SRI KRISHNA COLLEGE OF TECHNOLOGY

An Autonomous Institution, (Approved by AICTE and affiliated to Anna University) Accredited by NAAC with "A" grade

Kovaipudur, Coimbatore-641042 Tamil Nadu, India

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

22ITE04-DEVOPS

Name : ……

Register No. : ……

Degree & Branch : ……

Class & Semester : ……

Academic Year :……

SRI KRISHNA COLLEGE OF TECHNOLOGY

(An Autonomous Institution)
Approved by AICTE | Affiliated to Anna University Chennai|
Accredited by NBA - AICTE| Accredited by NAAC with 'A' Grade
KOVAIPUDUR, COIMBATORE 641042

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

22ITE04 – DEVOPS

REGISTER NUMBER:

Certified that this is the Bonafide record of work done by

Mr / Ms.……. of the **Fifth Semester** Computer Science and Engineering during the academic year **2024 – 2025 (ODD SEMESTER).**

FACULTY IN-CHARGE **HOD/CSE**

Submitted for the End Semester Practical Examination held on …… at **SRI KRISHNA COLLEGE OF TECHNOLOGY**, Kovaipudur, Coimbatore-42.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

SRI KRISHNA COLLEGE OF TECHNOLOGY

An Autonomous Institution, (Approved by AICTE and affiliated to Anna University) Accredited by NAAC with "A" grade

Kovaipudur, Coimbatore-641042 Tamil Nadu, India

RUBRIC ASSESSMENT FOR: 22ITE04 – DevOps

Items	Excellent	Good	Satisfactory	Needs Improvement
AIM & ALGORITHM (20 MARKS)	16-20	11-15	6-10	1-5
	Explained extensive theory in relation to experiment conducted, Compliment with illustration on the theory explained, Written in neat and with good flow argument.	Explained in a descriptive manner but not extensive. Illustrations are there but not to compliment the explanation. Written neatly with relevant arguments.	Explain in a satisfactory manner but short of description and Illustrations. Not enough argument though writing is ok.	Poor explanation. Direct copy from the book or manual! Poor writing presentation.
CODING (30 MARKS)	26-30	21-25	11-20	1-10
	Procedures are listed in clear concise but descriptive. Each step is numbered and is a complete sentence in passive tone past tense.	Procedures are listed in a logical order, but steps are not numbered and/or are not in complete sentences. Some mistakes in grammar.	Procedures are listed but are not in a logical order or are difficult to follow. Not descriptive enough.	Procedures do not accurately list the steps of the experiment. Contain many errors in language.
EXECUTION (30 MARKS)	26-30	21-25	11-20	1-10
	All-important trends and data comparisons have been interpreted correctly and discussed, good understanding of results is conveyed	Almost all of the results Son have been correctly interpreted and discussed, only minor improvements are needed	Some of the results have been correctly interpreted and discussed; partial but incomplete understanding of results is still evident	Very incomplete or incorrect interpretation of trends and comparison of data indicating a lack of understanding of results

	9-10	7-8	4-6	1-3
OUTPUT & RESULT (10 MARKS)	Experiment executed with precision; all materials used effectively; accurate data recorded.	Experiment mostly executed well; minor errors in materials or data; relevant observations.	Experiment lacked clarity; materials misused; significant errors in data; unclear observations	Poor execution; missing or incorrect materials; inaccurate data; irrelevant observations.
DOCUMENTATION &VIVA (10 MARKS)	9-10	7-8	4-6	1-3
	All-important conclusions have been clearly made, student shows good understanding	All-important conclusions have been drawn, could be better stated	Conclusions regarding major points are drawn, but many are misstated, indicating a lack of understanding	Conclusions missing or missing the important points

INDEX

Exp. No.	Date	Name of the Experiment	Page No.	Aim and Algorithm (20)	Coding (30)	Execution (30)	Output and Result (10)	Documentation and Viva (10)	Total (100)
1		Install and configure Jenkins in Cloud platform	1						
2		Create CI pipeline using Jenkins	4						
3		Create a CD pipeline in Jenkins and deploy in Cloud	7						
4		Build a simple application using Gradle	11						
5		Create Maven Build pipeline in Azure	14						
6		Run regression tests using Maven Build pipeline in Azure	17						
7		Create an Ansible playbook for a simple web application infrastructure	21						
8		Install Ansible and configure ansible roles and to write playbooks	24						

Average (100):

Faulty Incharge:

SRI KRISHNA COLLEGE OF TECHNOLOGY

(An Autonomous Institution) Approved by AICTE | Affiliated to Anna University Chennai| Accredited by NBA - AICTE| Accredited by NAAC with 'A' Grade

KOVAIPUDUR, COIMBATORE 641042

REG NO. & NAME OF STUDENT	:
SEMESTER & YEAR OF STUDY	:
COURSE NO. & NAME OF LABORATORY COURSE	:
EXPERIMENT NO.	:
TITLE OF EXPERIMENT	:
DATE OF EXPERIMENT	:

EVALUATION BY FACULTY MEMBER (BASED ON RUBRICS)

CRITERIA	MAXIMUM MARKS	MARKS SCORED BY STUDENT
AIM AND ALGORITHM	20	
CODING	30	
EXECUTION	30	
OUTPUT AND RESULT	10	
DOCUMENTATION AND VIVA	10	
TOTAL MARKS	100	

EXPERIMENT OBJECTIVE:

EXP NO: 1	Install and configure Jenkins in Cloud platform
DATE:	

