Qu1: Integration with Jenkins and GitHub Automate test execution via CI/CD.

Push a Selenium project to GitHub.

Setup a Jenkins pipeline to trigger on every push. Generate and email test reports using ExtentReports

Ans:

Create a selenium Test File in a Local repository Eg: sele.py:

import pytest

from selenium import webdriver

from selenium.webdriver.chrome.service import Service

from webdriver\_manager.chrome import ChromeDriverManager

def test\_open\_site():

driver = webdriver.Chrome(service=Service(ChromeDriverManager().install())) driver.get('https://selenium.dev/')

driver.quit()

Create a requirements.txt file with content:

selenium

webdriver-manager pytest

pytest-extent-reporter

Create a pytest.ini with content:

[pytest]

addopts =--extent-html=report.html

Create JenkinsFile with content:

pipeline {

agent any

stages {

stage('Checkout Code') { steps {

git branch: 'main', url: 'https://github.com/ShivRaiGithub/DevopsAssess.git'

}

}

stage('Install Dependencies') { steps {

sh 'pip3 install-r requirements.txt'

}

}

stage('Run Selenium Test') { steps {

sh 'pytest sele.py'

}

}

stage('Archive Test Report') {

steps {

archiveArtifacts artifacts: 'report.html', fingerprint: true

}

}

stage('Send Email') { steps {

emailext(

subject: 'Selenium Test Report', body: 'Test report.',

attachmentsPattern: 'report.html', to: ['sshaktirai@gmail.com'](mailto:%27sshaktirai@gmail.com)

)

}

}

}

}

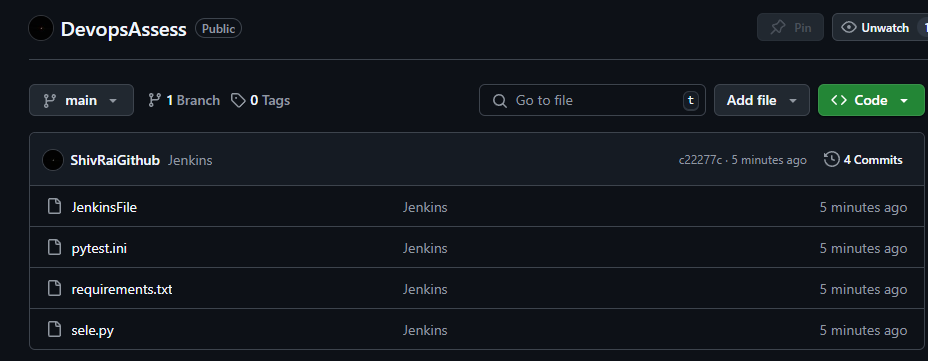
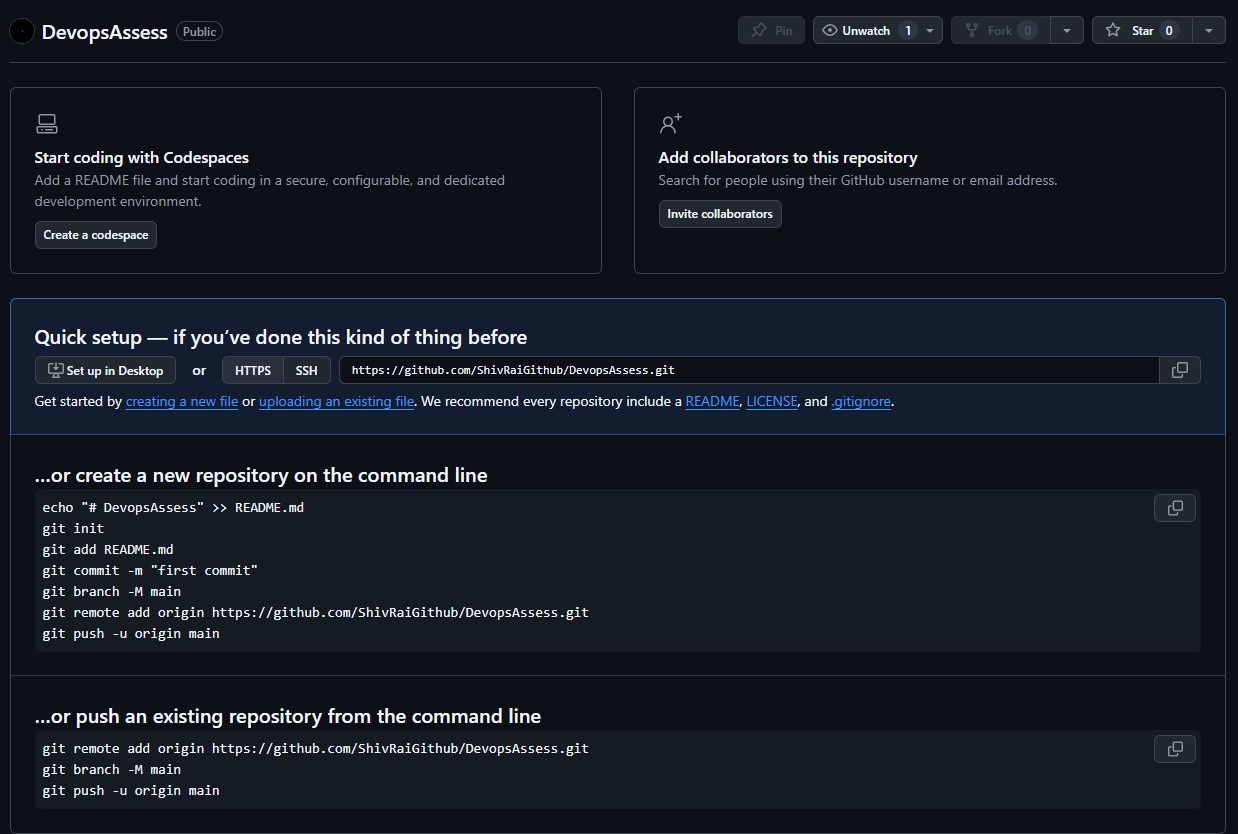
Create a new github repo

