ACS COLLEGE OF ENGINEERING

DEPARTMENT OF CYBER SECURITY & ENGINEERING

LAB MANUAL

Course Name: **DEVOPS**

Course Code: **BCSL657D**

SEMESTER – VI

Prepared by

MRS. K.P.Sangeetha

Dept. of Cyber Security,
ACSCE

Vision

The vision of the Computer Science and engineering department is to provide excellence knowledge and enrich the problem-solving skills of the students in the field of CSE with a focus to prepare the students for industry need, carry out research, recognized as innovative leader, responsible citizen and improve the environment.

Mission

- Prepare the students with strong fundamental concepts, analytical capability, programming and problem-solving skills.
- Create an ambience of education through faculty training, self-learning, sound academic practices and research endeavors.
- Provide opportunities to promote organizational and leadership skills in students through various extra- curricular and co-curricular events
- To make the students as for as possible industry ready to enhance their employability in the industries.
- To improve department industry collaboration through internship programme and interaction with professional society through seminar/workshops.
- Imbibe social awareness and responsibility in students to serve the society and protect environment.

CONTENTS

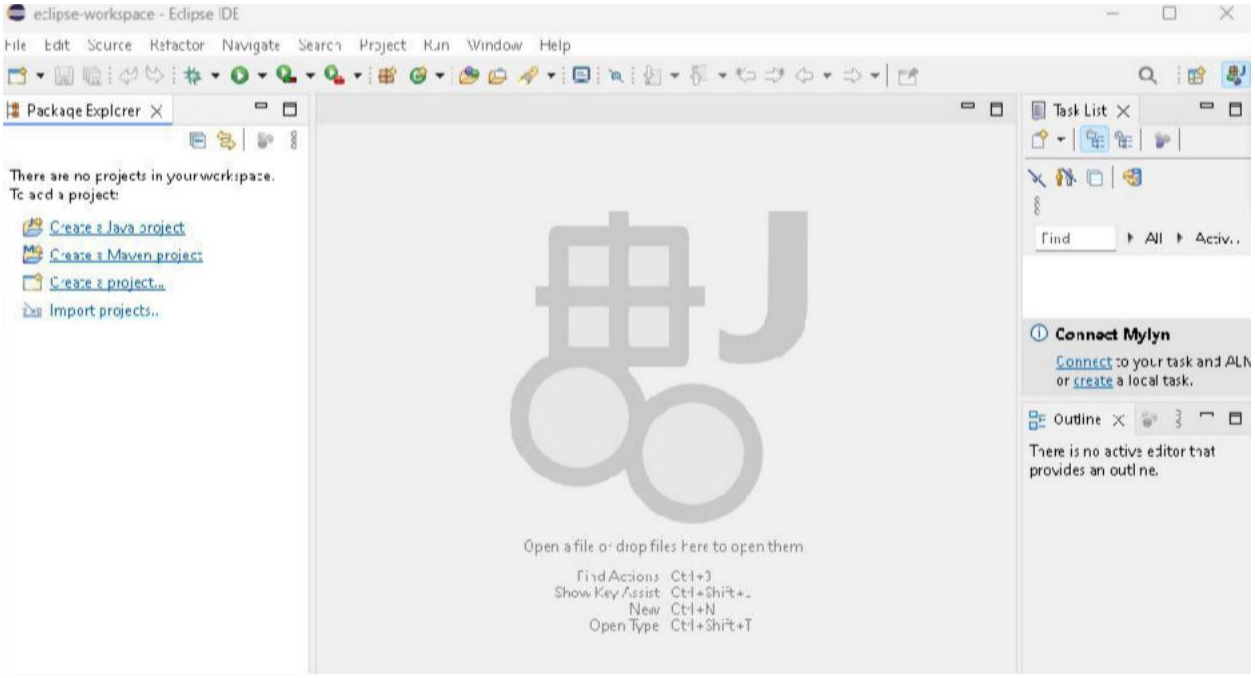
PROGRAM-1

Introduction to Maven and Gradle:Overview of Build AutomationTools, Key Differences Between Maven and Gradle, Installation and Setup

STEP1:Install Eclipse using this link
https://www.eclipse.org/downloads/ENTERPRISE JAVA AND
WEB DEVELOPERS

STEP2:Select ECLIPSE IDE FOR JAVA DEVELOPER OR ECLIPSE IDE FOR
STEP3:SELECT INSTALLATION FOLDER JAVA VERSION

STEP4:ONCE ECLIPSE IS INSTALLED THE SCREEN LOOKS AS IN BELOW



STEP5: Lets see Procedure to install MAVEN & GRADLE

a) **First make sure JDK current version is installed**
https://www.oracle.com/java/technologies/downloads/?er=221886#jdk23-windows
Then set environment variable path both user and system

- **Have a JDK installation on your system. Either set the JAVA_HOME environment variable pointing to your JDK installation or have the java executable on your PATH.**

b) **To install apache maven pls go to link as in below and download zip file of bin**
https://maven.apache.org/download.cgi

c) To unzip the Source zip archive Run in Windows cmd prompt unzip apache-maven-3.9.9-bin.zip

If don't want to run directly extract the file to Program Files

d) SetupaPATHinenvironmentalsettings

"Add the bin directory of the created directory apache-maven-3.9.9 to the PATH environment variable"

e) After environment variable is set

f) Run this command in CMD prompt

mvn --v(2 hyphen)

After running,you see the text screen as in below

```
Apache Maven 3.9.9 (8e8579a9e76f7d015ee5ec7bfcdc97d260186937)
Maven home: /opt/apache-maven-3.9.9
Java version: 1.8.0_45, vendor: Oracle Corporation
Java home: /Library/Java/JavaVirtualMachines/jdk1.8.0_45.jdk/Contents/Home/jre
Default locale: en_US, platform encoding: UTF-8
OS name: "mac os x", version: "10.8.5", arch: "x86_64", family: "mac"
```