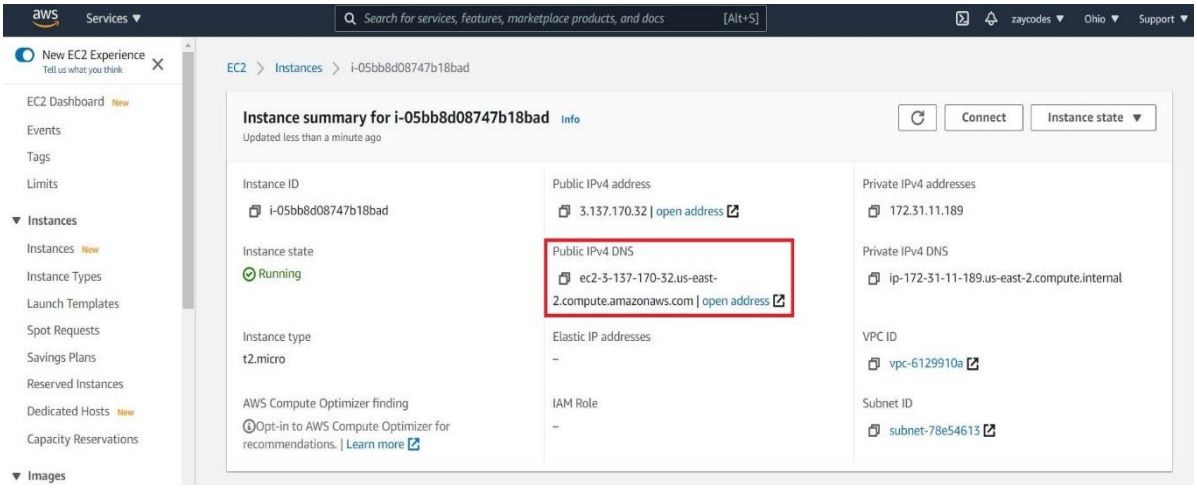
AIM: -

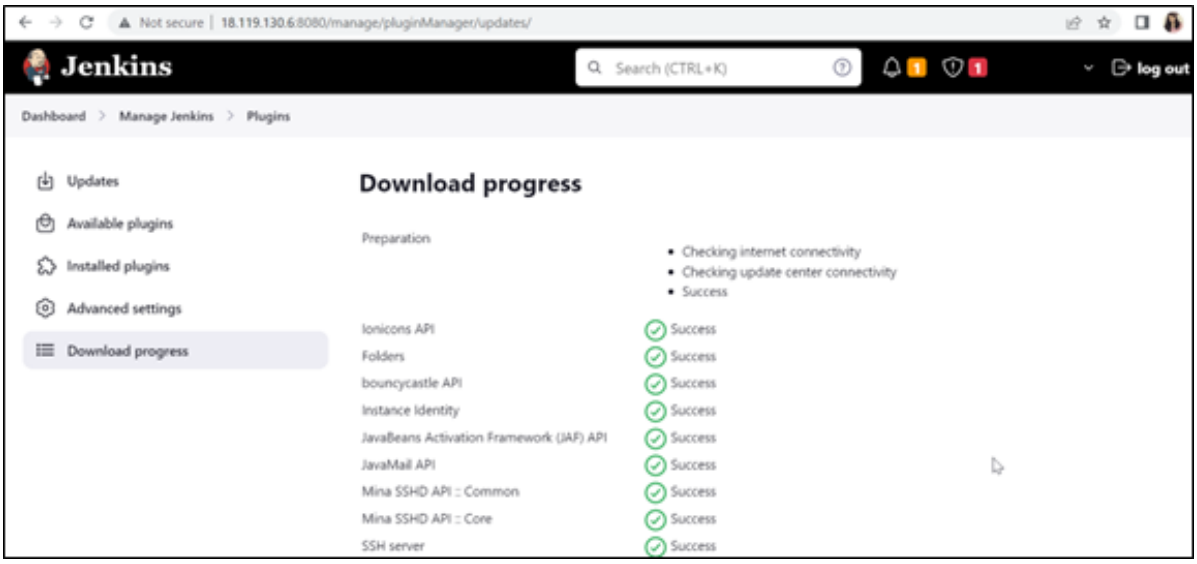
To install and configure Jenkins in Cloud platform

ALGORITHM / PROCEDURE: -

1. Launch a new EC2 instance in AWS, allowing inbound access on ports 80, 22, and 8080.
2. Connect to the instance via SSH using your key pair.
3. Update packages and install Java with sudo apt update && sudo apt install openjdk-11-jdk -y.
4. Add the Jenkins repository, then install Jenkins with sudo apt update && sudo apt install jenkins -y.
5. Start and enable Jenkins using sudo systemctl start jenkins and sudo systemctl enable jenkins.
6. Access Jenkins at http://your-instance-public-ip:8080 and unlock it using the initial password found in /var/lib/jenkins/secrets/initialAdminPassword.
7. Complete the setup by installing suggested plugins, creating an admin user, and setting your Jenkins URL.

OUTPUT: -


aws
Services
Search for services, features, marketplace products, and docs [Alt+S]
zaycodes
Ohio
Support
New EC2 Experience
Tell us what you think
EC2 Dashboard New
Events
Tags
Limits
Instances
Instances New
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts New
Capacity Reservations
Images
EC2 > Instances > i-05bb8d08747b18bad
Instance summary for i-05bb8d08747b18bad Info
Updated less than a minute ago
Connect
Instance state
Instance ID
i-05bb8d08747b18bad
Public IPv4 address
3.137.170.32 | open address
Private IPv4 addresses
172.31.11.189
Instance state
Running
Public IPv4 DNS
ec2-3-137-170-32.us-east-2.compute.amazonaws.com | open address
Private IPv4 DNS
ip-172-31-11-189.us-east-2.compute.internal
Instance type
t2.micro
Elastic IP addresses
VPC ID
vpc-6129910a
AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. | Learn more
IAM Role
Subnet ID
subnet-78e54613