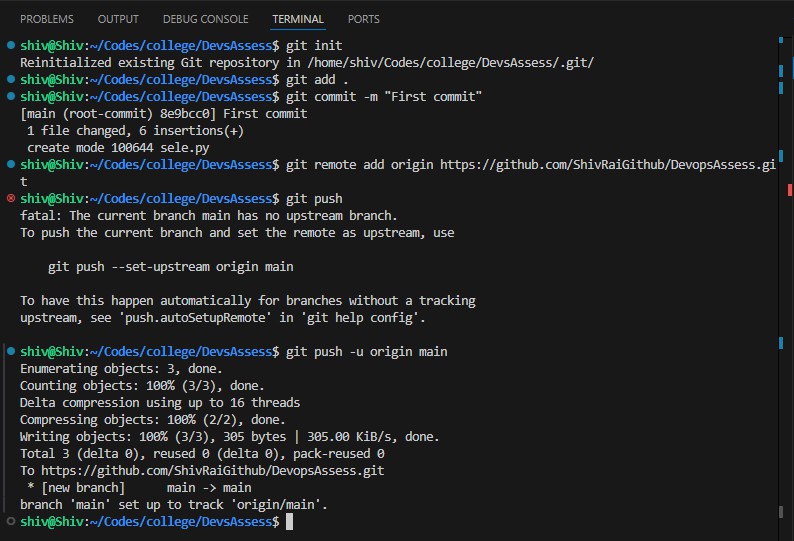
Push local repo to github using steps:

git init git add .

git commit-m "first commit" git branch-M main

git remote add origin https://github.com/ShivRaiGithub/DevopsAssess.git

git push-u origin main



Qu2: Version control basics and collaboration using Git.

Initialize a Git repository and perform basic commands (init, add, commit, status, log). Clone a repo, create a new branch, and simulate a merge conflict.

Resolve the conflict and push changes to GitHub.

Ans:

Create a folder

Create a new file file.txt with content:

