

Experiment 4 : To understand Continuous Integration, install and configure Jenkins with Maven/Ant/Gradle to set up a build job.

Aim : To understand the concept of **Continuous Integration (CI)** and implement it by installing and configuring **Jenkins** with **Maven, Ant, or Gradle** to automate the build process. This study aims to explore how Jenkins helps in setting up a CI pipeline, executing automated builds, and improving software development efficiency.

Theory :

Theory of Continuous Integration Using Jenkins with Maven, Ant, or Gradle

Introduction to Continuous Integration (CI)

Continuous Integration (CI) is a **software development practice** where developers frequently integrate their code changes into a shared repository. Each integration is verified using **automated builds and tests**, ensuring that issues are detected early. CI helps streamline the development process, reduces manual errors, and improves software quality.

Key principles of CI:

1. **Frequent Code Integration** – Developers merge changes multiple times a day.
2. **Automated Build Process** – Code is compiled, built, and tested automatically.
3. **Immediate Feedback** – Issues are detected early and fixed promptly.
4. **Consistent Environment** – CI ensures that software builds are reproducible across different environments.

To implement Continuous Integration, organizations use **CI tools like Jenkins**, which automates the build, test, and deployment process.

Jenkins: A CI/CD Automation Tool

Jenkins is an open-source **automation server** that enables developers to **automate software builds, tests, and deployments**. It supports integration with version control systems (Git, SVN) and build tools like **Maven, Ant, and Gradle**.

Key Features of Jenkins

- **Automated Builds:** Supports scheduled or triggered builds based on repository

changes.

- **Build Pipelines:** Allows chaining multiple jobs for end-to-end automation.
- **Plugin Support:** Offers 1,500+ plugins for integration with tools like Docker, Kubernetes, and Slack.
- **Scalability:** Can distribute builds across multiple nodes for faster execution.

Build Tools: Maven, Ant, and Gradle

Build tools are essential in CI to **compile source code, resolve dependencies, and generate deployable artifacts.**

1. Apache Maven

- A widely used **Java-based build automation tool.**
- Uses **POM.xml (Project Object Model)** to define project dependencies, build lifecycle, and plugins.
- Supports phases like **clean, compile, test, package, install, and deploy.**
- Command to build a project:
- `mvn clean install`

2. Apache Ant

- **Older than Maven**, but still used for Java builds.
- Uses an **XML-based build script (build.xml)** to define tasks.
- More flexible but requires explicit configurations.

Command to execute a build:

- `ant build`

3. Gradle

- **Newer build tool**, used for **Java, Kotlin, and Android development**.
- Uses a **Groovy or Kotlin-based build script** instead of XML.
- Faster than Maven due to its **incremental build mechanism**.
- Command to build a project:
- `gradle build`

Jenkins Integration with Maven, Ant, and Gradle

Jenkins can be configured to **automate builds** using these tools. The integration process involves:

