Lab (2)

1) What is Jenkins pipeline?

Jenkins Pipeline is a suite of plugins that supports implementing and integrating continuous delivery pipelines into Jenkins. A continuous delivery pipeline is an automated expression of your process for getting software from version control right through to your users and customers.

2) What scripting language is Jenkins pipeline syntax based on?

Jenkins pipeline syntax is based on Groovy scripting language

3) What are the different ways to trigger pipeline?

There are many different ways to trigger a pipeline. Some of the most common ways include:

Manually triggering a pipeline: This is the simplest way to trigger a pipeline. You can do this by clicking the "Run" button in the Jenkins UI, or by using the jenkins.model.Jenkins.instance.run() method in the Groovy API.

Triggering a pipeline on a schedule: This allows you to run a pipeline at a specific time or interval. You can do this by configuring a schedule trigger in the Jenkins UI, or by using the jenkins.triggers.Trigger.schedule() method in the Groovy API.

Triggering a pipeline on a change in a repository: This allows you to run a pipeline whenever a change is made to a repository. You can do this by configuring a git trigger in the Jenkins UI, or by using the jenkins.triggers.Trigger.github() method in the Groovy API.

Triggering a pipeline on an event: This allows you to run a pipeline in response to an event, such as a new issue being created or a build failing. You can do this by configuring an event trigger in the Jenkins UI, or by using the jenkins.triggers.Trigger.buildTrigger() method in the Groovy API.

Triggering a pipeline from another pipeline: This allows you to run a pipeline as a dependency of another pipeline. You can do this by configuring a pipeline resource trigger in the Jenkins UI, or by using the jenkins.triggers.Trigger.pipelineResourceTrigger() method in the Groovy API.

4) What is different between parameter and jenkins env variable?

Parameters and environment variables are both used to pass values to Jenkins pipelines. However, they have different purposes and are used in different ways.

Parameters are values that are passed to a pipeline when it is run. They are typically used to configure the pipeline, such as the branch to build or the environment to use. Parameters can be set in the Jenkins UI, in a build script, or in a pipeline.

Environment variables are values that are available to all stages of a pipeline. They can be used to store configuration information, such as the database password or the API key. Environment variables can be set in the Jenkins UI, in a build script, or in a pipeline.

5) What is organization folder job and what is used for?

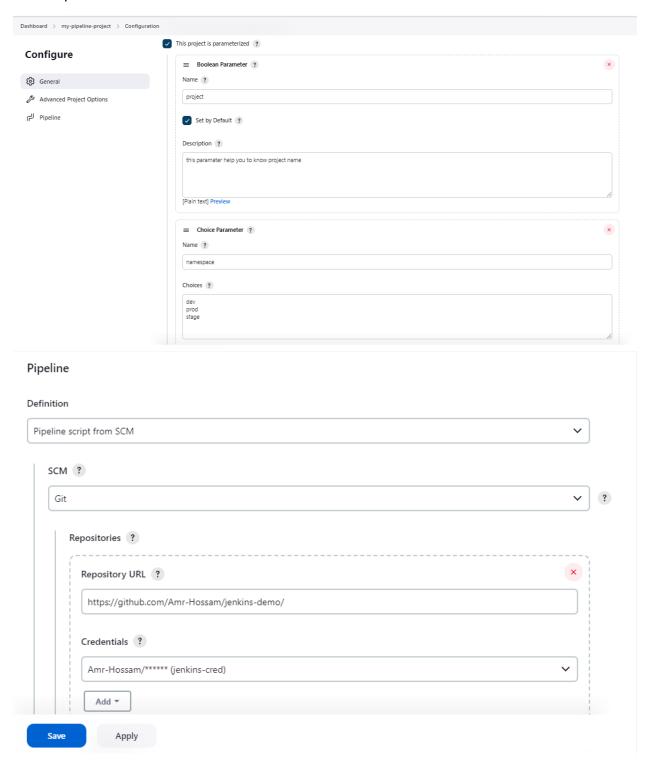
An Organization Folder Job is a type of job that allows you to automatically create and manage Multibranch Pipeline jobs for all repositories in an organization. This can be useful for large organizations with many repositories, as it can help to automate the process of creating and managing Jenkins jobs.

