

SARANATHAN COLLEGE OF ENGINEERING

(An Autonomous Institution)

Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai
Venkateswara Nagar, Panjappur, Tiruchirappalli-620012



Name: _____

Register No.: _____

Semester and Branch: _____

Subject Code and Name: _____

Bonafide record of the work done

in the laboratory during the period

_____ to _____

Head of the Department

Faculty in-charge

Date:

Submitted for the practical examination held at **SARANATHAN
COLLEGE OF ENGINEERING, Tiruchirappalli** on _____

Internal Examiner

External Examiner

TABLE OF CONTENTS

Ex. No.	Date	Experiment Name	Page No.	Signature
1.		Create Maven Build Pipeline in Azure	1	
2.		Run Regression Tests Using Maven Build Pipeline in Azure	9	
3.		Install Jenkins in Cloud	17	
4.		Create CI Pipeline using Jenkins	34	
5.		Create CD Pipeline using Jenkins and Deploy in Cloud	52	
6.		Create an Ansible Playbook for a Simple Web Application Infrastructure	55	
7.		Build a Simple Application Using Gradle	57	
8.		Install Ansible and Configure Ansible Roles and to Write Playbooks	62	

Staff In-charge



SARANATHAN COLLEGE OF ENGINEERING

An Autonomous Institution
Venkateswara Nagar, Panjappur
Tiruchirappalli – 620012

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Vision of the Department:

To evolve as a centre of academic excellence and advanced research in Computer Science and Engineering discipline.

Mission of the Department:

1. To inculcate in students a profound understanding of fundamentals related to discipline.
2. To inculcate skills, attitudes and their applications in solving real world problems with an inclination towards societal issues and research.
3. To promote research in the emerging areas of computer science and technology

Program Educational Objectives: (PEOs)

- PEO1:** Acquire strong foundation in the mathematical, scientific and engineering fundamentals necessary to formulate, solve and analyze engineering problems.
- PEO2:** Develop the ability to analyze the requirements of the software, understand the technical specifications, design and provide novel engineering solutions and efficient software/hardware designs.
- PEO3:** Have exposure to emerging cutting edge technologies, adequate training & opportunities to work as teams on multidisciplinary projects with effective communication skills and leadership qualities.
- PEO4:** Have awareness on the life-long learning and prepare them for research development and consultancy.
- PEO5:** Have a successful career and work with values & social concern bridging the digital divide and meet the requirements of Indian and multinational companies.

Program Specific Objectives: (PSOs)

- PSO1:** Foundation of mathematical concepts: To use mathematical methodologies to crack problem using suitable mathematical analysis, data structure and suitable algorithm.
- PSO2:** Foundation of Computer System: the ability to interpret the fundamental concepts and methodology of computer systems. Students can understand the functionality of hardware and software aspects of computer systems.
- PSO3:** Foundations of Software development: the ability to grasp the software development lifecycle and methodologies of software systems. Possess competent skills and knowledge of software design process. Familiarity and practical proficiency with a broad area of programming concepts and provide new ideas and innovations towards research

Program Outcomes: (POs)

S.No.	Program Outcomes
1	PO1: Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and engineering specialization to the solution of complex engineering problems.
2	PO2: Problem Analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3	PO3: Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4	PO4: Conduct Investigations of Complex Problems Use research-based knowledge and research methods including design of exercises, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5	PO5: Modern Tool Usage Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6	PO6: The Engineer and Society Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7	PO7: Environment and Sustainability Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8	PO8: Ethics Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9	PO9: Individual and Team Work Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10	PO10: Communication Communicate effectively on complex engineering activities with the engineering community and with society at large, communication such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11	PO11: Project management and finance Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12	PO12: Life-long learning Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change



SARANATHAN COLLEGE OF ENGINEERING

An Autonomous Institution

Approved by AICTE-New Delhi, Affiliated Anna University, Chennai
Venkateswara Nagar, Panjappur, Tiruchirappali -620012

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

2025-2026 ODD SEMESTER

Name of Lab Course	CCS342 / DEVOPS
Semester & Year	VII & IV
Name of the student	
Name of the Evaluator	Ms E Shapna Rani
Marks scored out of 20	

RUBRIC ASSESSMENT FOR LAB COURSES

Performance Indicators	Level 1 (0-1)	Level 2 (2-4)	Level 3 (5)
Pre lab questions, objectives (P-I)	Explanation to, the Pre lab questions and objectives of the experiment, is, where compared to the expectation of the faculty is not satisfactory.	Explanation to, the Pre lab questions and objectives of the experiment, is, where compared to the expectation of the faculty is partially satisfactory.	Explanation to, the Pre lab questions and objectives of the experiment, is, where compared to the expectation of the faculty is highly satisfactory.
Procedures (P-II)	Explanation to the procedure of the experiment, is, where compared to the expectation of the faculty is not satisfactory.	Explanation to the procedure of the experiment, is, where compared to the expectation of the faculty is partially satisfactory.	Explanation to the procedure of the experiment, is, where compared to the expectation of the faculty is highly satisfactory.
Data/ Observations (P-III)	Calculation of the observed values and validation of the results of the experiment inaccurate.	Calculation of the observed values and validation of the results of the experiment approximate	Calculation of the observed values and validation of the results of the experiment precise.
Post lab questions, Conclusions (P-IV)	Explanation to the Post lab questions and Conclusions of the experiment, is, where compared to the expectation of the faculty is not satisfactory.	Explanation to the Post lab questions and Conclusions of the experiment is, where compared to the expectation of the faculty partially satisfactory.	Explanation to the Post lab questions and Conclusions of the experiment, is, where compared to the expectation of the faculty is highly satisfactory.

Register No: _____

ASSESSMENT SHEET

S.No.	NAME OF THE EXERCISE	P-I (5)	P-II (5)	P-III (5)	P-IV (5)	Total (20)
1	Create Maven Build Pipeline in Azure					
2	Run Regression Tests Using Maven Build Pipeline in Azure					
3	Install Jenkins in Cloud					
4	Create CI Pipeline using Jenkins					
5	Create CD Pipeline using Jenkins and Deploy in Cloud					
6	Create an Ansible Playbook for a Simple Web Application Infrastructure					
7	Build a Simple Application Using Gradle					
8	Install Ansible and Configure Ansible Roles and to Write Playbooks					
TOTAL (OUT OF 20)						

SIGNATURE OF EVALUATOR

Course Outcomes:

CO Code	Course Outcomes
CO1	Understand different actions performed through Version control tools like Git.
CO2	Perform Continuous Integration and Continuous Testing and Continuous Deployment using Jenkins by building and automating test cases using Maven.
CO3	Perform Continuous Integration and Continuous Testing and Continuous Deployment using Jenkins by building and automating test cases using Gradle.
CO4	Ability to Perform Automated Continuous Deployment.
CO5	Ability to do configuration management using Ansible.
CO6	Understand to leverage Cloud-based DevOps tools using Azure DevOps.

CO-PO Mapping:

CO Code	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
CO1	3	3	3	2	3	-	-	-	-	-	-	-	2	2	2
CO2	3	3	3	2	3	-	-	-	-	-	-	-	2	2	2
CO3	3	3	3	2	3	-	-	-	-	-	-	-	2	2	2
CO4	3	3	3	2	3	-	-	-	-	-	-	-	2	2	2
CO5	3	3	3	2	3	-	-	-	-	-	-	-	2	2	2
CO6	3	3	3	2	3	-	-	-	-	-	-	-	2	2	2

CCS342 – DEVOPS

Syllabus

1. Create Maven Build pipeline in Azure
2. Run regression tests using Maven Build pipeline in Azure
3. Install Jenkins in Cloud
4. Create CI pipeline using Jenkins
5. Create a CD pipeline in Jenkins and deploy in Cloud
6. Create an Ansible playbook for a simple web application infrastructure
7. Build a simple application using Gradle
8. Install Ansible and configure ansible roles and to write playbooks

Ex. No: 1

CREATE MAVEN BUILD PIPELINE IN AZURE

Date:

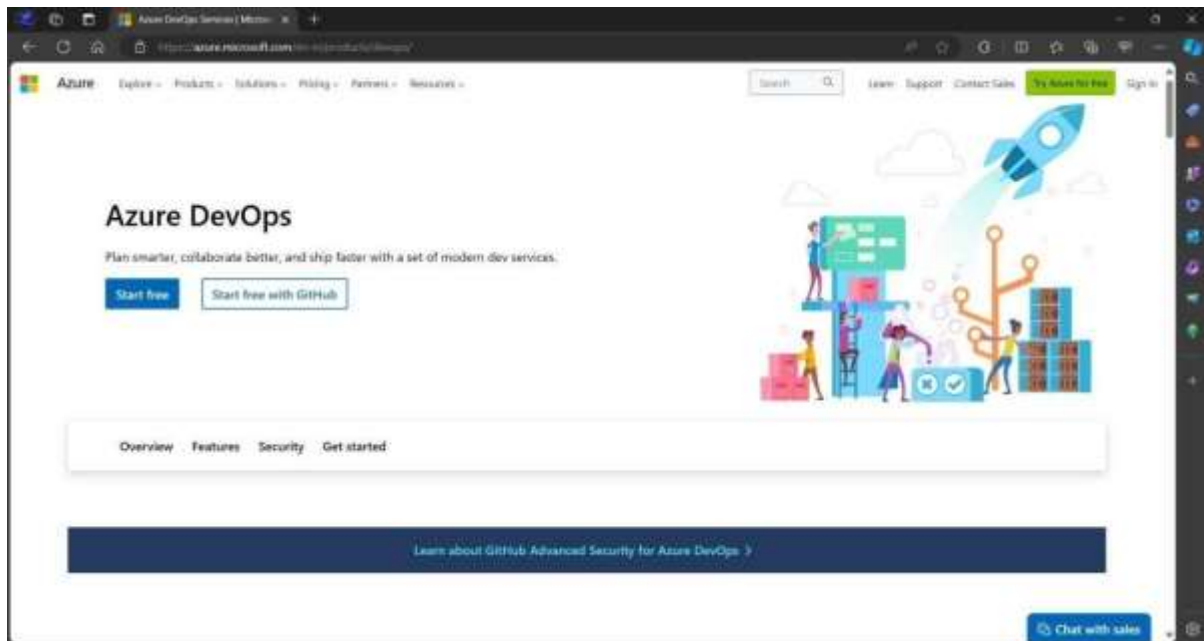
AIM

To build a maven build pipeline in Azure.

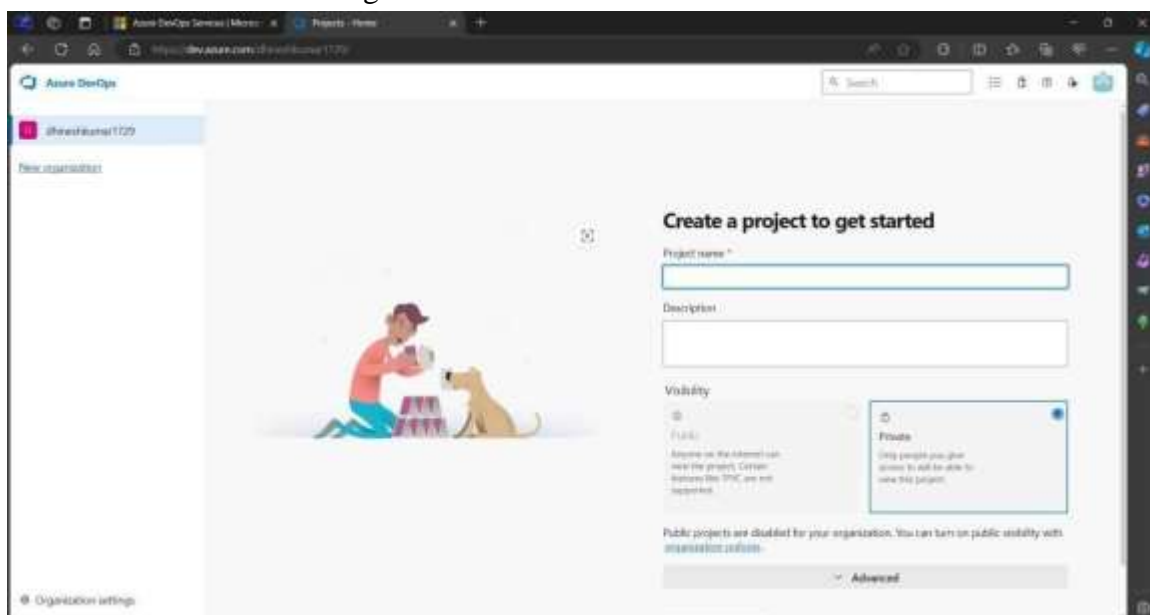
PROCEDURE

STEP 1: Create an account on the Azure DevOps website.

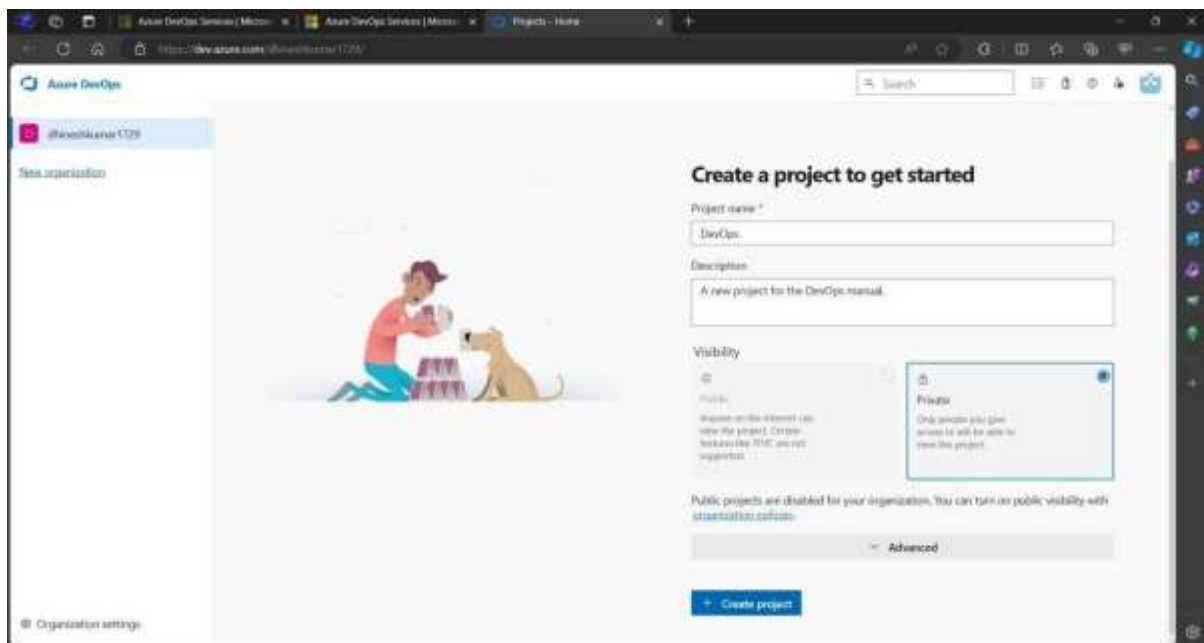
1. Go to <https://azure.microsoft.com/en-in/products/devops/> website and sign in using a GitHub account.



2. Create a new organization with a suitable name of your choice. Here dhineshkumar1729 is the organization name.



3. Create a new project named **DevOps** in private mode.



STEP 2: Installing IntelliJ IDEA

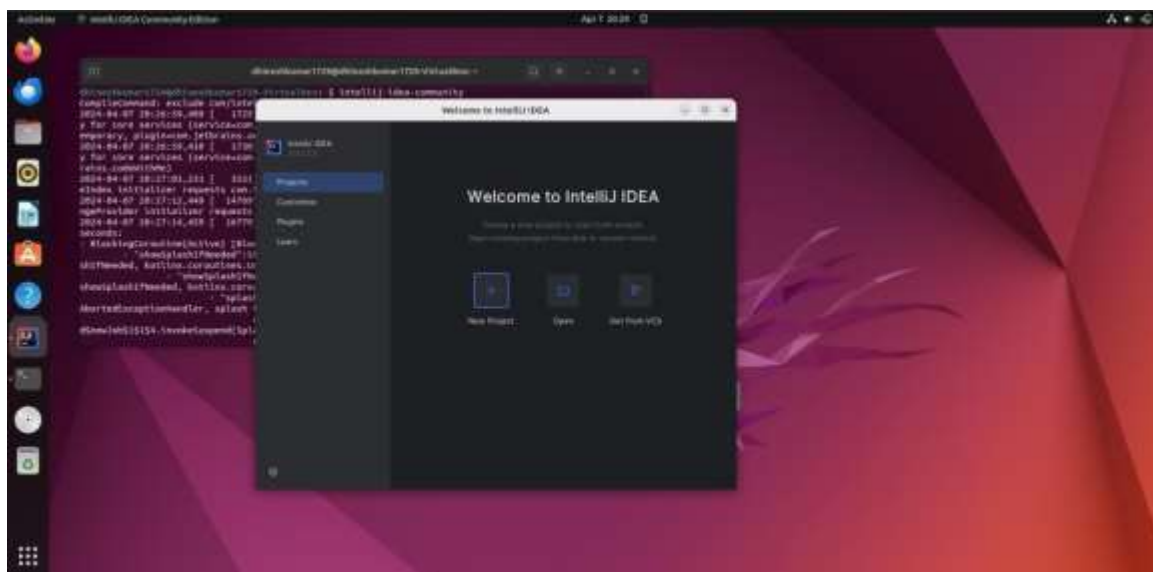
1. Download IntelliJ Idea from <https://www.jetbrains.com/idea/download/?section=linux> and activate a free trial for 30 days or through the following command.

\$ sudo snap install intellij-idea-community --community

2. Open the IDE using the below command in the terminal.

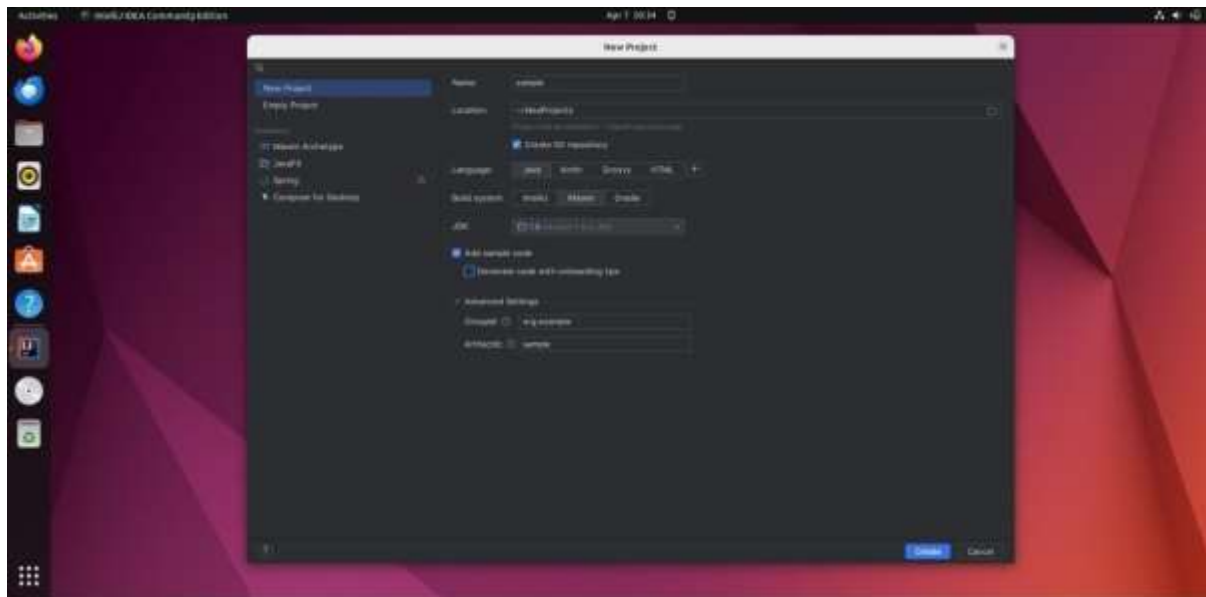
\$ intellij-idea-community

3. Java is a pre-requisite for using this IDE



STEP 3: Create a new Maven project using the IDE.

1. Create a project by clicking on the New Project option, choose an appropriate name and select the following features as mentioned in the image (Build system: Maven and Language: Java)



1. A sample code has been generated. Now press ALT + F12 to open a terminal inside the IDE.

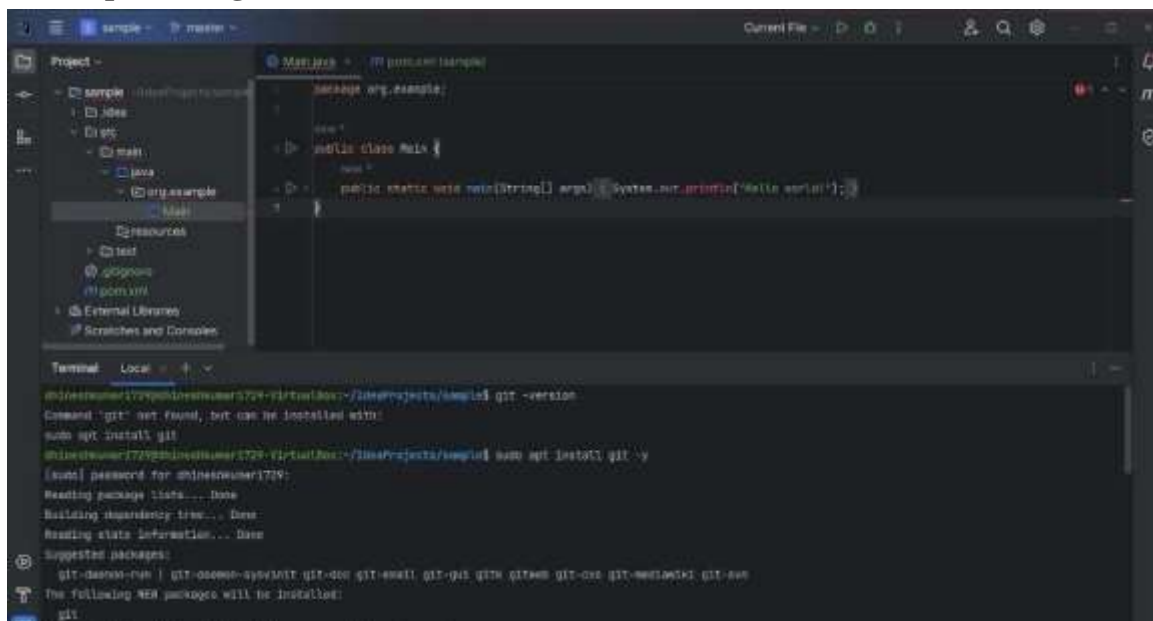
STEP 4: Install GIT and configure it

1. Check whether git is installed using the following command.

\$ git -version

2. Install git using

\$ sudo apt install git



3. Once git has been installed execute the following commands one by one.

```
$ git init
```

```
$ git add .
```

```
$ git commit -m "First Commit"
```

```
$ git branch -M main
```

1. Configure git using the below commands

```
$ git config --global user.name "Your_Name"
```

```
$ git config --global user.email "Your_Email_ID"
```

1. Go to <https://github.com> sign in with your account and create a new private repository with the same name as your maven project.



1. Copy the **SSH** URL for the created repository.

2. Now go back to IDE's terminal and execute the following commands

```
$ ssh-keygen -t rsa -b 4096 -C dhineshkumar24murugan007@gmail.com
```

(press ENTER for all questions)

```
$ cat ~/.ssh/id_rsa.pub (copy the printed SSH key)
```

1. Now go to the **SSH and GPG** section in the GitHub settings option.

2. Click on the **New SSH key** button.

3. Choose a suitable name and paste the SSH key into the provided space.

4. Now get back to the IDE's terminal and type the following command

```
$ git remote set-url origin git@github.com:dk172923/sample.git
```

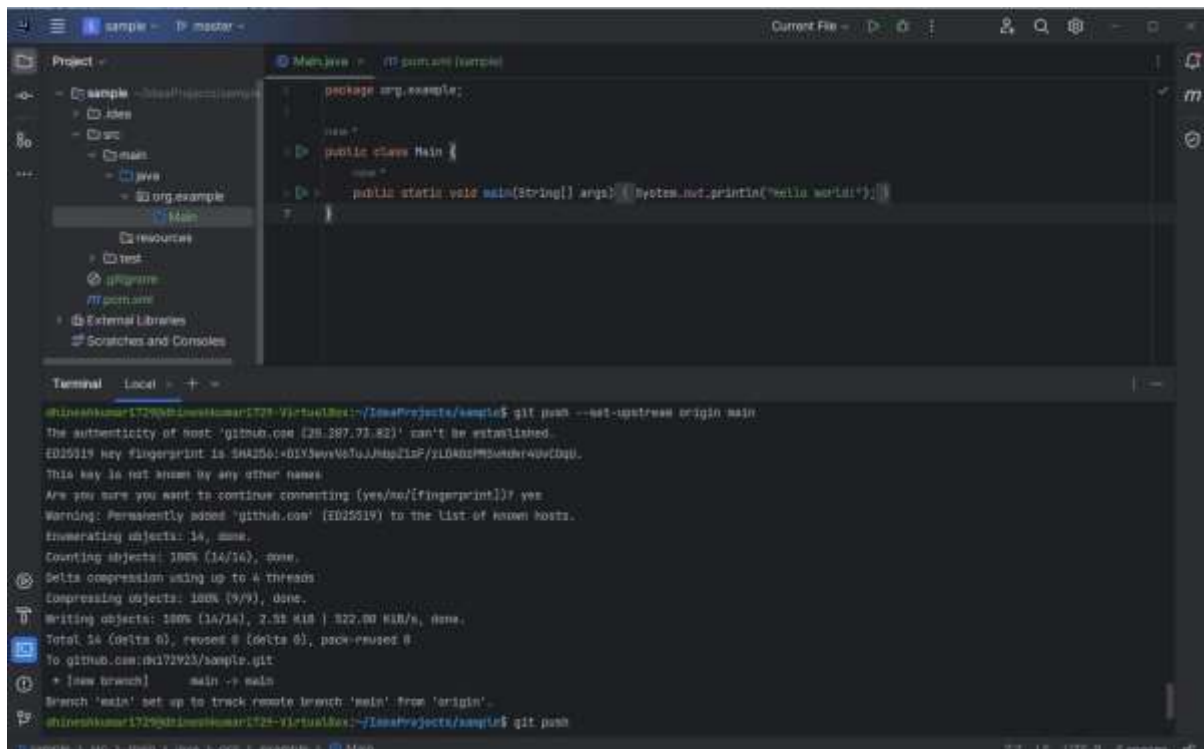
(Here, [git@github.com:dk172923/sample.git](ssh://git@github.com:dk172923/sample.git) is the remote URL to use SSH
dk172923 – GitHub ID

sample.git – repository name)

1. Now finally run the push commands to push the code to the remote repository from the local repository.

```
$ git push --set-upstream origin main (Type YES for prompted question)
```

```
$ git push
```



```
package org.example;

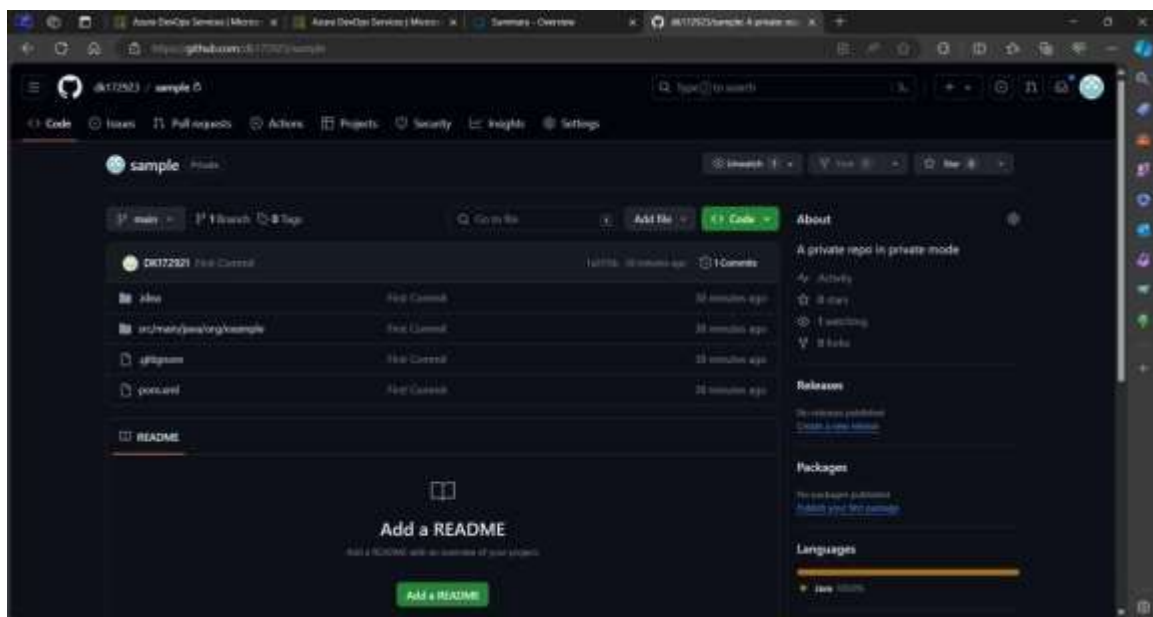
import java.io.*;

public class Main {

    public static void main(String[] args) {
        System.out.println("hello world!");
    }
}
```

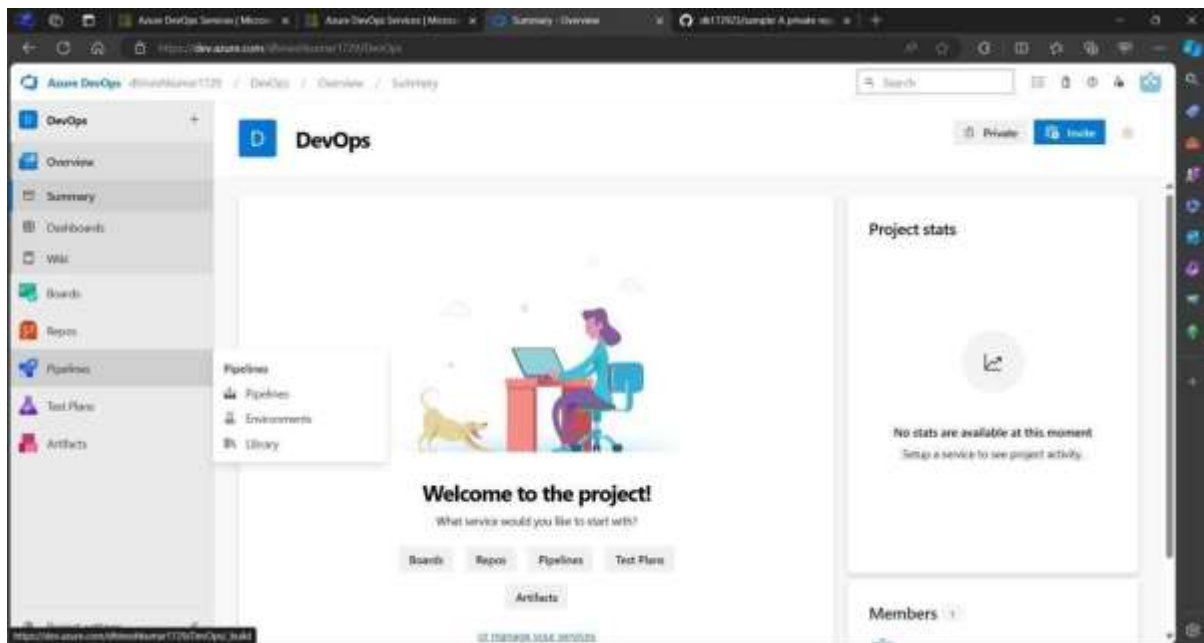
```
shinesh@kali:~/Documents/sample$ git push --set-upstream origin main
The authenticity of host 'github.com [20.207.73.82]' can't be established.
ED25519 key fingerprint is SHA256:01359v6t0Jh4ZiF/zLD8zP80u8dR40v4bq.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
Enumerating objects: 14, done.
Counting objects: 100% (14/14), done.
Delta compression using up to 4 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (14/14), 2.55 MiB | 322.00 KiB/s, done.
Total 14 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:shin3sh/sample.git
+ [new branch] main -> main
Branch 'main' set up to track remote branch 'main' from 'origin'.
shinesh@kali:~/Documents/sample$ git push
```

The code has been successfully pushed to the remote repository.