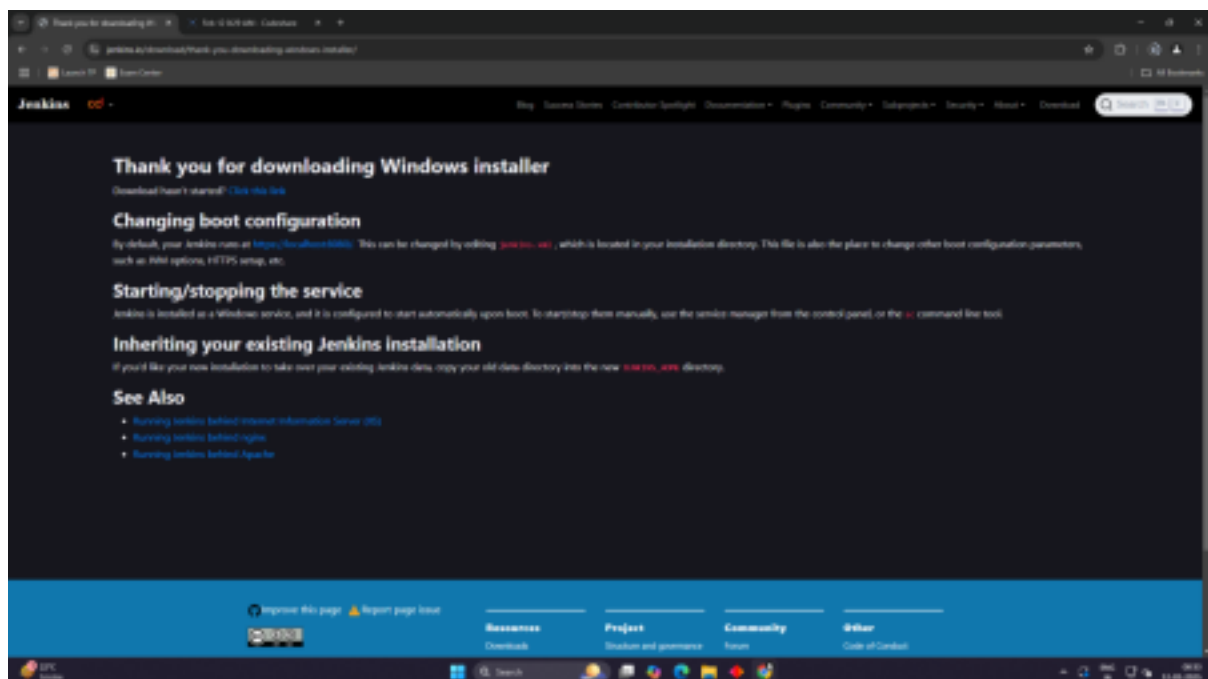
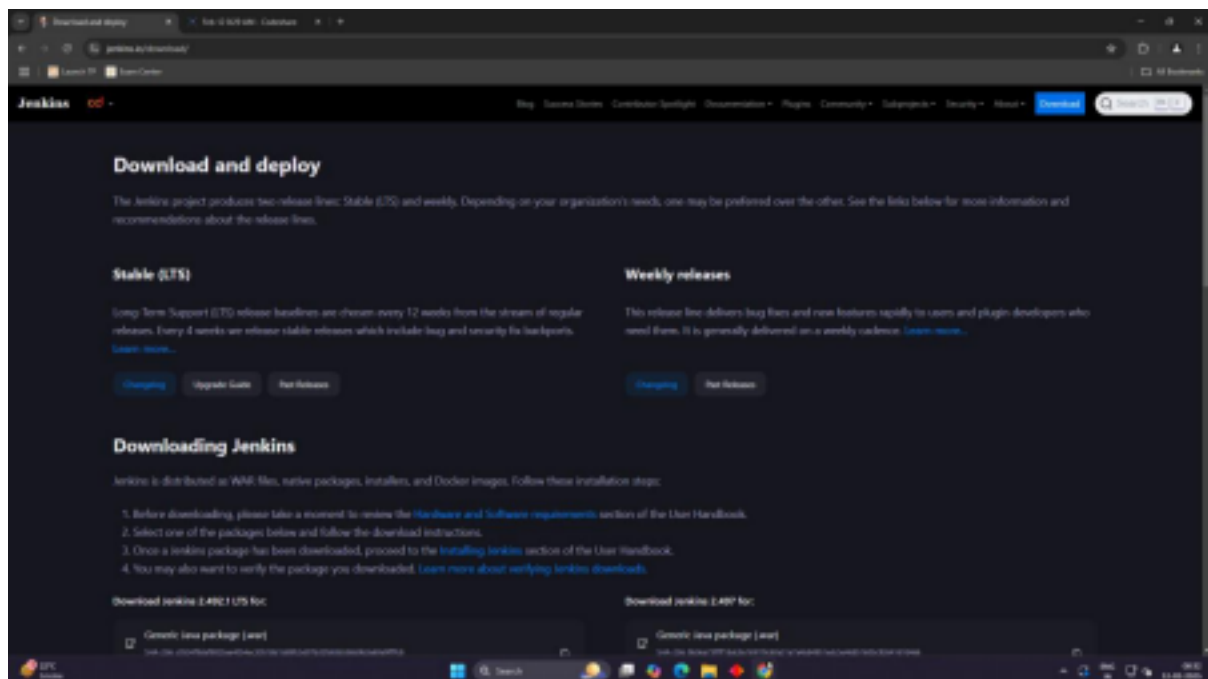
1. **Installing Jenkins** and setting up build tools.

