<u>Experiment 2</u>

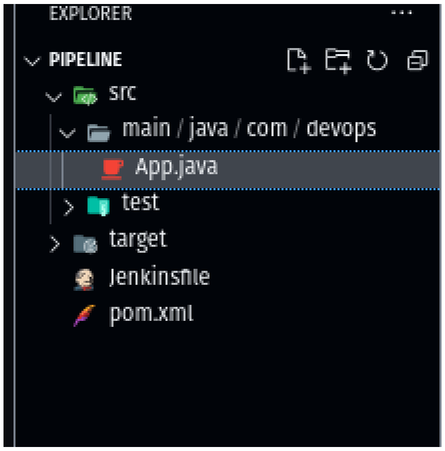
Creating a Jenkins Pipeline with a Jenkinsfile

Aim

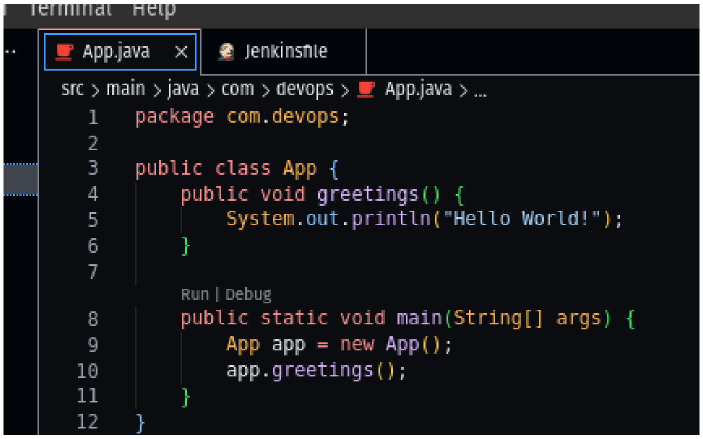
Create a Jenkins pipeline using a Jenkinsfile that builds a simple project, runs tests, and deploys the project to a designated environment