Not secure | 18.119.130.6:8080/manage/pluginManager/updates/
Jenkins
Search (CTRL+K)
log out
Dashboard > Manage Jenkins > Plugins
Updates
Available plugins
Installed plugins
Advanced settings
Download progress
Download progress
Preparation
Checking internet connectivity
Checking update center connectivity
Success
Ionicons API Success
Folders Success
bouncycastle API Success
Instance Identity Success
JavaBeans Activation Framework (JAF) API Success
JavaMail API Success
Mina SSHD API :: Common Success
Mina SSHD API :: Core Success
SSH server Success

RESULT: -

Thus we successfully installed and configured Jenkins in Cloud platform

SRI KRISHNA COLLEGE OF TECHNOLOGY

(An Autonomous Institution) Approved by AICTE | Affiliated to Anna University Chennai| Accredited by NBA - AICTE| Accredited by NAAC with 'A' Grade

KOVAIPUDUR, COIMBATORE 641042

REG NO. & NAME OF STUDENT	:
SEMESTER & YEAR OF STUDY	:
COURSE NO. & NAME OF LABORATORY COURSE	:
EXPERIMENT NO.	:
TITLE OF EXPERIMENT	:
DATE OF EXPERIMENT	:

EVALUATION BY FACULTY MEMBER (BASED ON RUBRICS)

CRITERIA	MAXIMUM MARKS	MARKS SCORED BY STUDENT
AIM AND ALGORITHM	20	
CODING	30	
EXECUTION	30	
OUTPUT AND RESULT	10	
DOCUMENTATION AND VIVA	10	
TOTAL MARKS	100	

EXPERIMENT OBJECTIVE:

EXP NO: 2	Create CI pipeline using Jenkins
DATE:	

AIM: -

To create CI pipeline using Jenkins

ALGORITHM / PROCEDURE: -

1. In Jenkins, create a new job by selecting New Item, entering a job name, choosing Pipeline, and clicking OK.
2. Under the Pipeline section, select Pipeline script from SCM and specify Git as the SCM, then add your repository URL and credentials if necessary.
3. In your code repository, create a Jenkinsfile in the root directory to define the pipeline stages.
4. Add stages to the Jenkinsfile for tasks like building, testing, and deploying your application.
5. Configure a webhook in your repository to trigger the Jenkins pipeline on every code push.
6. In Jenkins, save the job configuration, then click Build Now to test the pipeline.
7. Check each stage's output in Console Output to review the build, test, and deployment status, and access any saved artifacts.

CODING: -

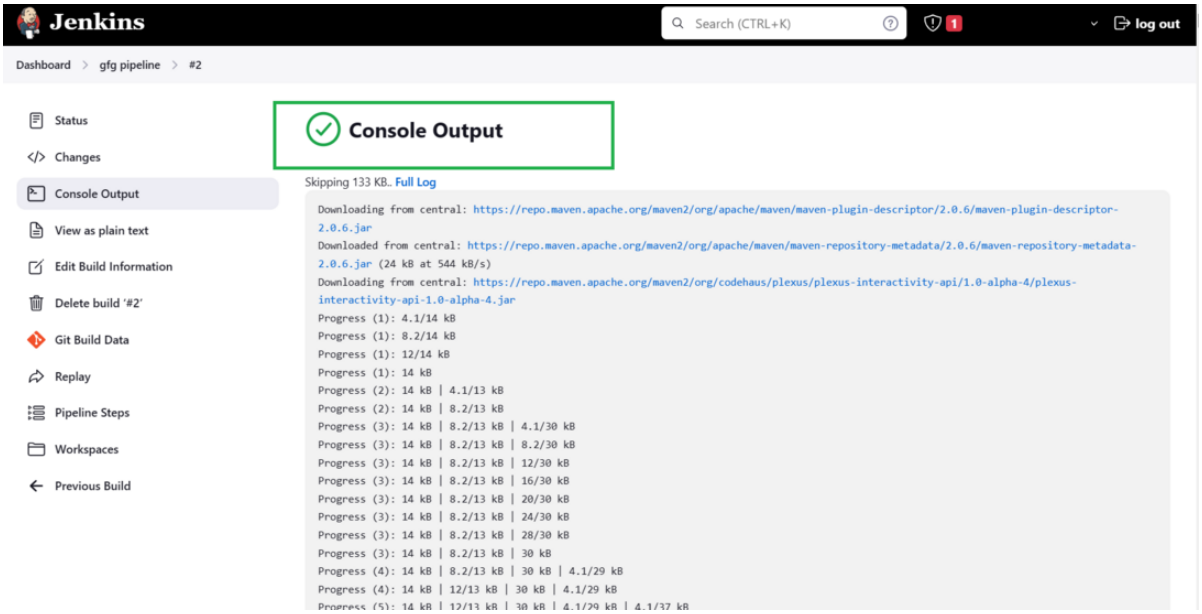
Jenkins File:

```
pipeline {
    agent any
    stages {
        stage('Clone') {
            steps {
                git 'https://github.com/your-repository.git' // Clone the repository
            }
        }
        stage('Build') {
```

```
        steps {
            sh './build-script.sh' // Run build commands (e.g., compile code)
        }
    }
    stage('Test') {
        steps {
            sh './test-script.sh' // Execute test scripts
        }
    }
    stage('Static Analysis') {
        steps {
            sh './lint-script.sh' // Perform static code analysis
        }
    }
    stage('Package') {
        steps {
            sh './package-script.sh' // Package application (e.g., JAR or ZIP)
        }
    }
    stage('Deploy') {
        steps {
            sh './deploy-script.sh' // Deploy to staging or production
        }
    }
    stage('Notify') {
        steps {
            mail to: 'team@example.com', subject: "Build ${currentBuild.fullDisplayName}", body:
"Build completed with status: ${currentBuild.result}"
        }
```

```
        }
    }
    post {
        always {
            archiveArtifacts artifacts: '**/target/*.jar', allowEmptyArchive: true // Save artifacts
        }
    }
}
```

OUTPUT: -



RESULT: -

Thus we successfully created the CI pipeline using jenkins.