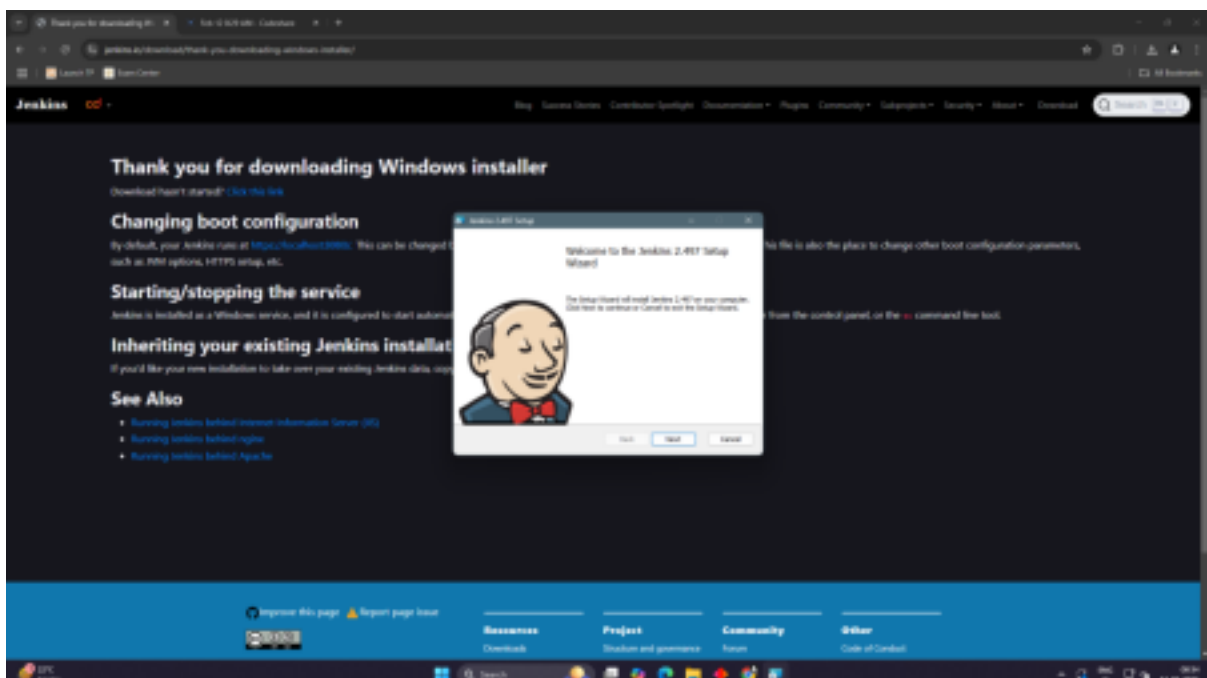
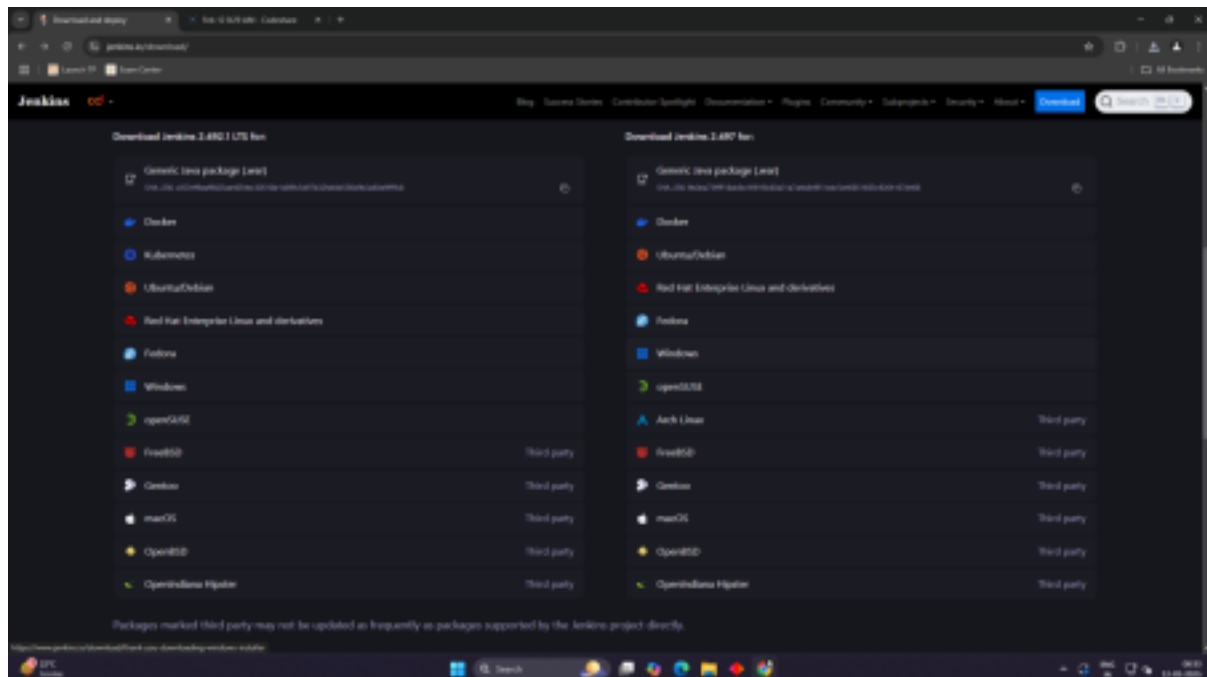
2. **Creating a job in Jenkins** that fetches source code from Git.
3. **Configuring build steps** to invoke Maven, Ant, or Gradle commands.
4. **Executing automated builds** and monitoring results.