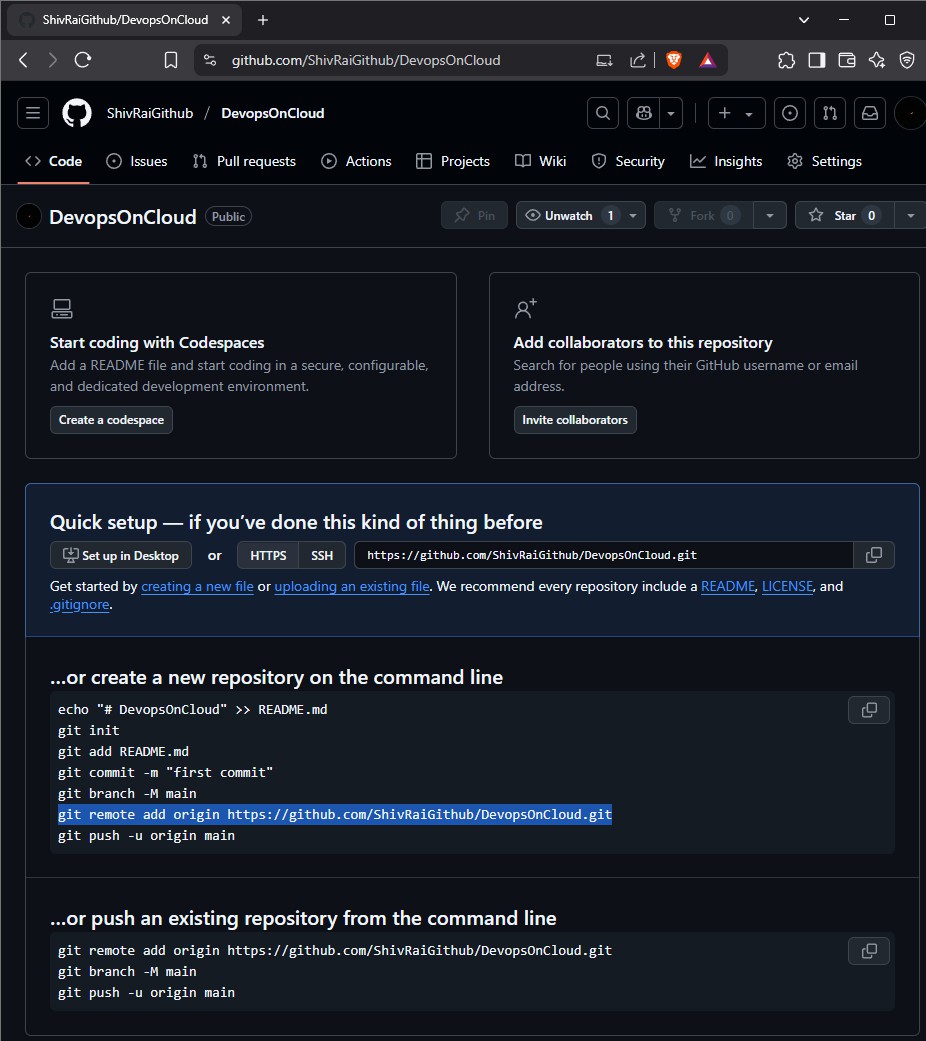
Hello world Welcome to Devops

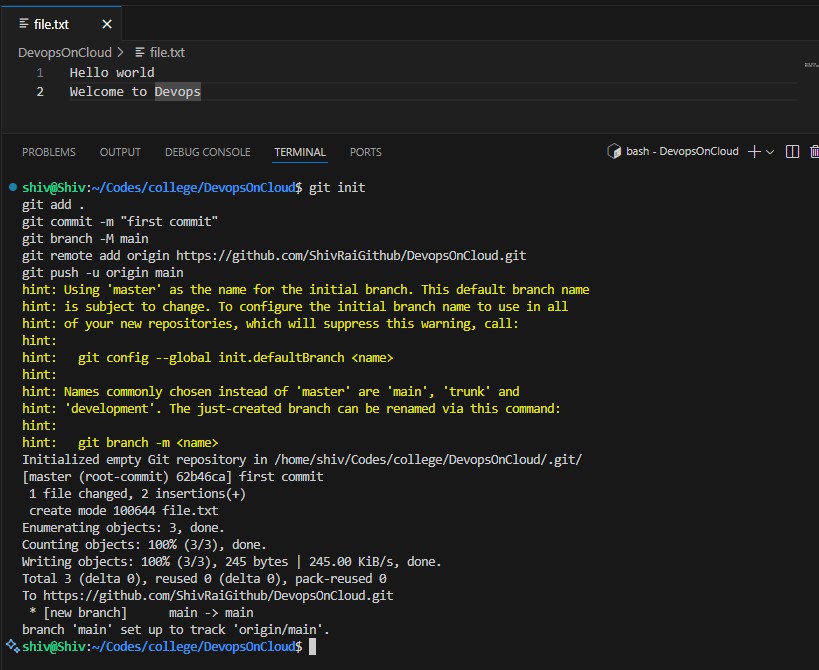
Push to Github using commands:

git init git add .

git commit-m "first commit" git branch-M main

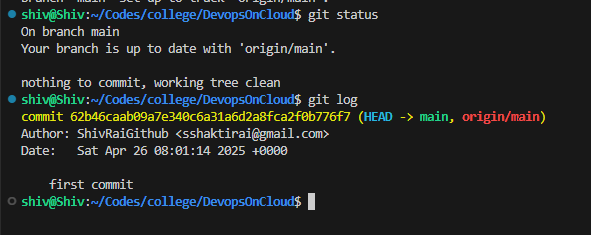
git remote add origin https://github.com/ShivRaiGithub/DevopsOnCloud.git git push-u origin main



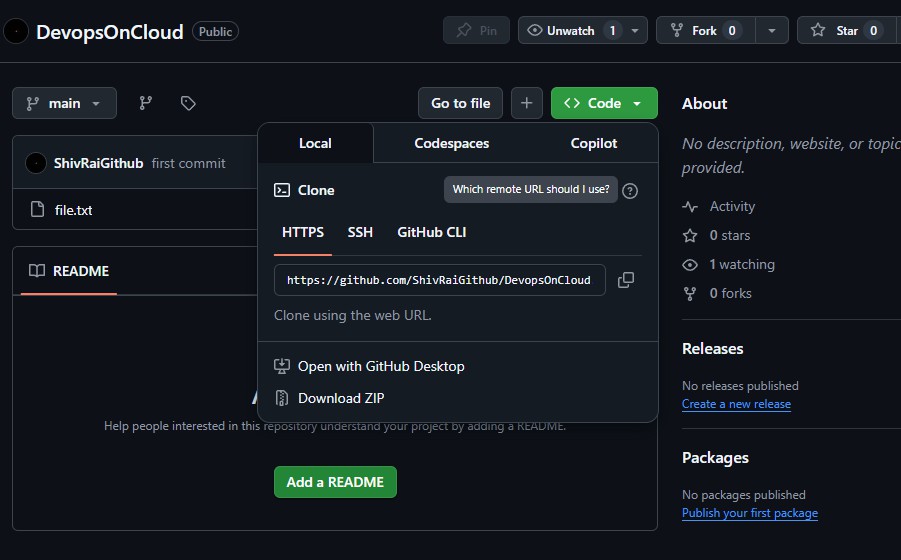


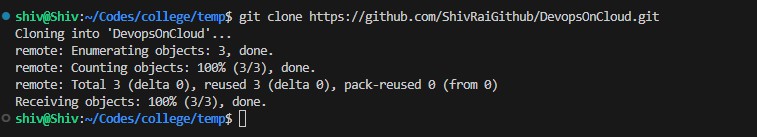
Use following commands to check the status and previous commits:

git status git log

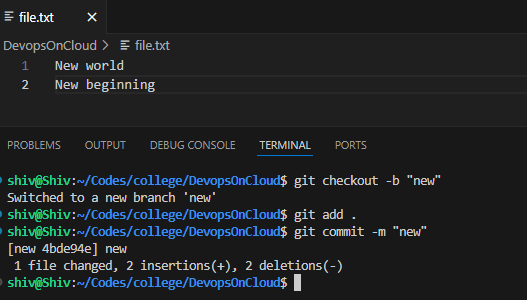


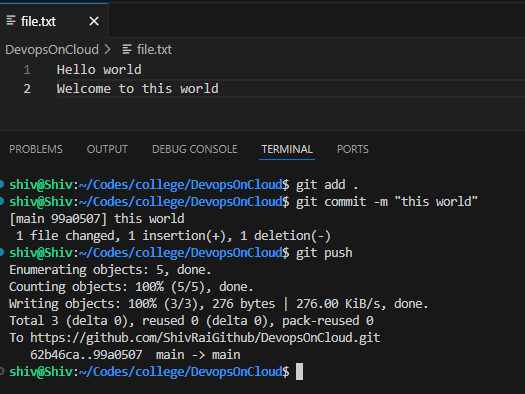
Use the following to clone:

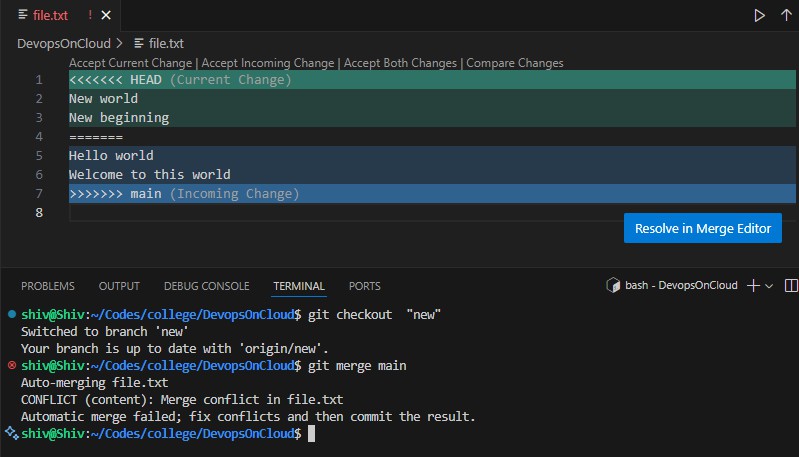
git clone https://github.com/ShivRaiGithub/DevopsOnCloud.git



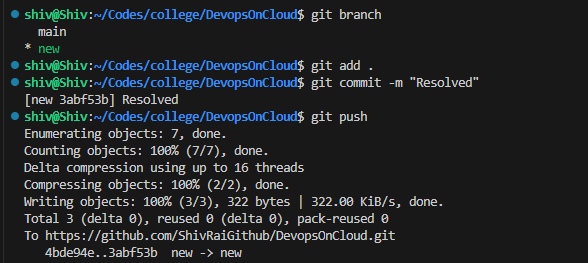
Make a new branch, do some changes and commit them:



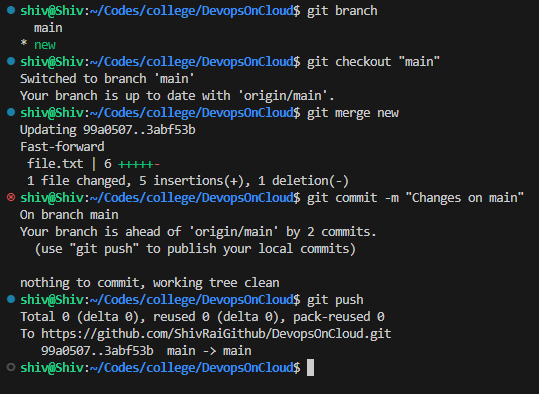
On the main branch do some changes and commit them:

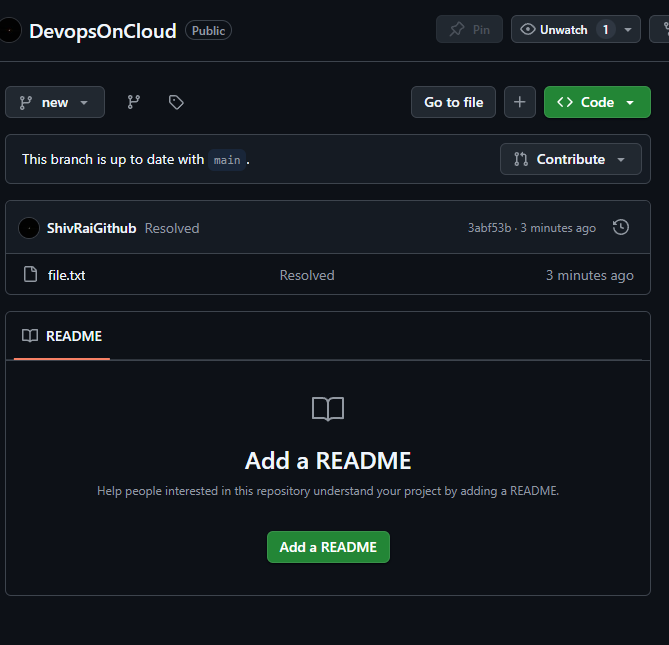
Switch to new branch and merge the new branch with main. Conflicts will occur. Resolve them

After resolution, push the changes



Push changes of main branch too:





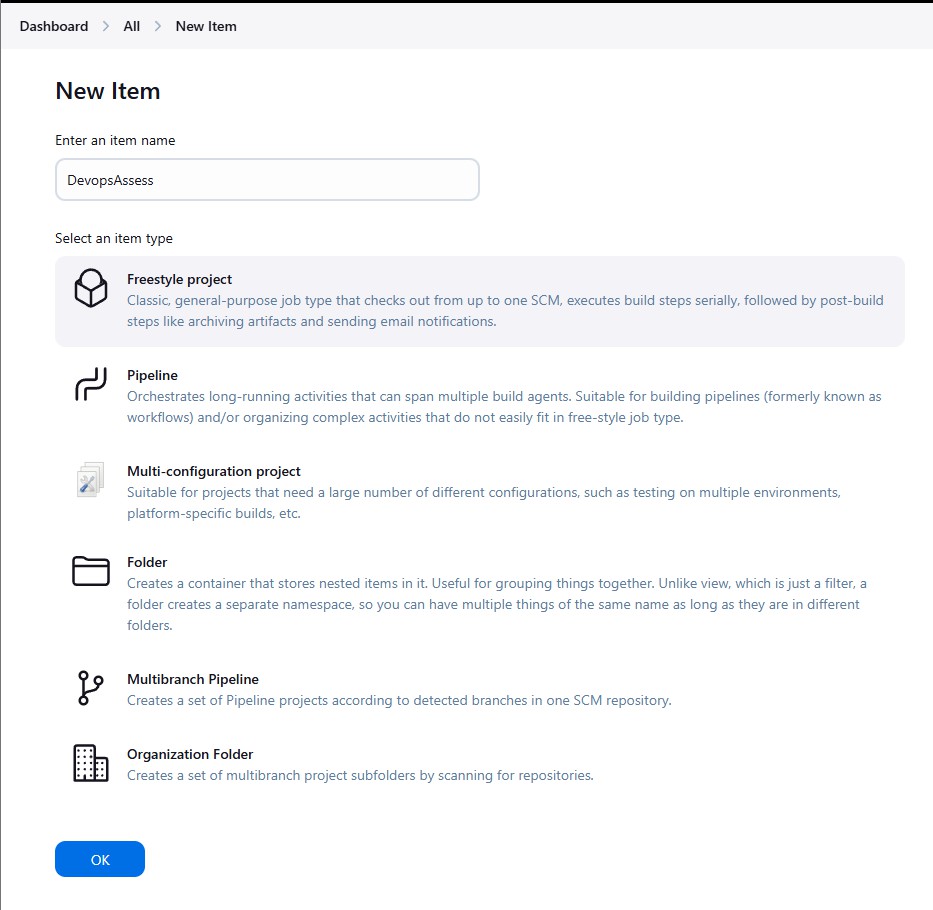
Qu3: Automate pulling code from GitHub and building it. Link Jenkins with your GitHub repo.