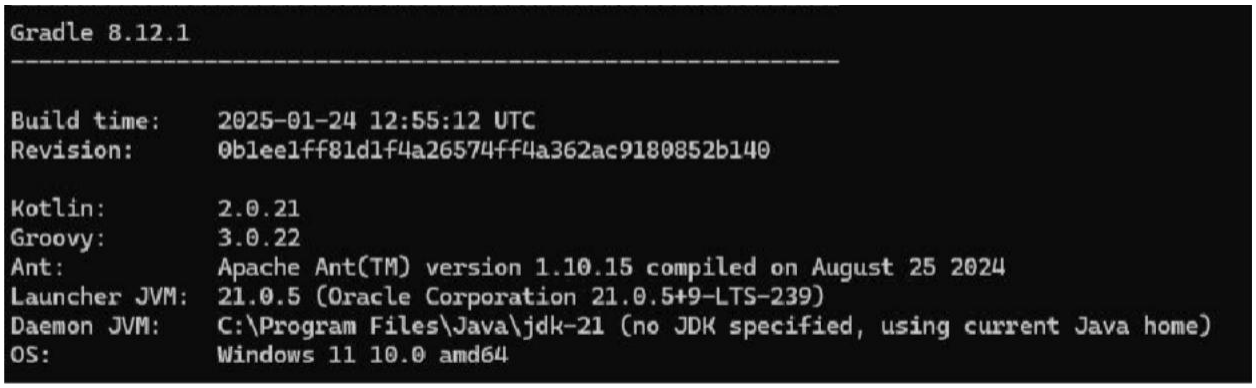
g) TO INSTALL GRADLE FOR WINDOWS follow procedure as in below

1. **Create an ew directoryC:\Gradle with File Explorer.**
2. **Open a second File Explorer window and go to the directory where the Gradle distribution was downloaded. Double-click the ZIP archive to expose the content. Drag the content folder gradle-8.12.1to your newly** created `C:\Gradle` folder.

Alternatively you can unpack the Gradle distribution ZIPintoC:\Gradle using an archiver tool of your choice or run command with path folder where the folder is created.

Unzip apache-maven-3.9.9-bin.zip Or can directly extract the zip file.

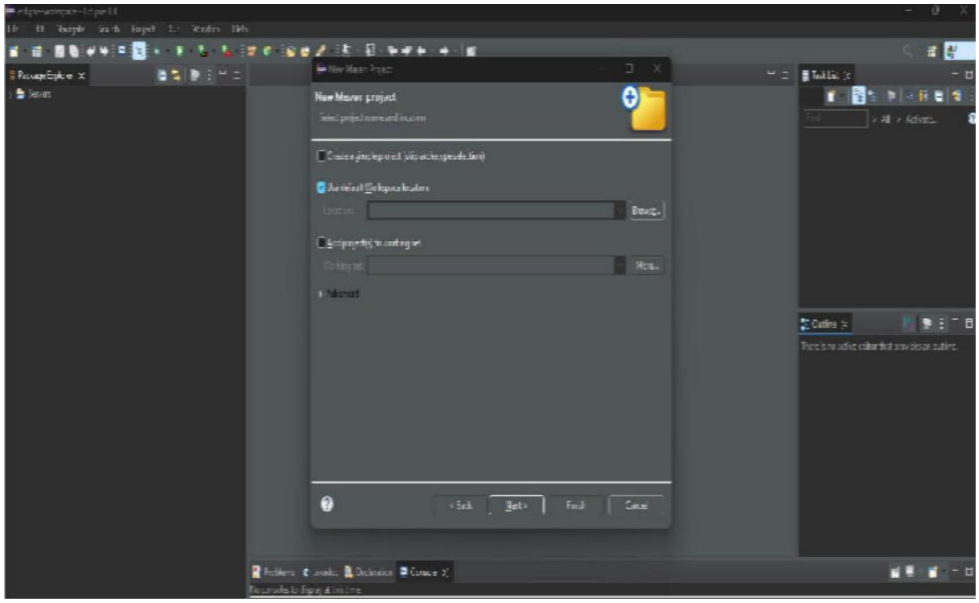
3. **Configure your system environment**
4. **Finally type the command gradle–v to check if the gradle is installed.**



PROGRAM-2

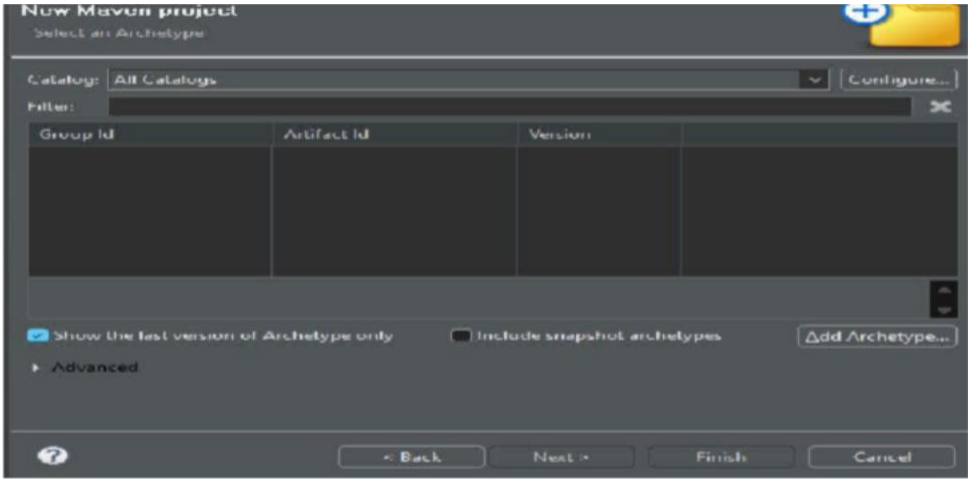
Working with Maven: Creating a Maven Project, Understanding the POMFile, Dependency Management and Plugins

STEP1:OPEN ECLIPSE THEN followth is navigation FIle ——————>
New---------- >Maven Project
After that Screen beas in bel