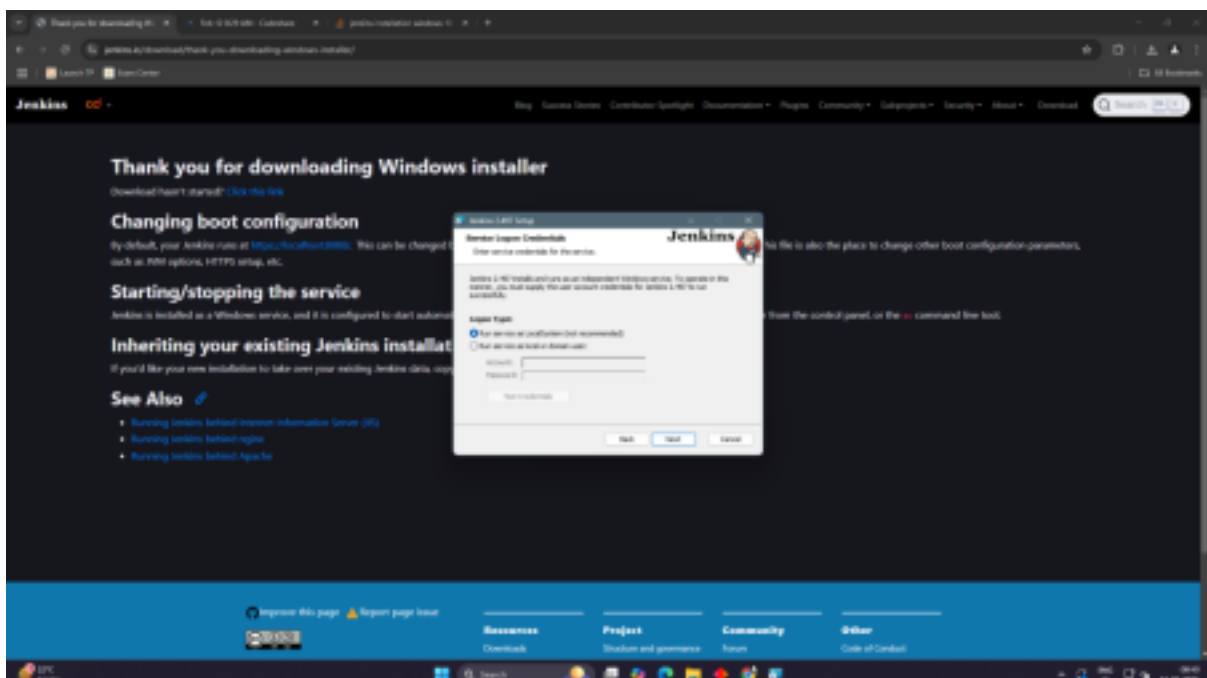
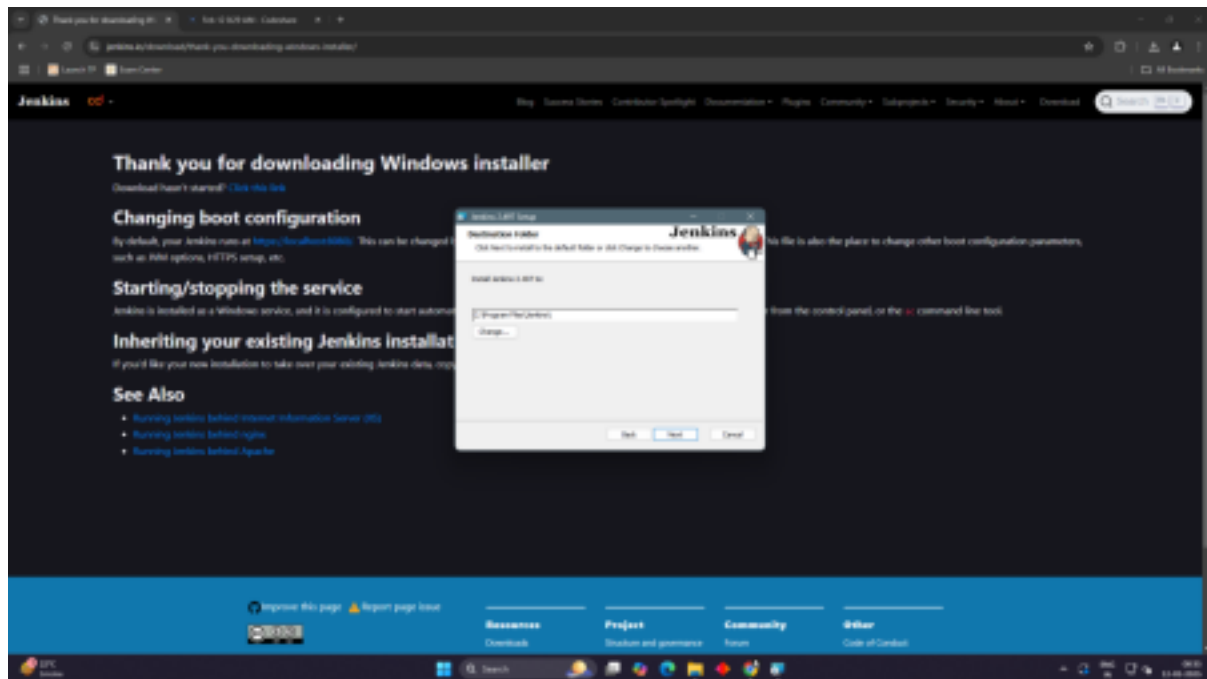
Advantages of Using Jenkins for CI

