

## ASSIGNMENT 5

AIM: To build Java program using Jenkins.

LO MAPPED: LO1, LO3

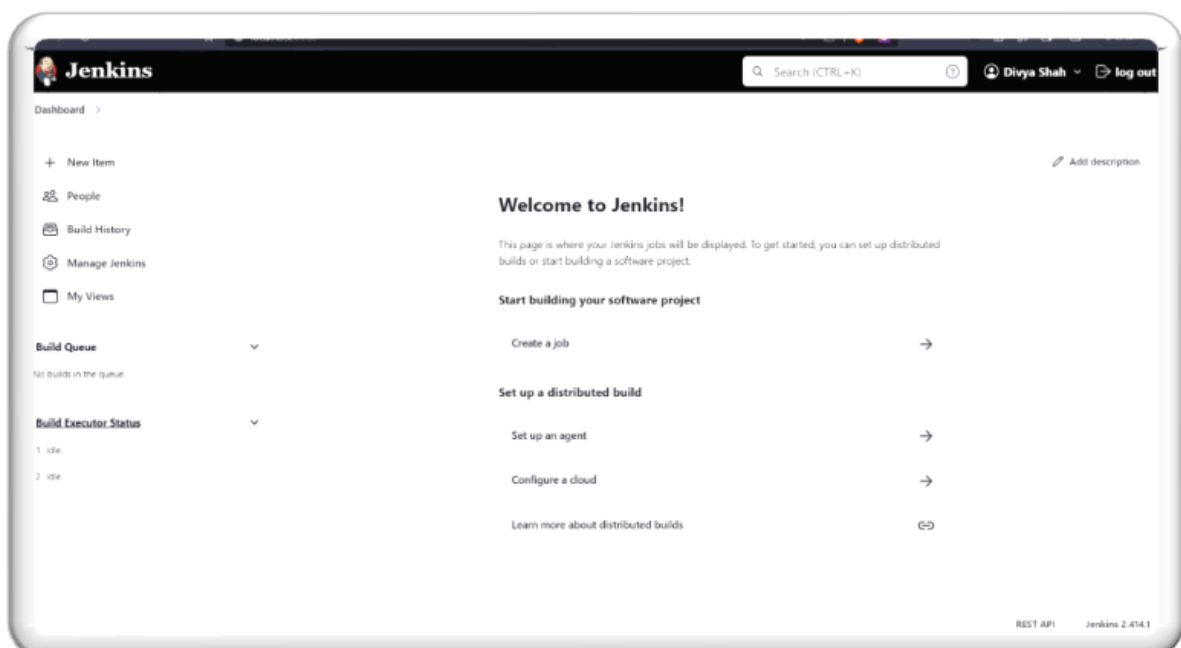
### THEORY:

Jenkins is an open-source automation server that facilitates continuous integration (CI) and continuous delivery (CD) of software projects. It helps automate the build, test, and deployment processes, allowing development teams to streamline their development workflow and deliver high-quality software more efficiently. Jenkins provides an extensible plugin architecture that enables integration with various tools and technologies.

### **Steps to Build a Java Program Using Jenkins:**

To build a Java program using Jenkins, you'll need to set up a Jenkins job that defines the build process. Below are the steps to accomplish this:


1. Install Jenkins: Download and install Jenkins on your system.




2. Create Job: Create a new freestyle project job in Jenkins.

**Enter an item name**


= Required field

**Freestyle project**


This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

**Pipeline**


Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

**Multi-configuration project**

Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

**Folder**

Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

**Multibranch Pipeline**

Creates a set of Pipeline projects according to detected branches in one SCM repository.

3. Source Code: Configure source code management (e.g., Git repository URL).

**Jenkins** Search (CTRL+K) Divya Shukla log out

Dashboard > FirstJavaProgram > Configuration

**Configure**

**General**

Enabled ☒

General

Source Code Management

Build Triggers

Build Environment

Build Steps

Post-build Actions

Description

Project url

Advanced

Advanced

Source Code Management

Discard old builds

☐ Discard old builds

☒ **GitHub project**

Project url

☐ This project is parameterised

☐ Throttle builds

☐ Execute concurrent builds if necessary

Dashboard > FirstJavaProgram >

Status

Changes

Workspace

Build Now

Configure

Delete Project

GitHub

Rename

Project FirstJavaProgram

Password Generator Java Project

Permalinks

General

Enabled ☒

Description

Hello World Java Program

Advanced ☐ Edited

☐ Quiet period ?