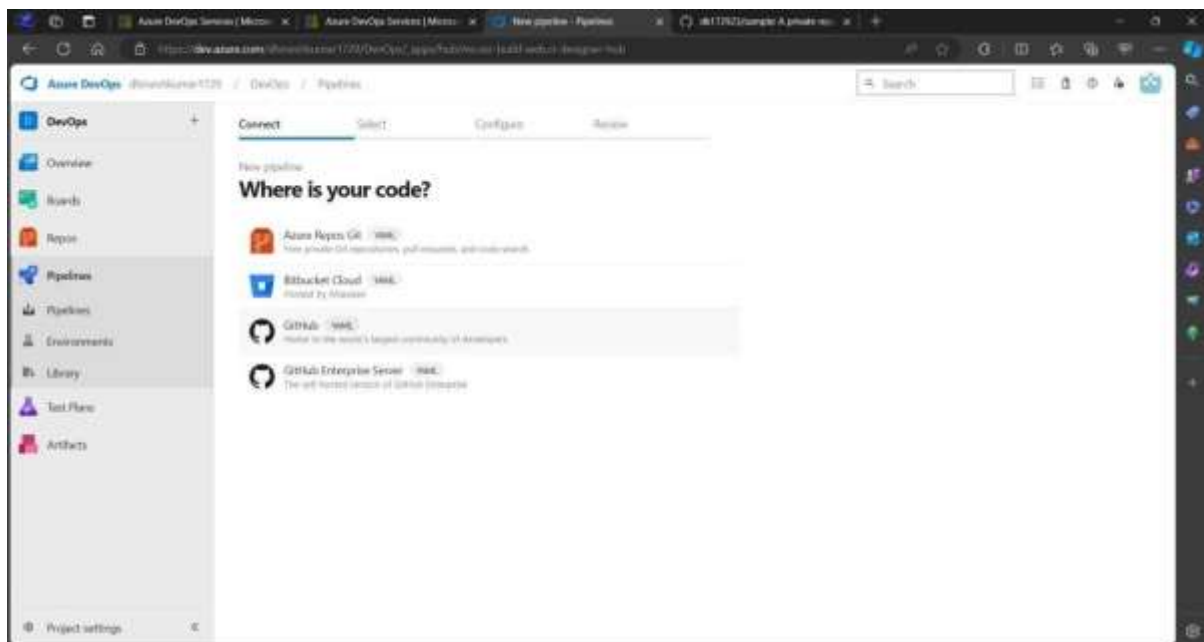
STEP 5: Create a pipeline for the created maven repository.

1. Click on Pipelines on the left pane of the Azure DevOps website.



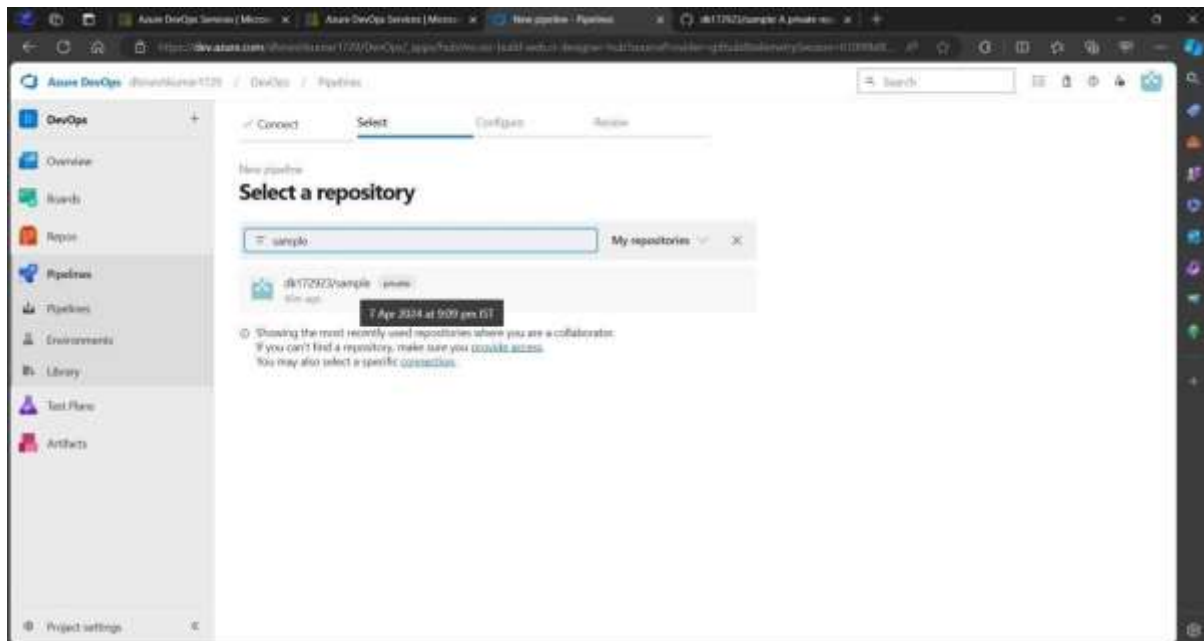
1. Click on the **Create Pipeline** button.

2. Choose the GitHub YAML option.

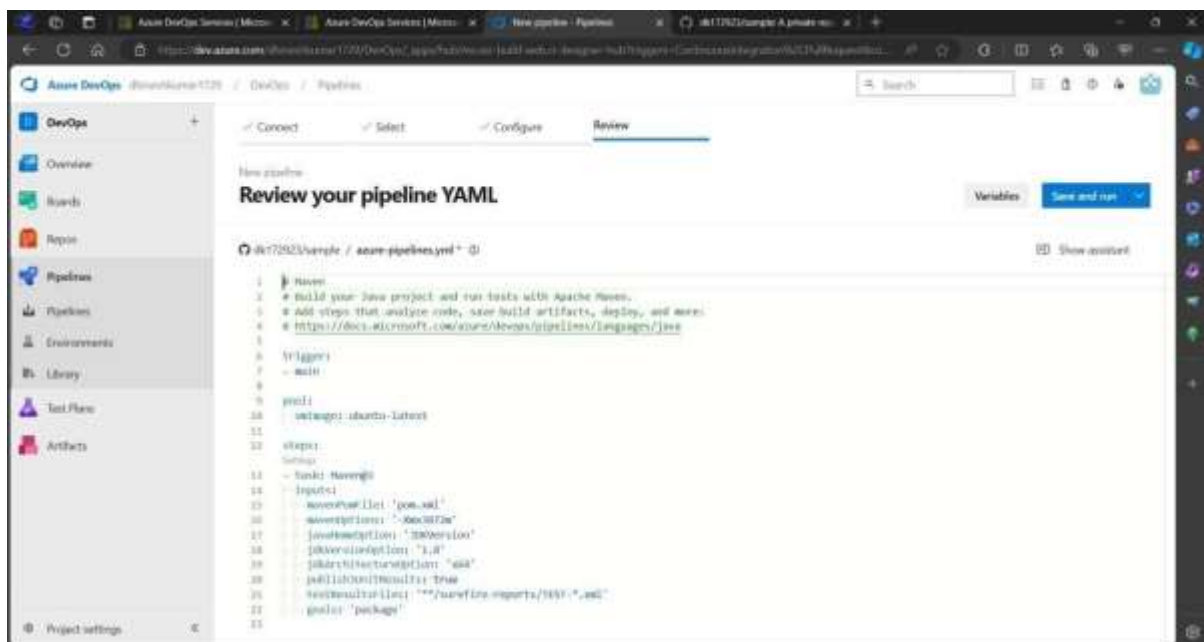


1. Select the created repository “sample” from the available repositories.

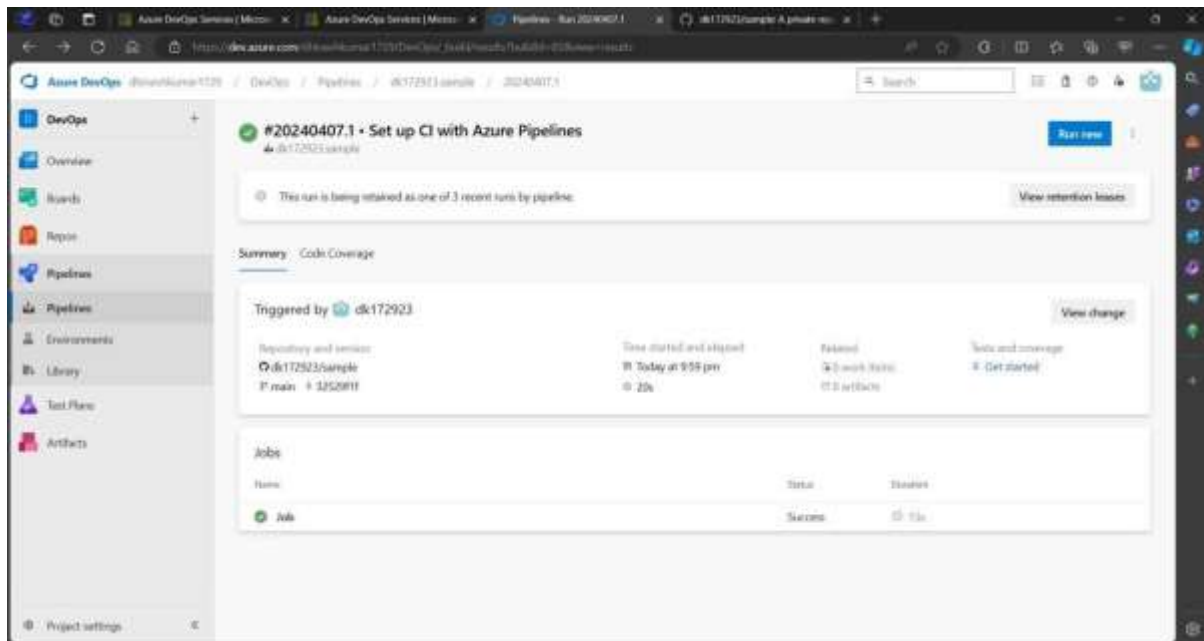
2. Give permissions if prompted to do so and sign in to your GitHub account if needed.



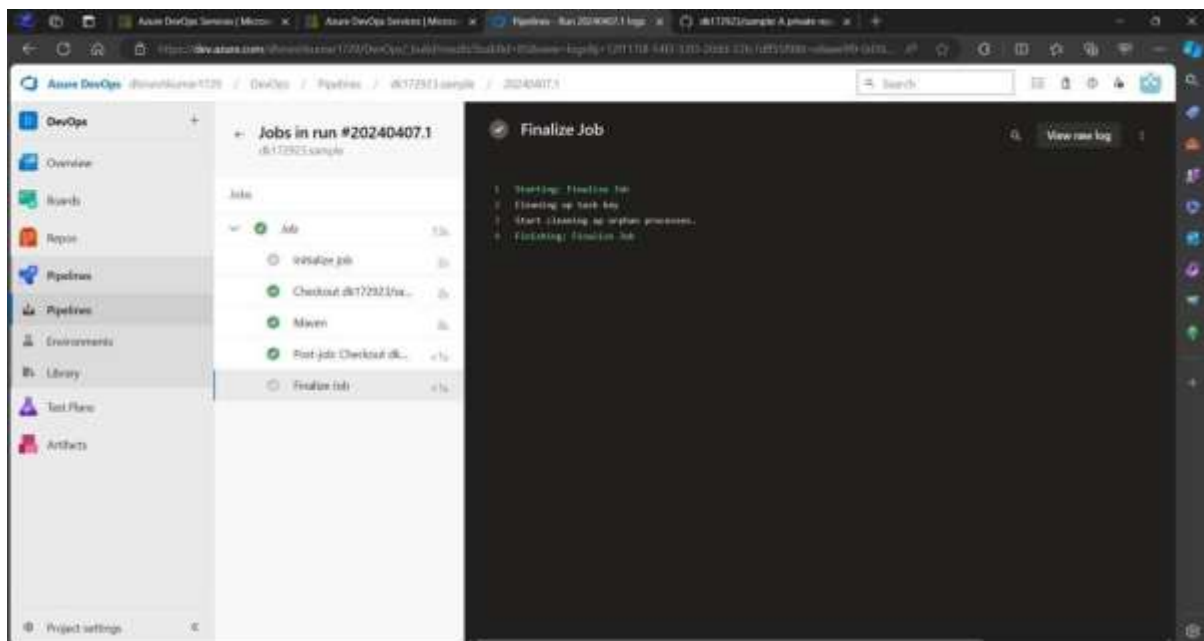
3. Choose Maven Pipeline from the given options.
4. A YAML file is automatically created based on the configuration details found in the pom.xml file in the repository.



1. Press the **Save and Run** button.
2. Create a commit message and press the run button again.
3. Click on the created Job from the Jobs table.



A successful job completion would be like this.



RESULT

Thus, a maven build pipeline has been created using Azure.

Ex. No: 2

RUN REGRESSION TESTS USING MAVEN BUILD

Date:

PIPELINE IN AZURE

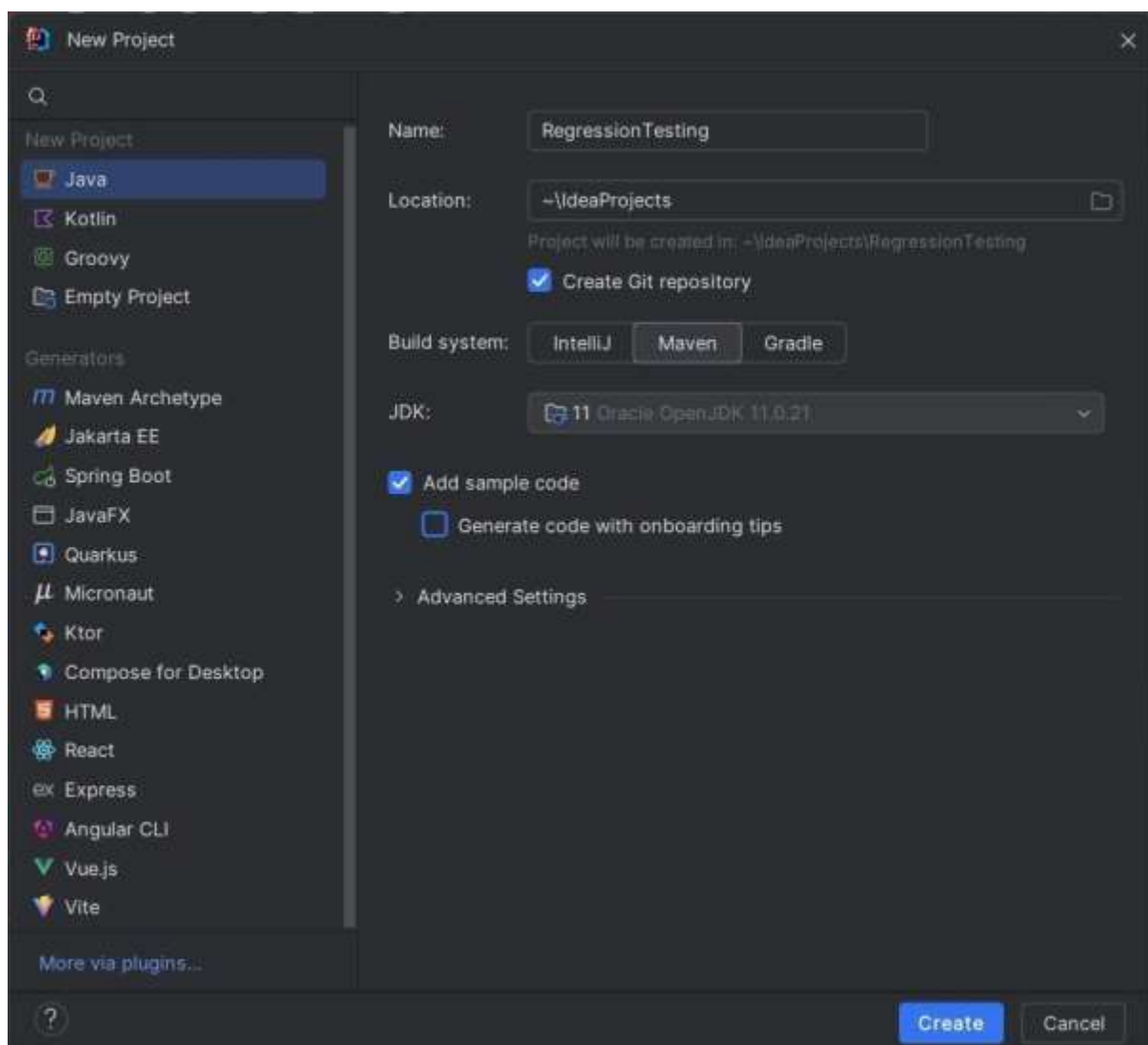
AIM

To run regression tests using the Maven build pipeline in Azure.

PROCEDURE

STEP 1: Create a new maven project in IntelliJ IDEA IDE.

1. Create a project with **RegressionTesting** as its name.
2. Choose the **Maven** build system and available JDK version.
3. Select the **Create Git Repository** option.



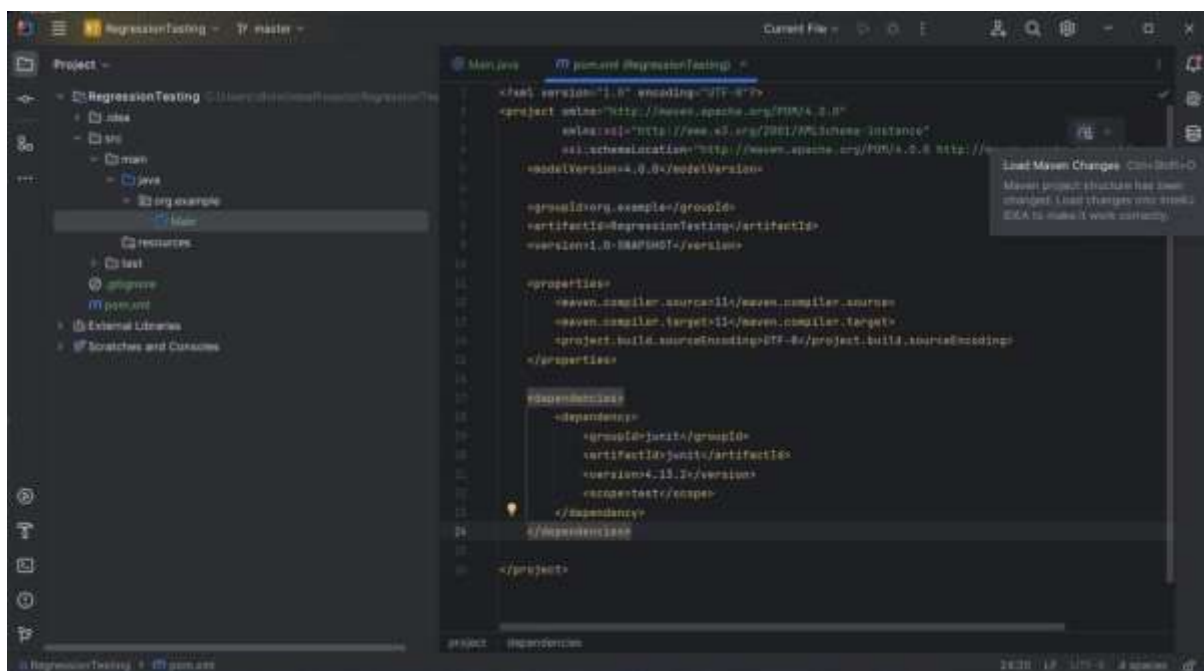
4. Wait for some time until the indexes of the project have been updated.
5. Add the following snippet of code into **pom.xml**

```

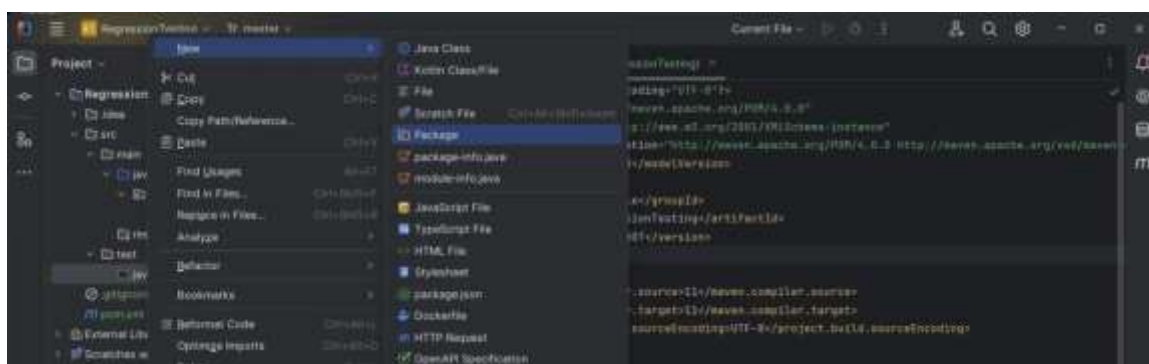
<dependencies>
  <dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.13.2</version>
    <scope>test</scope>
  </dependency>
</dependencies>

```

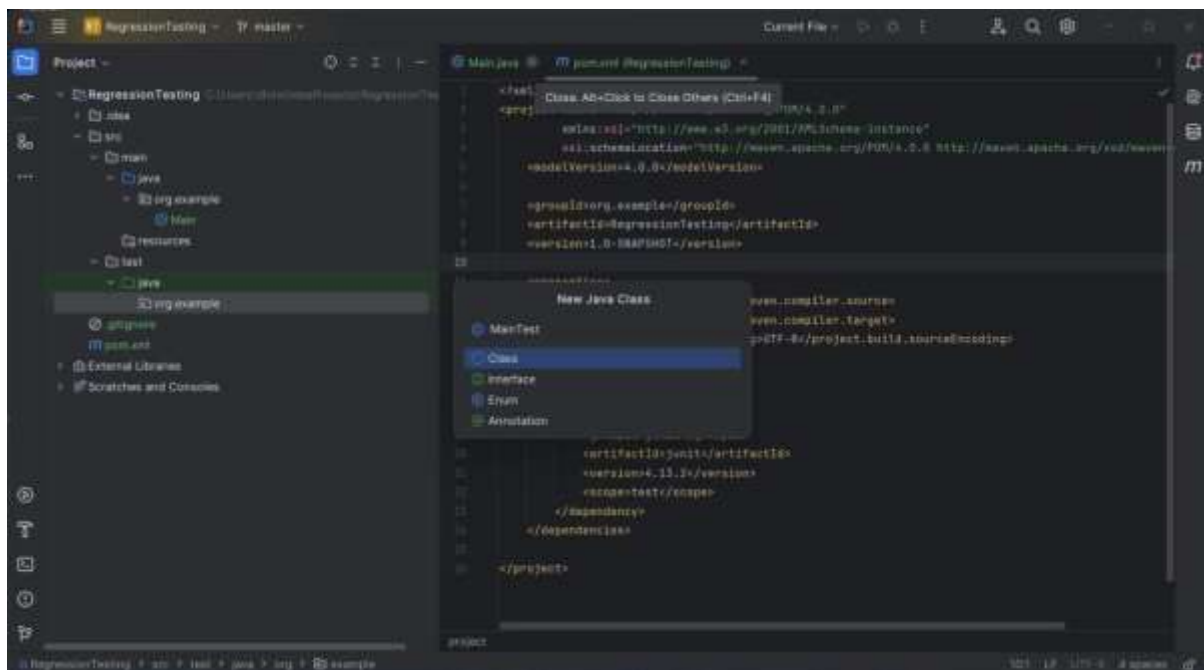
- Once the changes have been made click on the referred button to update the changes made to pom.xml or for simplicity just press **CTRL + SHIFT + O**



- Now, inside the java folder on the **Test** directory create a package named **org.example** (Right click on java folder inside the Test directory choose New and select package)



8. Right-click on the package and create a new class named **MainTest**



9. Paste the following into the MainTest.java file

```
package org.example;
import org.junit.Test;
import java.io.ByteArrayOutputStream;
import java.io.PrintStream;
import static org.junit.Assert.assertEquals;
public class MainTest {
    @Test
    public void testMainMethod() {
        // Redirect standard output to capture console output
        ByteArrayOutputStream outContent = new ByteArrayOutputStream();
        System.setOut(new PrintStream(outContent));

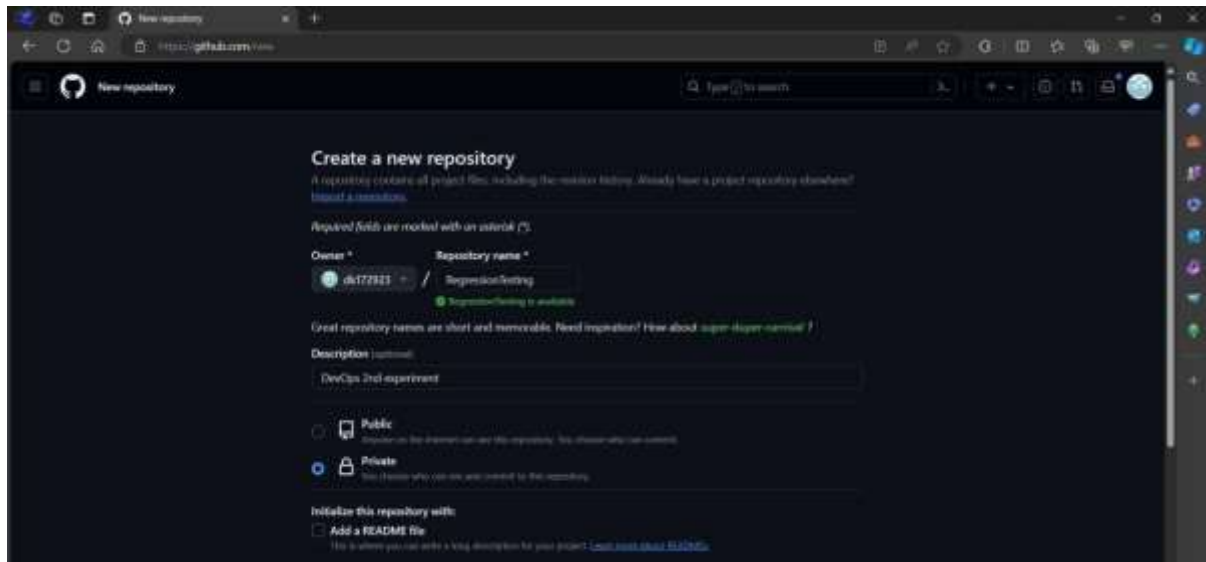
        // Call the main method
        Main.main(new String[]{});

        // Check if "Hello and welcome!" was printed
        assertEquals("Hello and welcome!", outContent.toString().trim());

        // Reset standard output
        System.setOut(System.out);
    }
}
```

STEP 2: Create a GitHub repository

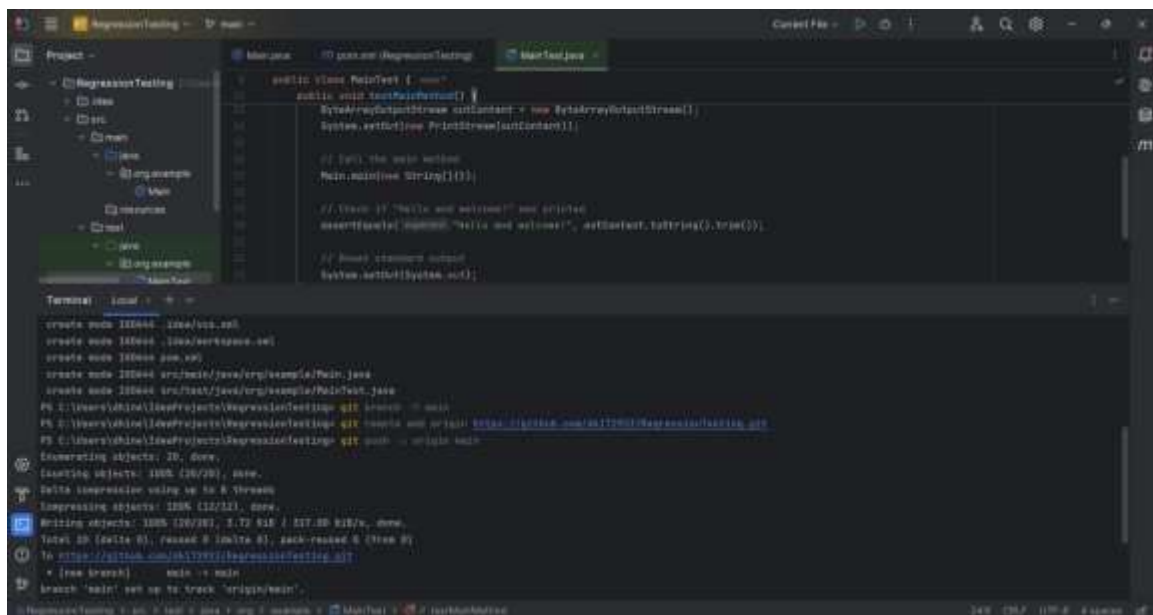
1. Create a private repository with RegressionTesting as its name.