SRI KRISHNA COLLEGE OF TECHNOLOGY

(An Autonomous Institution) Approved by AICTE | Affiliated to Anna University Chennai| Accredited by NBA - AICTE| Accredited by NAAC with 'A' Grade

KOVAIPUDUR, COIMBATORE 641042

REG NO. & NAME OF STUDENT	:
SEMESTER & YEAR OF STUDY	:
COURSE NO. & NAME OF LABORATORY COURSE	:
EXPERIMENT NO.	:
TITLE OF EXPERIMENT	:
DATE OF EXPERIMENT	:

EVALUATION BY FACULTY MEMBER (BASED ON RUBRICS)

CRITERIA	MAXIMUM MARKS	MARKS SCORED BY STUDENT
AIM AND ALGORITHM	20	
CODING	30	
EXECUTION	30	
OUTPUT AND RESULT	10	
DOCUMENTATION AND VIVA	10	
TOTAL MARKS	100	

EXPERIMENT OBJECTIVE:

EXP NO: 3	Create a CD pipeline in Jenkins and deploy in Cloud
DATE:	

AIM: -

To Create a CD pipeline in Jenkins and deploy in Cloud

ALGORITHM / PROCEDURE: -

1. In Jenkins, create a new Pipeline job by navigating to New Item, naming the job, selecting Pipeline, and clicking OK.
2. Under Pipeline settings, select Pipeline script from SCM and provide your Git repository URL, specifying the branch to deploy from.
3. Configure a Jenkinsfile in the repository that contains stages for building, testing, and deploying to the cloud.
4. In the Jenkinsfile, define build and test steps, followed by a deployment step that uses SSH or cloud CLI commands to deploy the application.
5. Set up a webhook in your version control system (e.g., GitHub or GitLab) to trigger the Jenkins pipeline whenever code is pushed to the main branch.
6. Test the pipeline by running Build Now in Jenkins, which will execute the build, test, and deployment stages automatically.
7. Monitor the pipeline in Jenkins to confirm successful deployment and check the application status on the cloud platform.

CODING: -

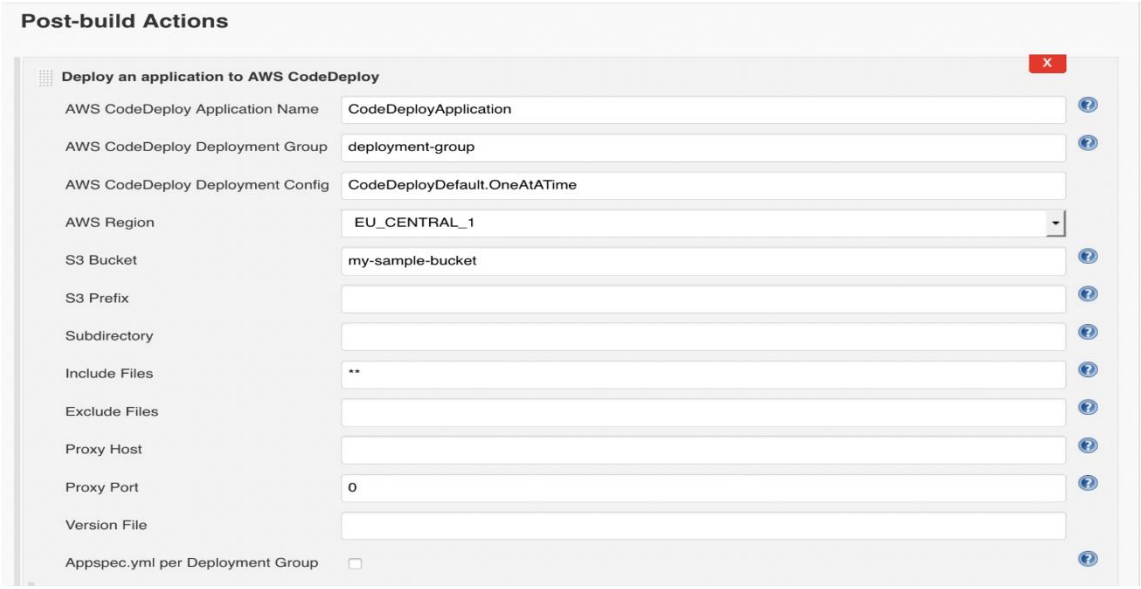
Jenkins File:

```
pipeline {
    agent any
    environment {
        DEPLOY_USER = 'ec2-user'                    // EC2 instance username
        DEPLOY_HOST = 'your-instance-public-ip'     // EC2 instance public IP
        DEPLOY_PATH = '/var/www/myapp'              // Directory on EC2 to deploy app
```

```
    SSH_KEY = credentials('aws-ec2-ssh-key')      // SSH key stored in Jenkins credentials
}
stages {
    stage('Clone') {
        steps {
            git 'https://github.com/your-repository.git'
        }
    }
    stage('Build') {
        steps {
            sh './build-script.sh' // Replace with build commands (e.g., Maven or npm)
        }
    }
    stage('Test') {
        steps {
            sh './test-script.sh' // Run tests
        }
    }
    stage('Package') {
        steps {
            sh './package-script.sh' // Package app files (e.g., create JAR or ZIP)
        }
    }
    stage('Deploy') {
        steps {
            sshagent(['aws-ec2-ssh-key']) { // Use Jenkins credentials for SSH
                sh """
                scp -o StrictHostKeyChecking=no -i $SSH_KEY target/myapp.jar \
                ${DEPLOY_USER}@${DEPLOY_HOST}:${DEPLOY_PATH}
```

```
                ssh -i $SSH_KEY ${DEPLOY_USER}@${DEPLOY_HOST} 'sudo systemctl restart
myapp'
                """ // Copy package and restart service
            }
        }
    }
    stage('Notify') {
        steps {
            mail to: 'team@example.com',
                subject: "Deployment ${currentBuild.fullDisplayName}",
                body: "Deployment completed with status: ${currentBuild.result}"
        }
    }
  }
  post {
    success {
        echo 'Deployment successful!'
    }
    failure {
        echo 'Deployment failed!'
    }
  }
}
```

OUTPUT: -



RESULT: -

Thus we successfully created a CD pipeline in Jenkins and deploy in Cloud.