☐ Retry Count ?

☐ Block build when upstream project is building ?

☐ Block build when downstream project is building ?

☒ Use custom workspace ?

Directory

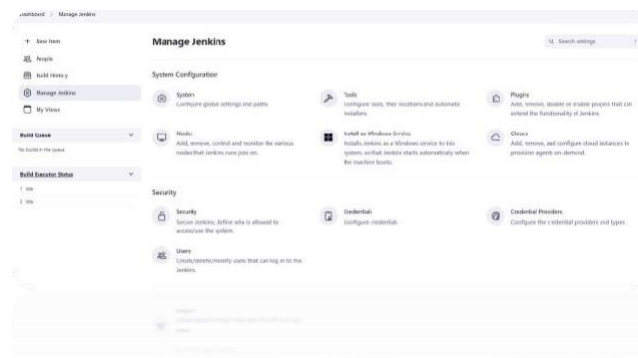
C:\Users\Asus\OneDrive\Documents\NetBeansProjects\MyFirstJavaApp\src\com\java\main\MyFirstJavaProgram.java

Display Name ?

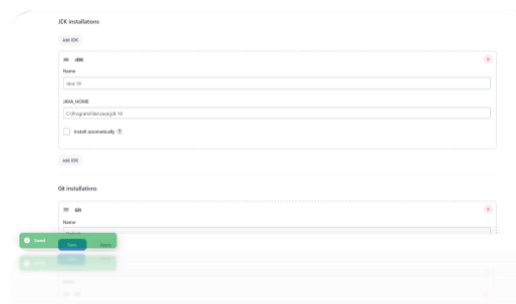
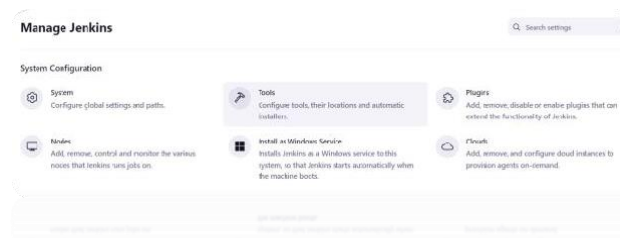
☐ Keep the build logs of dependencies ?



4. Build Trigger: Set up build triggers, like periodic or repository changes.



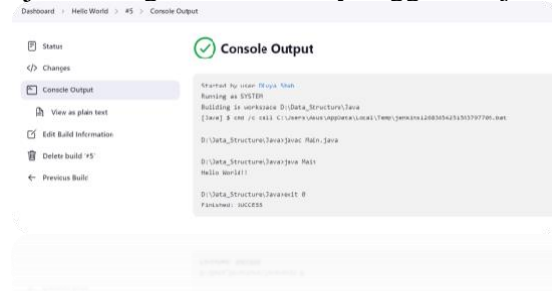
5. Build Steps: Add build step, e.g., "Invoke top-level Maven targets" for Maven projects with goals like clean install.



6. Post-Build Actions: Optionally configure actions after the build, like archiving artifacts or notifications.



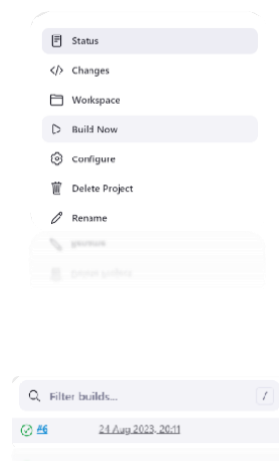
7. Save and Run: Save job settings and manually trigger the job.

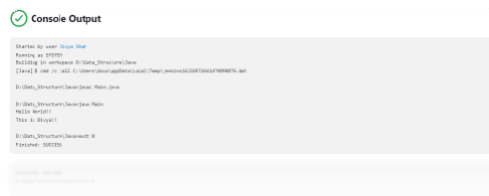


8. Monitor Progress: Watch the Jenkins dashboard for build progress.



9. View Artifacts: Access build artifacts if configured.





**CONCLUSION:** By this assignment we learned how to build a java program using Jenkins.