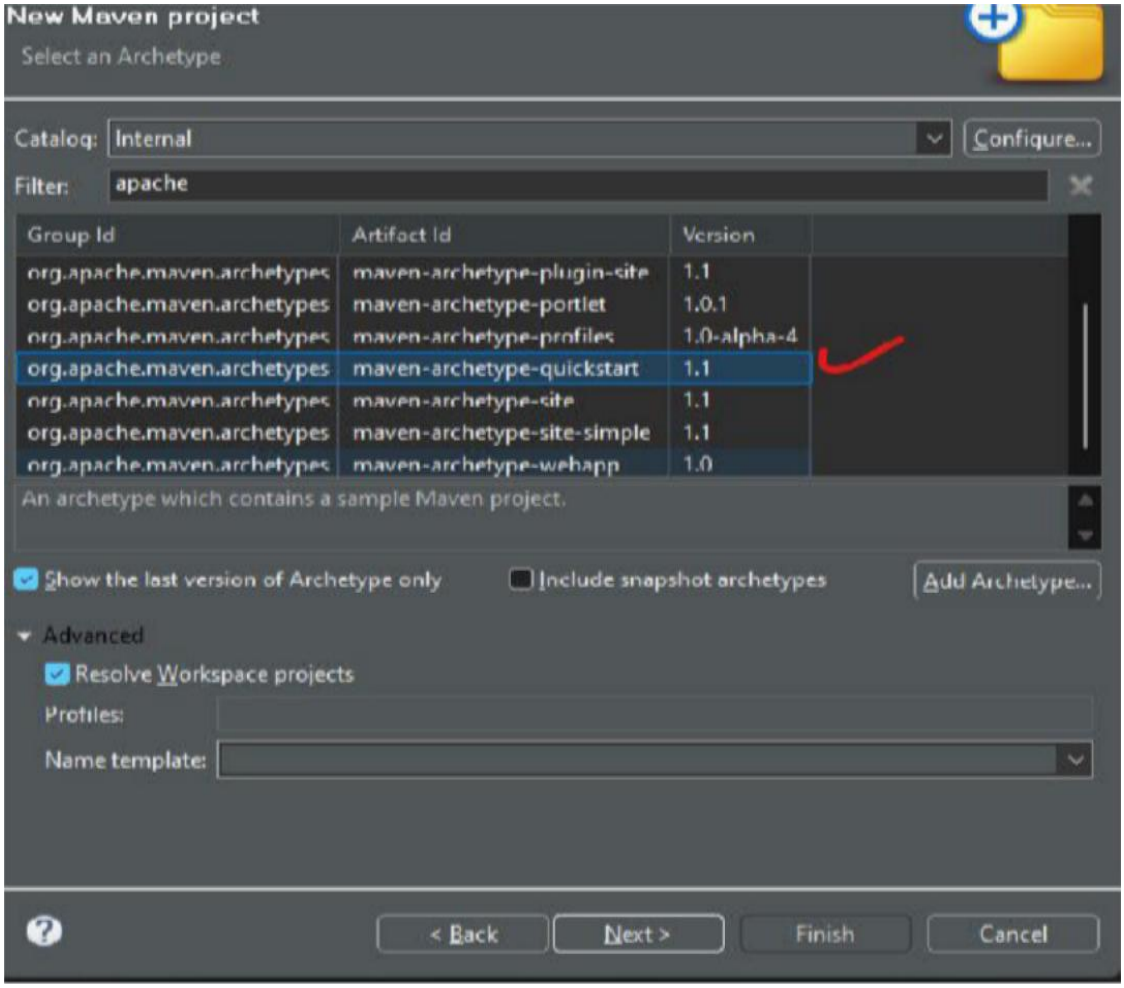
STEP2:Make sure Use default Workspace Location is selected, then click Next

The screen be as in below



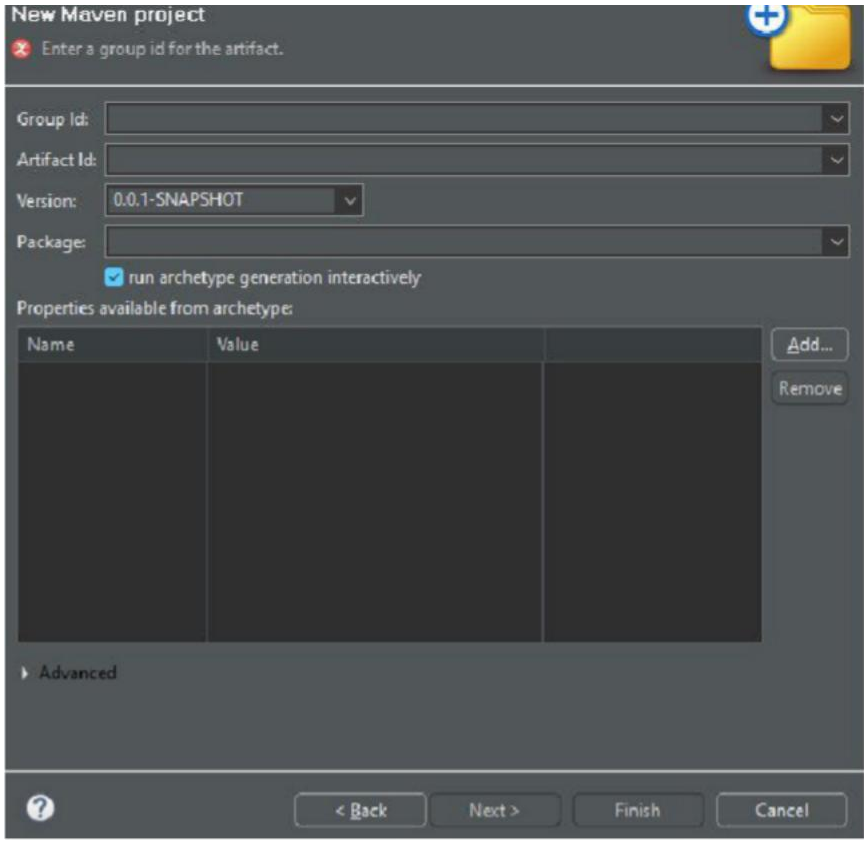
STEP3:In Screen shown above,click near the entry place of Filter and type “apache” or select catalog as Internal
We want as imple maven JAR based application. So, we will choose the “maven-archetype-quickstart” artifact to create the project

.