SRI KRISHNA COLLEGE OF TECHNOLOGY

(An Autonomous Institution) Approved by AICTE | Affiliated to Anna University Chennai| Accredited by NBA - AICTE| Accredited by NAAC with 'A' Grade

KOVAIPUDUR, COIMBATORE 641042

REG NO. & NAME OF STUDENT	:
SEMESTER & YEAR OF STUDY	:
COURSE NO. & NAME OF LABORATORY COURSE	:
EXPERIMENT NO.	:
TITLE OF EXPERIMENT	:
DATE OF EXPERIMENT	:

EVALUATION BY FACULTY MEMBER (BASED ON RUBRICS)

CRITERIA	MAXIMUM MARKS	MARKS SCORED BY STUDENT
AIM AND ALGORITHM	20	
CODING	30	
EXECUTION	30	
OUTPUT AND RESULT	10	
DOCUMENTATION AND VIVA	10	
TOTAL MARKS	100	

EXPERIMENT OBJECTIVE:

EXP NO: 4	Build a simple application using Gradle
DATE:	

AIM: -

To Build a simple application using Gradle

ALGORITHM / PROCEDURE: -

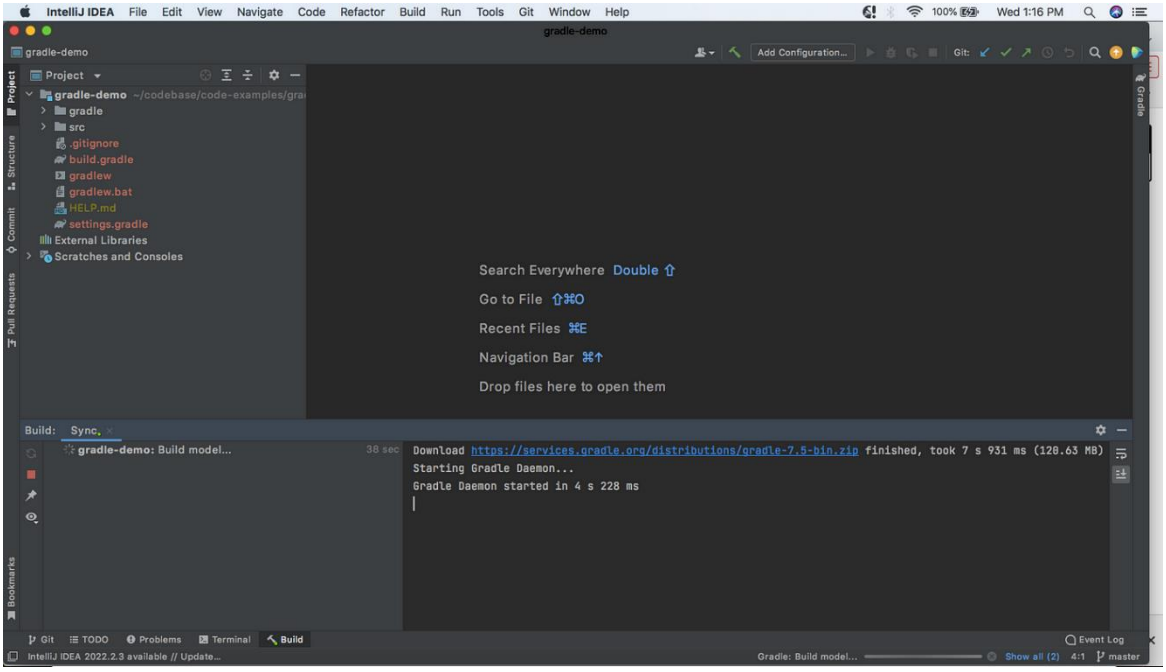
1. Start by creating a project directory, HelloWorldApp, and initialize it with gradle init --type java-application to generate the basic Java project structure.
2. Edit the src/main/java/App.java file, replacing its content with System.out.println("Hello, World!"); to display a simple message.
3. In build.gradle, confirm that the main class is set to App and ensure that dependencies and repositories are configured for testing, if needed.
4. Build the application by running gradle build and execute it with gradle run, which should output "Hello, World!" in the console.
5. Optionally, create a simple test in src/test/java/AppTest.java using JUnit, and run gradle test to verify the test setup.

