

Day 26 Task: Jenkins Declarative Pipeline:

One of the most important parts of your DevOps and CI/CD journey is a Declarative Pipeline Syntax of Jenkins.

Some terms:

- **What is Pipeline** - A pipeline is a collection of steps or jobs interlinked in a sequence.
- **Declarative:** Declarative is a more recent and advanced implementation of a pipeline as a code.
- **Scripted:** Scripted was the first and most traditional implementation of the pipeline as a code in Jenkins. It was designed as a general-purpose DSL (Domain Specific Language) built with Groovy.

Why you should have a Pipeline:

The definition of a Jenkins Pipeline is written into a text file (called a [Jenkinsfile](#)) which in turn can be committed to a project's source control repository.

This is the foundation of "Pipeline-as-code"; treating the CD pipeline as a part of the application to be versioned and reviewed like any other code.

Creating a [Jenkinsfile](#) and committing it to source control provides a number of immediate benefits:

Automatically creates a Pipeline build process for all branches and pull requests.

Code review/iteration on the Pipeline (along with the remaining source code).

Pipeline syntax:

```
pipeline {  
    agent any  
    stages {  
        stage('Build') {  
            steps {  
                //  
            }  
        }  
        stage('Test') {
```

```

    steps {
        //
    }
}

stage('Deploy') {
    steps {
        //
    }
}
}
}
}
}

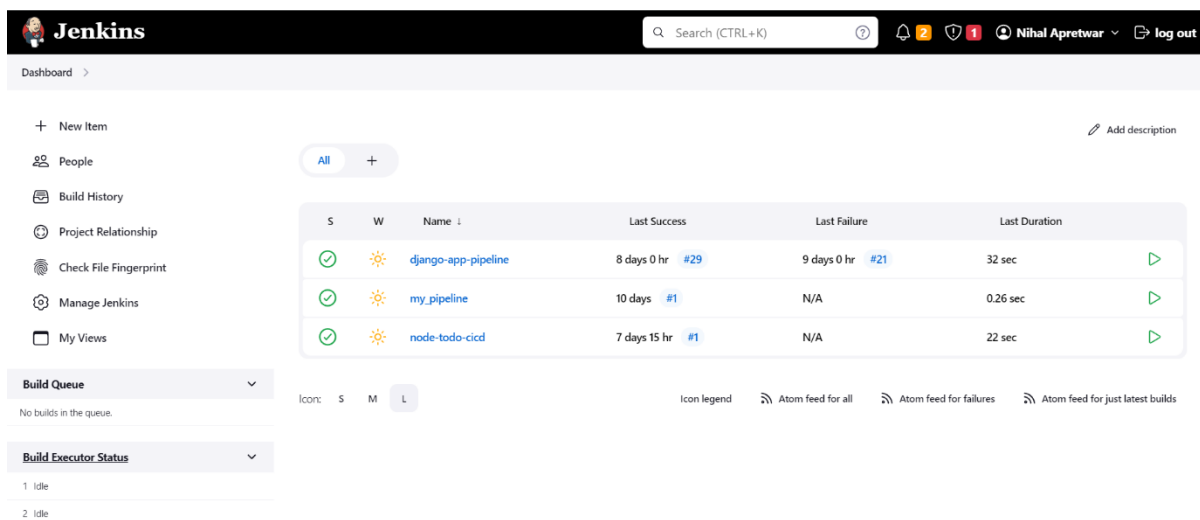
```

Task:

Create a New Job, this time select Pipeline instead of Freestyle Project.

Complete the example using the Declarative pipeline.

- Start jenkins server and click on “New Item” on home screen.