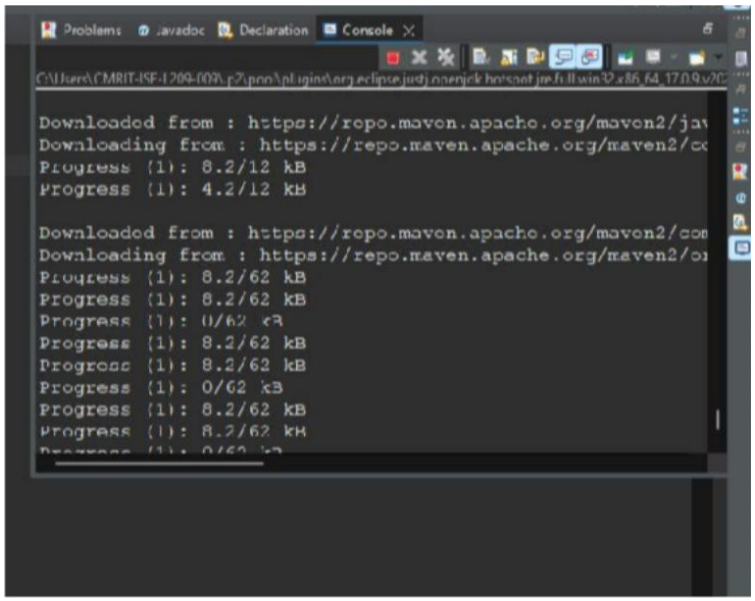
STEP4: Enter
Group Id:com.program2.maven
ArtifactId:program2-example-jar Keep snapshot as
it is

Package:com.program2.maven.program2



After entering above mentioned details click on Finish

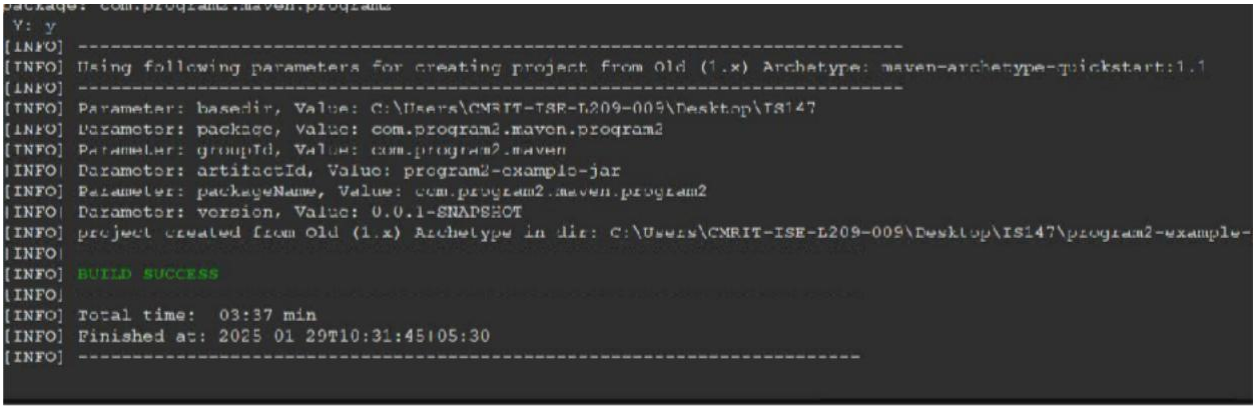
You be able to see the automation build happening for Maven Jar Project



It asks for Configuration confirmation just click Y

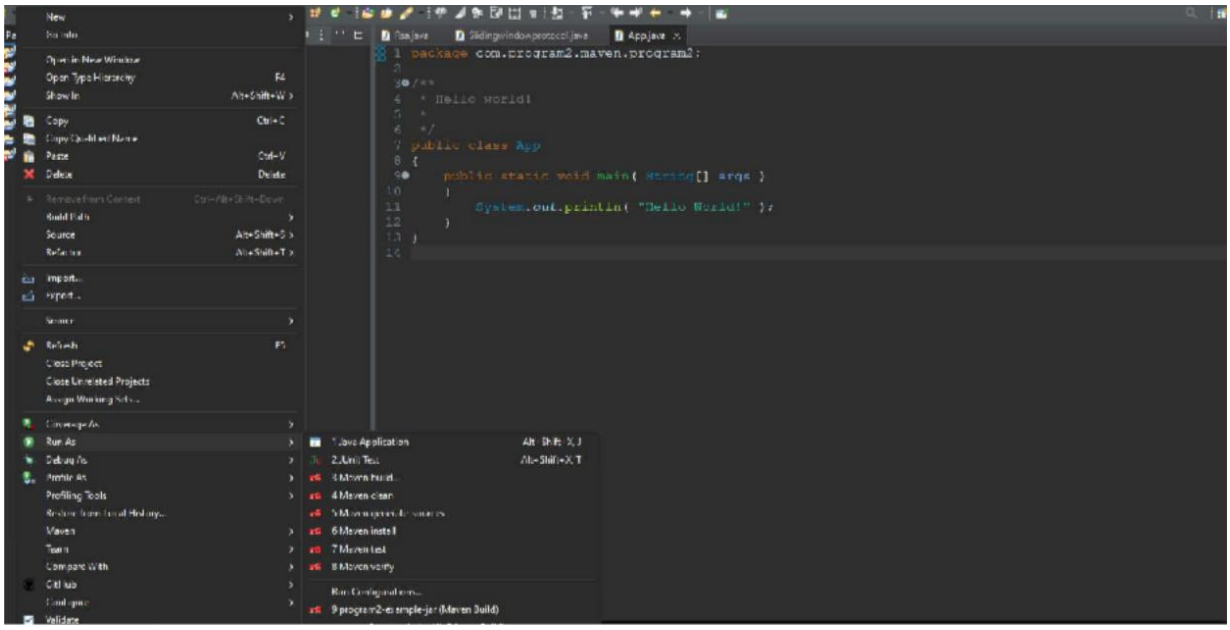
```
Confirmpropertiesconfiguration: groupId:
com.program2.maven artifactId: program2-
example-jar version: 0.0.1-SNAPSHOT
package:com.program2.maven.program2
```