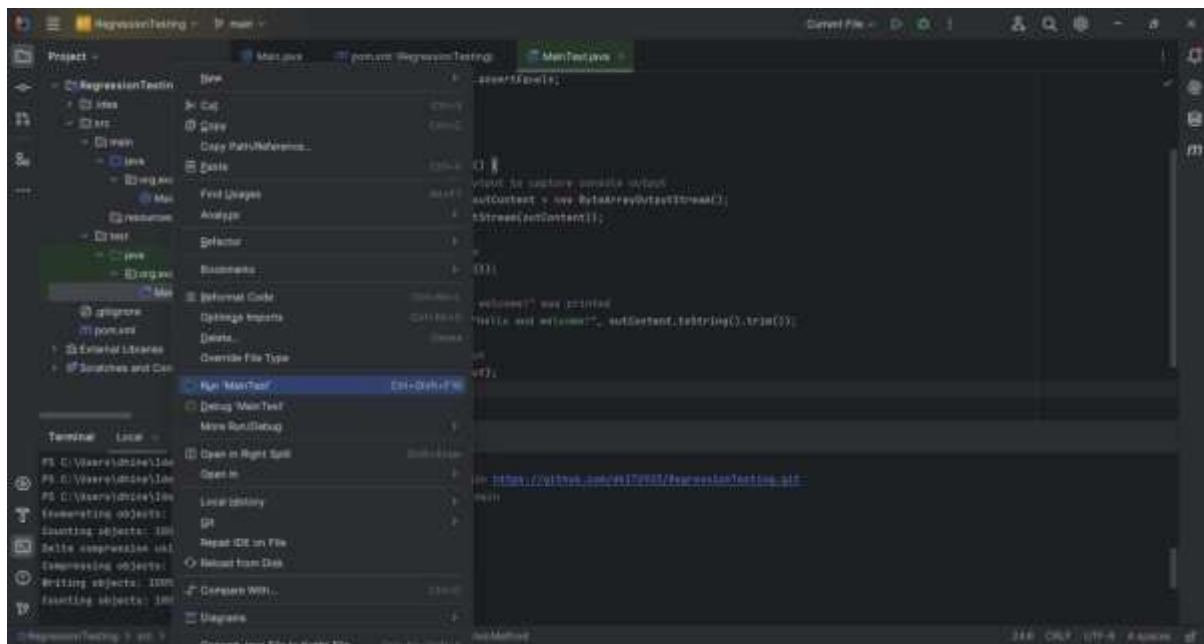
2. Execute the following commands in the terminal of IntelliJ IDEA

```
$ git init
$ git add .
$ git commit -m "regression"
$ git branch -M main
$ git remote add origin YOUR_REPOSITORY_LINK
$ git push -u origin main
```

NOTE: YOUR_REPOSITORY_LINK must contain either the HTTPS or SSH link to your respective GitHub repository.



3. The MainTest class can be executed by Right-Clicking on the class from the project structure and choosing **Run 'MainTest'**

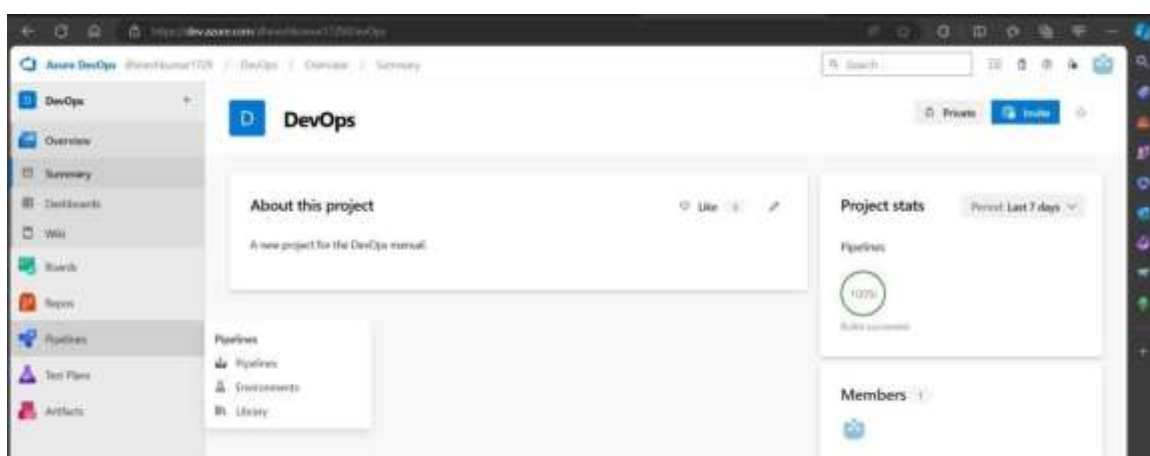


The following can be achieved on the Run tab.

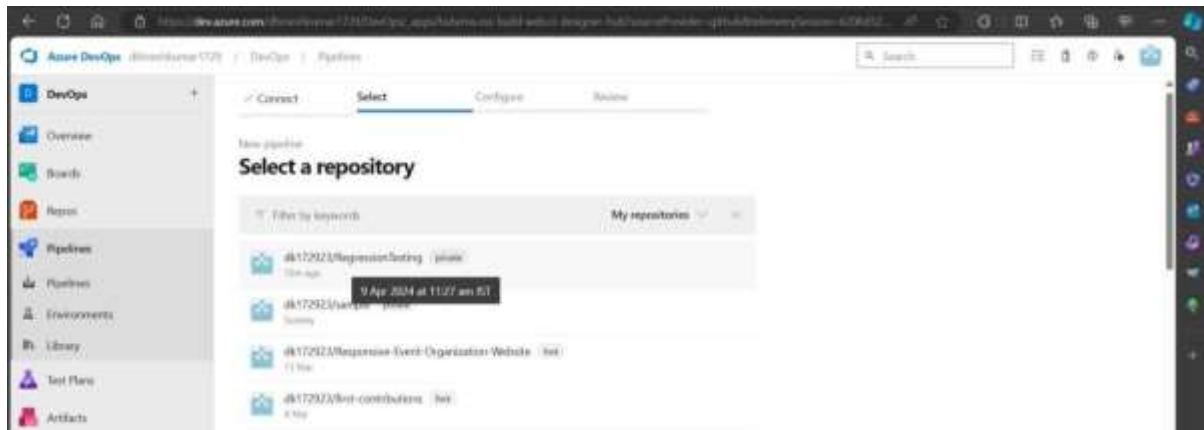


STEP 3: Create an Azure Pipeline

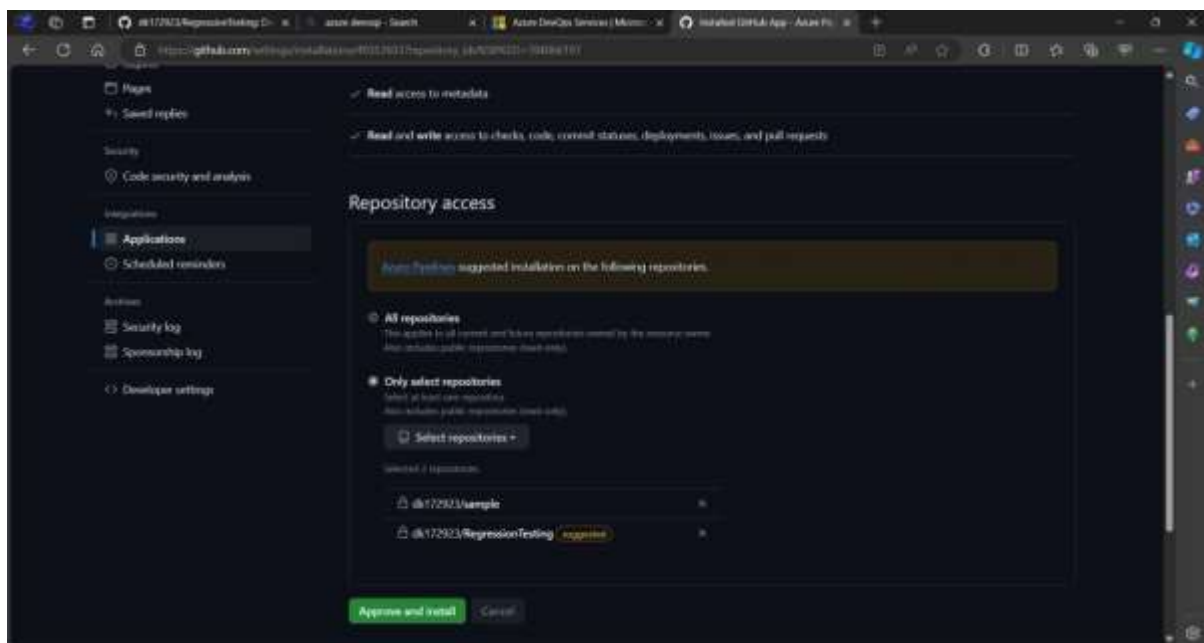
1. Open the Azure DevOps site



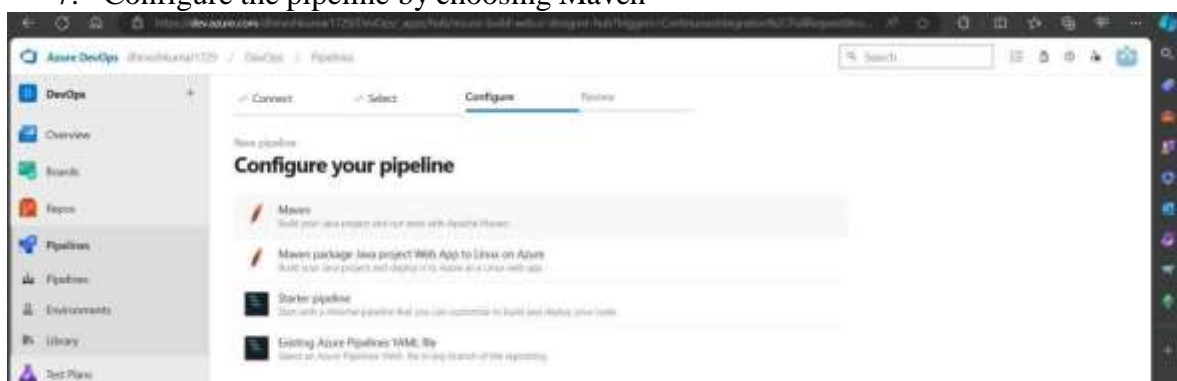
2. Click on existing project DevOps and choose Pipelines from the left pane.
3. Click on the **New Pipeline** button on the right.
4. Choose GitHub from the list.
5. Select the repository for RegressionTesting.



6. If prompted for permission then select Approve and Install on GitHub website.



7. Configure the pipeline by choosing Maven



8. On the next page an azure-pipeline.yml file is generated automatically. Replace this file with the following code

```
# Maven

# Build your Java project and run tests with Apache Maven.

# Add steps that analyze code, save build artifacts, deploy, and more:
# https://docs.microsoft.com/azure/devops/pipelines/languages/java

trigger:

- main

pool:

  vmImage: 'ubuntu-latest'

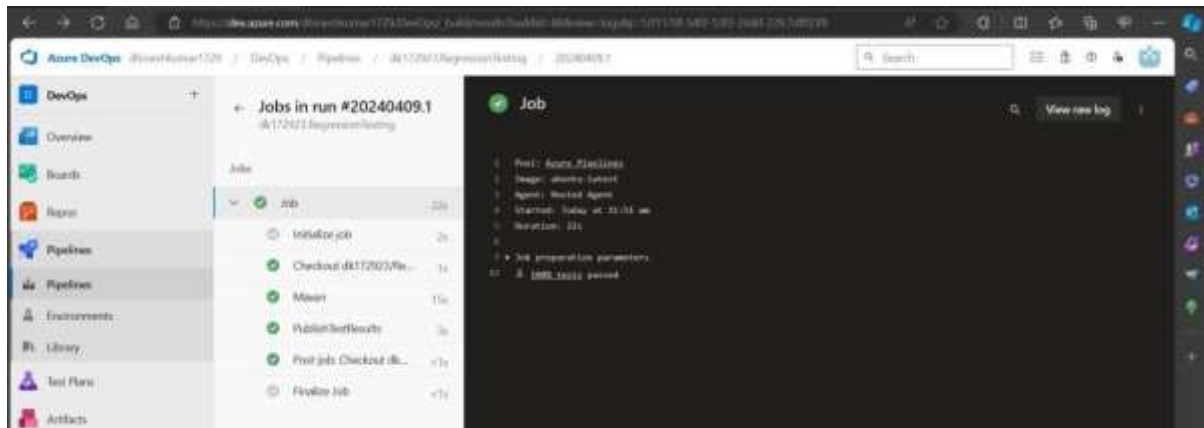
steps:

- task: Maven@3
  inputs:
    mavenPomFile: 'pom.xml'
    mavenOptions: '-Xmx3072m'
    javaHomeOption: 'JDKVersion'
    jdkVersionOption: '11'
    jdkArchitectureOption: 'x64'
    goals: 'clean test'

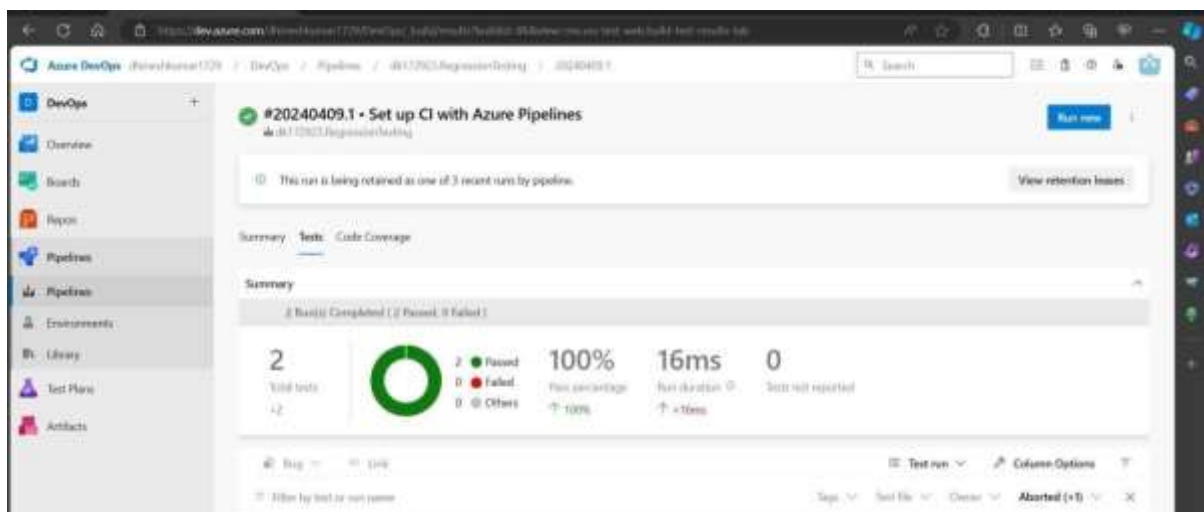
- task: PublishTestResults@2
  inputs:
    testResultsFiles: '**/surefire-reports/TEST-*.xml'
    testRunTitle: 'JUnit Test Results'
```

9. Click on the Save and Run button to commit the changes and execute the pipeline.

10. The following results can be obtained on the output page.



11. From the test results tab it can be found that the tests passed.



RESULT

Thus, regression tests have been run using the Maven Build Pipeline in Azure.

Ex. No: 3

INSTALL JENKINS IN CLOUD

Date:

AIM

To install Jenkins in the cloud (AWS).

PROCEDURE

STEP 1: Create an AWS account

1. Go to <https://aws.amazon.com/> and create an AWS Account.



2. Provide a root email ID and username for the user.

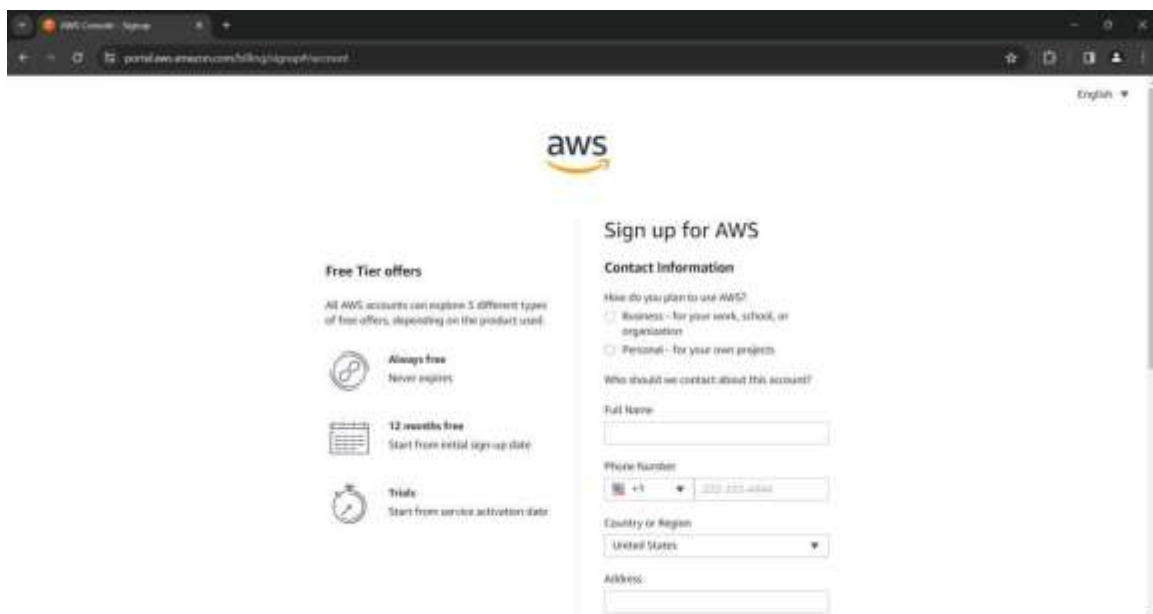


3. After Email verification set up your password.



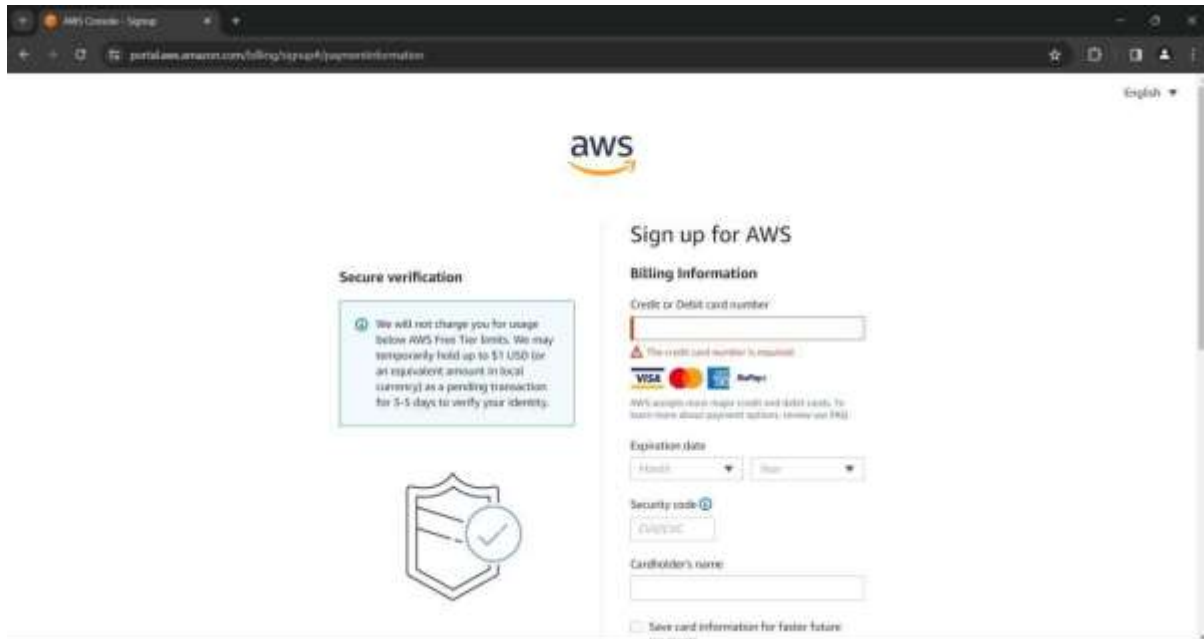
The screenshot shows the AWS sign-up page in a web browser. The URL is `portal.aws.amazon.com/billing/signup#/start|password`. The page features the AWS logo at the top center. On the left, there is a section titled "Explore Free Tier products with a new AWS account." with a link to `aws.amazon.com/free` and an illustration of a hand holding three cubes. On the right, the "Sign up for AWS" section is active, showing a "Create your password" step. A green message box states: "It's you! Your email address has been successfully verified." Below this, there are two password input fields: "Root user password" and "Confirm root user password". An orange "Continue (step 1 of 3)" button is visible, along with a "Sign in to an existing AWS account" link at the bottom.

4. Complete the sign-up process by providing additional information.



The screenshot shows the AWS sign-up page in a web browser. The URL is `portal.aws.amazon.com/billing/signup/#contact`. The page features the AWS logo at the top center. On the left, there is a section titled "Free Tier offers" with three options: "Always free" (Never expires), "12 months free" (Start from initial sign-up date), and "Trials" (Start from service activation date). On the right, the "Sign up for AWS" section is active, showing a "Contact information" step. It includes radio buttons for "Business - for your work, school, or organization" and "Personal - for your own projects". Below these are input fields for "Full Name", "Phone Number" (with a country code dropdown), "Country or Region" (a dropdown menu showing "United States"), and "Address".

5. Provide the debit card information for the identity verification process. A charge of 2 rupees will be debited.



Sign up for AWS

Billing information

Credit or Debit card number

The credit card number is required.

VISA MASTERCARD RuPay

AWS accepts major credit and debit cards. To learn more about payment options, review our FAQ.

Expiration date

Month Year

Security code

Cardholder's name

☐ Save card information for faster future purchases

6. Confirm your identity by providing the PAN card number associated with the name of the user.



Sign up for AWS

Confirm your identity

Name

Choose the name that you want to verify identity for.

Kartika B

Primary purpose of account registration

Choose one that best applies to you. If your account is for a business, select the one that applies to your business.

Personal use

Ownership type

Individual

Add document type

To verify your identity, the name on the document must match the name that you chose.

Permanent Account Number (PAN)

Permanent Account Number (PAN)

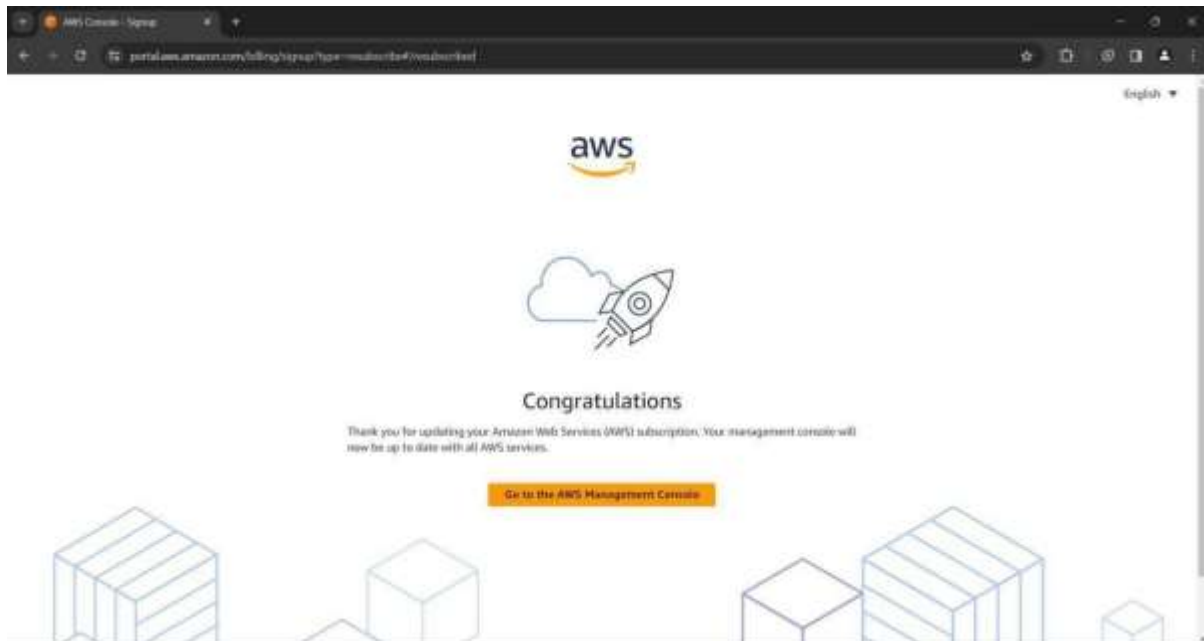
7. Verify the phone number and perform a security check.

The screenshot shows the AWS sign-up page for confirming identity. The URL in the browser is `portal.aws.amazon.com/signup?type=individual&id=verifyyouridentity`. The page features the AWS logo and a heading "Sign up for AWS". Below this is the section "Confirm your identity" with a sub-heading "Before you can use your AWS account, you must verify your phone number. When you continue, the AWS automated system will contact you with a verification code." There are two radio buttons for "How should we send you the verification code?": "Text message (SMS)" (selected) and "Voice call". Below these are fields for "Country or region code" (set to "United States (+1)") and "Mobile phone number". A "Security check" section is partially visible at the bottom, showing a CAPTCHA image with the text "8d 8c 7w".

8. Select Basic Support as the plan since it is free of charge.

The screenshot shows the AWS sign-up page for selecting a support plan. The URL in the browser is `portal.aws.amazon.com/signup?type=individual&id=selectsupport`. The page features the AWS logo and a heading "Sign up for AWS". Below this is the section "Select a support plan" with a sub-heading "Choose a support plan for your business or personal account. Compare plans and pricing estimates." There are three radio buttons for "Select a support plan": "Basic support - free" (selected), "Developer support - From \$25/month", and "Business support - From \$100/month". Each option has a list of features and an icon. The "Basic support - free" option includes features like "24x7 self-service access to AWS", "For account and billing issues only", and "Access to Personal Health Dashboard & Trusted Advisor". The "Developer support - From \$25/month" option includes features like "Recommended for developers experimenting with AWS", "Email access to AWS Support during business hours", and "12 Business-hour response time". The "Business support - From \$100/month" option includes features like "Recommended for running production workloads on AWS", "24x7 full support via email, phone, and chat", "1-hour response time", and "Full set of Trusted Advisor recommendations".