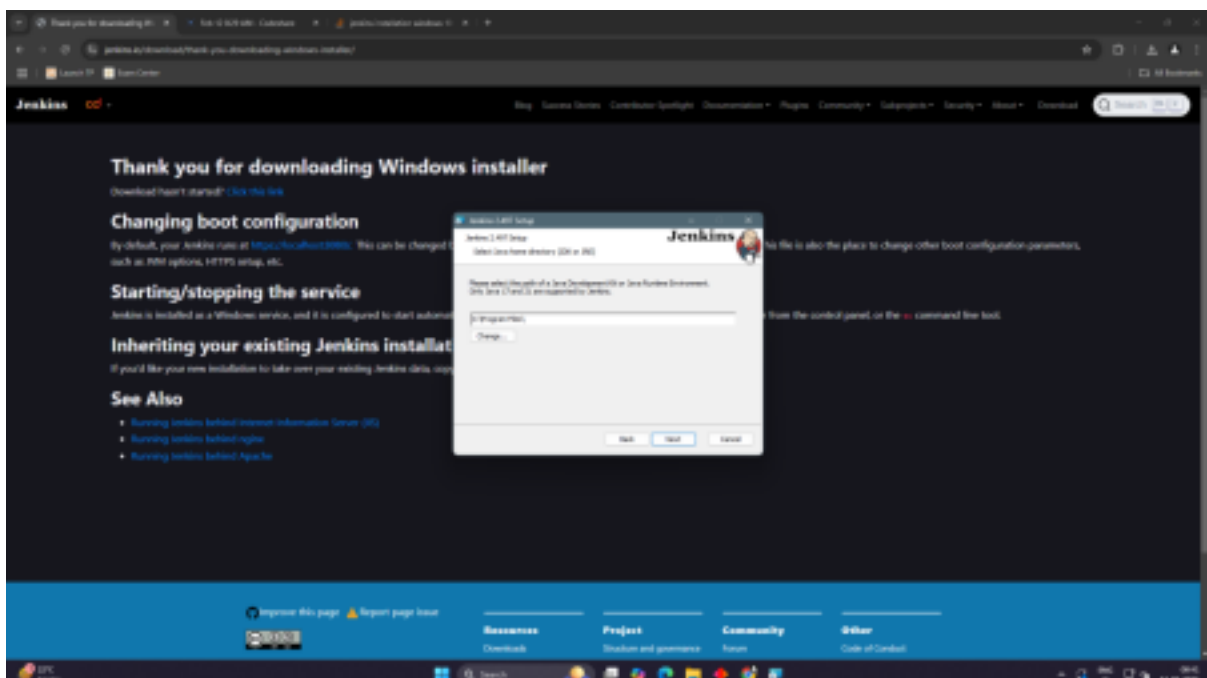
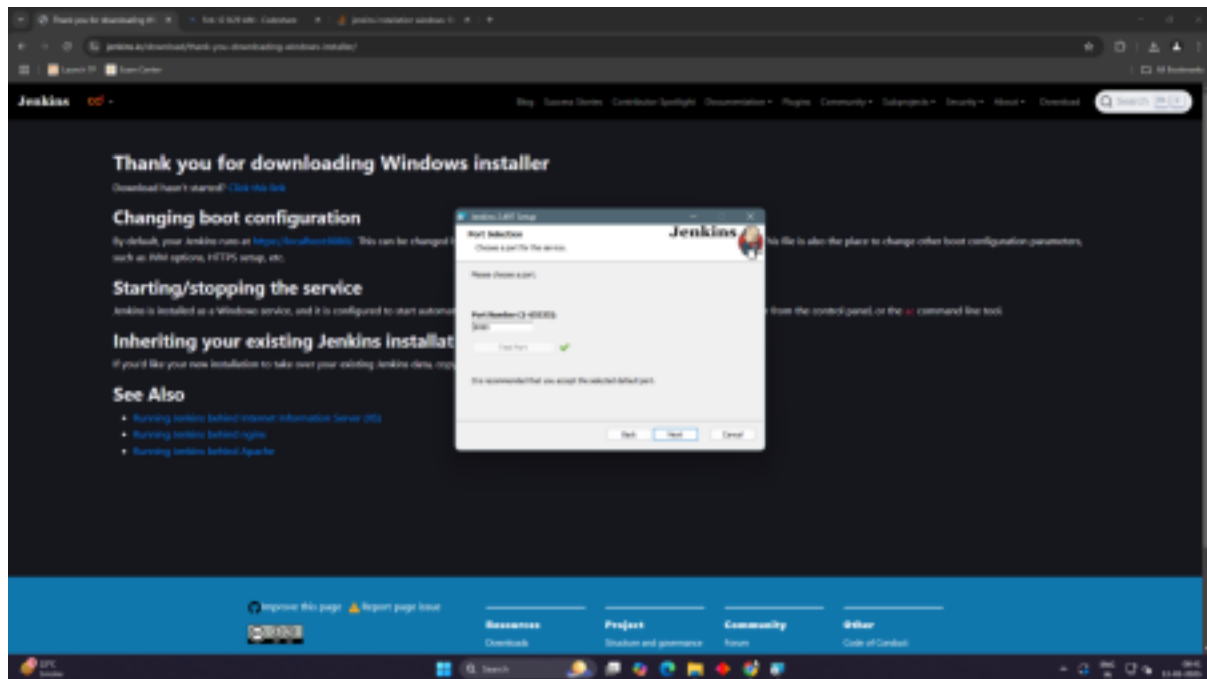
- **Faster Development Cycle** – Automated builds and testing reduce manual effort.
- **Early Bug Detection** – Continuous integration ensures quick issue identification.
- **Improved Collaboration** – Developers work on the latest stable codebase.
- **Efficient Deployment** – Jenkins supports integration with **Docker, Kubernetes, and cloud platforms**.