The RESULT be as in below

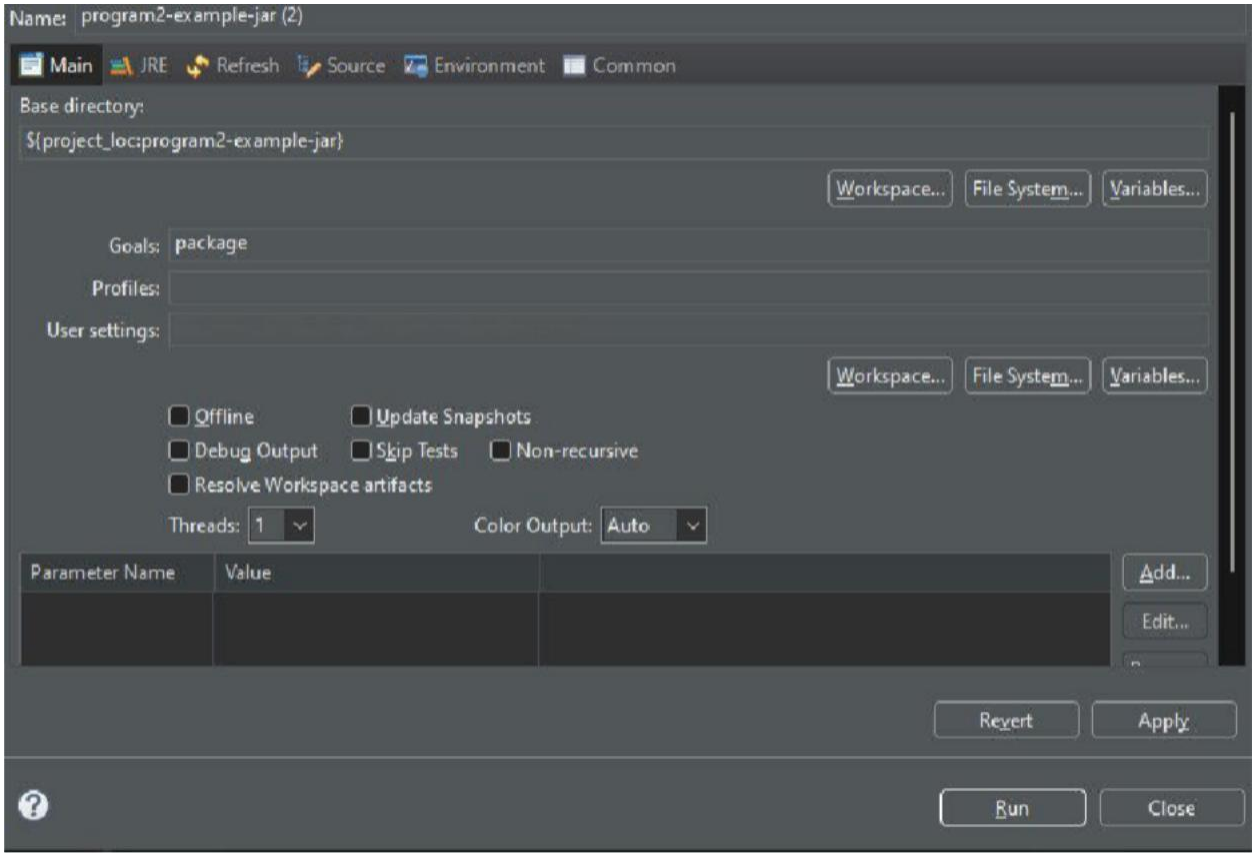


STEP5:Now its time to build Maven Project

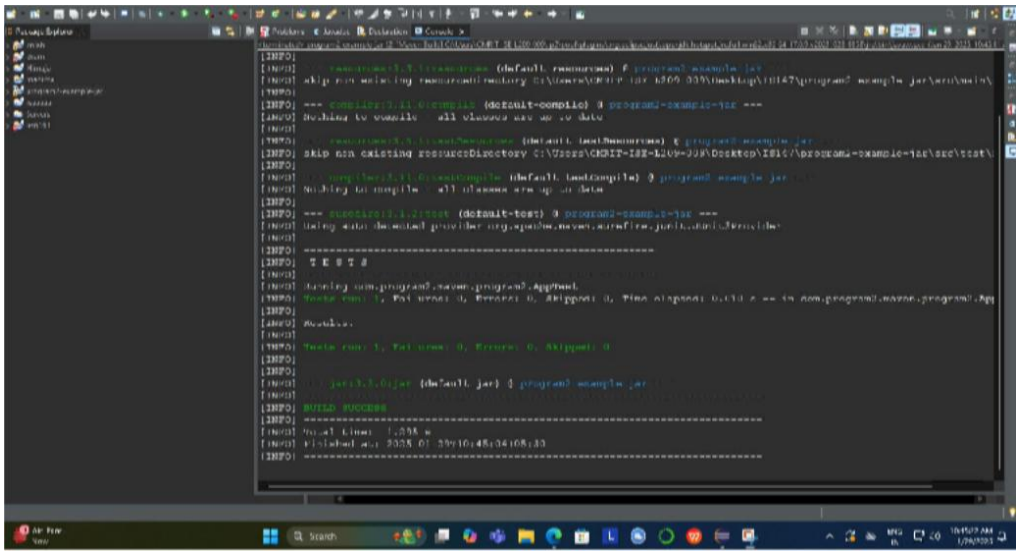
Goto Maven Project ----------->Right Click on the Project and select Maven Build