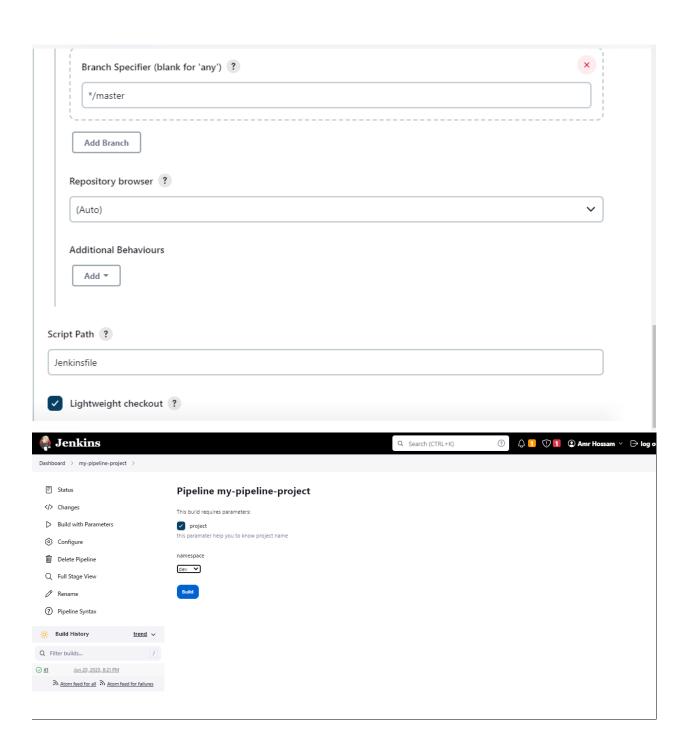
Once the Organization Folder Job has been created, Jenkins will automatically scan the organization's repositories for any repositories that contain a Jenkinsfile. If a Jenkinsfile is found, Jenkins will create a Multibranch Pipeline job for that repository.

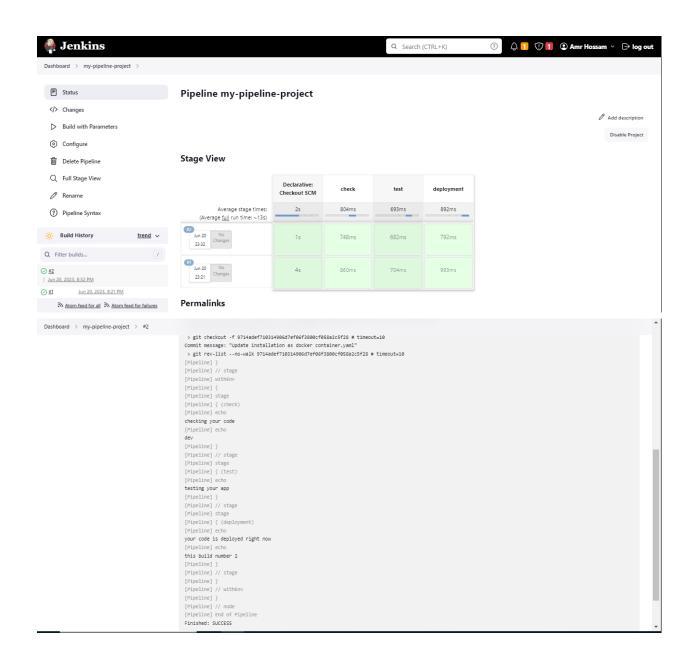
Organization Folder Jobs can be used to:

- Automate the process of creating and managing Jenkins jobs for large organizations.
- Provide a centralized view of all the Jenkins jobs for an organization.
- Make it easier to manage and configure Jenkins jobs for an organization.

6) Create Jenkins pipeline for your repo and use script file (jenkinsfile) to write pipeline syntax, include parameter functions and env variable in it?







7) Create another multibranch pipeline and filter branches to contain only (master, dev, test)?