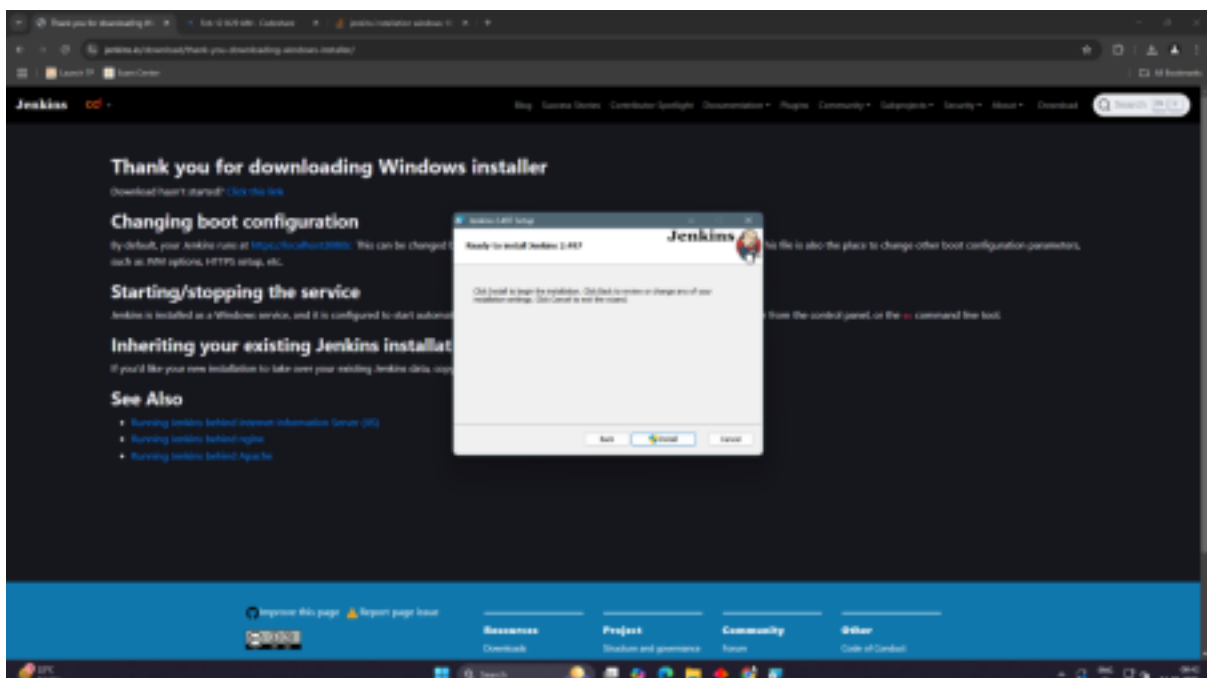
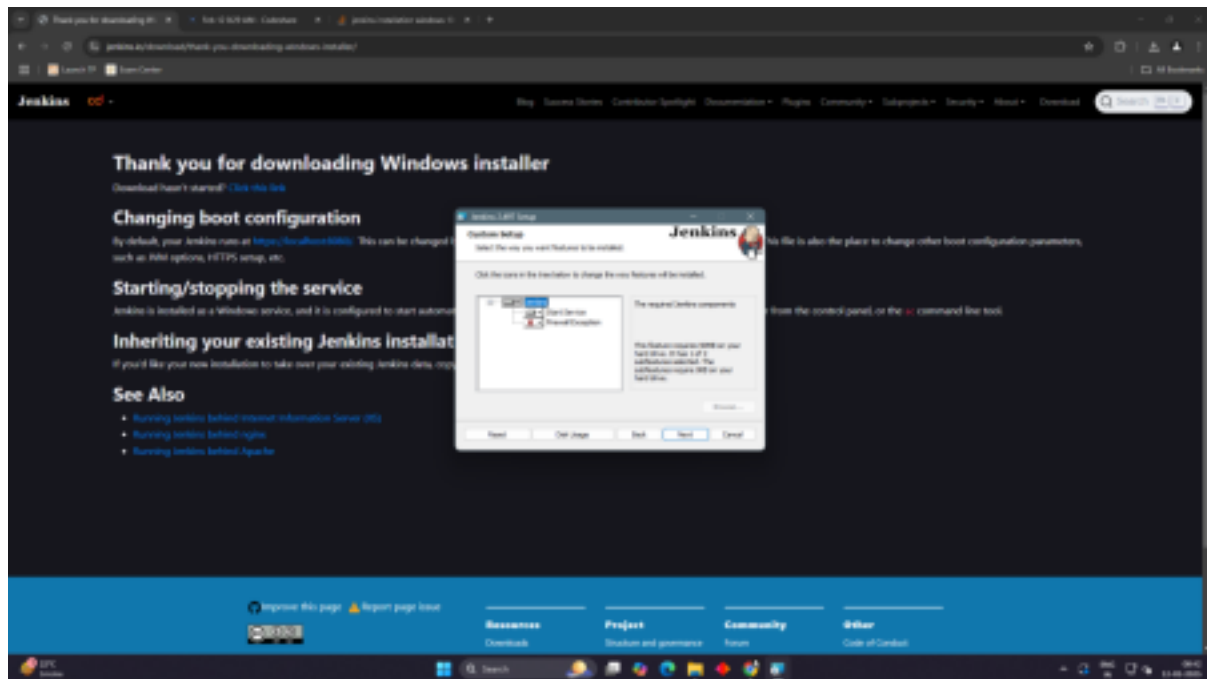
Implementation :