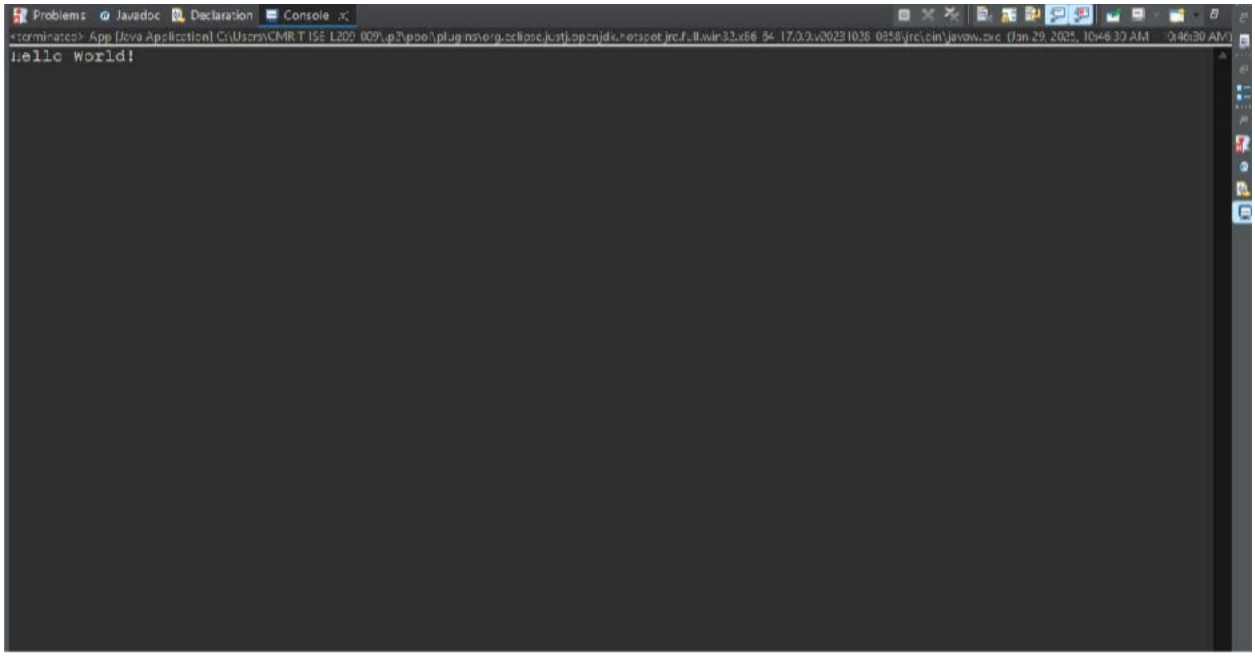
After the above procedure is done Select Goal as
package



And Click Run
The result be as in below



Now goto App.java finally run java application



Description

What is group Id in maven?

Group Id identifies a particular project uniquely across all projects, so we should follow an aming convention. A very simple and commonly used way of doing this is to use the reverse of your domain, i.e. com.javarewind.maven.

A good way of maintaining the integrity of group Id is to use the project structure. In case the project consists of multiple modules then every module should append an identifier to the parent groupId.
i.e.com.javarewind.maven,com.javarewind.spring,com.javarewind.struts..etc.

What is artifact Id in maven?

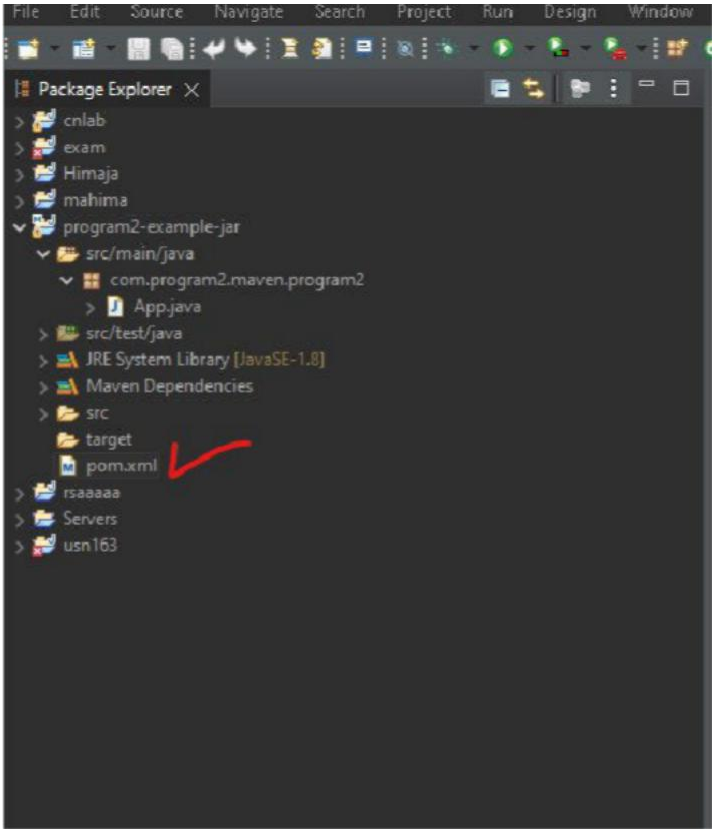
Artifact Id is then ame of warfile without version, if you are creating it by yourself you are free to took any name of your choice in lower case and without any strange symbol. But if this is a third party jar than we have to take the name of the jar as suggested by its distribution.

What is the arche type in maven?

Archetype is a Maven project templating tool kit which tells the maven the type of project we are going to create.Arche type enables them aven to create a template project of the user's choice so that the user can get the project up and running instantly.

