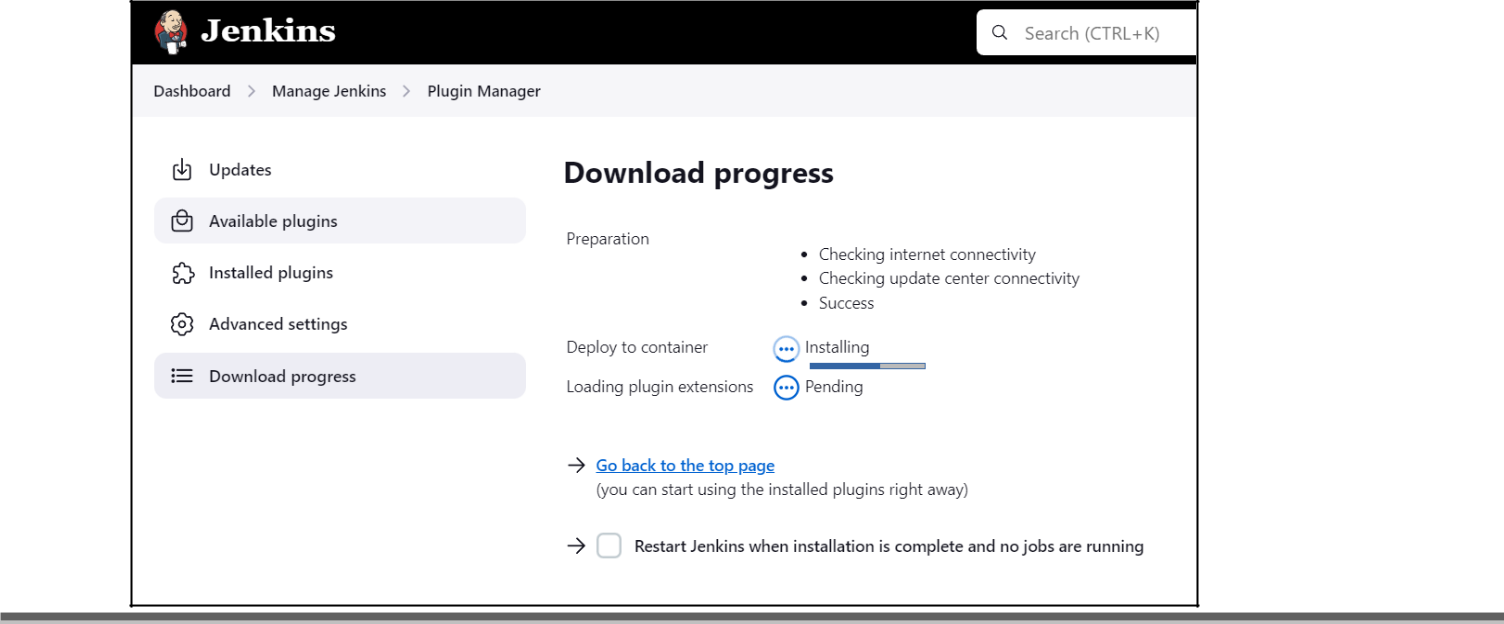
**Pre-requisites:** Tomcat Server

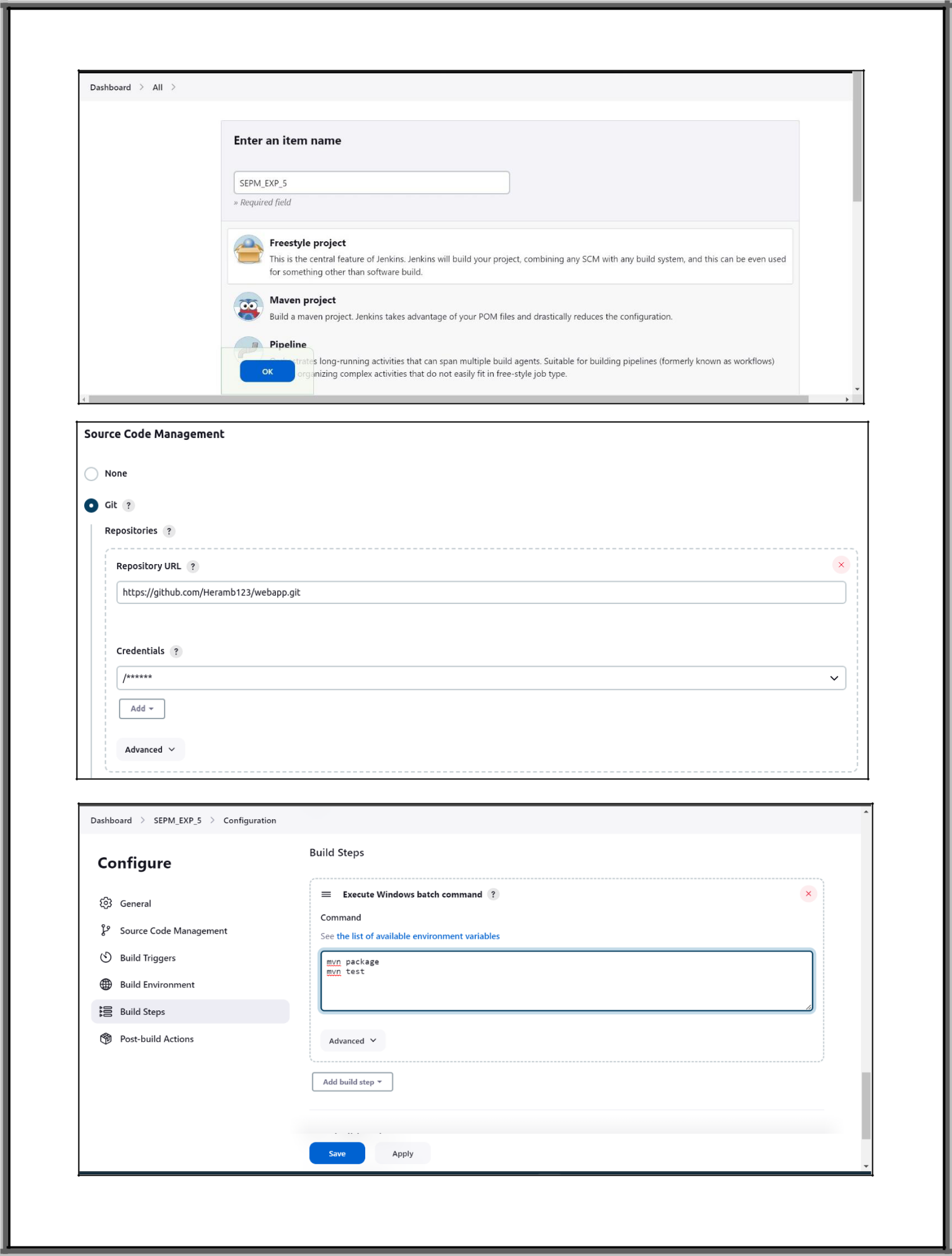
**Building First Pipeline**

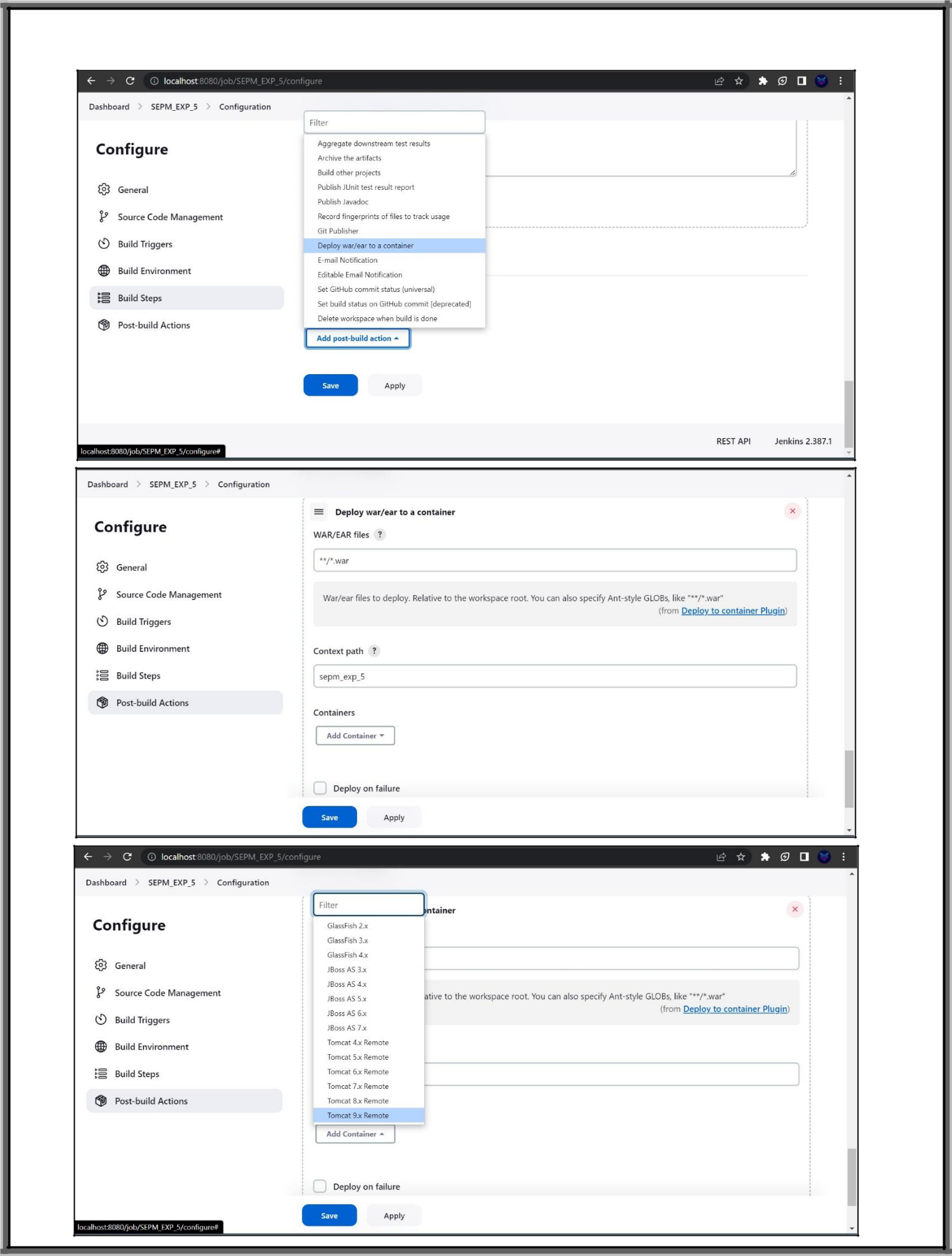




**Running Application on Tomcat:**







**Conclusion:**

We have studied Maven in Jenkins, and create a pipeline script to Test and deploy an application over a server. and its benefits. We also studied and installed Jenkins, along with Maven.