9. The AWS account has been successfully created and verified. Now click on **Go to the Aws Management Console** button.

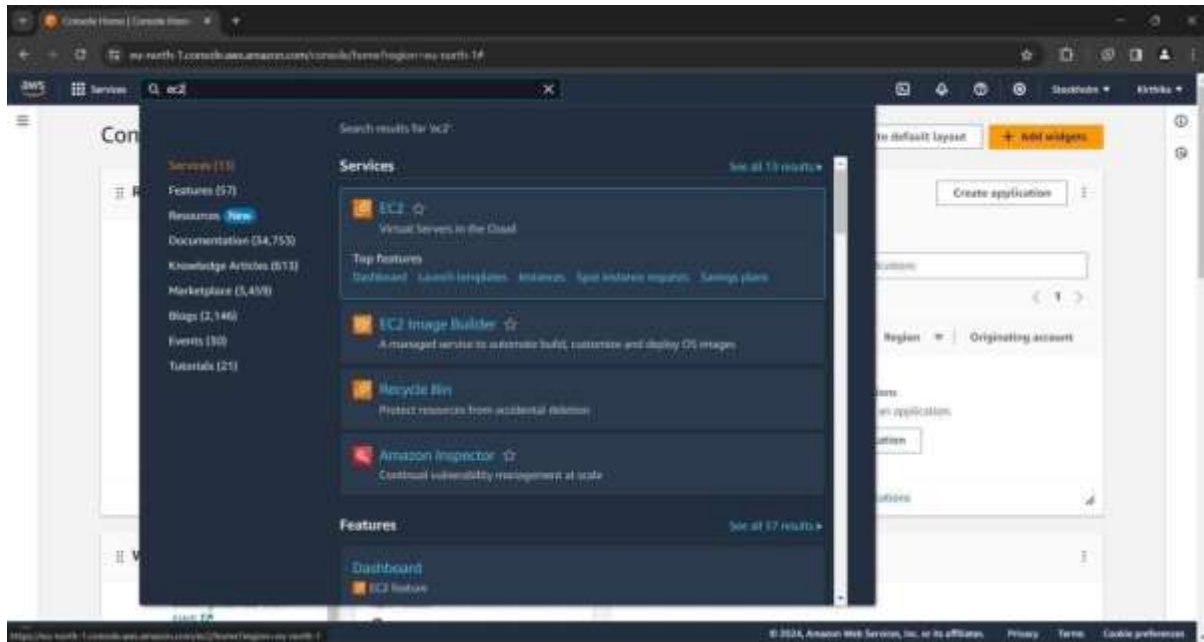


10. Sign in as a Root user by providing the appropriate email address and password.

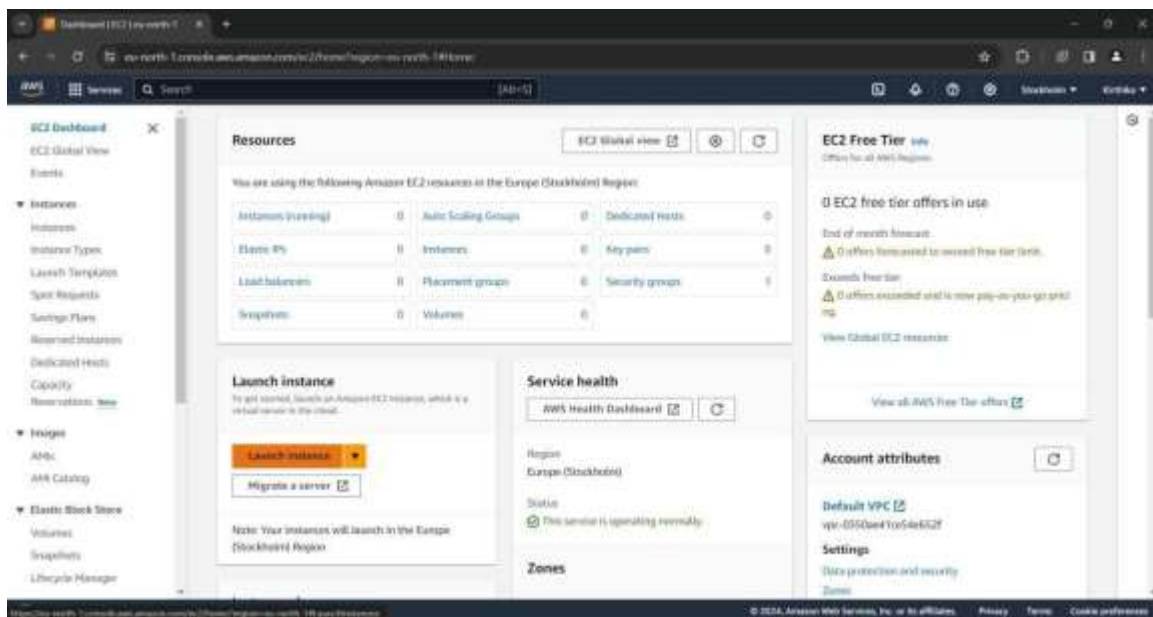


STEP 2: Create an EC2 instance.

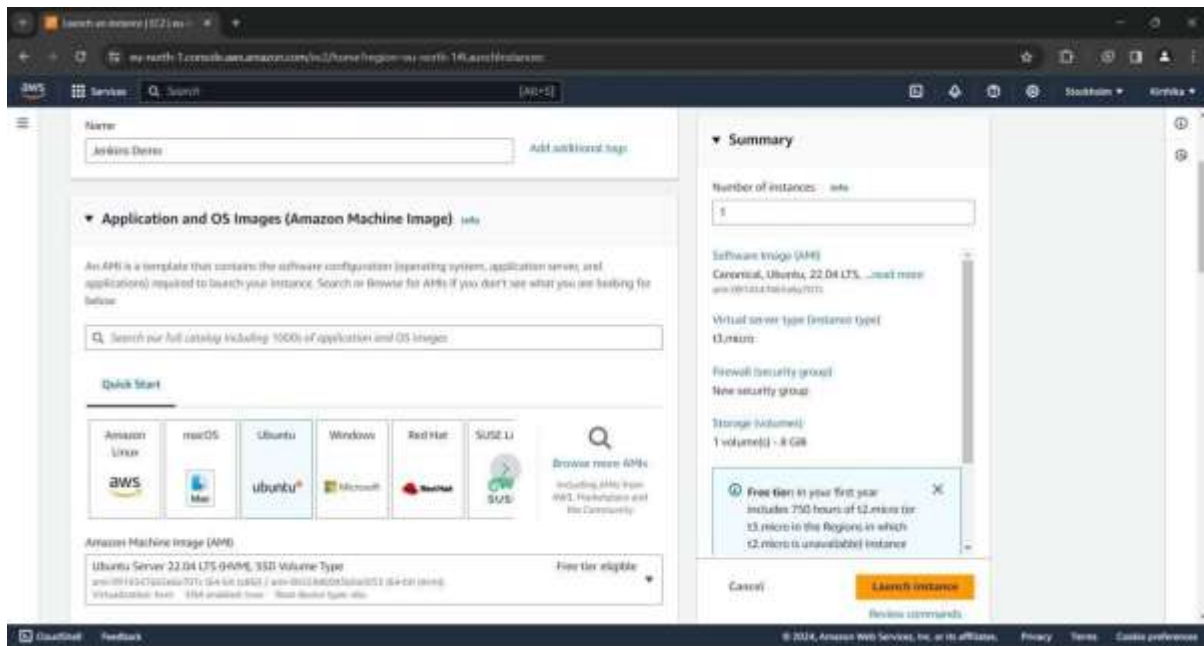
1. Search EC2 in the search bar and choose EC2 from the services tab.



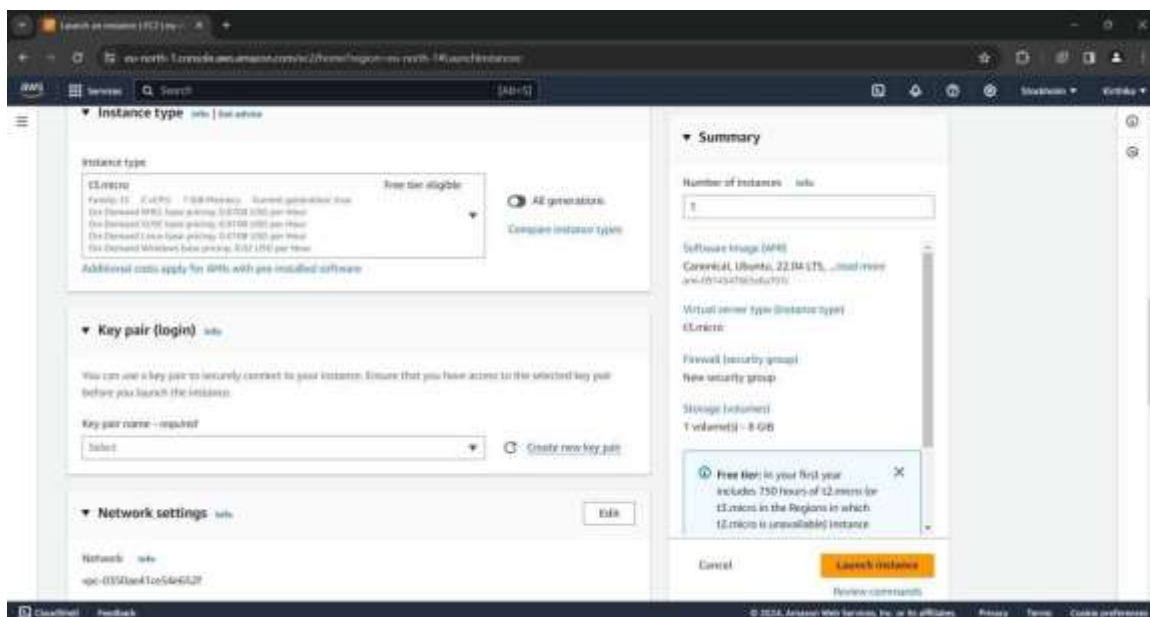
2. Click on the **Launch Instance** button.



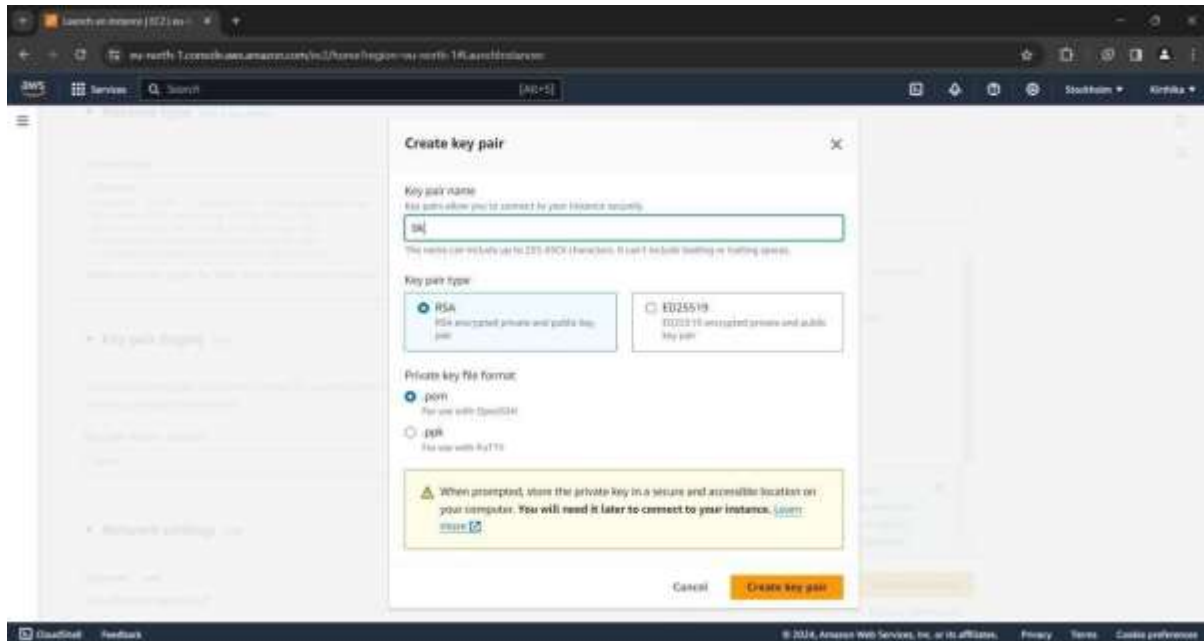
3. Provide a name for the instance for example **Jenkins Demo** and choose Ubuntu as the operating system.



4. Create a new key pair if you haven't created one yet.

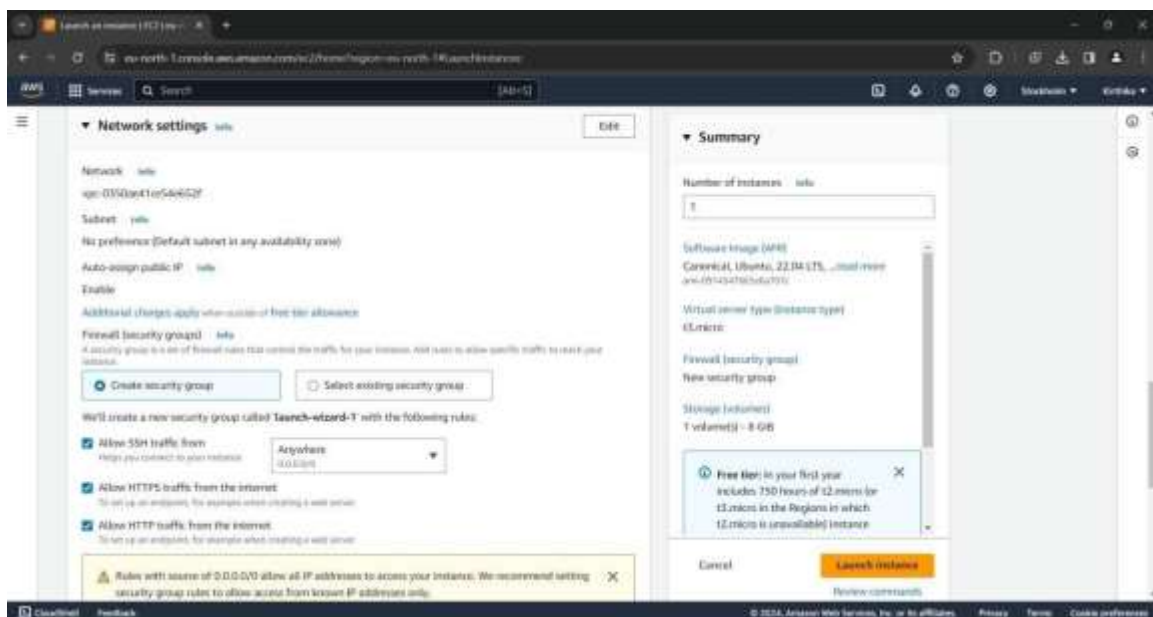


5. Give a suitable name for the key pair, select the key pair type as RSA and click on the **Create key pair** button at the bottom.

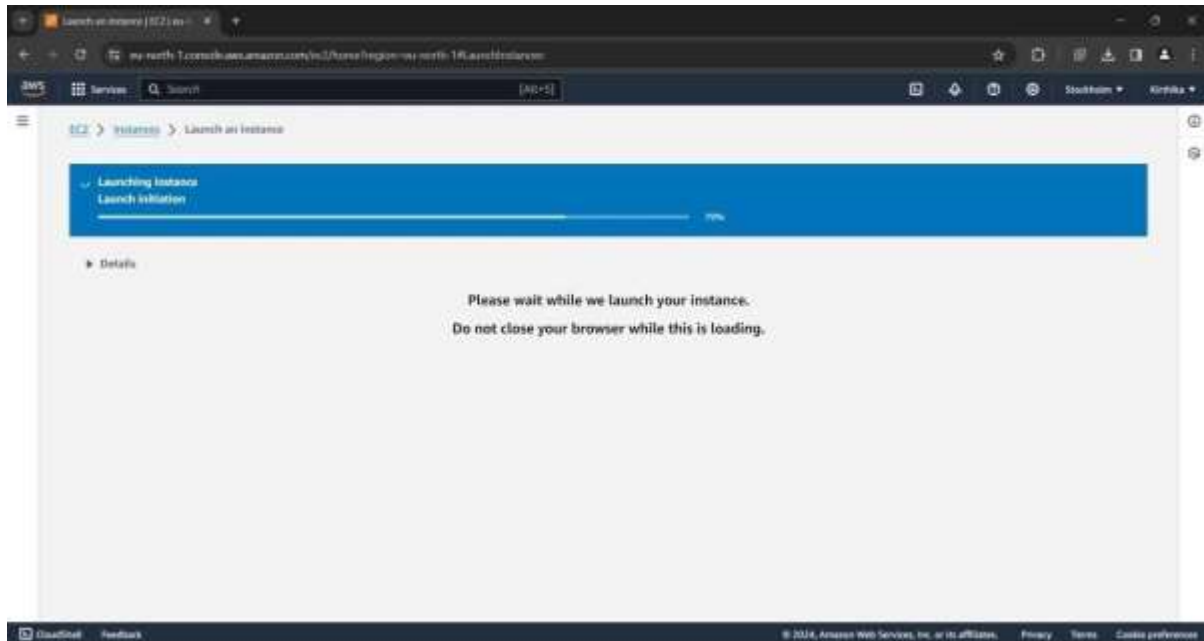


6. In the Network Settings tab select the following options.

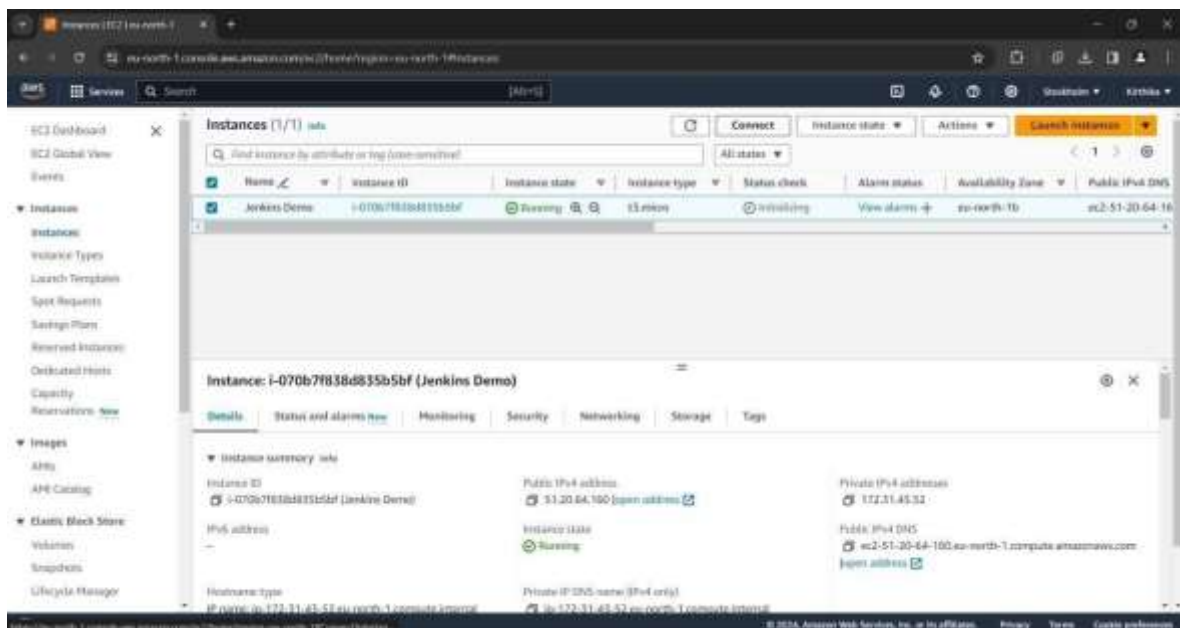
- Allow HTTPS traffic from the internet
- Allow HTTP traffic from the internet



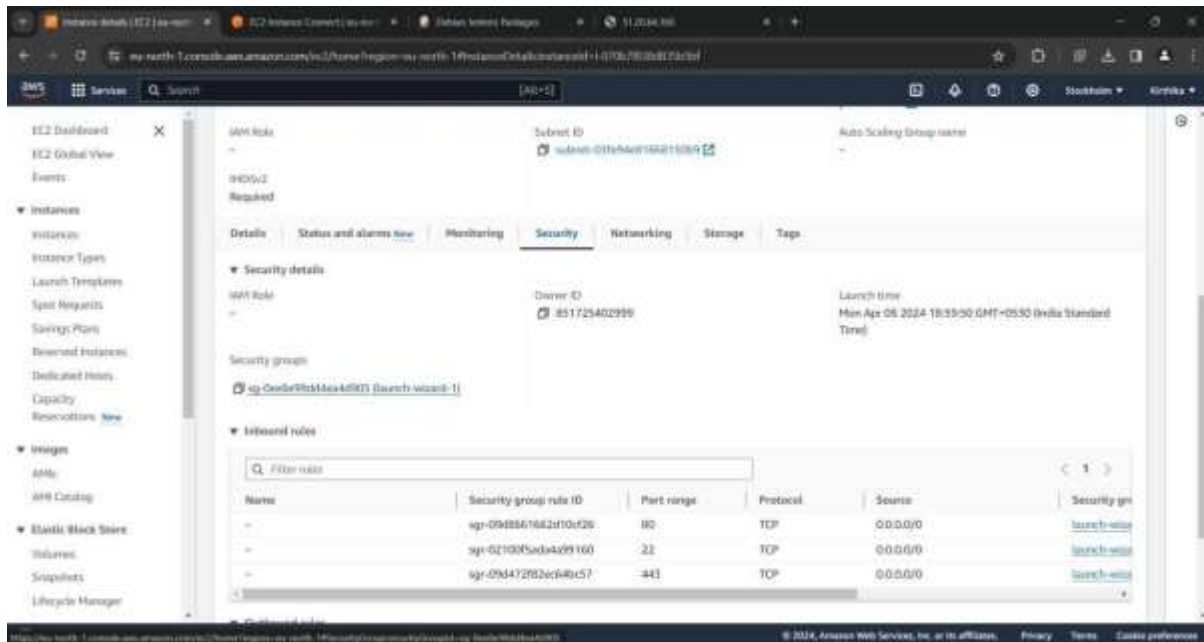
7. Wait while the instance is being launched. It may take some time.



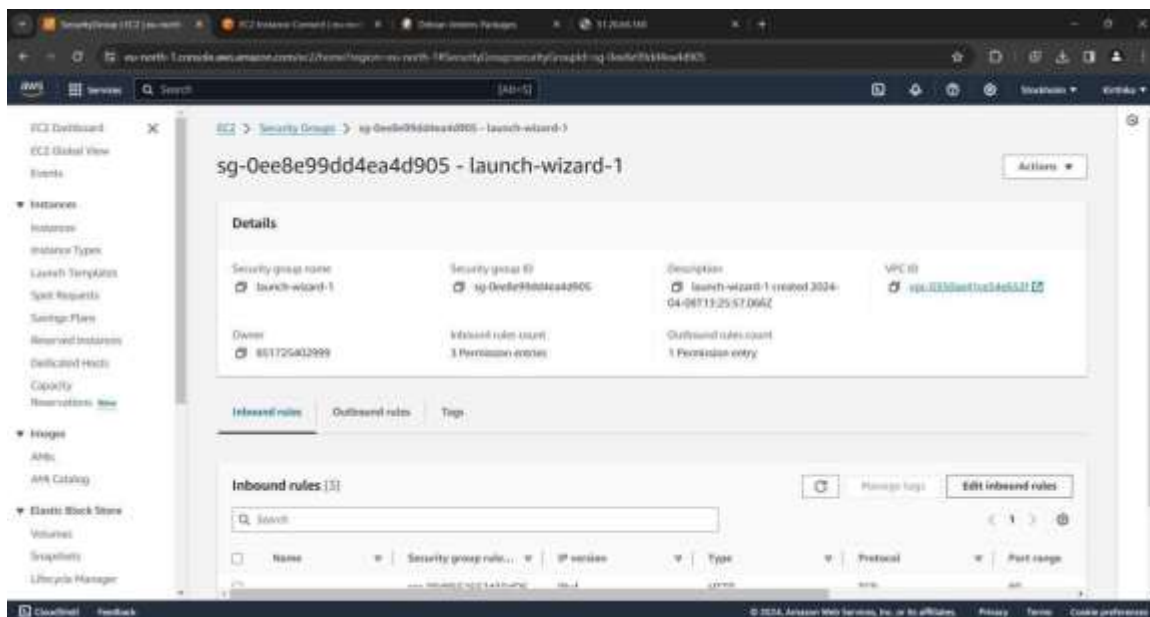
8. Select the Jenkins Demo instance and click on the **Connect** button.



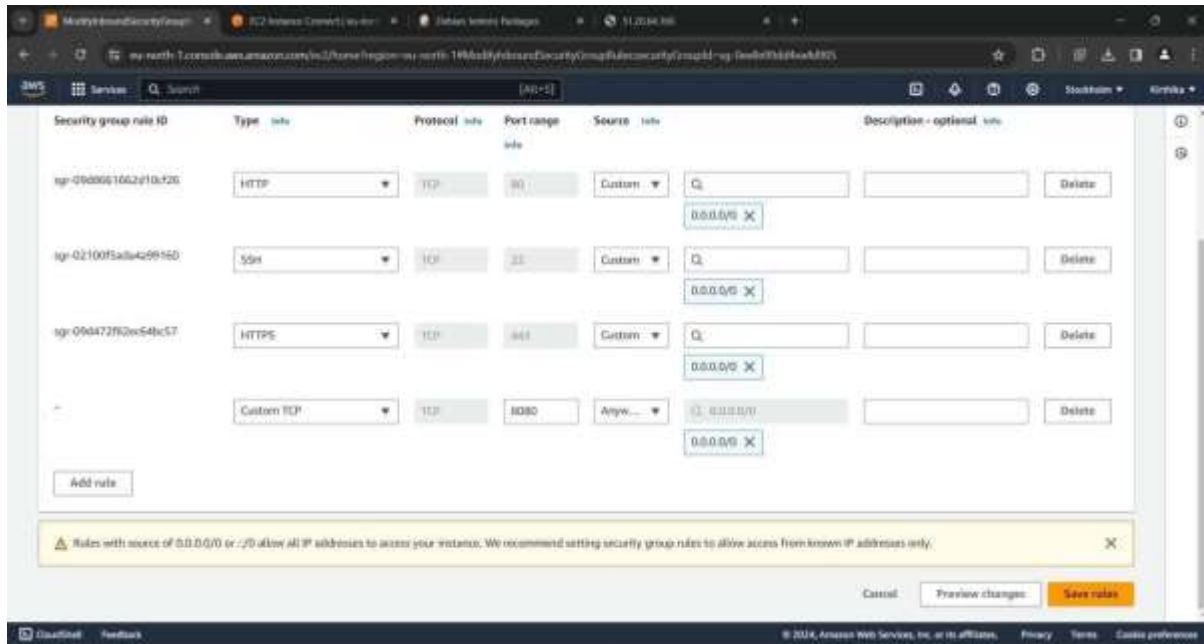
9. Click on the Security tab and choose the **Security Groups** option.



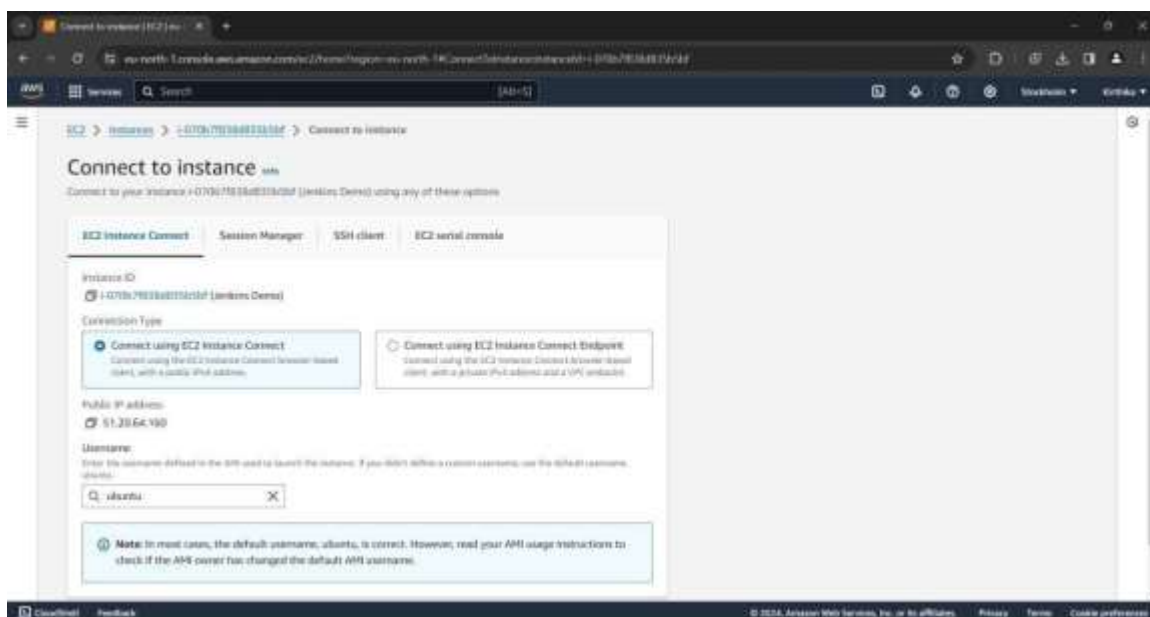
10. Click on the **Edit Inbound Rules** button.