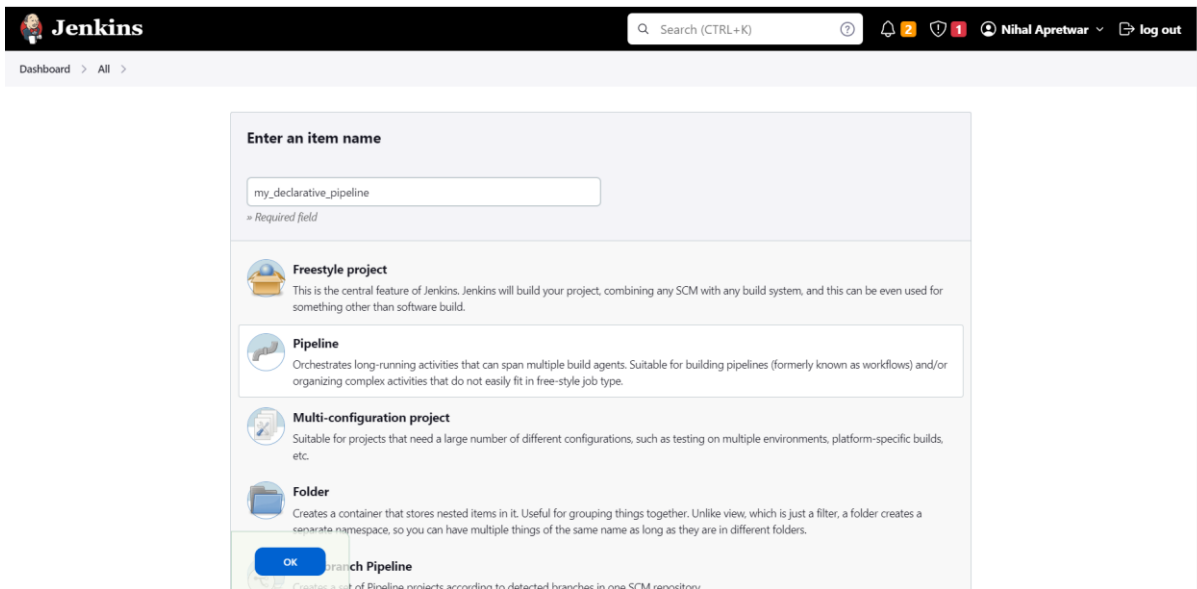


The screenshot shows the Jenkins dashboard with the following components:

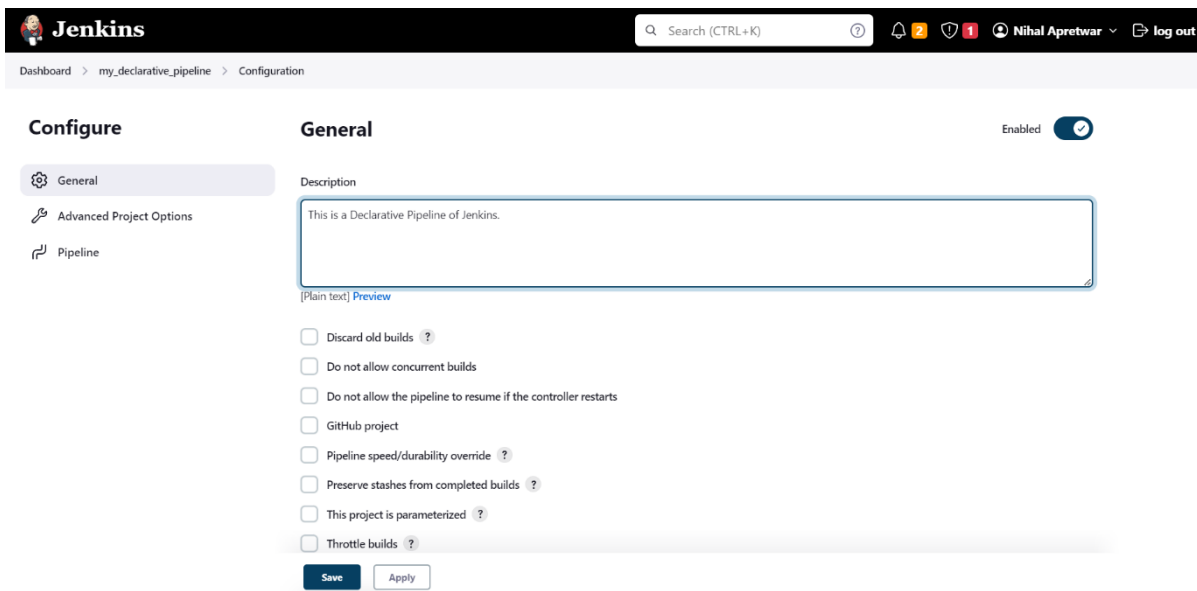
- Header:** Jenkins logo, search bar (Search (CTRL+K)), and user profile (Nihal Apretwar) with a log out button.
- Left Sidebar:**
 - + New Item
 - People
 - Build History
 - Project Relationship
 - Check File Fingerprint
 - Manage Jenkins
 - My Views
- Main Content Area:**
 - Build Queue:** No builds in the queue.
 - Build Executor Status:** 1 Idle, 2 Idle.
 - Pipeline Table:**

S	W	Name ↓	Last Success	Last Failure	Last Duration
✓	☀	django-app-pipeline	8 days 0 hr #29	9 days 0 hr #21	32 sec
✓	☀	my_pipeline	10 days #1	N/A	0.26 sec
✓	☀	node-todo-cicd	7 days 15 hr #1	N/A	22 sec