“archetype:generate” generates anew project from the provided arche type or updates the actual project I fusing a partial arche type. Maven provides a number of predefined archtypes, see more details from Maven Documentation.

HOW POM.XML LOOK IS AS IN SCREEN BELOW




```
http://maven.apache.org/xsd/maven-4.0.0.xsd (xsi:schemaLocation with catalog)
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instanc
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>com.program2.maven</groupId>
  <artifactId>program2 example jar</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>jar</packaging>

  <name>program2-example-jar</name>
  <url>http://maven.apache.org</url>

  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  </properties>

  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>3.8.1</version>
      <scope>test</scope>
    </dependency>
  </dependencies>
</project>
```

PROGRAM3 :Working with Gradle: Setting Up a Gradle Project, Understanding Build Scripts(Groovy and Kotlin DSL),Dependency Management and Task Automation

STEP1:Let's do this in cmdprompt Goto Command Prompt

Then first make anew directory the command is mkdir pgm3

For changing to a current directory the command is cd pgm3

Now run gradleinit
After execution of command the screen shows as in below where we opt for build type select as 1

After selecting application type next it asks for Implementation language select as groovy

After selecting Implementation language it will ask for Java version and project name

```
Enter selection (default: Java) [1..6] 3

Enter target Java version (min: 7, default: 21): 21

Project name (default: gradletest): gradletest
```

After providing version and project name
Select application structure as Single application structure and Domain Specific Language as Kotlin

```
Select application structure:
  1: Single application project
  2: Application and library project
Enter selection (default: Single application project) [1..2] 1

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

After every procedure is over it shows Build successful

```
Generate build using new APIs and behavior (some features may change in the next minor release)? (default: no) [yes, no] yes


> Task :init
Learn more about Gradle by exploring our Samples at https://docs.gradle.org/8.12.1/samples/sample_building_groovy_applications.html

BUILD SUCCESSFUL in 2m 2s
1 actionable task: 1 executed
```

STEP2: Now its time to Build the script Just type the
command as:
gradlewrun

It will take atleast 3-5minutes to run the configuration script we have set through steps finally the output be as in below If You want to see the structure of an application run the command as tree

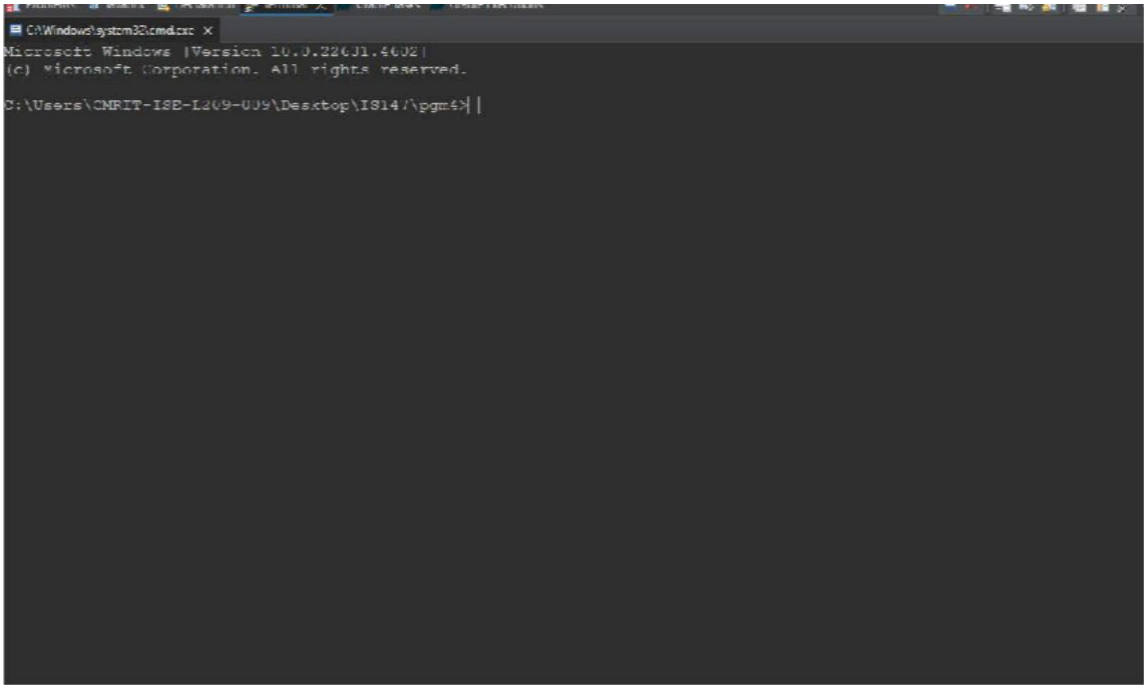
```
C:\Users\CMRIT-ISE-L209-009\gradletest>gradlew run
Calculating task graph as no cached configuration is available for tasks: run