11. Click on **Add rule** at the left bottom, enter the port range as 8080 and choose 0.0.0.0/0



12. Click on the Connect button at the bottom after choosing Connect using **EC2 Instance Connect**.



2. Execute the following command to install Jenkins.

```
$ sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
```

```
$ echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
/etc/apt/sources.list.d/jenkins.list > /dev/null
```

```
$ sudo apt-get update
```

```
$ sudo apt-get install fontconfig openjdk-17-jre
```

```
$ sudo apt-get install Jenkins
```

These commands can also be found at <https://pkg.jenkins.io/debian-stable/> for Debian releases.

3. The following commands will start Jenkins.

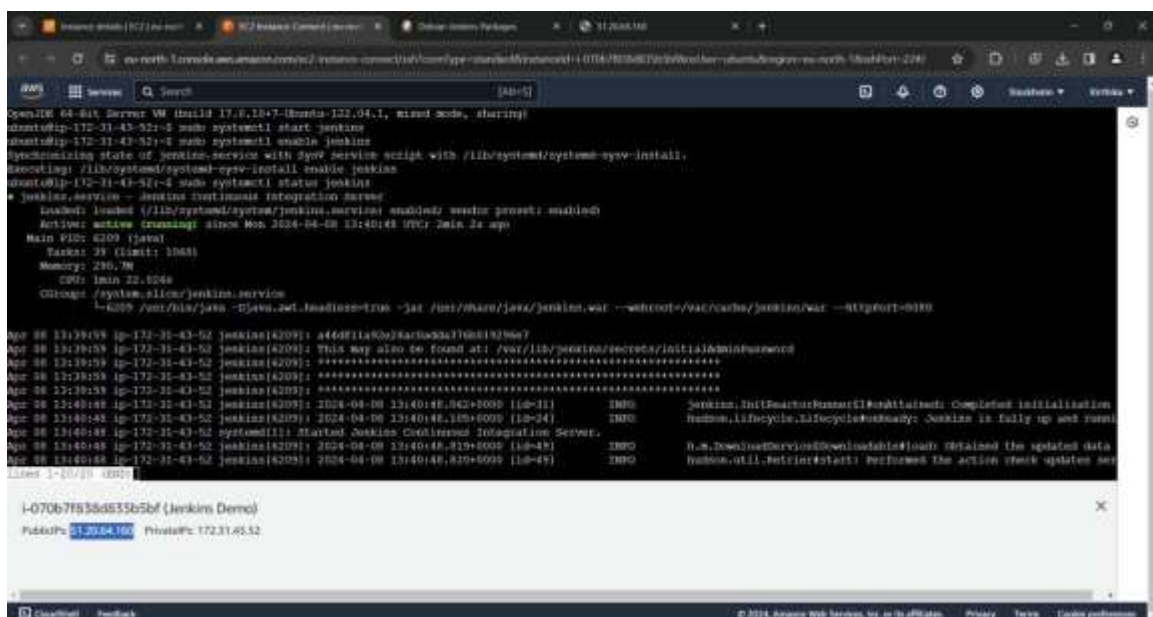
```
$ sudo systemctl start Jenkins
```

```
$ sudo systemctl enable Jenkins
```

```
$ sudo systemctl status Jenkins
```

4. Copy the public IP address found below the terminal and copy the address. Open a new tab, paste the IP address and add **:8080** at the last.

Example: 51.20.64.160:8080



STEP 4: Configuring Jenkins

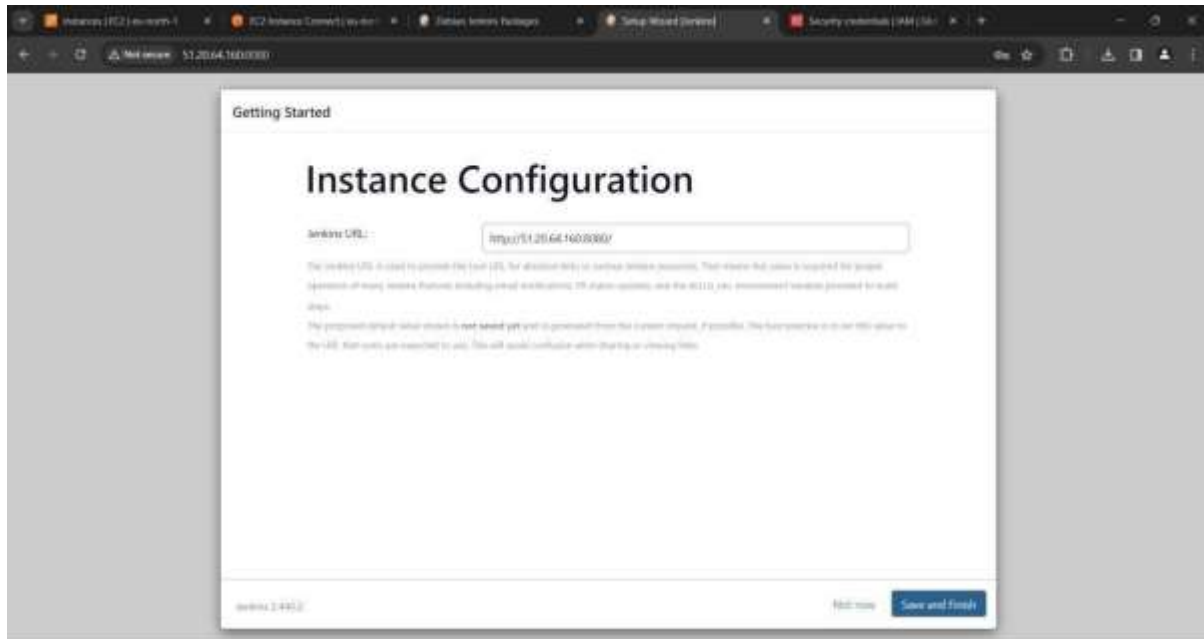
1. Jenkins will be initially locked. The password will be located at the location provided on the screen. Copy it.
2. Go back to the terminal and execute
`$ sudo cat /var/lib/Jenkins/secrets/initialAdminPassword`
3. Copy the password from the terminal, paste it on the Jenkins webpage and press Continue.



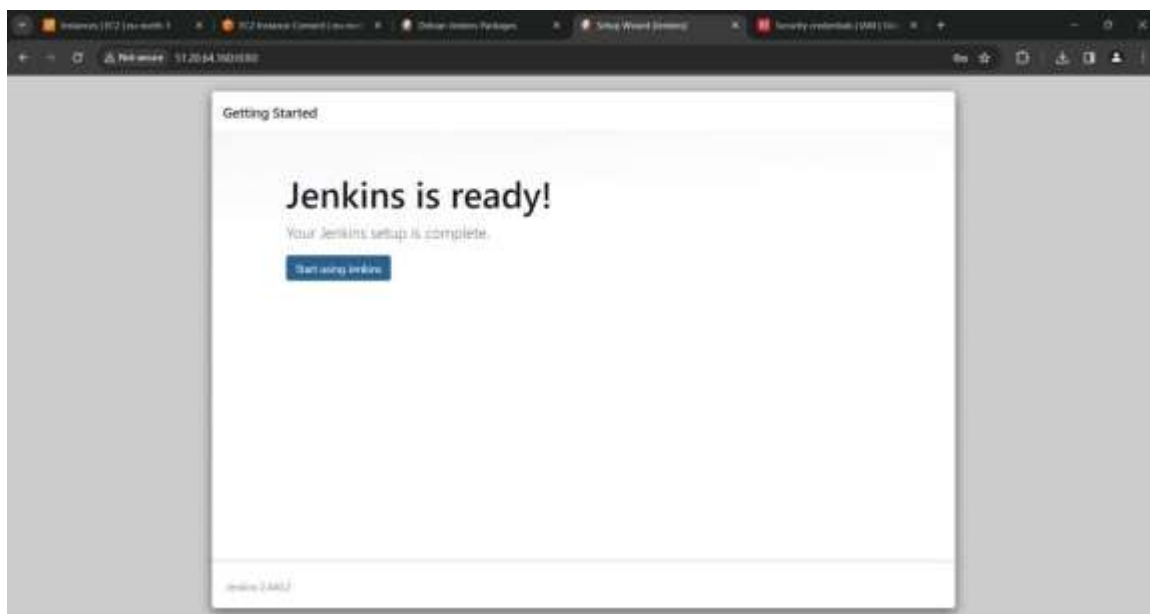
4. Select Installed suggested plugins to install the most used plugins in Jenkins.



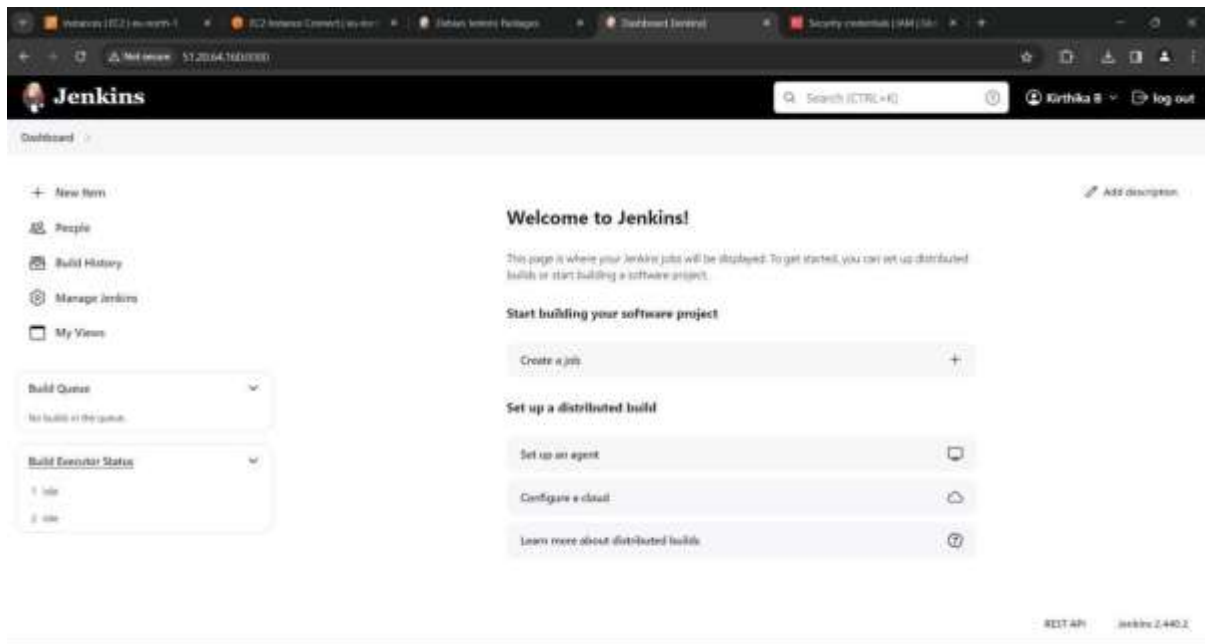
7. Check whether the Jenkins URL is the same as the one mentioned in the address bar and press **Save and Finish**.



8. Now Jenkins has been successfully configured on the cloud. Click on Start using Jenkins to work with it.



9. The Jenkins web page would look like below. Jobs can be built here.



RESULT:

Thus, Jenkins has been installed on the cloud.

Ex. No: 4

CREATE CI PIPELINE USING JENKINS

Date:

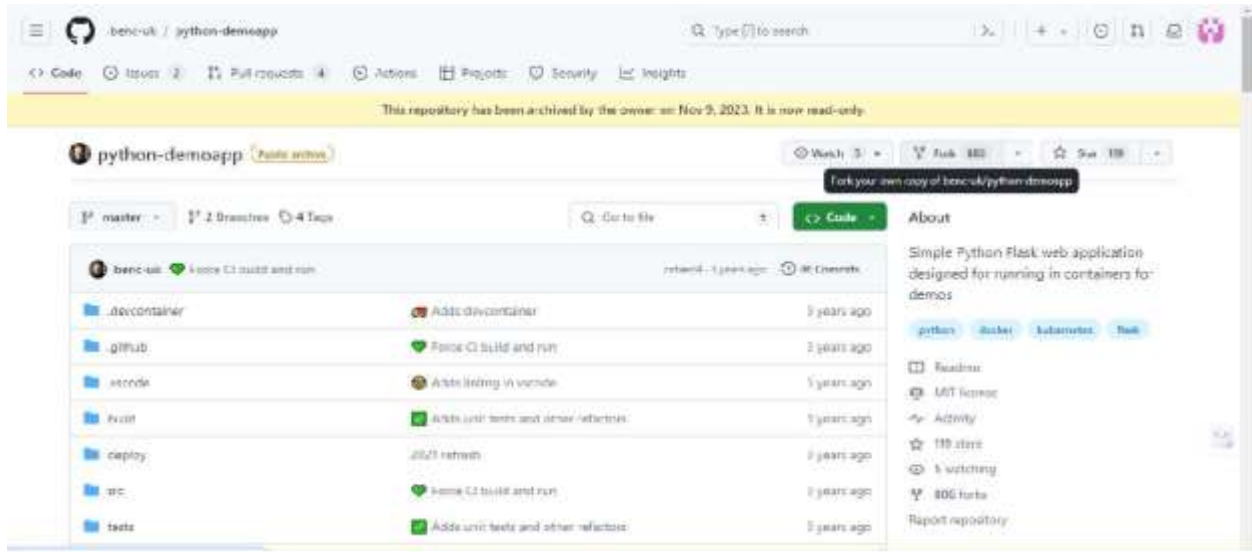
AIM:

To create a CI pipeline using Jenkins.

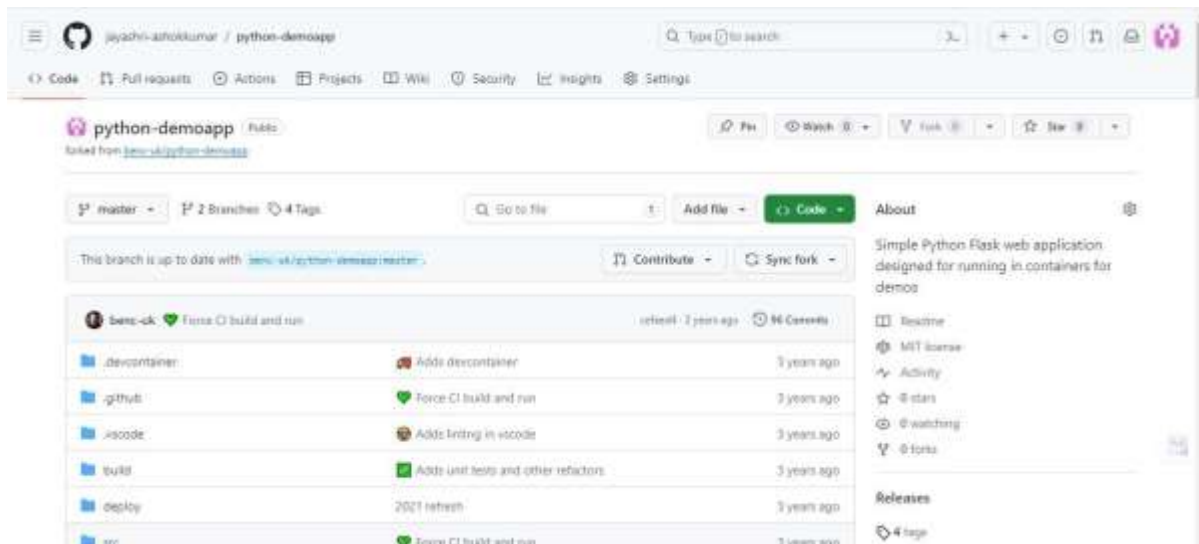
PROCEDURE:

STEP 1: Click and open this GitHub repository (<https://github.com/benc-uk/python-demoapp>). And also click fork to make the changes in your own account.

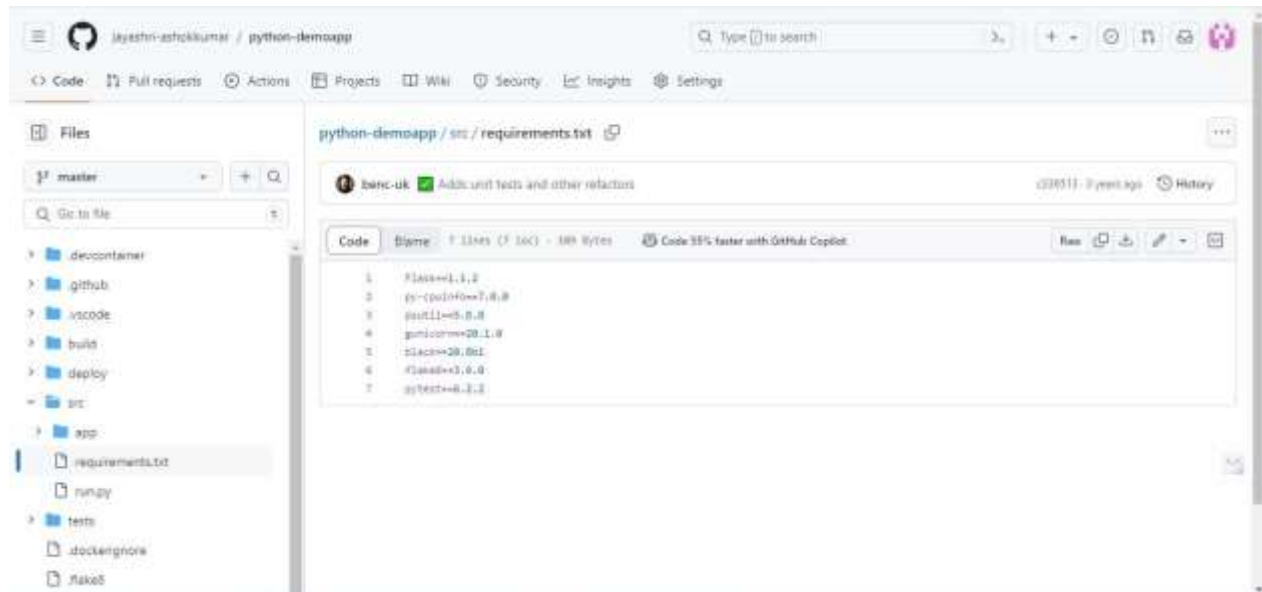
(Note: The username and password in GitHub, Jenkins and Docker Hub website should be the same.)



STEP 2: Now you have moved into your own GitHub account.



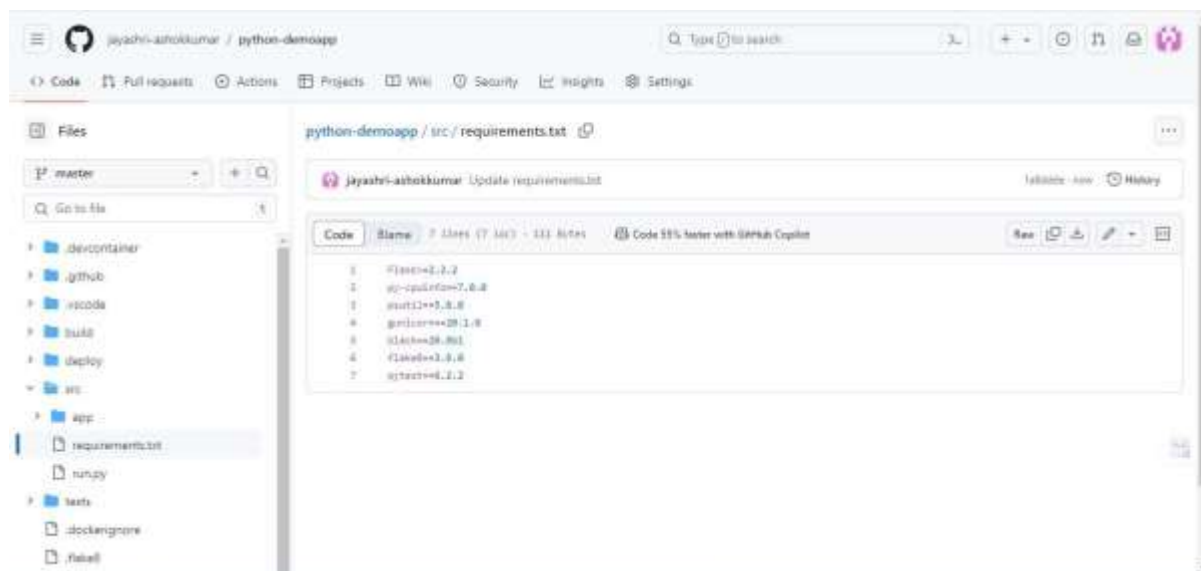
STEP 3: Click the src directory, choose requirements.txt and you will be able to see the below code.



The screenshot shows the GitHub web interface for the repository 'jayashri-ashokkumar / python-demoapp'. The left sidebar displays the file tree with the 'src' directory expanded, showing 'requirements.txt' selected. The main content area displays the code for 'python-demoapp / src / requirements.txt'. The code is as follows:

```
1 Flask==1.1.2
2 py-cpuinfo==7.0.0
3 psutil==5.8.0
4 gunicorn==20.1.0
5 flask==20.0.0
6 flask==3.0.0
7 pytest==6.2.2
```

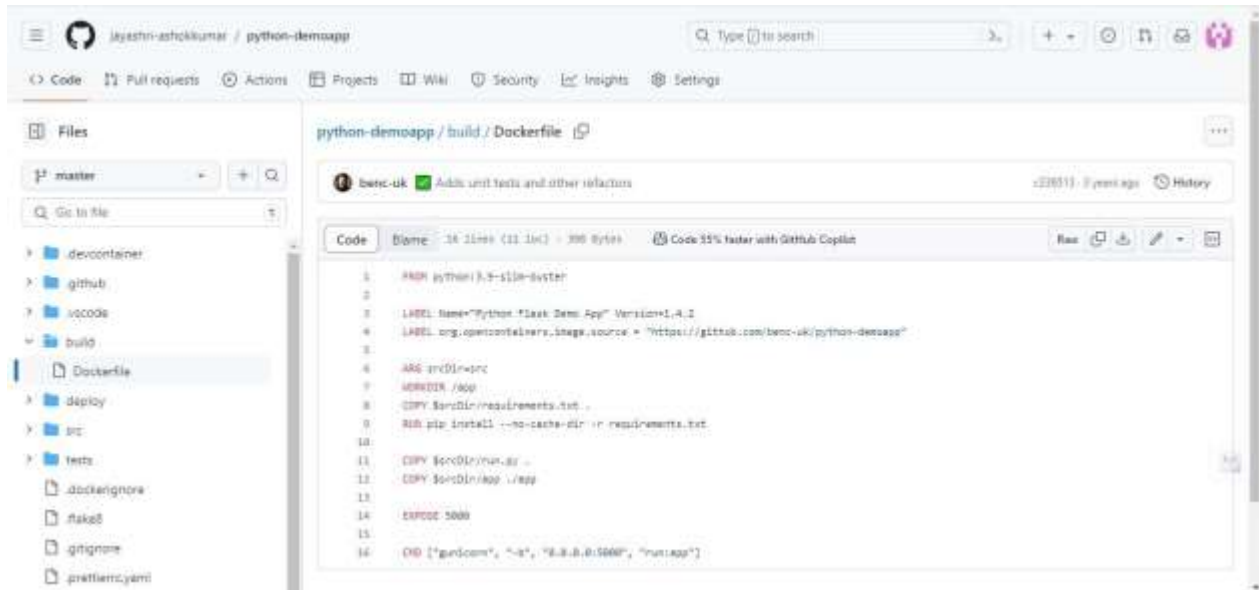
STEP 4: Now, you can make the below changes in the code.



The screenshot shows the GitHub web interface for the repository 'jayashri-ashokkumar / python-demoapp'. The left sidebar displays the file tree with the 'src' directory expanded, showing 'requirements.txt' selected. The main content area displays the code for 'python-demoapp / src / requirements.txt'. The code is as follows:

```
1 Flask==2.0.2
2 py-cpuinfo==7.0.0
3 psutil==5.8.0
4 gunicorn==20.1.0
5 flask==20.0.0
6 flask==3.0.0
7 pytest==6.2.2
```

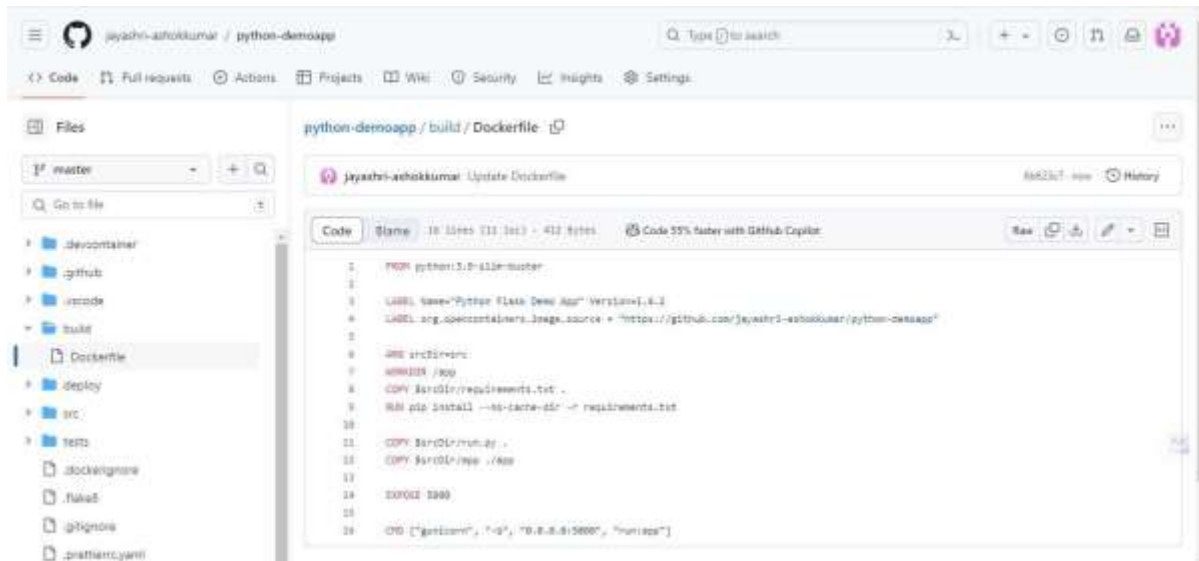
STEP 5: Click build directory and choose Dockerfile and you will be able to see the below code.



The screenshot shows a GitHub repository named 'python-demoapp' by user 'jayashri-ashokkumar'. The 'Files' sidebar on the left shows the directory structure, with 'build' selected and 'Dockerfile' highlighted. The main content area displays the 'Dockerfile' code, which is 16 lines long (11 lines of code, 308 bytes). The code is as follows:

```
1 FROM python:3.9-slim-buster
2
3 LABEL Name="Python Flask Demo App" Version=1.0.1
4 LABEL org.opencontainers.image.source = "https://github.com/benc-uk/python-demoapp"
5
6 ARG srcDir=/src
7 WORKDIR /app
8 COPY $srcDir/requirements.txt .
9 RUN pip install --no-cache-dir -r requirements.txt
10
11 COPY $srcDir/run.py .
12 COPY $srcDir/app /app
13
14 EXPOSE 5000
15
16 CMD ["gunicorn", "-s", "0.0.0.0:5000", "runapp"]
```

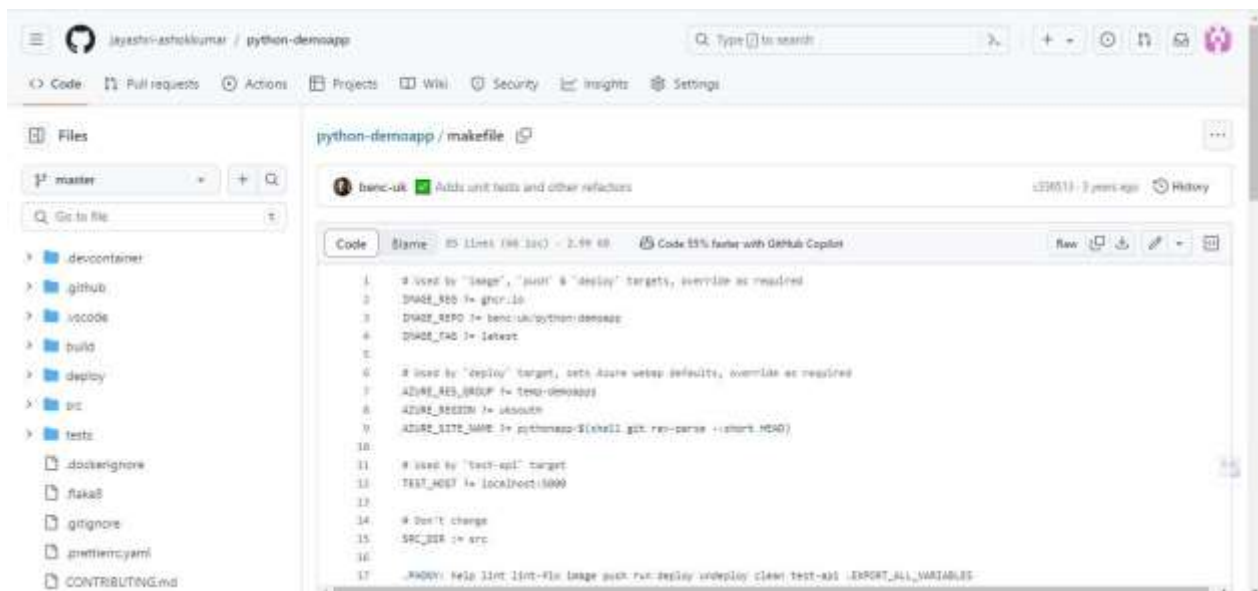
STEP 6: Replace the above link with your own GitHub repository link.