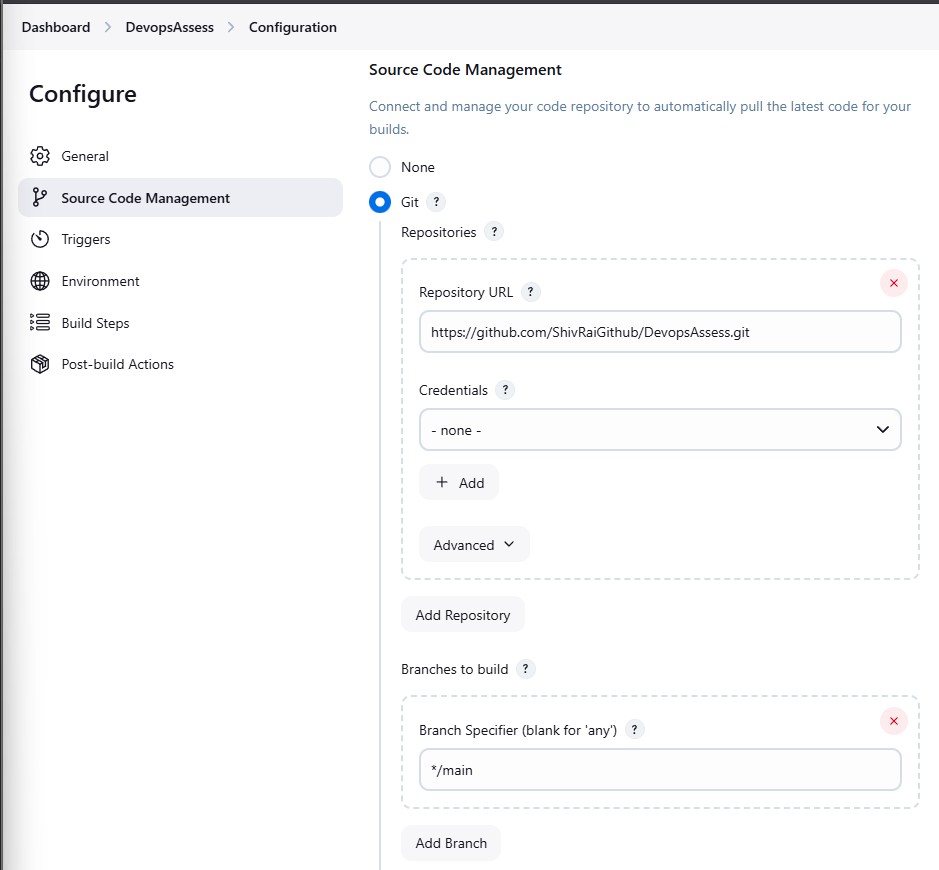
Use Git plugin in the Source Code Management section.

Add a shell script in the build step to print a list of files (ls -la). Trigger the job manually.

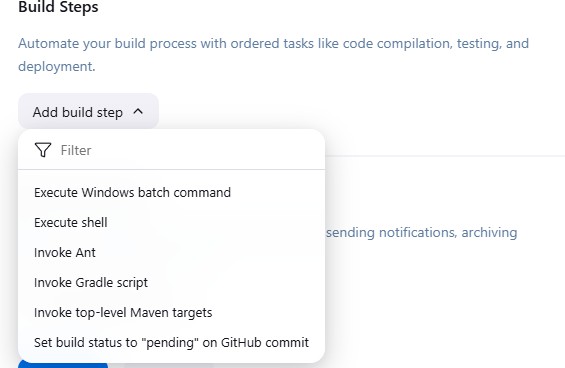
Ans:

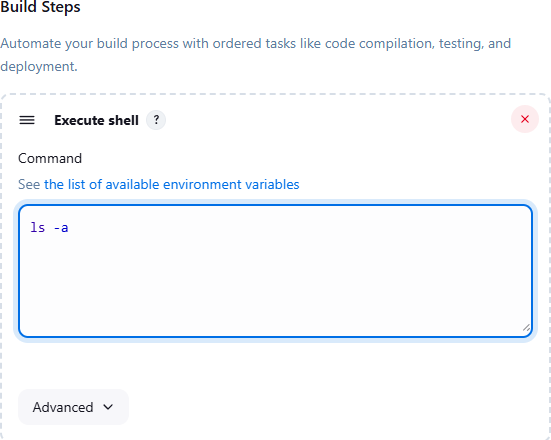
Go to Jenkins Dashboard Make a New Item:



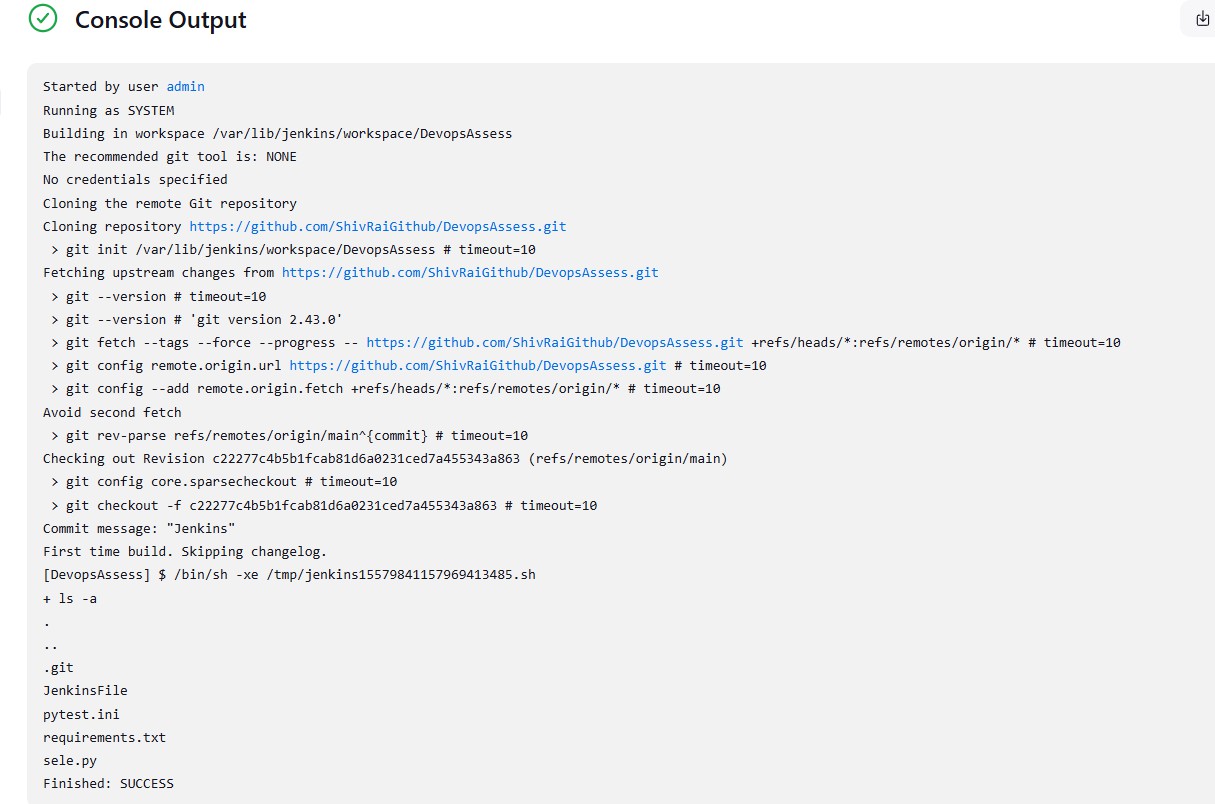
In source code management of Configuration, select git and provide github repo details:

Add the build command to show the files in Build Steps:





Save the Configuration.

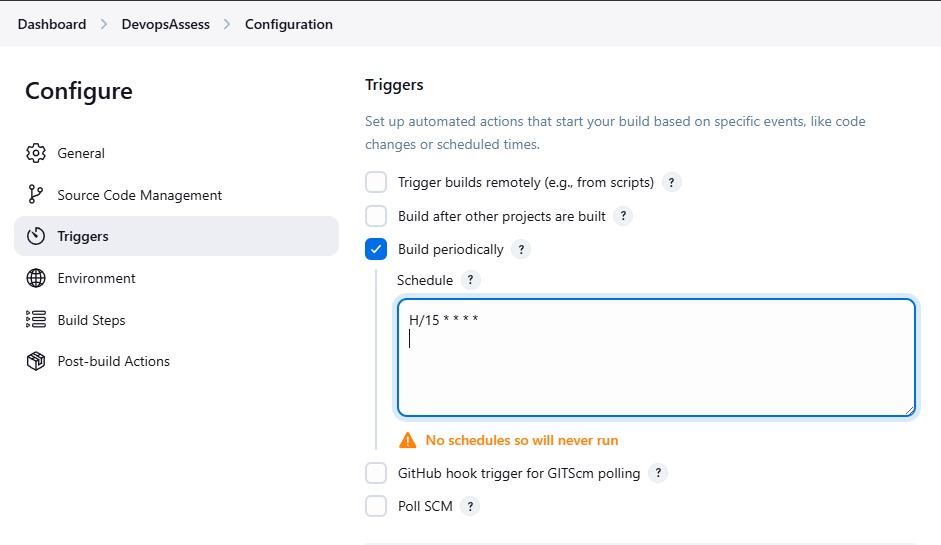
Go to dashboard, click on job, click on Build Now. Check console logs.

Qu4: Scheduled Jobs (CRON Trigger) Automate job execution at intervals.

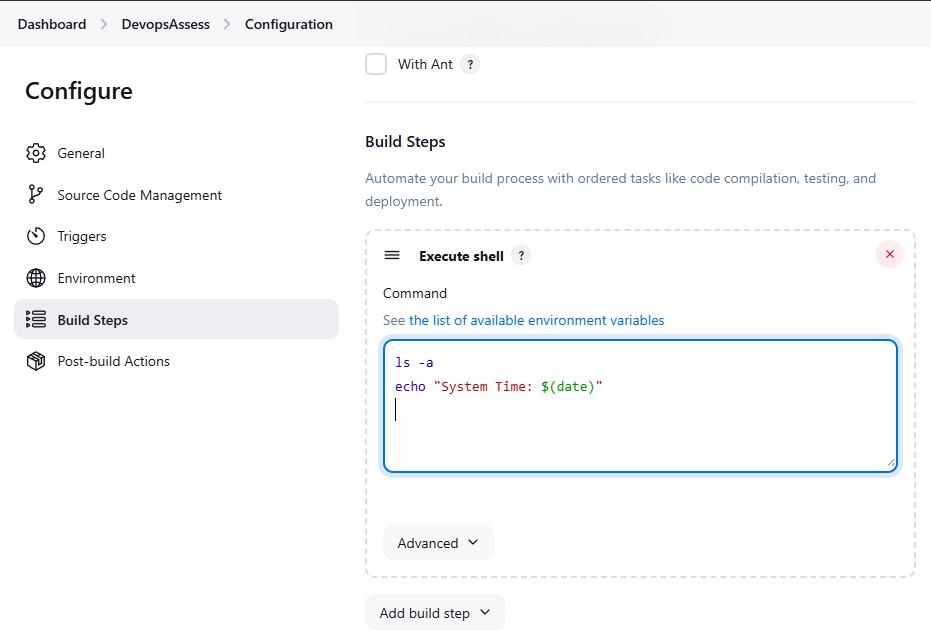
Set Build Trigger using Jenkins’ cron syntax (H/15 \* \* \* \* for every 15 mins). Print the system time during execution to confirm the job is running.