1. Create a maven project

a. Create the project

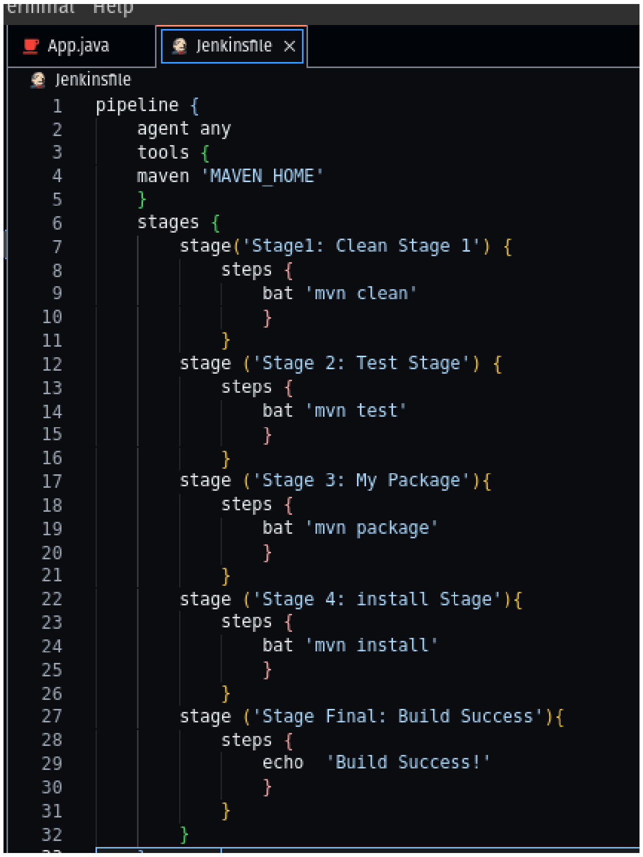


b. Create a package DevOps inside src/main/java/ and add App.java file



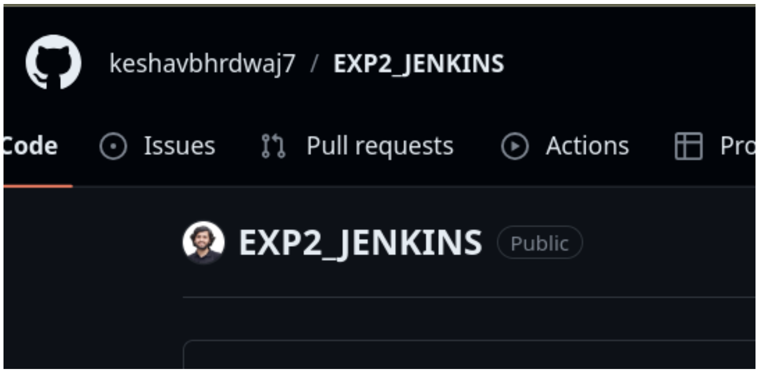
```
package com.devops;

public class App {
    public void greetings() {
        System.out.println("Hello World!");
    }

    public static void main(String[] args) {
        App app = new App();
        app.greetings();
    }
}
```

c. Create a Jenkinsfile and write the steps to be executed in it



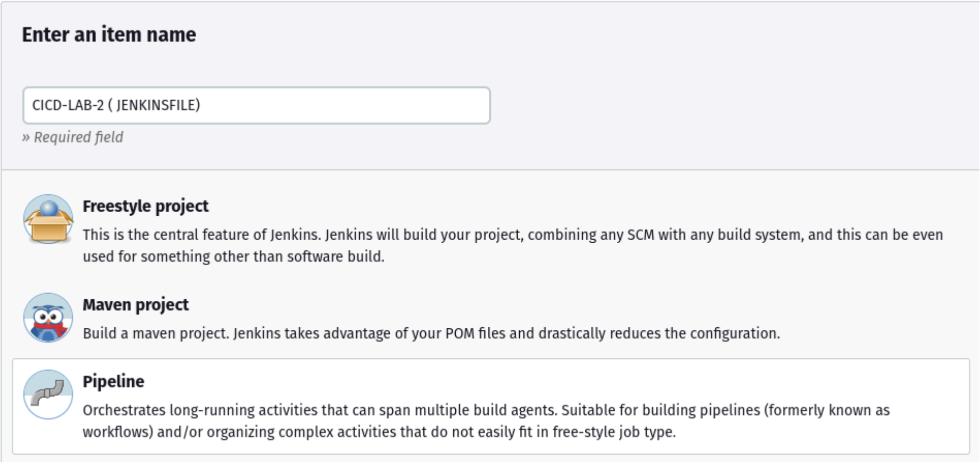
```
pipeline {
    agent any
    tools {
    maven 'MAVEN_HOME'
    }
    stages {
        stage('Stage1: Clean Stage 1') {
            steps {
                bat 'mvn clean'
                }
            }
        stage ('Stage 2: Test Stage') {
            steps {
                bat 'mvn test'
                }
            }
        stage ('Stage 3: My Package'){
            steps {
                bat 'mvn package'
                }
            }
        stage ('Stage 4: install Stage'){
            steps {
                bat 'mvn install'
                }
            }
        stage ('Stage Final: Build Success'){
            steps {
                echo  'Build Success!'
                }
            }
        }
```

2. Create a GitHub Repository.



3. Create a Jenkins Pipeline

a. Create a pipeline



Enter an item name

CICD-LAB-2 (JENKINSFILE)

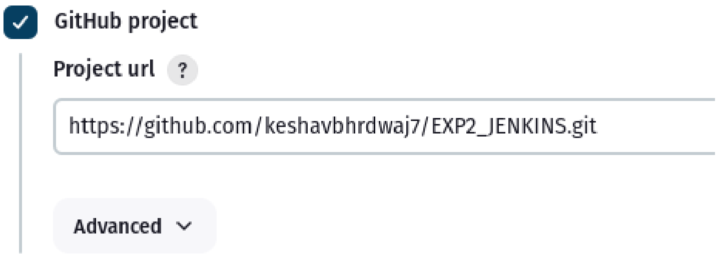
» *Required field*

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.