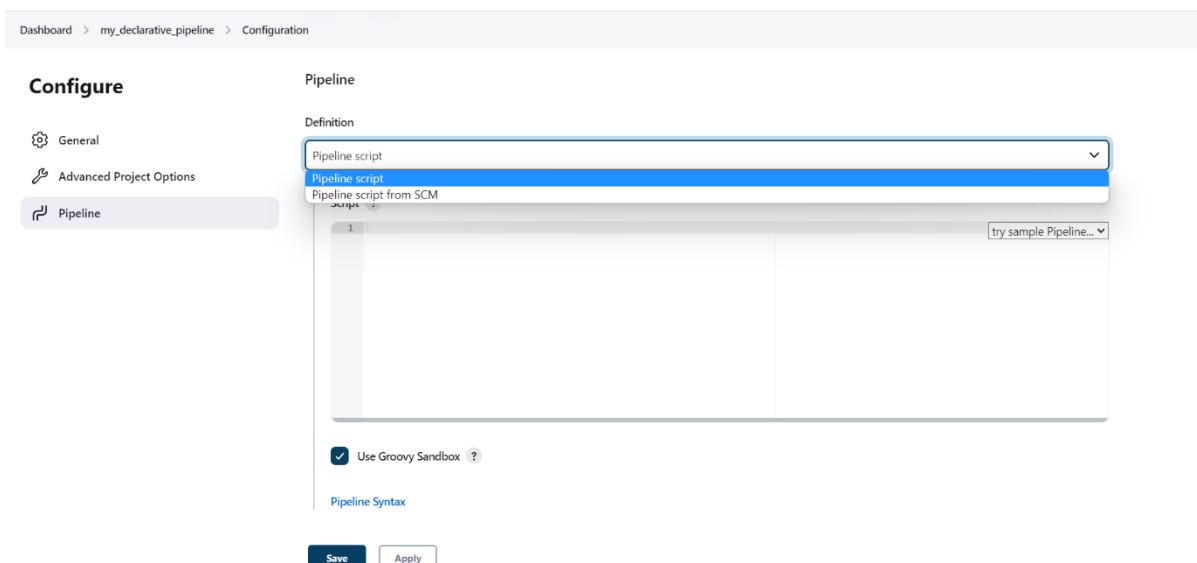
- Enter the desired name and after entering name, select Pipeline and click on ok button.



- Go to project configuration page and give the desired description for the job.



- Now go to the pipeline session, in definition select Pipeline script



- Start typing the code for building a declarative pipeline and click on the save button.

Configure

- General
- Advanced Project Options
- Pipeline

Definition

Pipeline script

Script ?


```
1 pipeline {
2   agent any
3
4   stages {
5     stage('Hello') {
6       steps {
7         echo 'Hello world this is my declarative pipeline'
8       }
9     }
10  }
11 }
```

☒ Use Groovy Sandbox ?[Pipeline Syntax](#)

Save

Apply

- Now build the project.

 **Jenkins**

Search (CTRL+K) ?

Dashboard > my_declarative_pipeline >

Status

</> Changes

Build Now

Configure

Delete Pipeline

Full Stage View

Rename

Pipeline Syntax

Pipeline my_declarative_pipeline

This is a Declarative Pipeline of Jenkins.

Stage View

Average stage times:	
(Average full run time: ~4s)	
#2	Feb 11 16:26 No Changes
#1	Feb 11 16:12 No Changes

Hello

302ms

302ms


Build History

trend

Filter builds...

#2 Feb 11, 2023, 10:56 AM

- Now you see the job will be build successfully.

 **Jenkins**

Search (CTRL+K) ?

Dashboard > my_declarative_pipeline > #2

Status

</> Changes

Console Output

View as plain text

Edit Build Information

Delete build '#2'

Restart from Stage

Replay

Pipeline Steps

Workspaces

Previous Build

Console Output

Started by user Nihal Apretwar

[Pipeline] Start of Pipeline

[Pipeline] node

Running on Jenkins in /var/lib/jenkins/workspace/my_declarative_pipeline

[Pipeline] {

[Pipeline] stage

[Pipeline] { (Hello)

[Pipeline] echo

Hello world this is my declarative pipeline

[Pipeline] }

[Pipeline] // stage

[Pipeline] }

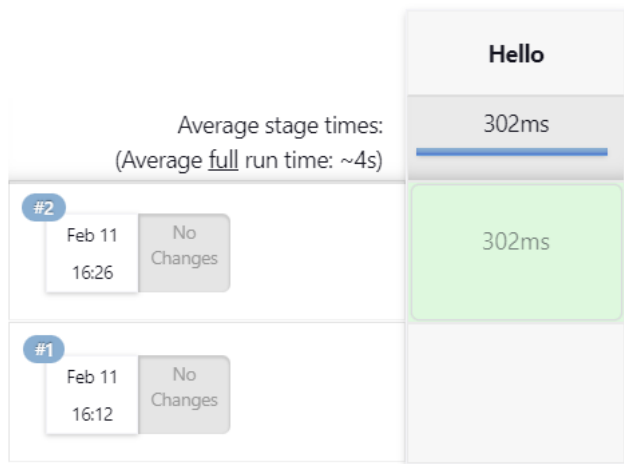
[Pipeline] // node

[Pipeline] End of Pipeline

Finished: SUCCESS



my_declarative_pipeline - Stage View



Thank you for reading!

Happy Learning 😊