CODING: -

App.java

```
public class App {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```

OUTPUT: -



RESULT: -

Thus we successfully developed a simple application using Gradle.

SRI KRISHNA COLLEGE OF TECHNOLOGY

(An Autonomous Institution) Approved by AICTE | Affiliated to Anna University Chennai| Accredited by NBA - AICTE| Accredited by NAAC with 'A' Grade

KOVAIPUDUR, COIMBATORE 641042

REG NO. & NAME OF STUDENT	:
SEMESTER & YEAR OF STUDY	:
COURSE NO. & NAME OF LABORATORY COURSE	:
EXPERIMENT NO.	:
TITLE OF EXPERIMENT	:
DATE OF EXPERIMENT	:

EVALUATION BY FACULTY MEMBER (BASED ON RUBRICS)

CRITERIA	MAXIMUM MARKS	MARKS SCORED BY STUDENT
AIM AND ALGORITHM	20	
CODING	30	
EXECUTION	30	
OUTPUT AND RESULT	10	
DOCUMENTATION AND VIVA	10	
TOTAL MARKS	100	

EXPERIMENT OBJECTIVE:

EXP NO: 5	Create Maven Build pipeline in Azure
DATE:	

AIM: -

To Create Maven Build pipeline in Azure

ALGORITHM / PROCEDURE: -

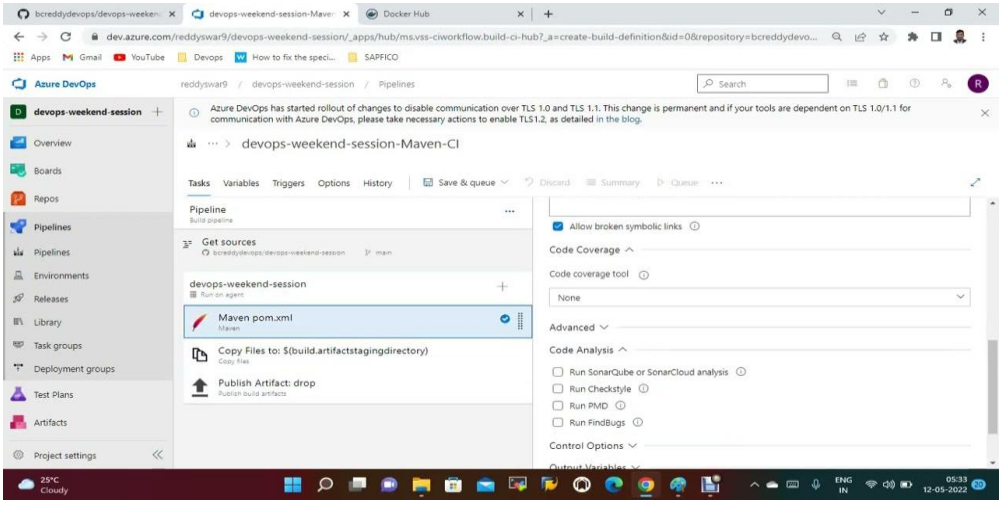
1. Create a New Azure Pipeline: In Azure DevOps, go to the Pipelines section and click New Pipeline.
2. Choose Repository: Select the repository where your Maven project is stored, such as GitHub or Azure Repos.
3. Select Pipeline Template: Choose Maven from the available templates or select Starter Pipeline if you want to manually configure it.
4. Configure YAML File: If you selected Starter Pipeline, configure the pipeline by editing the YAML file to define the build process.
5. Add Maven Build Task: In the YAML file, use the maven task to specify the goals like clean install or clean package to build the project.
6. Set Up Build Triggers: Configure triggers to run the pipeline on code changes, such as when code is pushed to a specific branch.
7. Run and Monitor: Save the pipeline, then run it manually or based on triggers, and monitor the build process through Azure DevOps.

CODING: -

```
trigger:
- main  # Change to your branch name, e.g., 'master' or 'develop'
pool:
  vmImage: 'ubuntu-latest'  # Or any other VM image like 'windows-latest'
steps:
- task: MavenAuthenticate@0  # Authenticate with a repository if necessary
  inputs:
    mavenFeed: 'your-feed-name'
```

```
- task: Maven@3  # Maven task to build the project
  inputs:
    mavenPomFile: 'pom.xml'  # Path to the Maven pom.xml file
    options: '-X'  # Optional: extra Maven options, e.g., verbose mode
    goals: 'clean install'  # Or 'clean package' depending on the goal
    javaHomeOption: 'JDKVersion'  # Choose the JDK version (e.g., '1.8', '11')
    jdkVersionOption: '1.8'
    mavenVersionOption: 'Default'
- task: PublishBuildArtifacts@1  # Optional: Publish build artifacts
  inputs:
    PathtoPublish: '$(Build.ArtifactStagingDirectory)'
    ArtifactName: 'drop'  # Name of the artifact
    publishLocation: 'Container'
```

OUTPUT: -



RESULT: -

Thus we successfully created Maven Build pipeline in Azure.

SRI KRISHNA COLLEGE OF TECHNOLOGY

(An Autonomous Institution) Approved by AICTE | Affiliated to Anna University Chennai| Accredited by NBA - AICTE| Accredited by NAAC with 'A' Grade

KOVAIPUDUR, COIMBATORE 641042

REG NO. & NAME OF STUDENT	:
SEMESTER & YEAR OF STUDY	:
COURSE NO. & NAME OF LABORATORY COURSE	:
EXPERIMENT NO.	:
TITLE OF EXPERIMENT	:
DATE OF EXPERIMENT	:

EVALUATION BY FACULTY MEMBER (BASED ON RUBRICS)

CRITERIA	MAXIMUM MARKS	MARKS SCORED BY STUDENT
AIM AND ALGORITHM	20	
CODING	30	
EXECUTION	30	
OUTPUT AND RESULT	10	
DOCUMENTATION AND VIVA	10	
TOTAL MARKS	100	

EXPERIMENT OBJECTIVE:

EXP NO: 6	Run regression tests using Maven Build pipeline in Azure
DATE:	

AIM: -

To Run regression tests using Maven Build pipeline in Azure.

ALGORITHM / PROCEDURE: -

1. Create or Modify Azure Pipeline: If you don't have an existing pipeline, create a new one in Azure DevOps as described earlier. If you already have a pipeline, modify it to include the test steps.
2. Choose Repository: Select the repository that contains your Maven project with the tests.
3. Define Pipeline Template: Choose Maven as your pipeline template or use the Starter Pipeline and manually define the pipeline.
4. Update azure-pipelines.yml: Add Maven commands to run the regression tests after building the project. The regression tests are usually executed by Maven through mvn test or mvn verify.
5. Add Maven Test Goals: In the azure-pipelines.yml, configure the goals to include test or verify to run the tests.
6. Configure Test Results: Optionally, you can publish test results to Azure DevOps for reporting purposes.
7. Save and Run the Pipeline: Once the pipeline is configured, save it and run it to execute the regression tests along with the build.

CODING: -