Maven project
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

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.

b. Add the project url

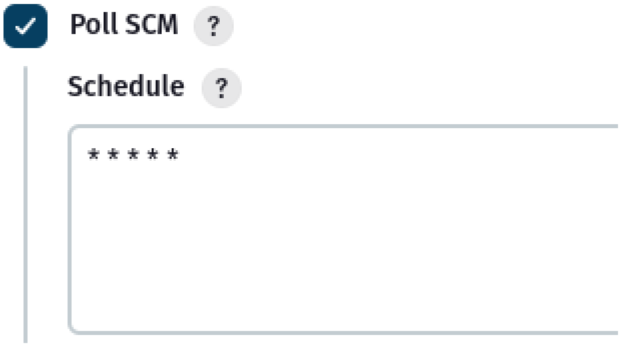


☑ GitHub project

Project url ?

https://github.com/keshavbhrdwaj7/EXP2_JENKINS.git

c. Configure the build trigger to poll scm and set it to perform build every minute

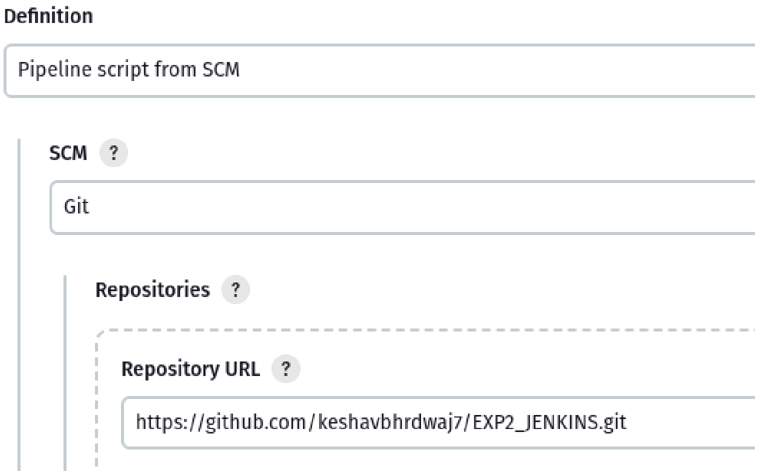


☑ Poll SCM ?