The screenshot shows the same GitHub repository, but the 'Dockerfile' has been updated. The commit message is 'Update Dockerfile' by 'jayashri-ashokkumar'. The code is now 16 lines long (11 lines of code, 412 bytes). The updated code is as follows:

```
1 FROM python:3.9-slim-buster
2
3 LABEL Name="Python Flask Demo App" Version=1.0.1
4 LABEL org.opencontainers.image.source = "https://github.com/jayashri-ashokkumar/python-demoapp"
5
6 ARG srcDir=/src
7 WORKDIR /app
8 COPY $srcDir/requirements.txt .
9 RUN pip install --no-cache-dir -r requirements.txt
10
11 COPY $srcDir/run.py .
12 COPY $srcDir/app /app
13
14 EXPOSE 5000
15
16 CMD ["gunicorn", "-s", "0.0.0.0:5000", "runapp"]
```

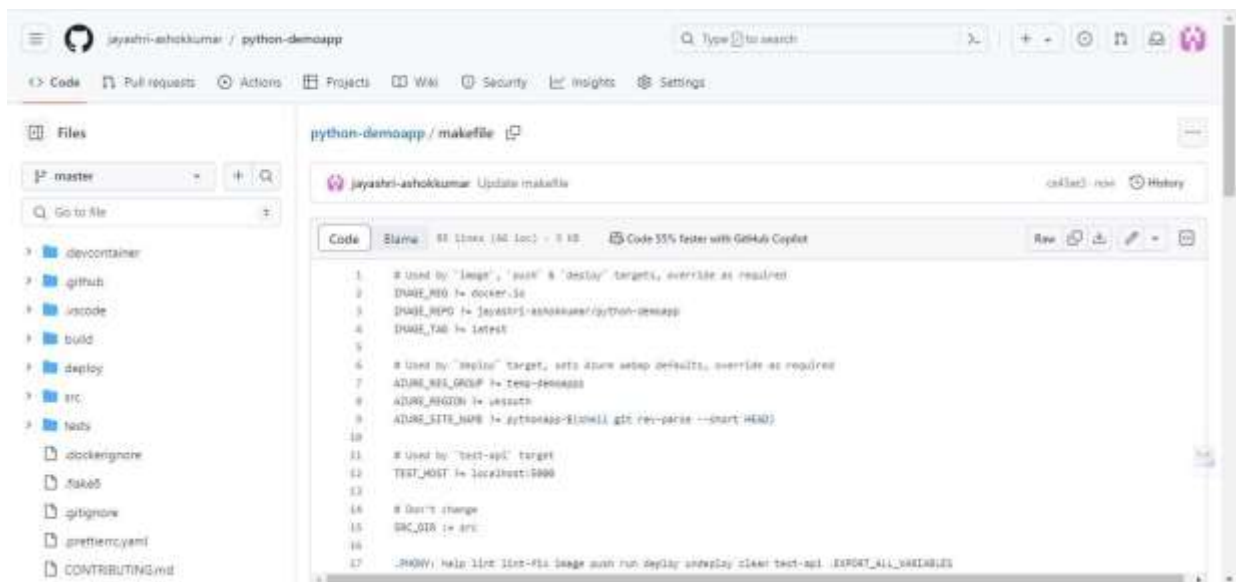
STEP 7: Click on makefile, and you will be able to see the below code.



The screenshot shows the GitHub interface for the repository 'jayashri-ashokkumar / python-demoapp'. The 'Files' tab is selected, and the file 'makefile' is open. The code is as follows:

```
1 # Used by 'image', 'push' & 'deploy' targets, override as required
2 IMAGE_REPO := ghcr.io
3 IMAGE_REPO := jayashri-ashokkumar/python-demoapp
4 IMAGE_TAG := latest
5
6 # Used by 'deploy' target, sets Azure webapp defaults, override as required
7 AZURE_RES_GROUP := temp-demoapp
8 AZURE_REGION := usouth
9 AZURE_SITE_NAME := pythonapp-$(shell git rev-parse --short HEAD)
10
11 # Used by 'test-api' target
12 TEST_HOST := localhost:5000
13
14 # Don't change
15 SRC_DIR := src
16
17 .PHONY: help lint lint-fls image push run deploy undeploy clean test-api .EXPORT_ALL_VARIABLES
```

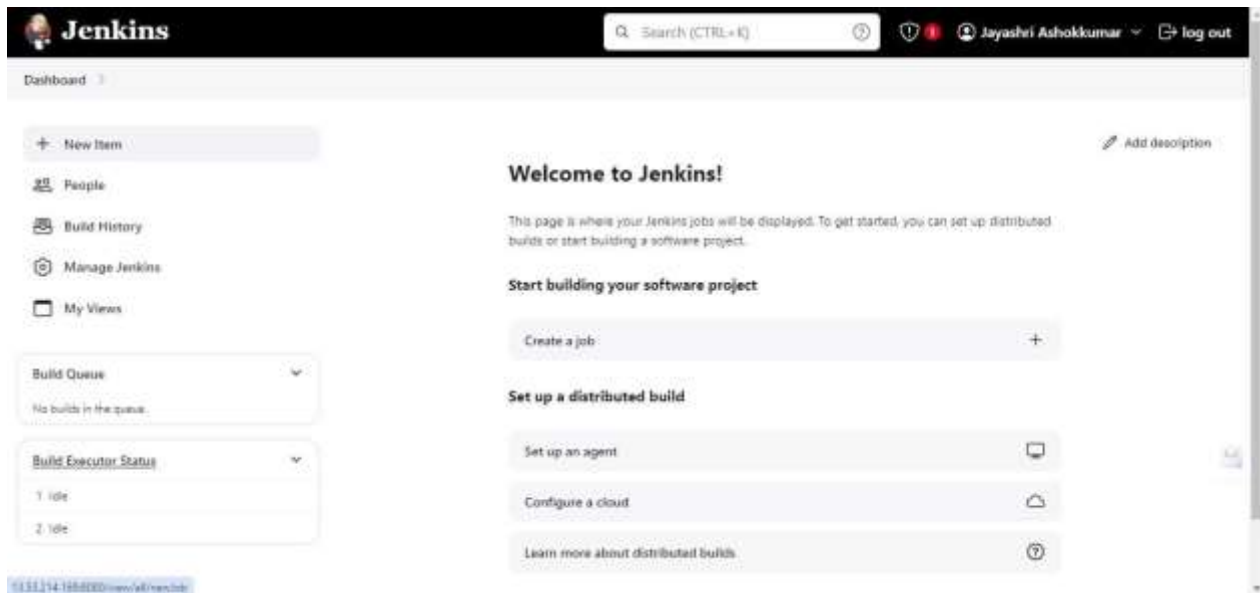
STEP 8: Replace the above name in the code with your own GitHub name and also replace the ghcr.io with docker.io.



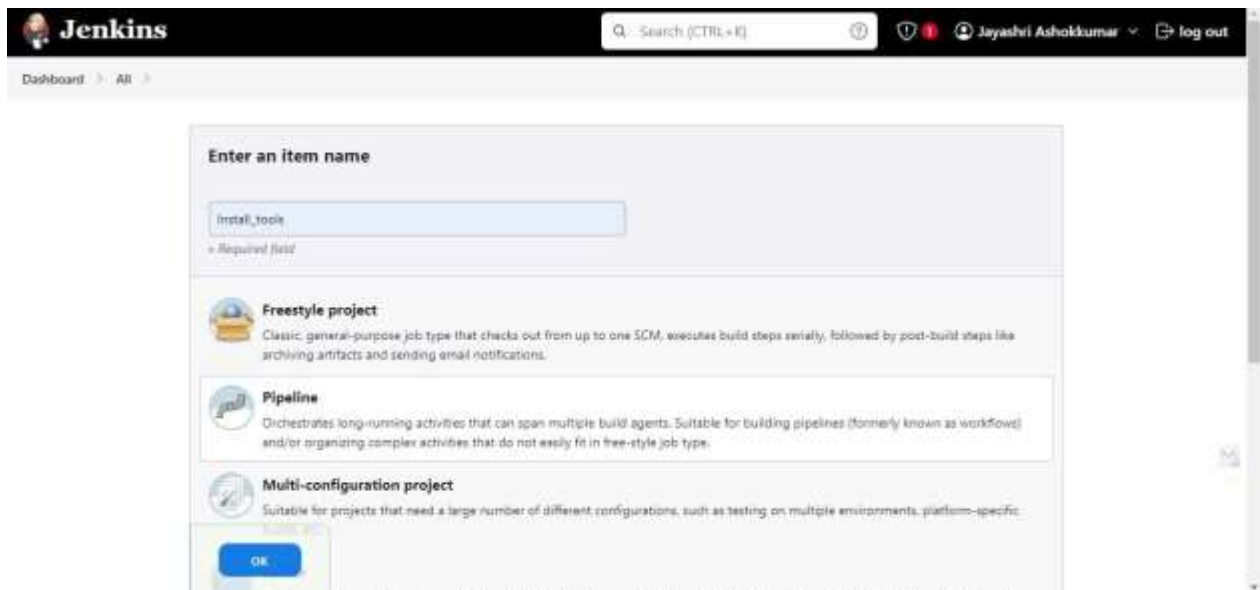
The screenshot shows the same GitHub repository, but the 'makefile' has been updated. The code is as follows:

```
1 # Used by 'image', 'push' & 'deploy' targets, override as required
2 IMAGE_REPO := docker.io
3 IMAGE_REPO := jayashri-ashokkumar/python-demoapp
4 IMAGE_TAG := latest
5
6 # Used by 'deploy' target, sets Azure webapp defaults, override as required
7 AZURE_RES_GROUP := temp-demoapp
8 AZURE_REGION := usouth
9 AZURE_SITE_NAME := pythonapp-$(shell git rev-parse --short HEAD)
10
11 # Used by 'test-api' target
12 TEST_HOST := localhost:5000
13
14 # Don't change
15 SRC_DIR := src
16
17 .PHONY: help lint lint-fls image push run deploy undeploy clean test-api .EXPORT_ALL_VARIABLES
```

STEP 9: Open the Jenkins page, and click on the new item that is present in the dashboard.



STEP 10: Now Enter the item name and select pipeline, then click on OK.



STEP 11: Click > Install_tools > Configure > Pipeline. In the definition section of Pipeline, Choose Pipeline script and the script area will be visible in which you will type the below code.

```

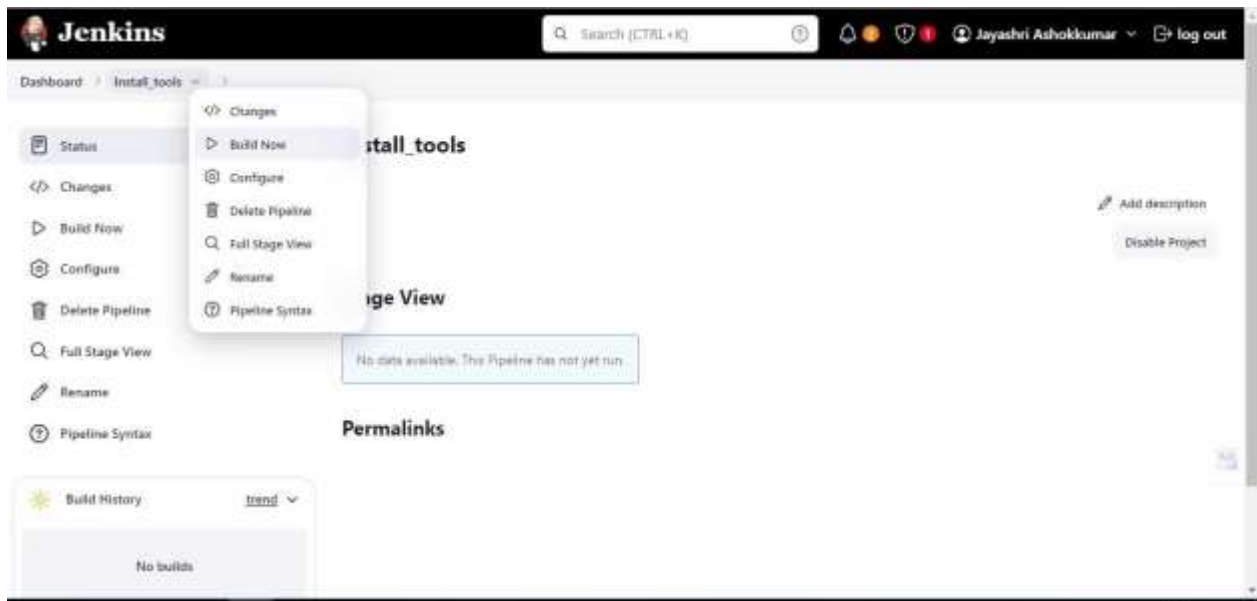
pipeline {
  agent any
  stages {
    stage('Python Version') {
      steps {
        sh "sudo apt update"
        sh "sudo apt -y upgrade"
        sh "sudo apt install python3.10-venv -y"
        sh "python3 -v"
      }
    }
  }
  stage('Docker Version'){
    steps{
      sh "docker -v"
    }
  }
  stage('Install Make'){
    steps{
      sh "sudo apt install make -y"
      sh "sudo apt install make-guile -y"
    }
  }
}

```

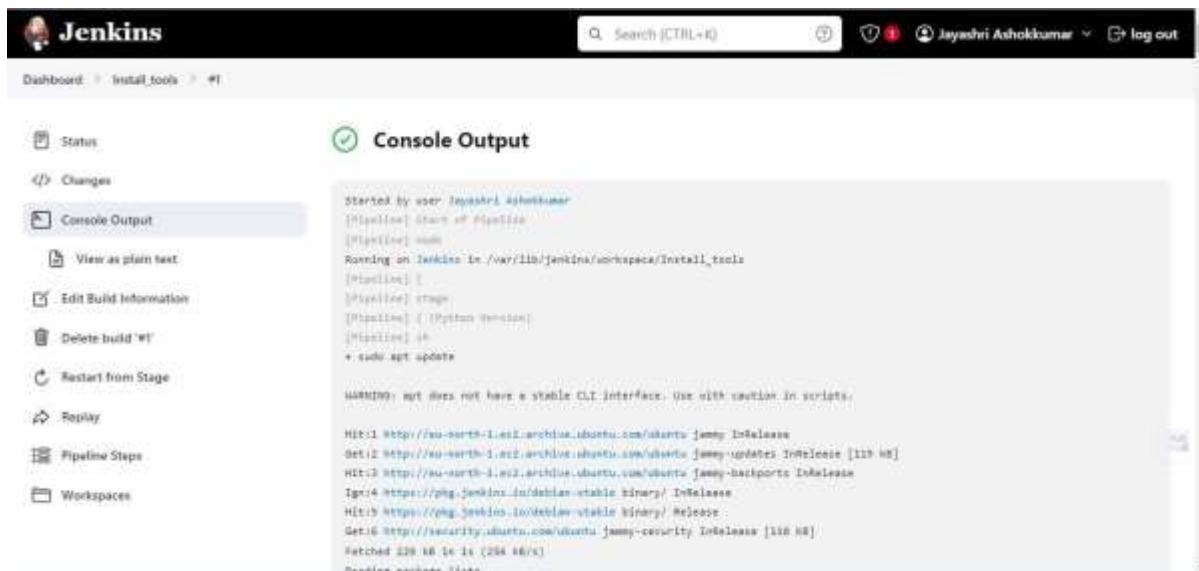
Now, Click on Save.

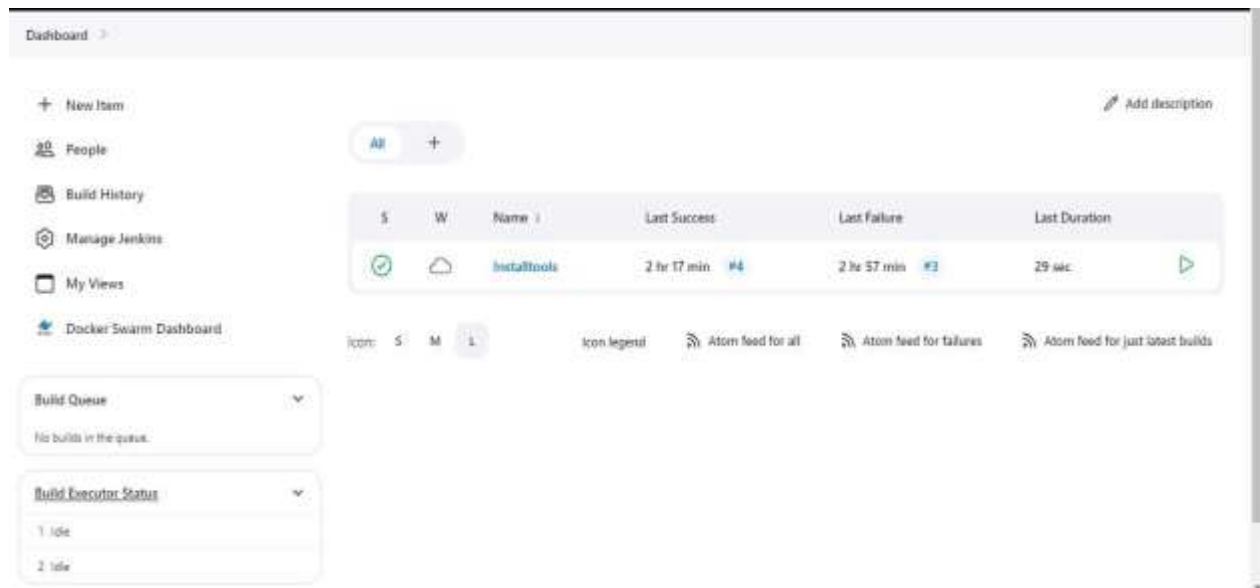


STEP 12: Click on the Install_tools and select the build now.

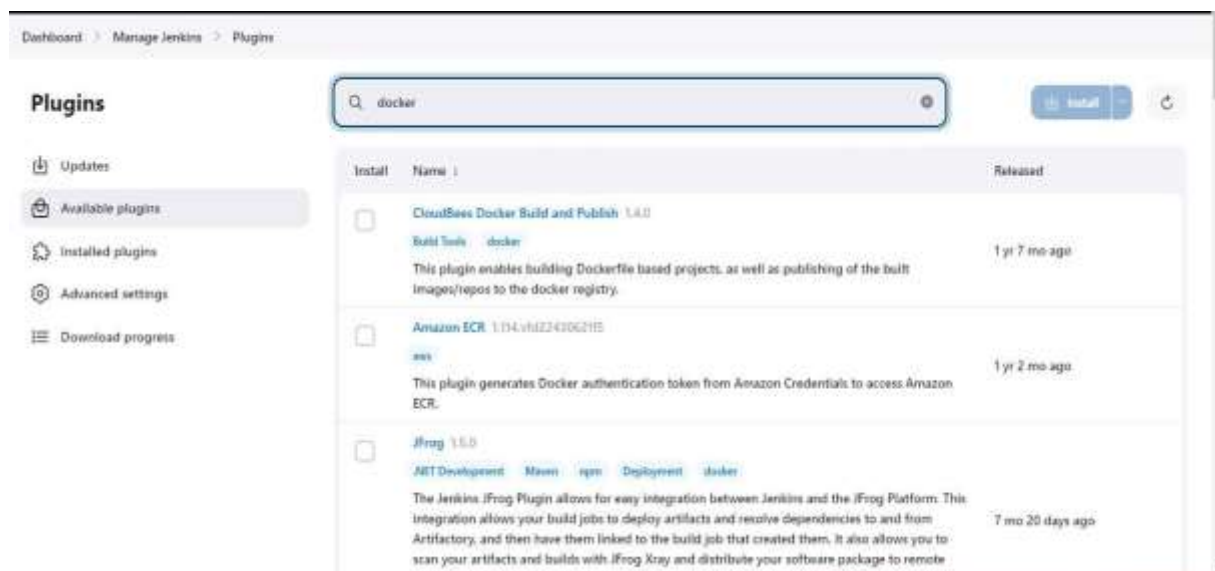


STEP 13: In the Build history section, click on the builds then you will be able to see the console output as 'FINISHED: SUCCESS'. Now The Pipeline is built.

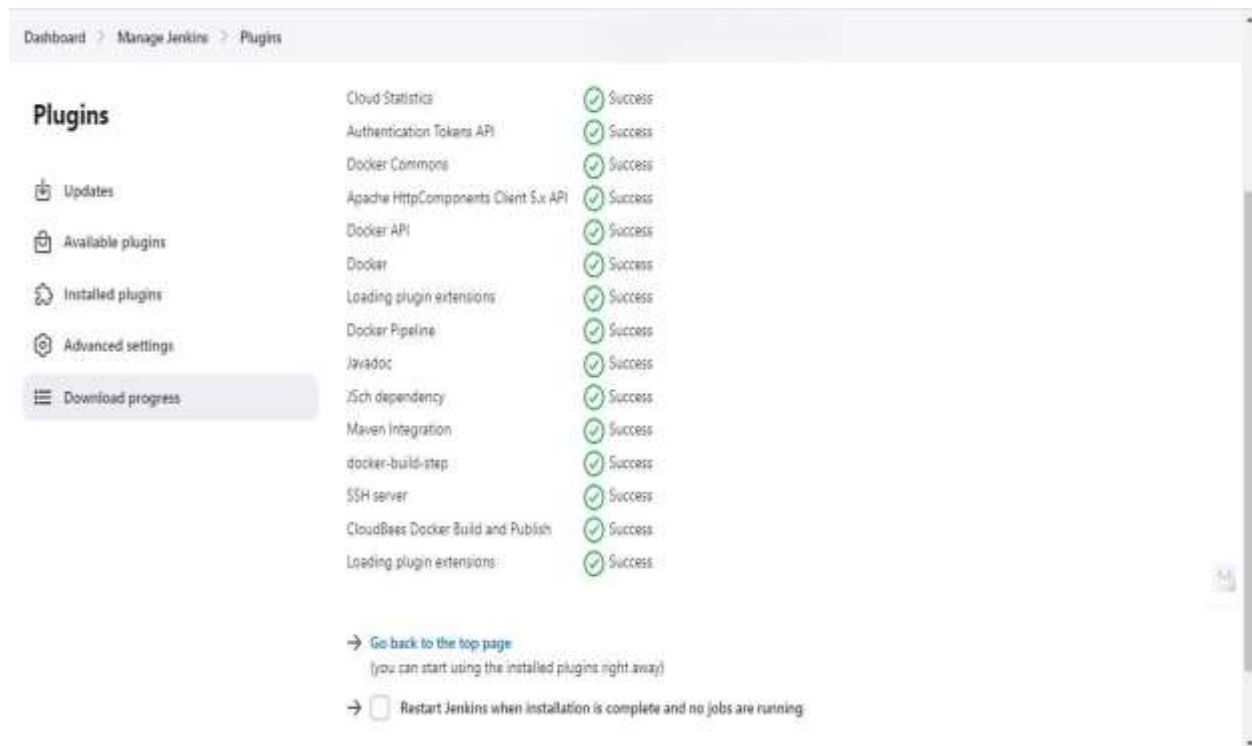




STEP 14: Click > Dashboard > Manage Jenkins> Plugins. In Plugins, Select Available Plugins. Here, search for docker and select the associated plugins (Docker, docker pipeline, docker build step, CloudBees Docker Build and Publish) then install it.



STEP 15: In Download progress, you will be able to see the set of plugins that were installed.



STEP 16: Run the following commands in the AWS ubuntu terminal.

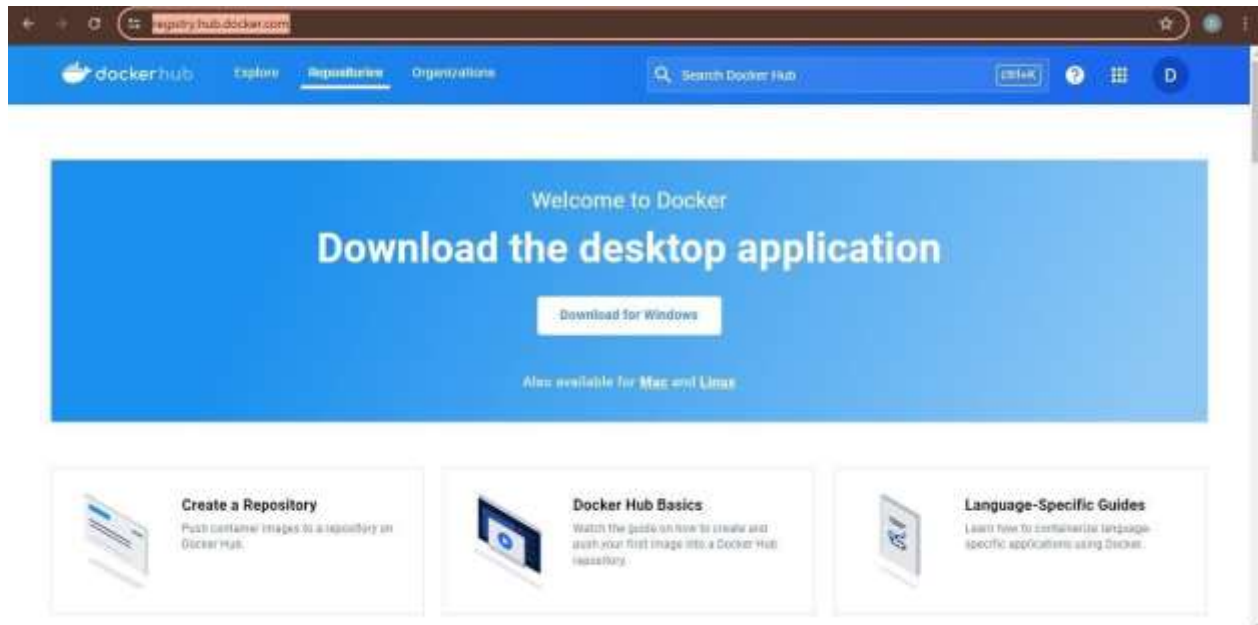
1. To Create the Docker Group:
`$sudo groupadd docker`
2. To Add the Jenkins User to the Docker Group:
`$sudo usermod -aG docker jenkins`
(Replace 'jenkins' with the actual username of the Jenkins user)
3. To Verify the Group Membership:
`$groups Jenkins`
4. To install the docker:
`$sudo apt install docker.io`
5. To restart jenkins:
`$sudo systemctl restart Jenkins`

STEP 17: Click > Manage Jenkins> Tools. Scroll for the docker section and perform the following actions in the image.

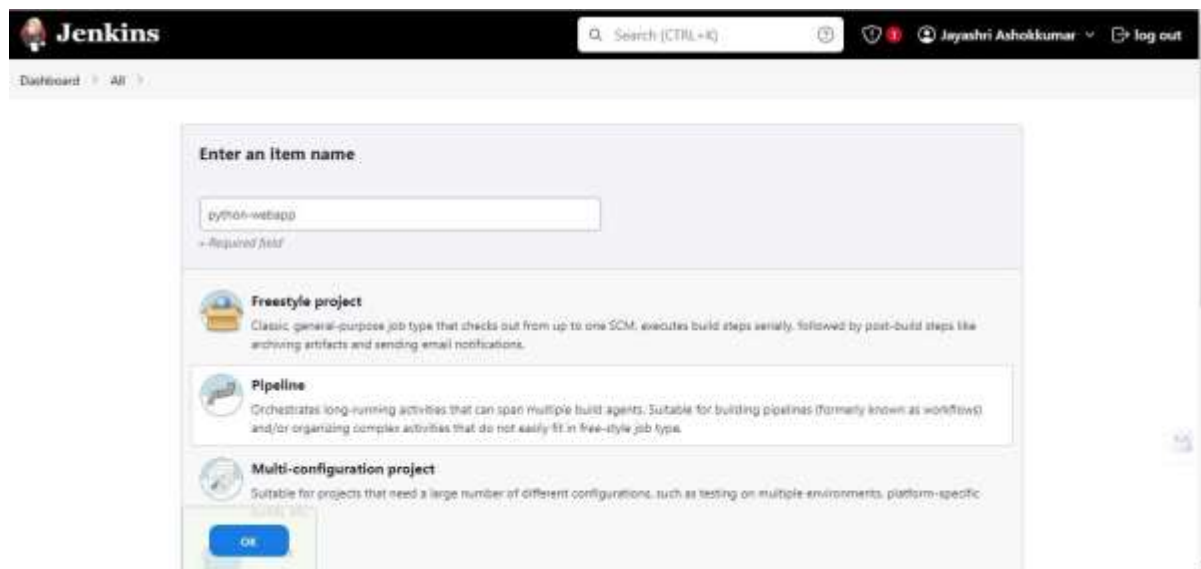


STEP 18: Open Browser and search for docker hub, then create an account in docker hub.