```
trigger:

- main  # Adjust this to your default branch, like 'master' or 'develop'


pool:

  vmImage: 'ubuntu-latest'  # You can change to 'windows-latest' if required


steps:
```

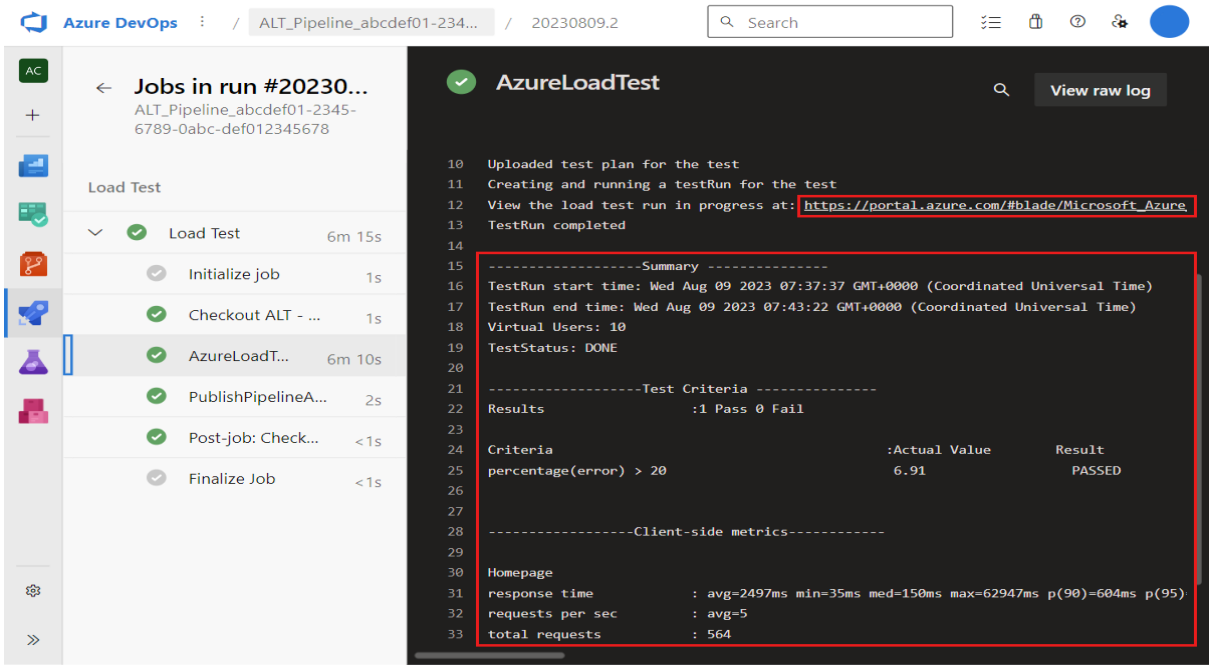
```
- task: MavenAuthenticate@0  # Authenticate with your Maven feed (if necessary)
  inputs:
    mavenFeed: 'your-feed-name'

- task: Maven@3  # Maven task to run build and tests
  inputs:
    mavenPomFile: 'pom.xml'  # Path to your Maven project file
    options: '-X'  # Optional: Enable verbose output
    goals: 'clean verify'  # Run clean and verify to compile, test, and package
    javaHomeOption: 'JDKVersion'  # Choose JDK version (e.g., '1.8', '11')
    jdkVersionOption: '1.8'
    mavenVersionOption: 'Default'

- task: PublishTestResults@2  # Publish test results to Azure DevOps
  inputs:
    testResultsFiles: '**/target/test-*.xml'  # Path to test result files
    testRunTitle: 'Maven Regression Tests'  # Title for test run