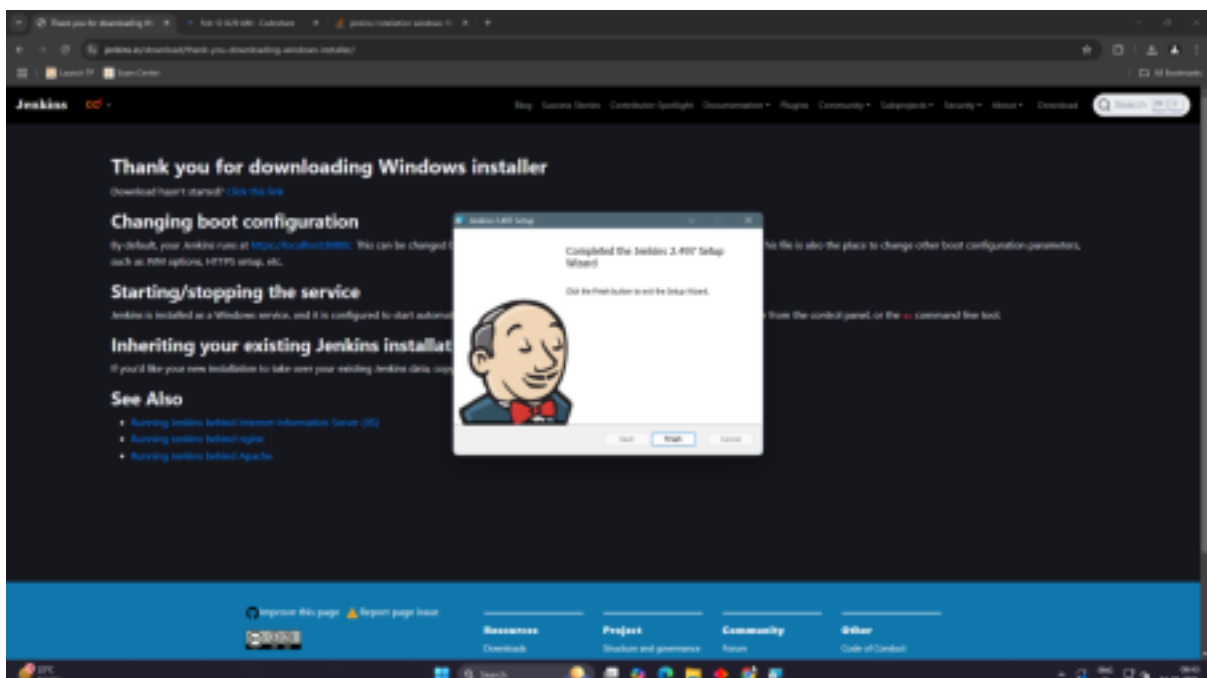
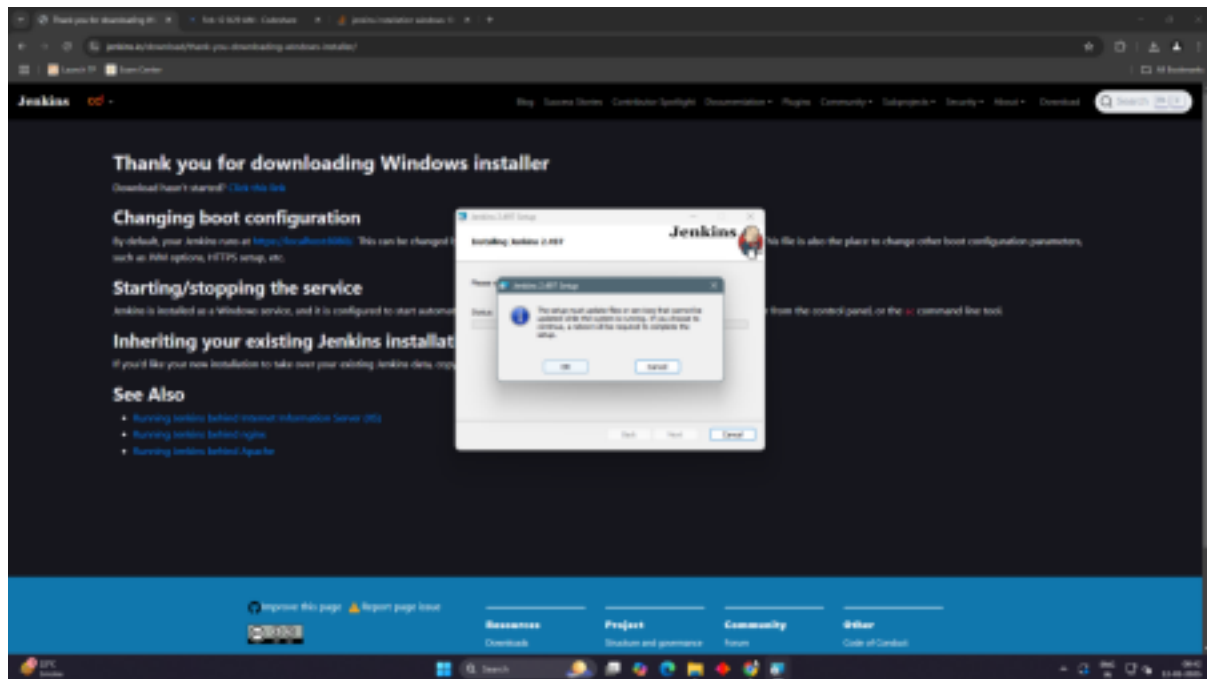