Schedule ?

```
* * * * *
```

d. Specify the path to Jenkinsfile in the project



Definition
Pipeline script from SCM

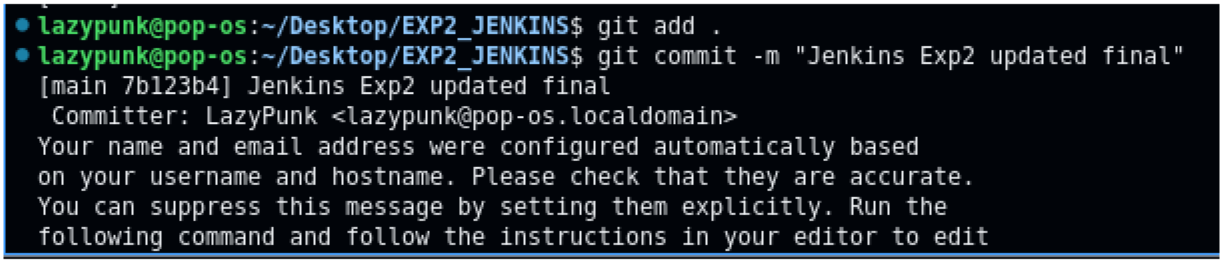
SCM
Git

Repositories

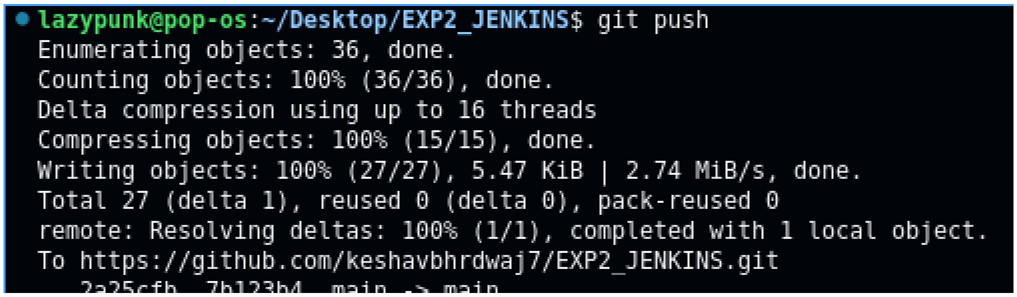
Repository URL
https://github.com/keshavbhrdwaj7/EXP2_JENKINS.git

e. Save the pipeline

4. Commit and push the changes of the project on GitHub

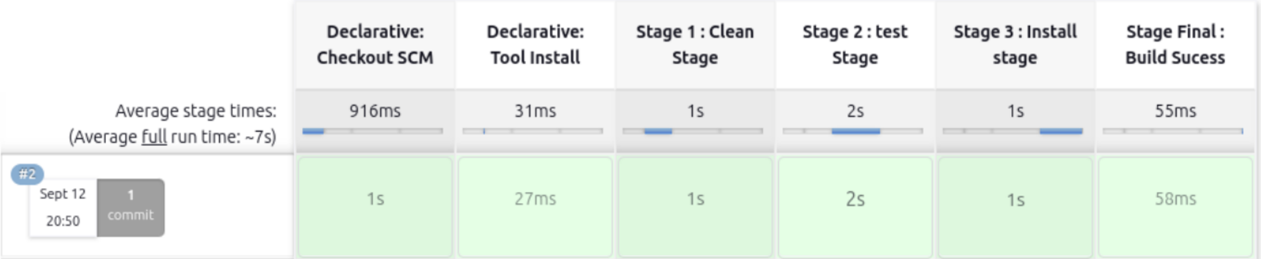


```
● lazypunk@pop-os:~/Desktop/EXP2_JENKINS$ git add .
● lazypunk@pop-os:~/Desktop/EXP2_JENKINS$ git commit -m "Jenkins Exp2 updated final"
  [main 7b123b4] Jenkins Exp2 updated final
   Committer: LazyPunk <lazypunk@pop-os.localdomain>
  Your name and email address were configured automatically based
  on your username and hostname. Please check that they are accurate.
  You can suppress this message by setting them explicitly. Run the
  following command and follow the instructions in your editor to edit
```



```
● lazypunk@pop-os:~/Desktop/EXP2_JENKINS$ git push
  Enumerating objects: 36, done.
  Counting objects: 100% (36/36), done.
  Delta compression using up to 16 threads
  Compressing objects: 100% (15/15), done.
  Writing objects: 100% (27/27), 5.47 KiB | 2.74 MiB/s, done.
  Total 27 (delta 1), reused 0 (delta 0), pack-reused 0
  remote: Resolving deltas: 100% (1/1), completed with 1 local object.
  To https://github.com/keshavbhrdwaj7/EXP2_JENKINS.git
     2a25cfb..7b123b4  main -> main
```

5. An automatic build is triggered



	Declarative: Checkout SCM	Declarative: Tool Install	Stage 1 : Clean Stage	Stage 2 : test Stage	Stage 3 : Install stage	Stage Final : Build Sucess
Average stage times: (Average full run time: ~7s)	916ms	31ms	1s	2s	1s	55ms
#2 Sept 12 20:50 1 commit	1s	27ms	1s	2s	1s	58ms