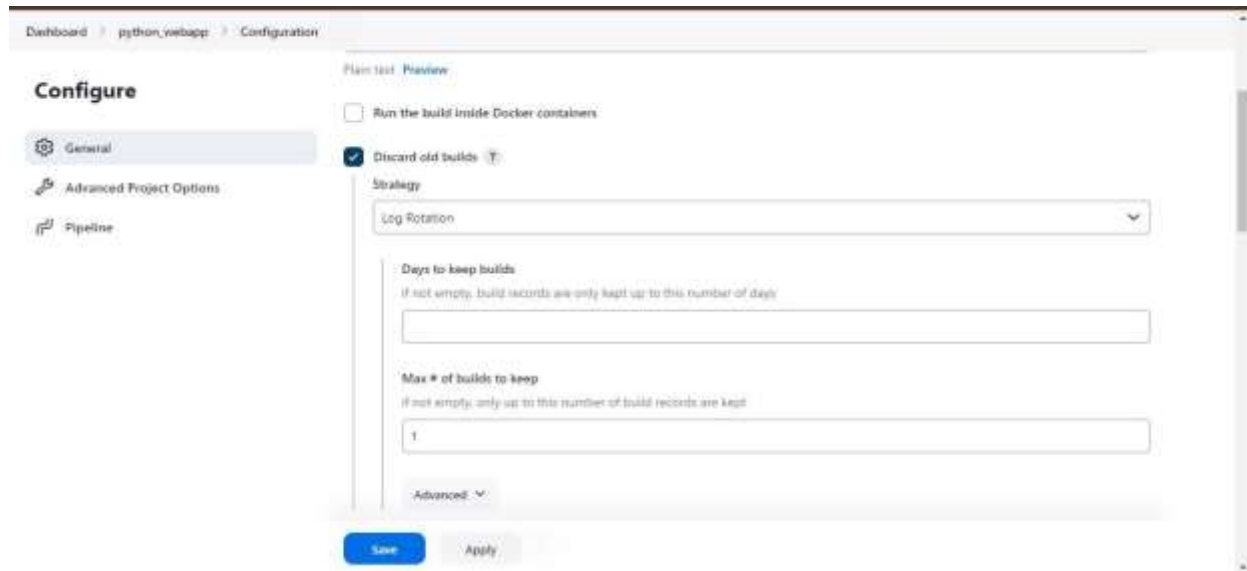




STEP 19: Click on the new item in the Jenkins dashboard. Now Enter the item name and select pipeline, then click on OK.



STEP 20: Click > dashboard > python-webapp > configure. In the general section, select 'Discard old builds' and set the max of builds to 1.



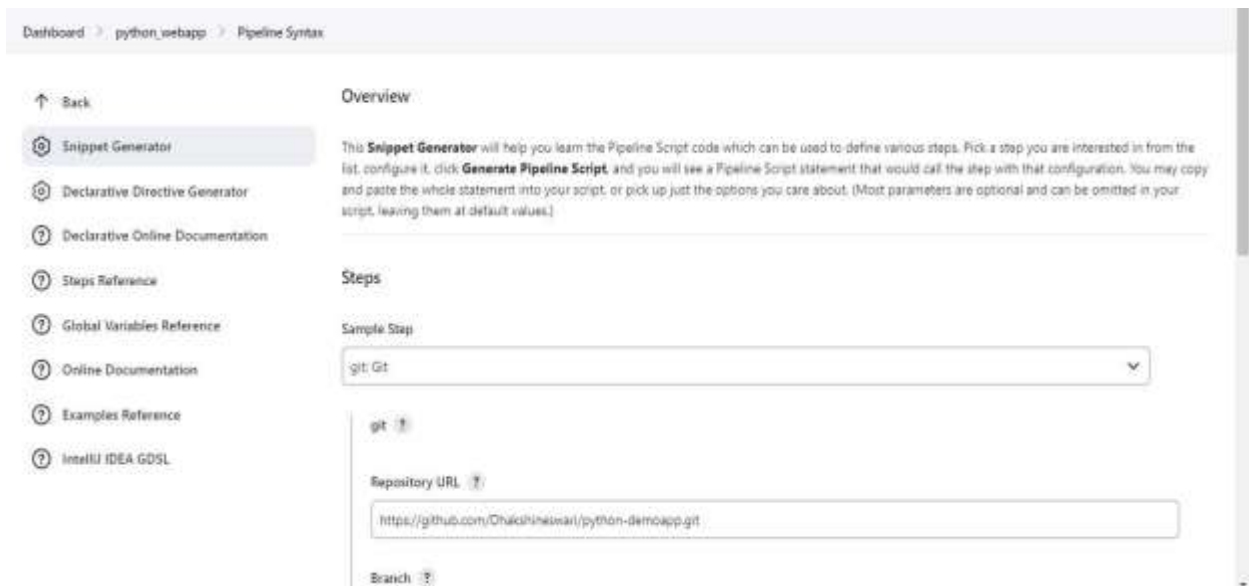
STEP 21: Scroll down for Pipeline. In the Script area, run the below code.

```
pipeline {
  agent any
  stages {
    stage('Git Checkout') {
      steps {
        git changelog: false, poll: false, url: ''
      }
    }
    stage('Docker Build') {
      steps {
        sh "make image"
      }
    }
    stage('Docker Push') {
      steps {
        script{
          withDockerRegistry(credentialsId: ' ', toolName: ' ') {
            sh "make push"
          }
        }
      }
    }
  }
}
```



STEP 22: Click on pipeline syntax and Select the Snippet generator

1. In the Sample step, choose the git: Git.



2. Paste your GitHub URL in the URL section.
3. Type as 'main' in Branch and deselect the options.

Dashboard > python_webapp > Pipeline Syntax

Examples Reference
IntelliJ IDEA GD5L

git ⓘ

Repository URL ⓘ
https://github.com/Dhakeshwaran/python-demoapp.git

Branch ⓘ
main

Credentials ⓘ
-> none -> ▼
+ Add +

☐ Include in polling? ⓘ

☐ Include in changelog? ⓘ

Generate Pipeline Script

4. Click on Generate pipeline script.

Dashboard > python_webapp > Pipeline Syntax

Include ⓘ
☐ Include in polling? ⓘ
☐ Include in changelog? ⓘ

Generate Pipeline Script

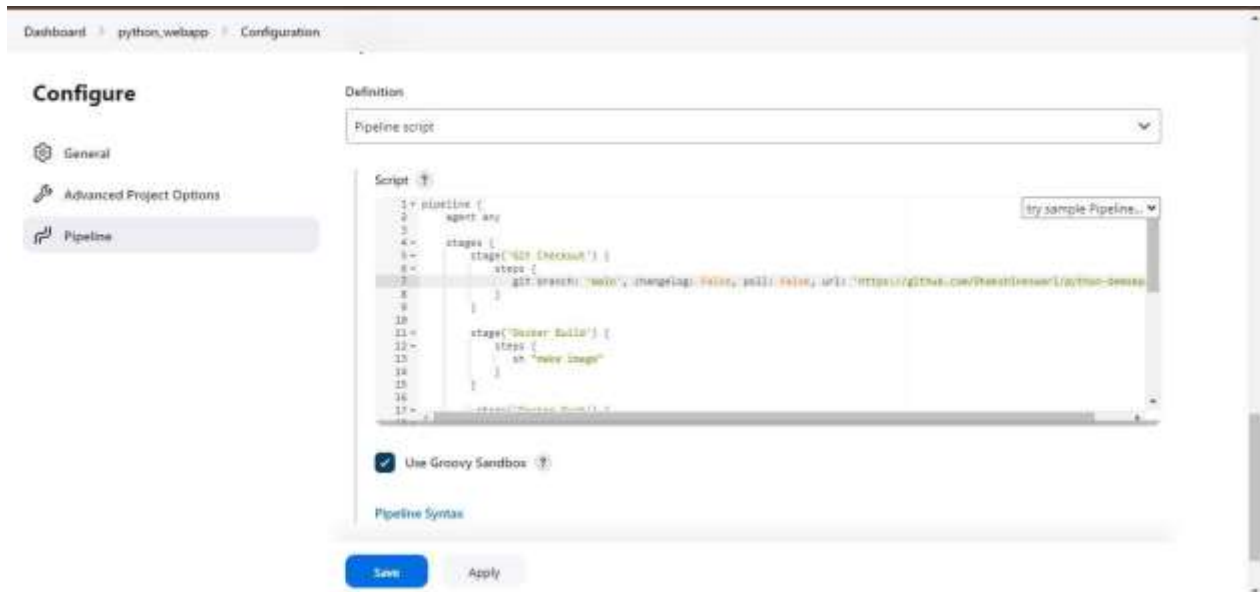
```
git branch: 'main', changelog: false, poll: false, url: 'https://github.com/Dhakeshwaran/python-demoapp.git'
```

Global Variables

There are many features of the Pipeline that are not steps. These are often exposed via global variables, which are not supported by the snippet generator. See the [Global Variables Reference](#) for details.

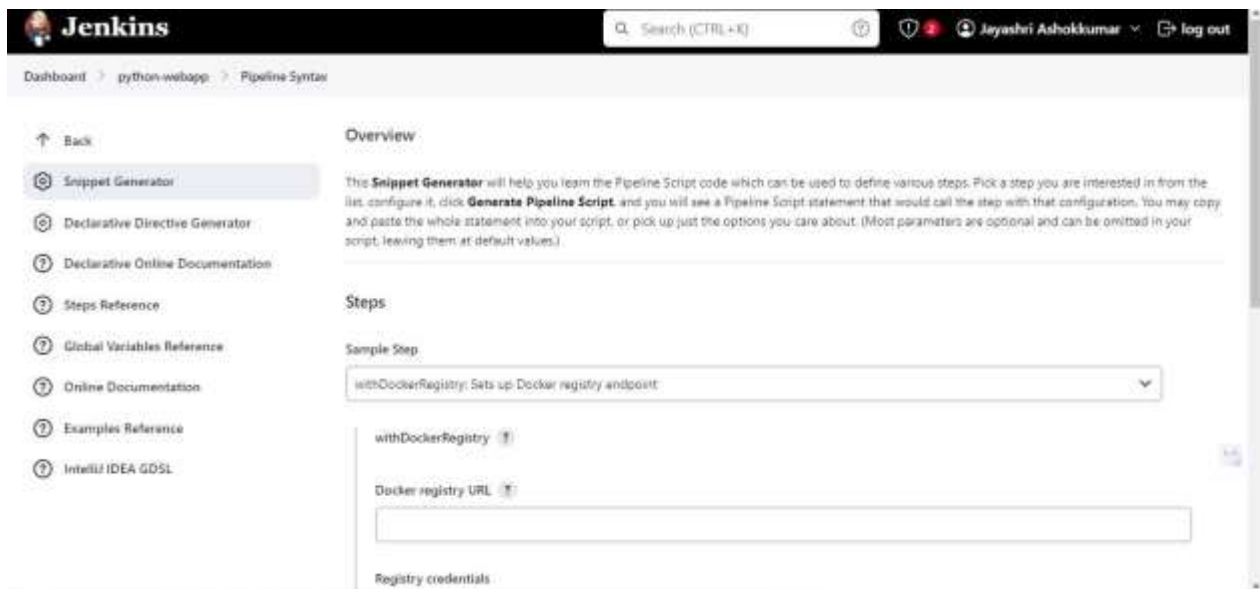
Jenkins 2.440.3

STEP 23: In Step 21, paste the above generator pipeline script in the URL portion of the code.



STEP 24: Again Click on pipeline and select the Snippet generator.

1. In the Sample step, choose the below option.



2. Make the Registry Credentials null and click Jenkins.

The screenshot shows the Jenkins Pipeline Syntax page. On the left, there is a sidebar with links: Global Variables Reference, Online Documentation, Examples Reference, and IntelliJ IDEA GDOL. The main area is titled 'Sample Step' and shows a dropdown menu with 'withDockerRegistry: Sets up Docker registry endpoint.' selected. Below this, there are fields for 'Docker registry URL', 'Registry credentials' (set to 'noah'), and 'Jenkins Credentials Provider' (set to 'Jenkins'). A 'Generate Pipeline Script' button is at the bottom.

3. Now, provide the username and password (GitHub) and click save.

The screenshot shows a dialog box titled 'Jenkins Credentials Provider: Jenkins'. It has fields for 'Username' (filled with 'jayashri_ashokkumar'), 'Password' (filled with 'password'), 'ID' (filled with 'docker-cred'), and 'Description' (filled with 'docker-cred'). There is a checkbox 'Treat username as secret' which is unchecked. At the bottom, there are 'Cancel' and 'Add' buttons.

4. The Credentials will have been added in the Registry Credentials section select the docker below, then click on Generate pipeline script.

Dashboard > python-webapp > Pipeline Syntax

Intelli IDEA GDOL

Docker registry URL

Registry credentials

Registry credentials: jayashri_ashokkumar/***** (docker-cred) [v]

+ Add

Docker installation

Docker installation: docker [v]

Generate Pipeline Script

```
// This step should not normally be used in your script. Consult the inline help for details.
withDockerRegistry(credentialsId: 'f3fed2f0-5a1d-4494-525b-14fb9043a35c', toolName: 'docker') {
    // some block
}
```

Copy

STEP 25: In Step 21, paste the above generator pipeline script in the Credentials ID and tool name portion of the code.

Dashboard > python-webapp > Configuration

Configure

General

Advanced Project Options

Pipeline

Pipeline

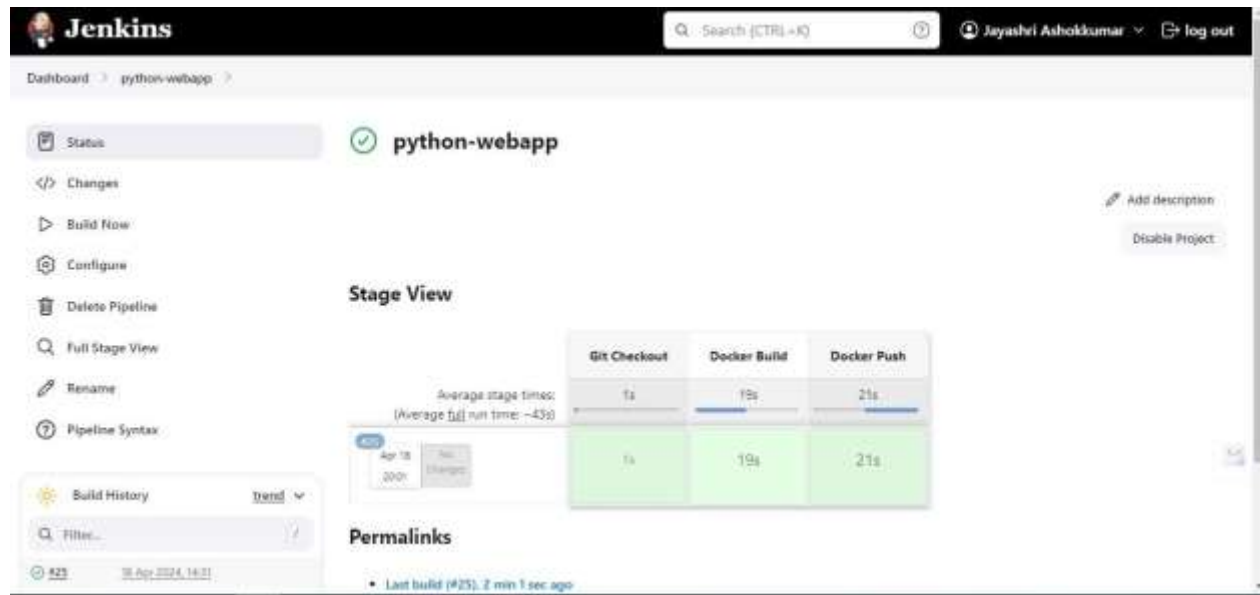
Pipeline Definition

Definition: Pipeline script [v]

Script

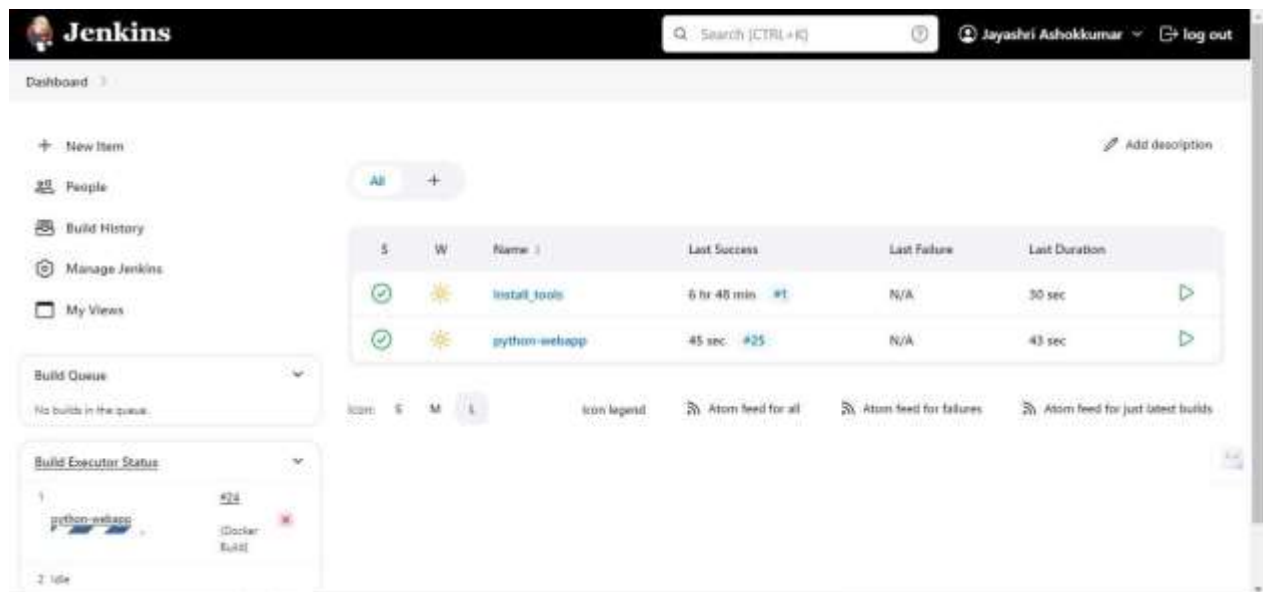
```
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
```

STEP 26: Click > dashboard > python-webapp > build now.



The screenshot shows the Jenkins dashboard for a project named 'python-webapp'. The left sidebar contains navigation links: Status, Changes, Build Now, Configure, Delete Pipeline, Full Stage View, Rename, and Pipeline Syntax. The main area displays the 'Stage View' for the 'python-webapp' pipeline. It shows three stages: 'Git Checkout' (1s), 'Docker Build' (19s), and 'Docker Push' (21s). The average stage time is 1s, and the average full run time is ~43s. Below the stage view, there is a 'Permalinks' section with a link to the last build (#25) from 2 min 1 sec ago. The top navigation bar includes a search bar, a user profile for 'Jayashri Ashokkumar', and a 'log out' button.

STEP 27: Now CI Pipeline has been built Successfully.



The screenshot shows the Jenkins dashboard with a list of builds. The left sidebar contains navigation links: New Item, People, Build History, Manage Jenkins, and My Views. The main area displays a table of builds. The table has columns: S, W, Name, Last Success, Last Failure, and Last Duration. The builds listed are 'install_tools' and 'python-webapp'. The 'python-webapp' build is highlighted in blue. Below the table, there is a 'Build Queue' section showing 'No builds in the queue.' and a 'Build Executor Status' section showing the status of the 'python-webapp' build. The top navigation bar includes a search bar, a user profile for 'Jayashri Ashokkumar', and a 'log out' button.

S	W	Name	Last Success	Last Failure	Last Duration
✓	☀	install_tools	6 hr 45 min #1	N/A	30 sec
✓	☀	python-webapp	45 sec #25	N/A	43 sec

RESULT:

Thus, the CI Pipeline has been successfully created.

Ex. No: 5 CREATE A CD PIPELINE IN JENKINS AND DEPLOY IN CLOUD

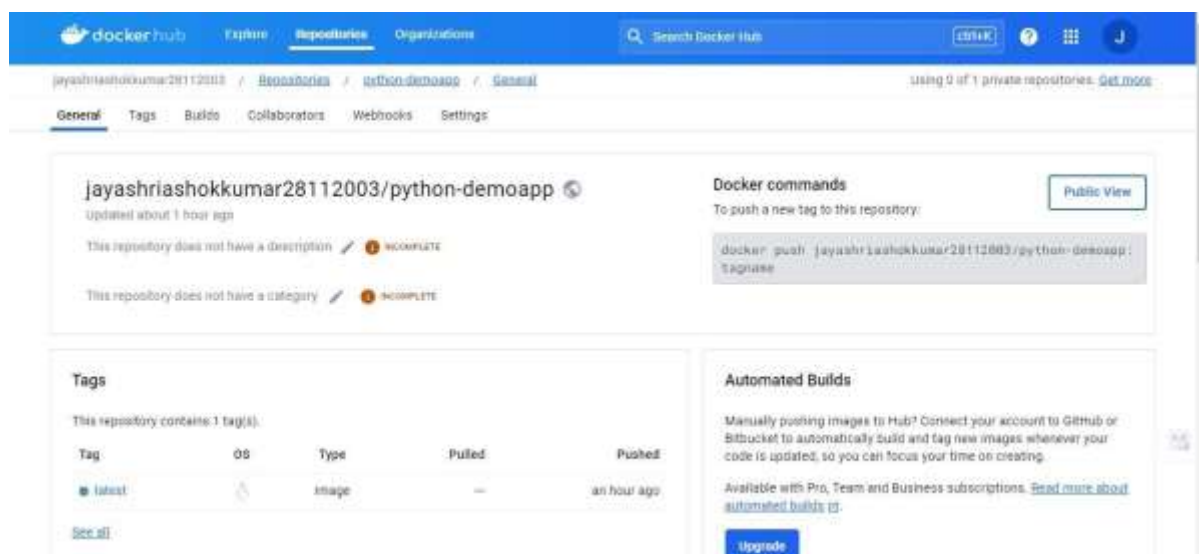
Date:

AIM:

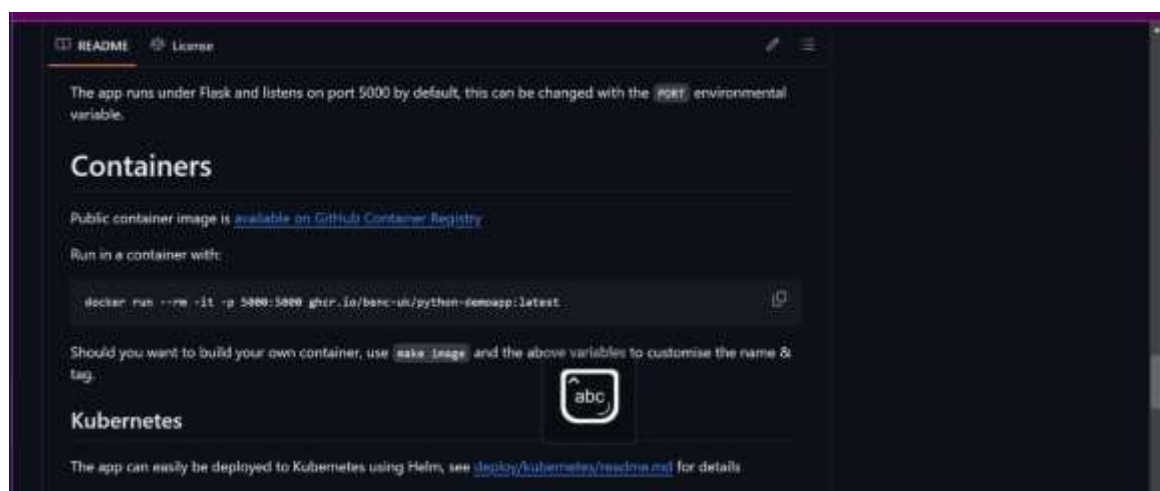
To create a CD pipeline in Jenkins and deploy it in the Cloud.

PROCEDURE:

STEP 1: Click Repositories > pythondemoapp > General. There, you will be able to see a command under Docker commands. Copy that command.

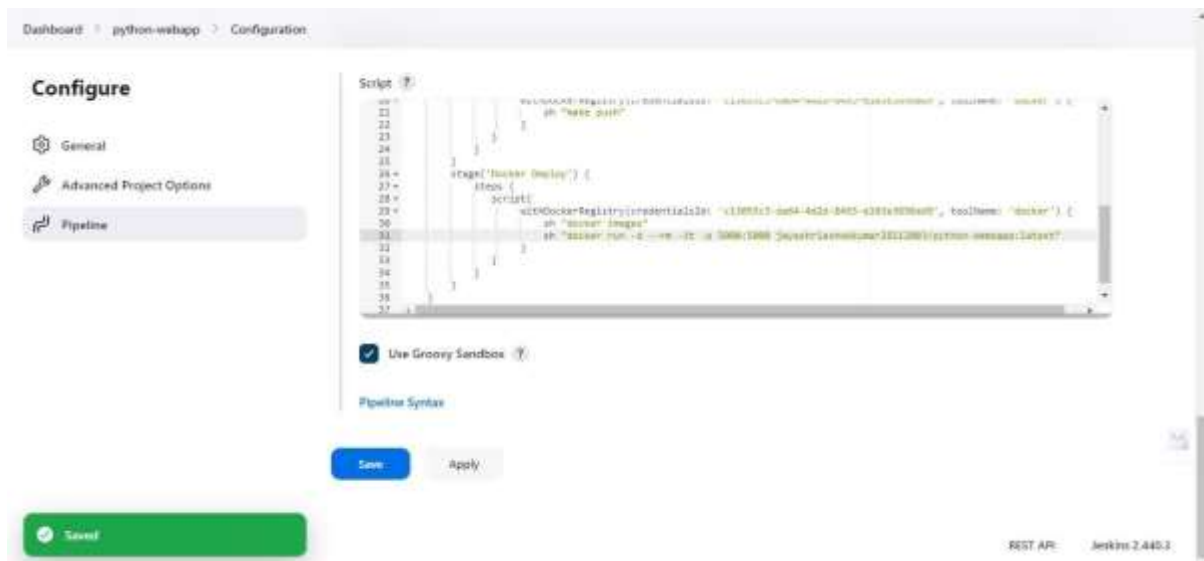


STEP 2: Now, go to your GitHub account. Scroll down to Containers and copy the command again for future use.

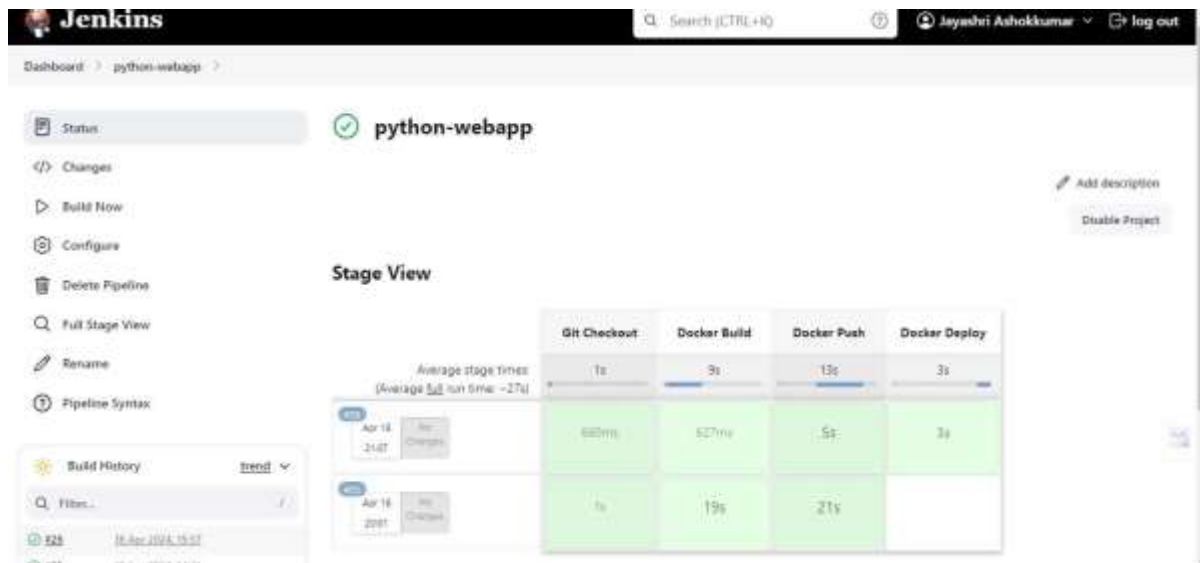


STEP 3: Now add the code (combination of the above commands) below to the code through which the CI Pipeline was built.