- task: PublishBuildArtifacts@1  # Optionally publish build artifacts
  inputs:
    PathtoPublish: '$(Build.ArtifactStagingDirectory)'
    ArtifactName: 'drop'
    publishLocation: 'Container'
```

OUTPUT: -



RESULT: -

Thus we successfully Run regression tests using Maven Build pipeline in Azure.

SRI KRISHNA COLLEGE OF TECHNOLOGY

(An Autonomous Institution) Approved by AICTE | Affiliated to Anna University Chennai| Accredited by NBA - AICTE| Accredited by NAAC with 'A' Grade

KOVAIPUDUR, COIMBATORE 641042

REG NO. & NAME OF STUDENT	:
SEMESTER & YEAR OF STUDY	:
COURSE NO. & NAME OF LABORATORY COURSE	:
EXPERIMENT NO.	:
TITLE OF EXPERIMENT	:
DATE OF EXPERIMENT	:

EVALUATION BY FACULTY MEMBER (BASED ON RUBRICS)

CRITERIA	MAXIMUM MARKS	MARKS SCORED BY STUDENT
AIM AND ALGORITHM	20	
CODING	30	
EXECUTION	30	
OUTPUT AND RESULT	10	
DOCUMENTATION AND VIVA	10	
TOTAL MARKS	100	

EXPERIMENT OBJECTIVE:

EXP NO: 7	Create an Ansible playbook for a simple web application infrastructure
DATE:	

AIM: -

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

ALGORITHM / PROCEDURE: -

1. Create an Inventory File: Define your web servers in an Ansible inventory file with their IP addresses or hostnames.
2. Write the Playbook: Create a playbook in YAML format that installs Apache, deploys the web application, and starts the Apache service.
3. Install Apache: Use Ansible's apt module to install Apache on the web servers.
4. Deploy Web Application: Use the copy module to transfer your web application's files (e.g., index.html) to the server's web directory.
5. Ensure Apache is Running: Use the service module to ensure Apache is started and enabled on boot.

CODING: -

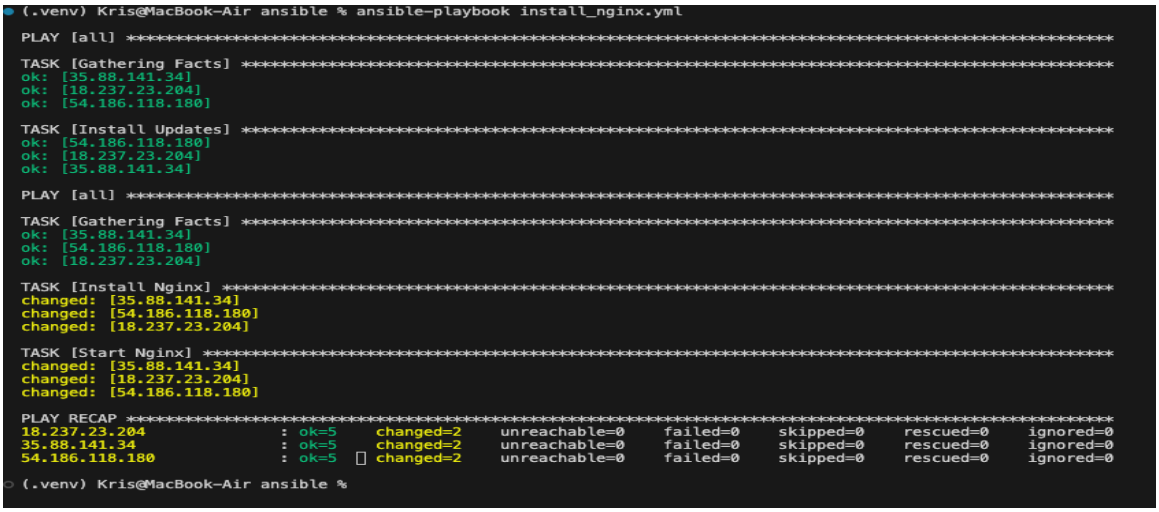
```
---
- name: Set up simple web application infrastructure
  hosts: web_servers
  become: true  # Use sudo to execute tasks as a privileged user
  tasks:


    - name: Install Apache web server
      apt:
        name: apache2
        state: present
        update_cache: yes
```

```
- name: Copy web application files to the web server
  copy:
    src: /path/to/your/webapp/index.html  # Local path to the web app
    dest: /var/www/html/index.html  # Destination on the server

- name: Ensure Apache service is running
  service:
    name: apache2
    state: started
    enabled: yes
```

OUTPUT: -



RESULT: -

Thus we successfully Created an Ansible playbook for a simple web application infrastructure.

SRI KRISHNA COLLEGE OF TECHNOLOGY

(An Autonomous Institution) Approved by AICTE | Affiliated to Anna University Chennai| Accredited by NBA - AICTE| Accredited by NAAC with 'A' Grade

KOVAIPUDUR, COIMBATORE 641042

REG NO. & NAME OF STUDENT	:
SEMESTER & YEAR OF STUDY	:
COURSE NO. & NAME OF LABORATORY COURSE	:
EXPERIMENT NO.	:
TITLE OF EXPERIMENT	:
DATE OF EXPERIMENT	:

EVALUATION BY FACULTY MEMBER (BASED ON RUBRICS)

CRITERIA	MAXIMUM MARKS	MARKS SCORED BY STUDENT
AIM AND ALGORITHM	20	
CODING	30	
EXECUTION	30	
OUTPUT AND RESULT	10	
DOCUMENTATION AND VIVA	10	
TOTAL MARKS	100	

EXPERIMENT OBJECTIVE:

EXP NO: 8	Install Ansible and configure ansible roles and to write playbooks
DATE:	

AIM: -

To Install Ansible and configure ansible roles and to write playbooks.

ALGORITHM / PROCEDURE: -

1. Install Ansible: Install Ansible on your machine using a package manager like apt, yum, or pip, depending on your operating system.
2. Create Ansible Roles: Roles are a way of organizing your playbooks into reusable components. You can create roles using the ansible-galaxy command.
3. Structure the Roles Directory: A role typically consists of a directory structure that includes tasks, handlers, templates, files, defaults, and vars.
4. Write Playbooks: A playbook is a YAML file that defines the tasks to be executed. You can reference roles in your playbook to apply the tasks defined in the role.
5. Run the Playbook: Use the ansible-playbook command to run your playbooks, applying the defined roles to the specified hosts.

CODING: -

```
---
- name: Install Nginx
  apt:
    name: nginx
    state: present
    update_cache: yes
- name: Start and enable Nginx service
  service:
    name: nginx
    state: started
    enabled: yes
```

OUTPUT: -

```
ubuntu@ip-10-111-4-53:~$ ansible-playbook apache.yml --check

PLAY [update web servers] ******************************************************

TASK [Gathering Facts] *********************************************************
ok: [10.111.4.18]

TASK [ensure apache is at the latest version] **********************************
ok: [10.111.4.18]

PLAY RECAP *********************************************************************
10.111.4.18                : ok=2    changed=0    unreachable=0    failed=0
```

RESULT: -

Thus we successfully installed Ansible and configure ansible roles and to write playbooks.