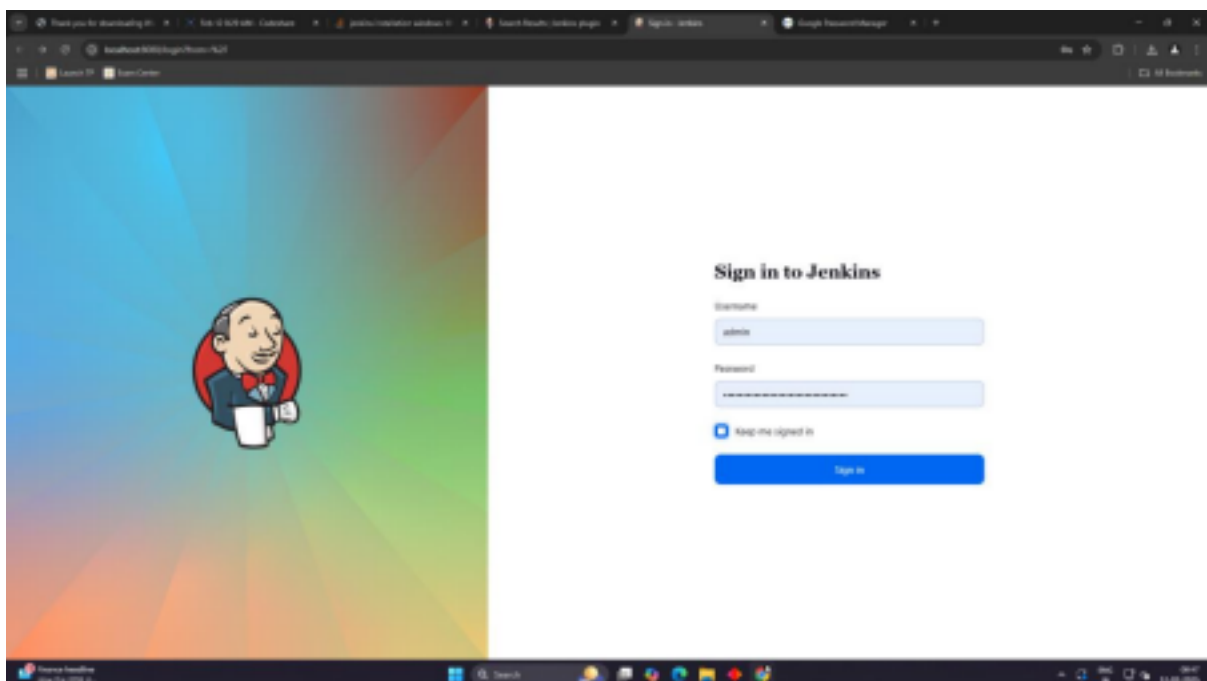
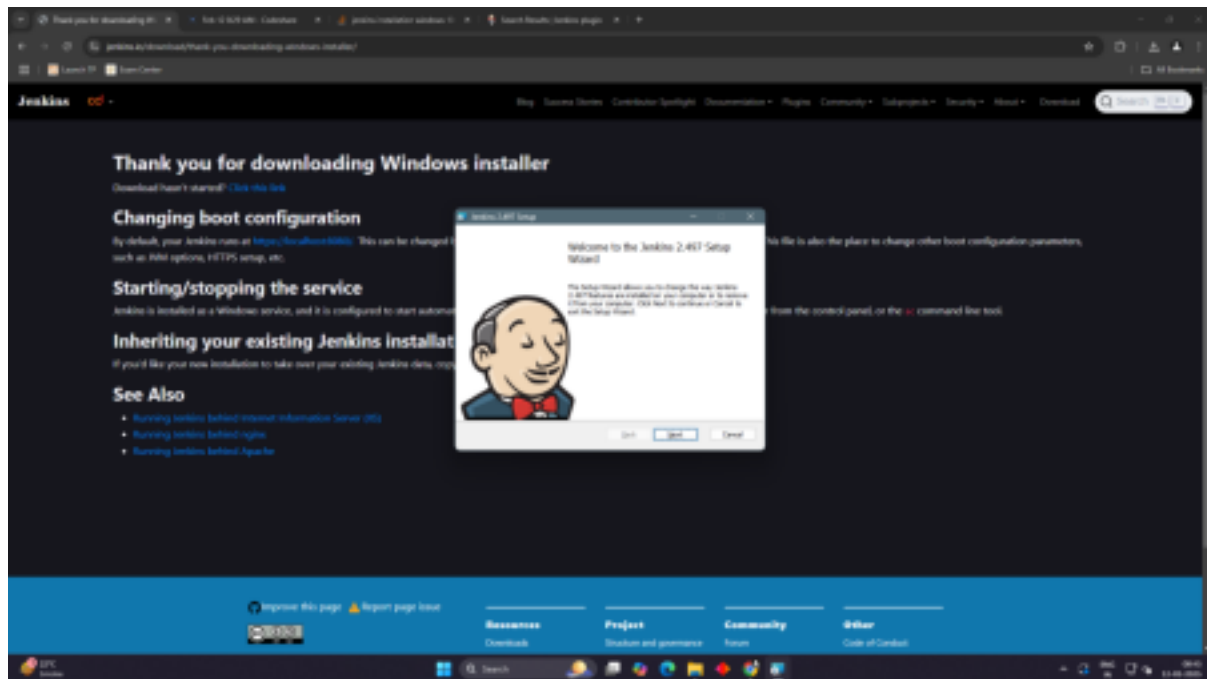


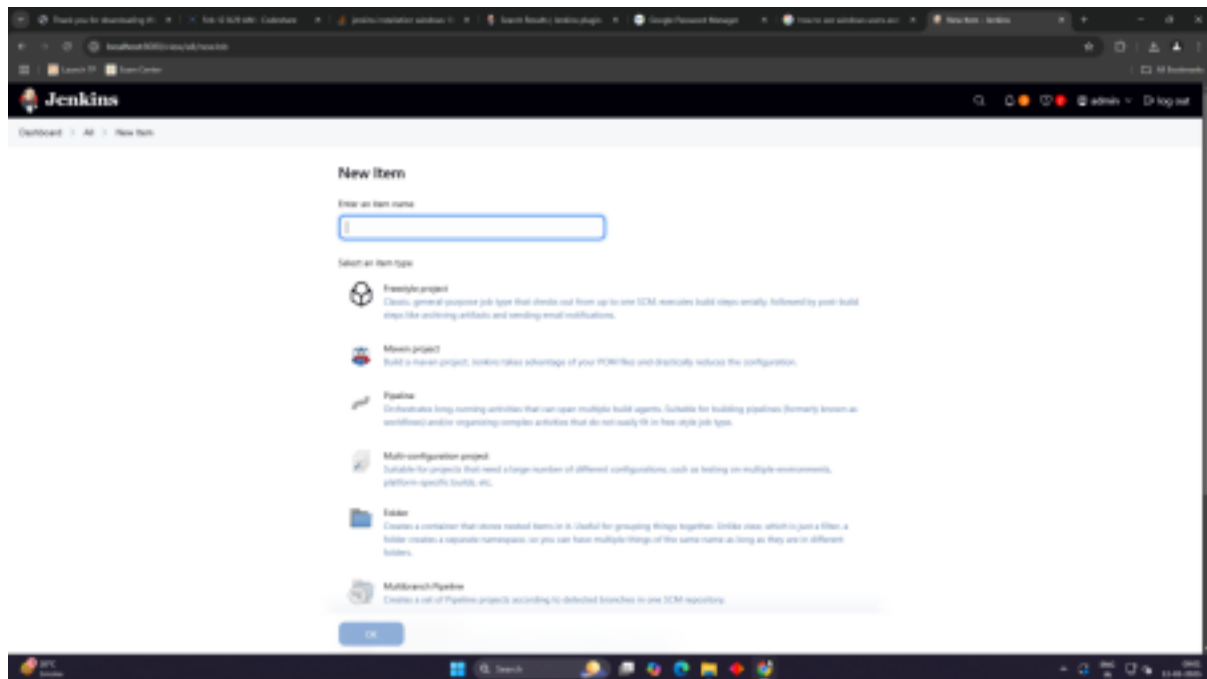












Conclusion : Thus we have successfully installed and configured Jenkins.