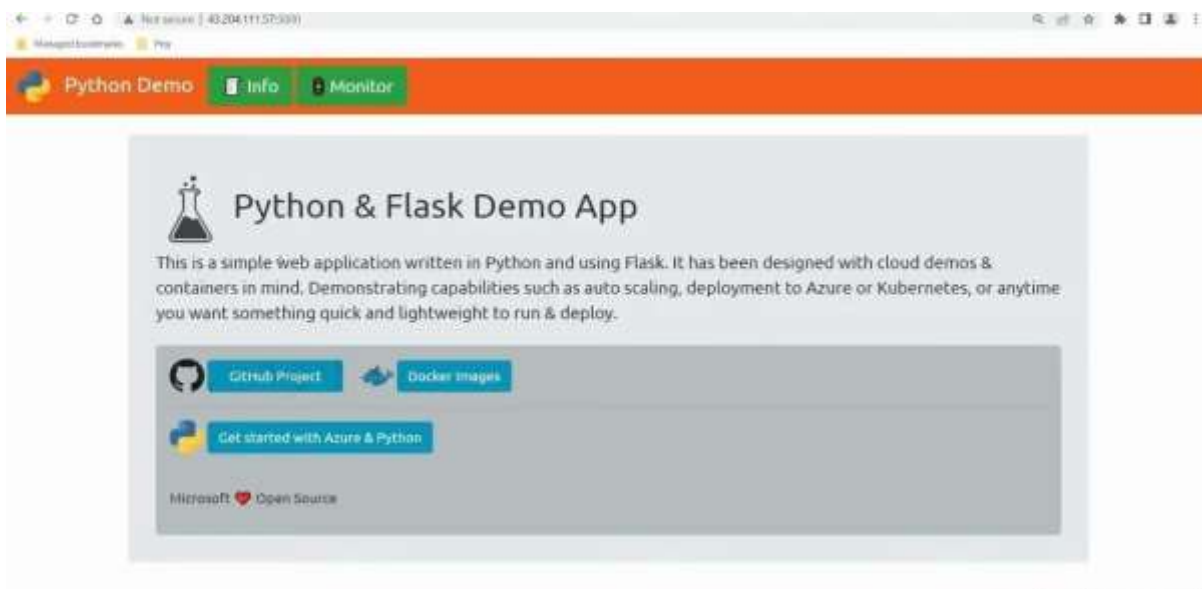
```
stage('Docker Deploy') {  
    steps {  
        script{  
            withDockerRegistry(credentialsId: 'dbc54abf-de2b-42b8-a0d2-f5539c9b8997',  
toolName: 'docker') {  
                sh "docker images"  
                sh "docker run -d --rm -it -p 80:5000 jayashriashokkumar28112003/python-  
webapp:latest"  
            }  
        }  
    }  
}
```



STEP 4: Click on Dashboard > python-webapp > Build Now



STEP 5: To access the Python web app copy and paste the public IP of the AWS instance and include port 5000 at the end of it. (For example: 41.204.111.57:5000)



RESULT:

Thus, the CD pipeline has been successfully created in Jenkins and deployed in the Cloud.

Ex. No: 6 CREATE AN ANSIBLE PLAYBOOK FOR A SIMPLE WEB
Date: APPLICATION INFRASTRUCTURE

AIM

To Create an Ansible Playbook for a simple web application infrastructure.

PROCEDURE

STEP 1: Create a directory named **ansible-practice1** and then navigate into that directory with the cd command.

The following commands are used to create a directory,

\$ cd ~

\$ mkdir ansible-practice1

\$ cd ansible-practice1

To add content to playbook use the following commands,

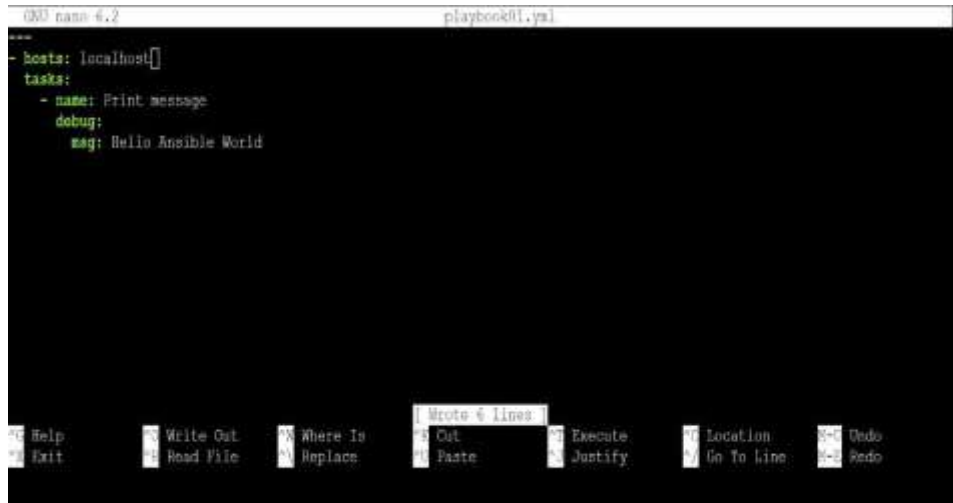
\$ nano playbook01.yml

```
ubuntu@ip-172-31-43-52:~/ansible-practice$ cd
ubuntu@ip-172-31-43-52:~$ mkdir ansible-practice1
ubuntu@ip-172-31-43-52:~$ cd ansible-practice1
ubuntu@ip-172-31-43-52:~/ansible-practice1$ nano playbook01.yml
ubuntu@ip-172-31-43-52:~/ansible-practice1$
```

STEP 2: Now create a playbook that contains the following content to be executed.
The content to be added in the playbook (**playbook01.yml**),

```
---
- hosts: localhost
  tasks:
    - name: Print message
      debug:
        msg: Hello Ansible World
```

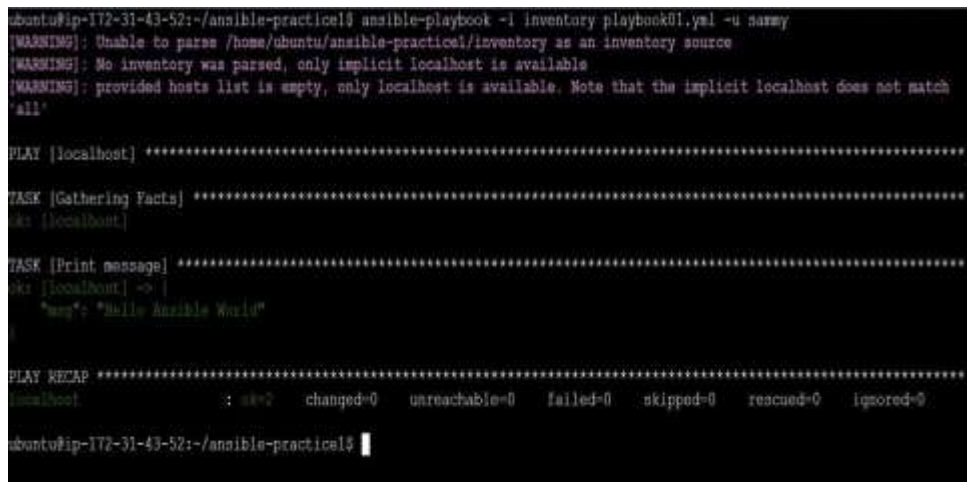
NOTE: The host name in the content must be changed to localhost.



```
---
- hosts: localhost
  tasks:
    - name: Print message
      debug:
        msg: Hello Ansible World
```

STEP 3: Run **ansible-playbook** with the same connection arguments, use an inventory file named **inventory** and the **sammy** user to connect to the remote server. The command to run the ansible playbook,

```
$ ansible-playbook -i inventory playbook-01.yml -u sammy
```



```
ubuntu@ip-172-31-43-52:~/ansible-practice1$ ansible-playbook -i inventory playbook01.yml -u sammy
[WARNING]: Unable to parse /home/ubuntu/ansible-practice1/inventory as an inventory source
[WARNING]: No inventory was parsed, only implicit localhost is available
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'

PLAY [localhost] *****

TASK [Gathering Facts] *****
ok: [localhost]

TASK [Print message] *****
ok: [localhost] => |
  "msg": "Hello Ansible World"

PLAY RECAP *****
localhost                : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

ubuntu@ip-172-31-43-52:~/ansible-practice1$
```

RESULT:

Thus, creating and executing an ansible playbook for a simple web application infrastructure has been executed successfully.

Ex. No: 7

BUILD A SIMPLE APPLICATION USING GRADLE

Date:

AIM

To build a simple application using Gradle.

PROCEDURE

Step 1: Create directory: `java_application`.

Step 2: Change to directory: `java_application`.

Step 3: Run `gradle init` command.

Step 4: Select project type (application).

Step 5: Choose implementation language (Java).

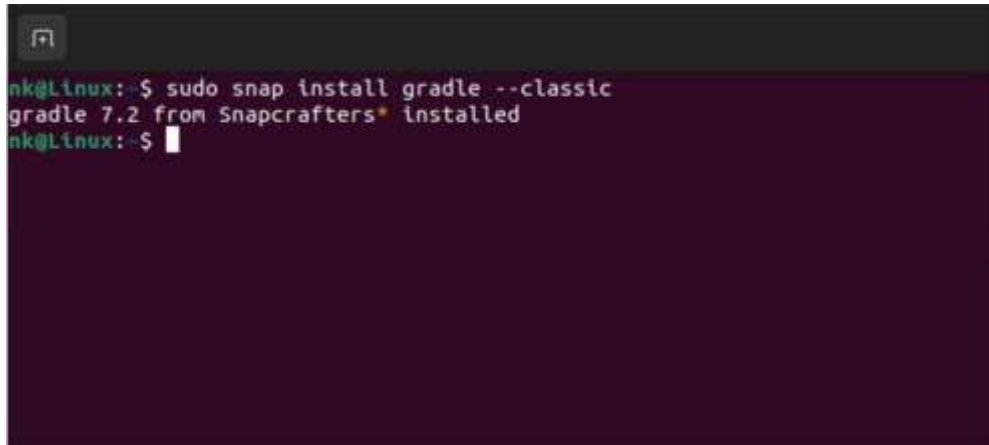
Step 6: Select build script language (Groovy).

Step 7: Choose testing framework (JUnit 4).

Step 8: Enter project name.

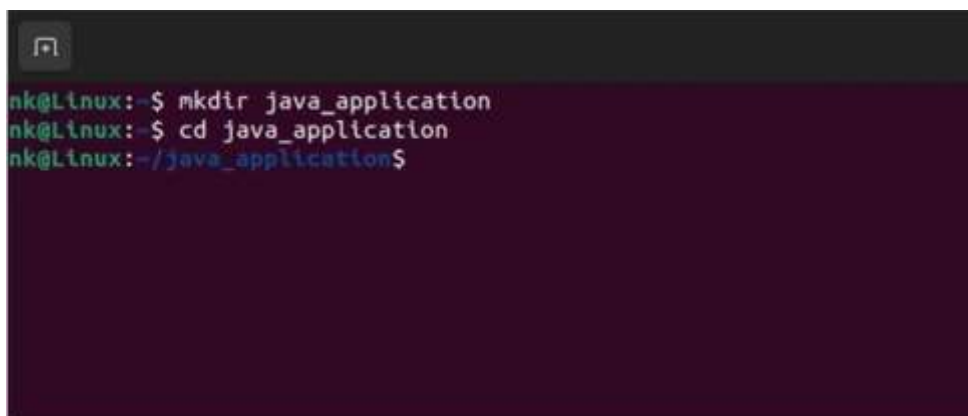
Step 9: Enter source package.

1. Install gradle using the following command



```
nk@Linux:~$ sudo snap install gradle --classic
gradle 7.2 from Snapcrafters* installed
nk@Linux:~$
```

2. Create a directory named **java_application** and change directory:



```
nk@Linux:~$ mkdir java_application
nk@Linux:~$ cd java_application
nk@Linux:~/java_application$
```

3. Run the **gradle init** command to create a new Gradle project

```
nk@Linux:~/java_application$ gradle init

Welcome to Gradle 7.2!

Here are the highlights of this release:
- Toolchain support for Scala
- More cache hits when Java source files have platform-specific line endings
- More resilient remote HTTP build cache behavior

For more details see https://docs.gradle.org/7.2/release-notes.html

Starting a Gradle Daemon (subsequent builds will be faster)

Select type of project to generate:
 1: basic
 2: application
 3: library
 4: Gradle plugin
Enter selection (default: basic) [1..4]
```

4. Select the project type (application, library, etc.) by typing `2` and pressing Enter for an application.

```
Select type of project to generate:
 1: basic
 2: application
 3: library
 4: Gradle plugin
Enter selection (default: basic) [1..4] 2
```

5. Choose the implementation language (Java, Kotlin, etc.) by typing `3` and pressing Enter for Java.

```
Select implementation language:
 1: C++
 2: Groovy
 3: Java
 4: Kotlin
 5: Scala
 6: Swift
Enter selection (default: Java) [1..6] 3
```

6. Select the default build script language (Groovy or Kotlin) by typing `1` and pressing Enter for Groovy.

```
Split functionality across multiple subprojects?:
  1: no - only one application project
  2: yes - application and library projects
Enter selection (default: no - only one application project) [1..2] 1

Select build script DSL:
  1: Groovy
  2: Kotlin
Enter selection (default: Groovy) [1..2]
```

7. Choose the testing framework (JUnit 4, TestNG, etc.) if prompted. The default is JUnit4

```
Select test framework:
  1: JUnit 4
  2: TestNG
  3: Spock
  4: JUnit Jupiter
Enter selection (default: JUnit Jupiter) [1..4]
```

8. Enter the project name when prompted (default is the directory name) & Enter the source package when prompted (default is the directory name).

```
nk@Linux: ~/java_application

- More resilient remote HTTP build cache behavior
For more details see https://docs.gradle.org/7.2/release-notes.html
Starting a Gradle Daemon (subsequent builds will be faster)

Select type of project to generate:
  1: basic
  2: application
  3: library
  4: Gradle plugin
Enter selection (default: basic) [1..4] 2

Select implementation language:
  1: C++
  2: Groovy
  3: Java
  4: Kotlin
  5: Scala
  6: Swift
Enter selection (default: Java) [1..6] 3

Split functionality across multiple subprojects?:
  1: no - only one application project
  2: yes - application and library projects
Enter selection (default: no - only one application project) [1..2] 1

Select build script DSL:
  1: Groovy
  2: Kotlin
Enter selection (default: Groovy) [1..2] 1

Select test framework:
  1: JUnit 4
  2: TestNG
  3: Spock
  4: JUnit Jupiter
Enter selection (default: JUnit Jupiter) [1..4]

Project name (default: java_application):
Source package (default: java_application):

> Task :init
Get more help with your project: https://docs.gradle.org/7.2/samples/sample_building_java_applications.html

BUILD SUCCESSFUL in 1m 23s
2 actionable tasks: 2 executed
nk@Linux: ~/java_application$
```

OUTPUT



```
dhineshkumar1729@dhineshkumar1729-VirtualBox:~$ cd java_application/
dhineshkumar1729@dhineshkumar1729-VirtualBox:~/java_application$ ./gradlew run
Downloading https://services.gradle.org/distributions/gradle-7.2-bin.zip
.....10%.....20%.....30%.....40%.....50%.....
60%.....70%.....80%.....90%.....100%
Starting a Gradle Daemon, 1 incompatible Daemon could not be reused, use --status
s for details

> Task :app:run
Hello World!

BUILD SUCCESSFUL in 37s
2 actionable tasks: 2 executed
dhineshkumar1729@dhineshkumar1729-VirtualBox:~/java_application$
```

RESULT

Thus, the Java application was successfully created using Gradle.


```

ubuntu@ip-172-31-43-52:~$ sudo apt-add-repository --yes --update ppa:ansible/ansible
Repository: 'deb https://ppa.launchpadcontent.net/ansible/ansible/ubuntu/ jammy main'
Description:
Ansible is a radically simple IT automation platform that makes your applications and systems easier to deploy. Avoid writing scripts or custom code to deploy and update your applications- automate in a language that approaches plain English, using SSH, with no agents to install on remote systems.
http://ansible.com/

If you face any issues while installing Ansible PPA, file an issue here:
https://github.com/ansible-community/ppa/issues
More info: https://launchpad.net/~ansible/+archive/ubuntu/ansible
Adding repository.
Adding deb entry to /etc/apt/sources.list.d/ansible-ubuntu-ansible-jammy.list
Adding disabled deb-src entry to /etc/apt/sources.list.d/ansible-ubuntu-ansible-jammy.list
Adding key to /etc/apt/trusted.gpg.d/ansible-ubuntu-ansible.gpg with fingerprint 6125E2A9c77F2810F87BD15B93C4A3FD7BB9C3
67
Hit:1 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Ign:4 https://pkg.jenkins.io/debian-stable binary/ InRelease
Hit:5 https://pkg.jenkins.io/debian-stable binary/ Release
Hit:6 http://security.ubuntu.com/ubuntu jammy-security InRelease
Get:7 https://ppa.launchpadcontent.net/ansible/ansible/ubuntu jammy InRelease [18.0 kB]
Get:9 https://ppa.launchpadcontent.net/ansible/ansible/ubuntu jammy/main amd64 Packages [1120 B]
Get:10 https://ppa.launchpadcontent.net/ansible/ansible/ubuntu jammy/main Translation-en [752 B]
Fetched 19.9 kB in 1s (19.8 kB/s)
Reading package lists... Done

```

Step 3: Now install the ansible latest version using the command,

`$ sudo apt install ansible`

```

ubuntu@ip-172-31-43-52:~$ sudo apt install ansible
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ansible is already the newest version (9.4.0-1ppa-jammy).
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
ubuntu@ip-172-31-43-52:~$

```

Now we are going to navigate the directory path to,

`$ cd /etc/ansible/`

Then, we will check the latest version of the ansible by,

`$ ansible --version`

```

ubuntu@ip-172-31-43-52:~$ cd /etc/ansible/
ubuntu@ip-172-31-43-52:/etc/ansible$ ansible --version
ansible [core 2.16.5]
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/home/ubuntu/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  ansible collection location = /home/ubuntu/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.10.12 (main, Nov 20 2023, 19:14:05) [GCC 11.4.0] (/usr/bin/python3)
  jinja version = 3.0.3
  libyaml = True
ubuntu@ip-172-31-43-52:/etc/ansible$

```

Step 4: To check the list of directories and files present in the ansible, run the command as,

`$ ls -la`

```

ubuntu@ip-172-31-43-52:/etc/ansible$ ls -la
total 20
drwxr-xr-x  3 root root 4096 Apr 12 13:51 .
drwxr-xr-x 103 root root 4096 Apr 12 13:51 ..
-rw-r--r--  1 root root  614 Mar 26 17:11 ansible.cfg
-rw-r--r--  1 root root 1175 Mar 26 17:11 hosts
drwxr-xr-x  2 root root 4096 Mar 26 17:11 roles
ubuntu@ip-172-31-43-52:/etc/ansible$

```


Then check the host file by,

\$ sudo nano hosts

```
GNU nano 6.2 hosts
# This is the default ansible 'hosts' file.
#
# It should live in /etc/ansible/hosts
#
# - Comments begin with the '#' character
# - Blank lines are ignored
# - Groups of hosts are delimited by [header] elements
# - You can enter hostnames or ip addresses
# - A hostname/ip can be a member of multiple groups
#
# Ex 1: Ungrouped hosts, specify before any group headers:
## green.example.com
## blue.example.com
## 192.168.190.1
## 192.168.190.10
#
# Ex 2: A collection of hosts belonging to the 'webserver' group:
#
# Help      Write Out  Where Is  Read 54 lines  Execute  Location  Undo
# Exit     Read File  Replace   Cut           Justify  Go To line  Redo
```

Step 5: Create an ansible role from scratch and run ansible-galaxy command.

\$ ansible-galaxy init my-role

\$ ls

```
ubuntu@ip-172-31-43-52:/etc/ansible$ sudo ansible-galaxy init role-name
- Role role-name was created successfully
ubuntu@ip-172-31-43-52:/etc/ansible$ ls
ansible.cfg  hosts  playbooks  role-name  roles
ubuntu@ip-172-31-43-52:/etc/ansible$
```

To view the role directory structure, run the tree command followed by the role name.

```
ubuntu@ip-172-31-43-52:/etc/ansible$ tree role-name/
role-name/
├── README.md
├── defaults
│   └── main.yml
├── files
├── handlers
│   └── main.yml
├── meta
│   └── main.yml
├── tasks
│   └── main.yml
├── templates
├── tests
│   ├── inventory
│   └── test.yml
└── vars
    └── main.yml

8 directories, 8 files
ubuntu@ip-172-31-43-52:/etc/ansible$
```


Step 6: Create three ansible roles as follows:

The prerequisites role – Installs git.

The postgresql role – Installs postgresql service.

The apache role – Installs the Apache webserver.

Use the following commands to create the above,

```
$ sudo ansible-galaxy init prerequisites
```

```
$ sudo ansible-galaxy init PostgreSQL
```

```
$ sudo ansible-galaxy init apache
```

```
$ ls
```

```
ubuntu@ip-172-31-43-52:/etc/ansible$ sudo ansible-galaxy init prerequisites
- Role prerequisites was created successfully
ubuntu@ip-172-31-43-52:/etc/ansible$ sudo ansible-galaxy init PostgreSQL
- Role PostgreSQL was created successfully
ubuntu@ip-172-31-43-52:/etc/ansible$ sudo ansible-galaxy init apache
- Role apache was created successfully
ubuntu@ip-172-31-43-52:/etc/ansible$ ls
PostgreSQL  ansible.cfg  apache  hosts  playbooks  prerequisites  role-name  roles
ubuntu@ip-172-31-43-52:/etc/ansible$
```

Step 7: Define each role that you have created. To achieve this, you need to edit the main.yml file located in the ‘tasks’ folder for each role.

First for prerequisites,

```
$ cd prerequisites/tasks/
```

Then edit the main.yml file by the command,

```
$ sudo nano main.yml
```

```
ubuntu@ip-172-31-43-52:/etc/ansible$ cd prerequisites/tasks/
ubuntu@ip-172-31-43-52:/etc/ansible/prerequisites/tasks$ sudo nano main.yml
ubuntu@ip-172-31-43-52:/etc/ansible/prerequisites/tasks$
```

Then add the content to this main.yml file as

```
# tasks file for prerequisites
```

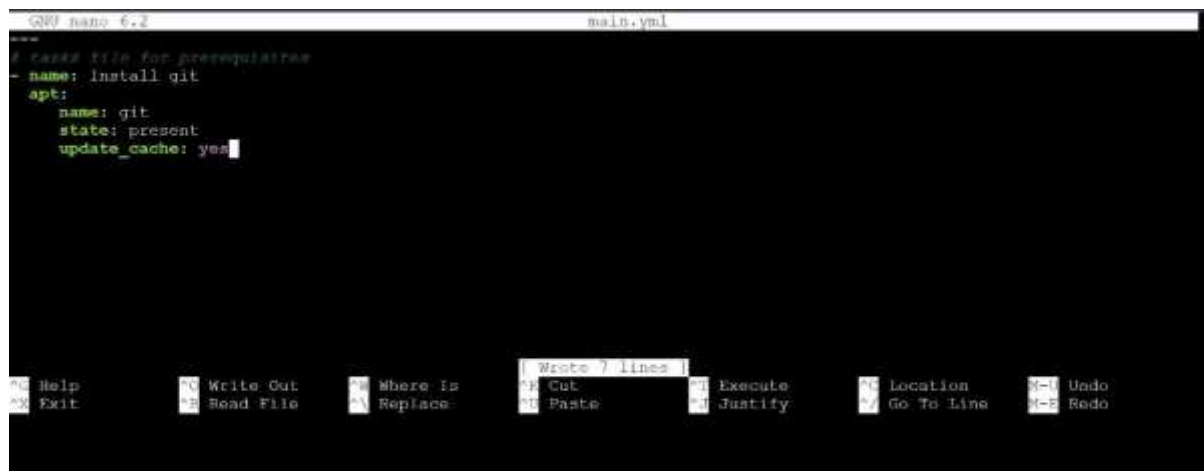
```
- name: Install git
```

```
  apt:
```

```
    name: git
```

```
    state: present
```

```
    update_cache: yes
```

A screenshot of a terminal window with the nano 6.2 editor. The title bar shows 'GNU nano 6.2' and 'main.yml'. The editor content is a YAML file for prerequisites. The first task is named 'install git' and has properties: name: git, state: present, and update_cache: yes. The bottom status bar shows various keyboard shortcuts for editor functions like Help, Exit, Write Out, Read File, etc.

```
GNU nano 6.2 main.yml
# tasks file for prerequisites
- name: install git
  apt:
    name: git
    state: present
    update_cache: yes
```

Next for PostgreSQL,

\$ cd PostgreSQL/tasks/

\$ sudo nano main.yml

Then add the content to this main.yml file as,

tasks file for PostgreSQL

- name: Install PostgreSQL

apt:

name: postgresql

state: present

update_cache: yes

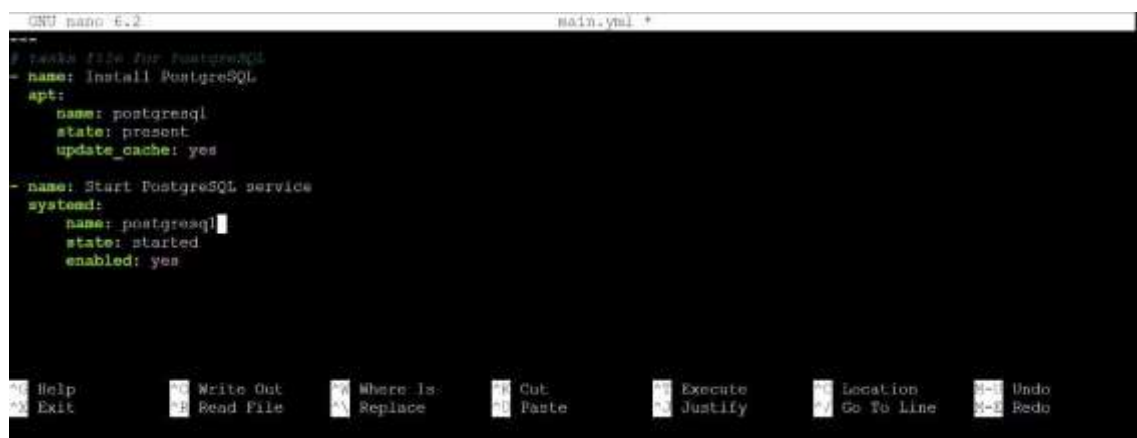
- name: Start PostgreSQL service

systemd:

name: postgresql

state: started

enabled: yes

A screenshot of a terminal window with the nano 6.2 editor. The title bar shows 'GNU nano 6.2' and 'main.yml *'. The editor content is a YAML file for PostgreSQL tasks. The first task is named 'Install PostgreSQL' and has properties: name: postgresql, state: present, and update_cache: yes. The second task is named 'Start PostgreSQL service' and has properties: name: postgresql, state: started, and enabled: yes. The bottom status bar shows various keyboard shortcuts for editor functions like Help, Exit, Write Out, Read File, etc.

```
GNU nano 6.2 main.yml *
# tasks file for PostgreSQL
- name: Install PostgreSQL
  apt:
    name: postgresql
    state: present
    update_cache: yes

- name: Start PostgreSQL service
  systemd:
    name: postgresql
    state: started
    enabled: yes
```

Finally for the apache web server,

```
$ cd apache/tasks/
```

```
$ sudo nano main.yml
```

Then add the content to this main.yml file as,

```
# tasks file for apache
```

```
- name: install Apache web server
```

```
  apt:
```

```
    name: apache2
```

```
    state: present
```

```
    update_cache: yes
```



```
GNU nano 2.2.6 main.yml
# tasks file for apache
- name: install Apache web server
  apt:
    name: apache2
    state: present
    update_cache: yes
```

Lastly, we are going to create a playbook file called stack.yml and call the roles by the command,

```
$ sudo nano stack.yml
```

Then add the content to this main.yml file as

```
- hosts: localhost
```

```
become: true
```

```
roles:
```

- prerequisites
- PostgreSQL
- apache

```
GNU nano 6.2 stack.yml
---
- hosts: localhost
  become: true
  roles:
    - prerequisites
    - PostgreSQL
    - apache

Help      Write Out  Where To  Write 7 lines  Execute   Location  Undo
Exit      Read File    Replace   Cut           Justify   Go To Line Redo
```

Step 8: Ensure that our roles are working as expected, and run the ansible playbook.

Run the following command,

```
$ sudo ansible-playbook stack.yml
```

```
ubuntu@ip-172-31-43-52:/etc/ansible$ sudo ansible-playbook stack.yml
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'

PLAY [localhost] *****

TASK [Gathering Facts] *****
ok: [localhost]

TASK [prerequisites : Install git] *****
ok: [localhost]

TASK [PostgreSQL : Install PostgreSQL] *****
changed: [localhost]

TASK [PostgreSQL : Start PostgreSQL service] *****
ok: [localhost]

TASK [apache : install Apache web server] *****
changed: [localhost]

PLAY RECAP *****
localhost                : ok=5    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

ubuntu@ip-172-31-43-52:/etc/ansible$
```

Then check the versions to ensure that the packages were installed successfully,

```
$ git --version
```

```
$ apachectl -v
```

```
ubuntu@ip-172-31-43-52:/etc/ansible$ apachectl -v
Server version: Apache/2.4.52 (Ubuntu)
Server built:   2024-04-10T17:45:18
ubuntu@ip-172-31-43-52:/etc/ansible$ git --version
git version 2.34.1
ubuntu@ip-172-31-43-52:/etc/ansible$
```

RESULT

Thus, installing ansible, configuring ansible roles and writing playbooks have been executed successfully.