Ans:

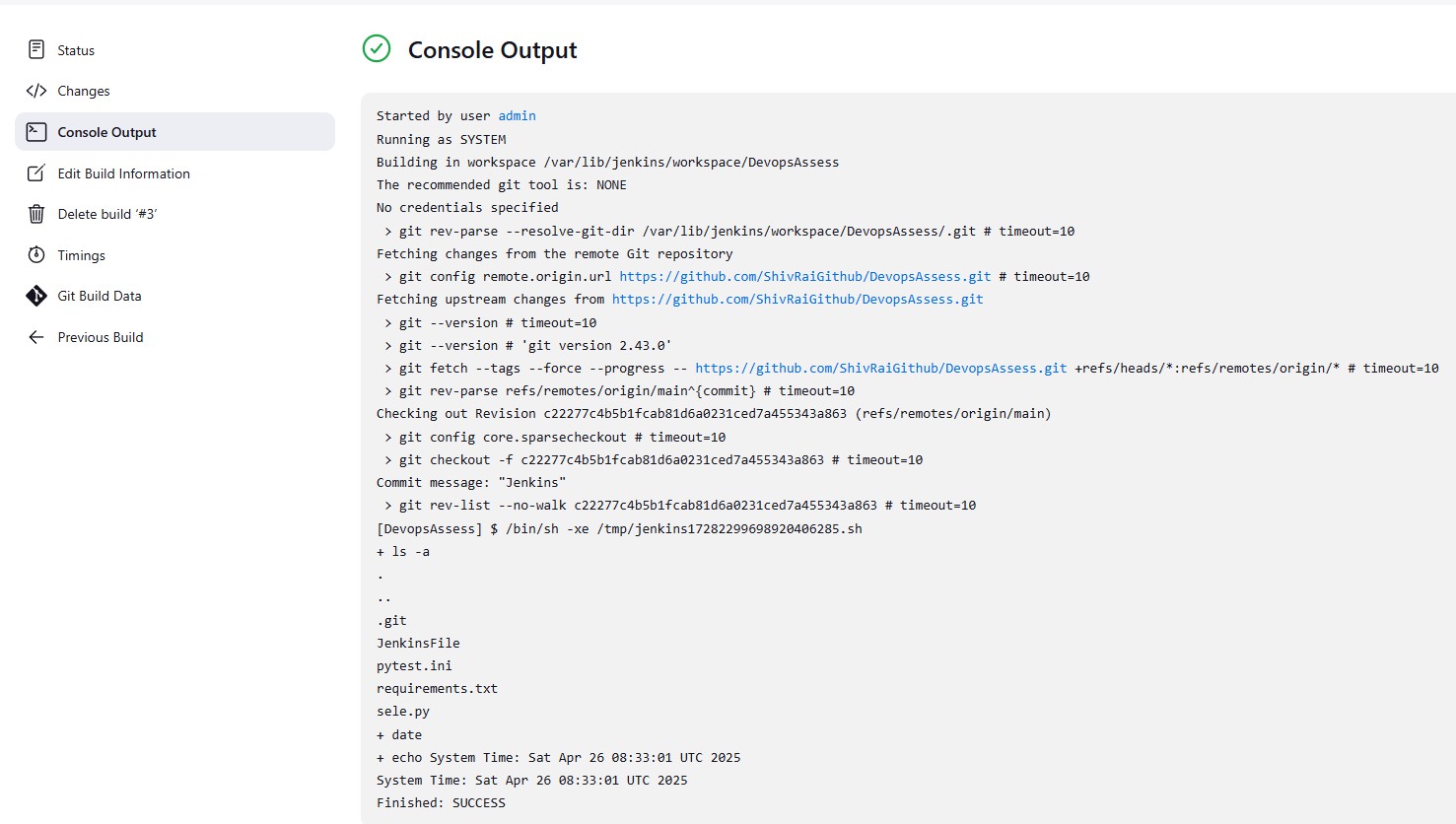
Go to Jenkins Dashboard, select job.

Go to Configure, select Build Periodically under Triggers. Put: H/15 \* \* \* \*

Within Build Steps, add line: echo "System Time: $(date)"



Save the configuration.

The output of next build will show the files of the repo as well as system time