> Task :app:run
Hello World!
```

```
C:\Users\CMRIT-ISE-L209-009\gradletest>tree
Folder PATH listing for volume Windows
Volume serial number is CA13-EB9B
C:.
├---.gradle
|   └---8.12.1
|       ├---checksums
|       ├---executionHistory
|       ├---expanded
|       ├---fileChanges
|       ├---fileHashes
|       └---vcsMetadata
├---buildOutputCleanup
|   ├---configuration-cache
|   |   └---87330868-729b-48ed-8a38-57771bbaae57
|   └---8qybc7c3ykh3sf9t2sllkie4w
├---vcs-1
├---.settings
├---app
|   ├---build
|   |   ├---classes
|   |   |   └---groovy
|   |   |       └---main
|   |   |           └---org
|   |   |               └---example
|   |   ├---generated
|   |   |   └---sources
|   |   |       └---annotationProcessor
|   |   |           └---groovy
|   |   |               └---main
|   |   └---tmp
|   |       └---compileGroovy
|   |           └---groovy-java-stubs
|   └---src
|       ├---main
|       |   ├---groovy
|       |   |   └---org
|       |   |       └---example
|       |   └---resources
|       └---test
|           ├---groovy
|           |   └---org
```

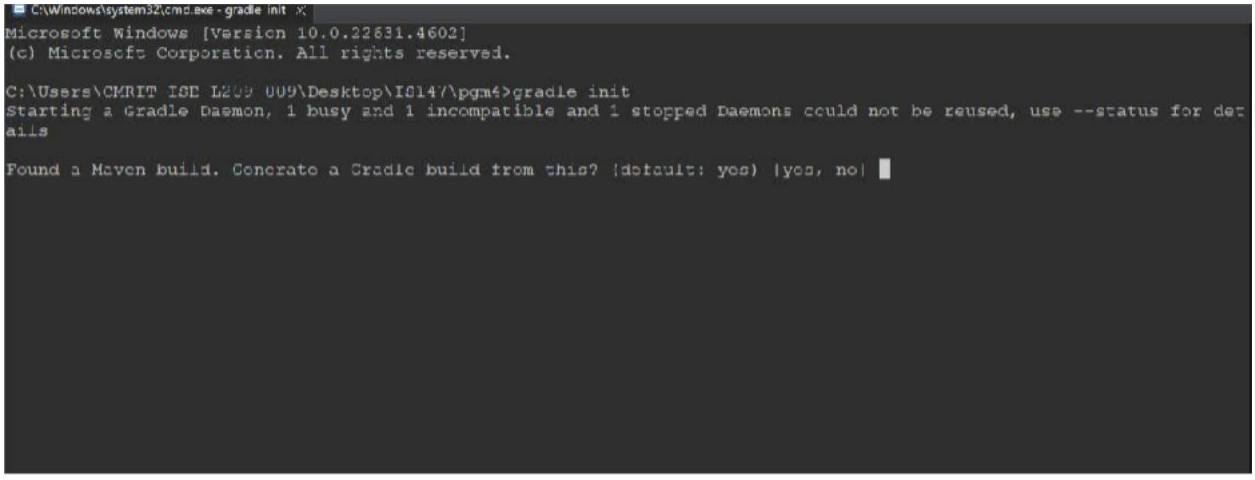
PROGRAM 4:Practical Exercise :Build and Run a Java Application with Maven, Migrate the Same Application to Gradle

STEP1:First create a Maven Project as in PROGRAM2 then build the project and run java application you will get Hello World Message

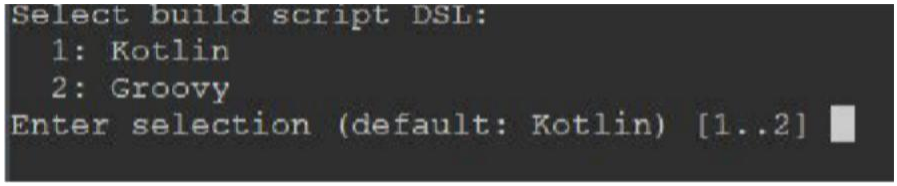
STEP2:Then to migrate to gradle use shortcutKey Ctrl+Alt+Shif+T Toget Terminal screen as in below:



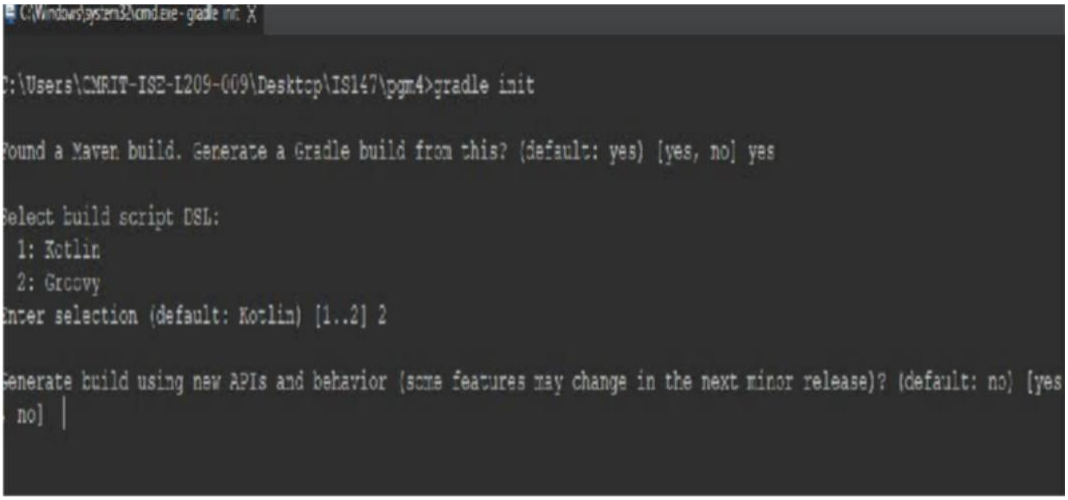
STEP3:Type command gradle init it will ask for migrate from mavento gradle type yes



STEP4:After the above comman disvalidated to yes it prompts to select Domain Specific Language as in screen below select2 (as we have done for Kotlin)

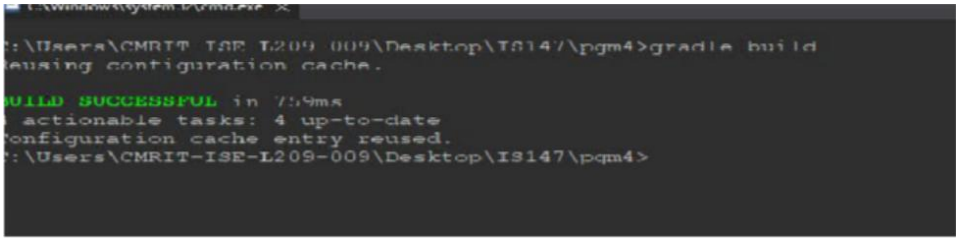


STEP5:After selecting Groovy it asks for validating prompt for API Generator just validate as yes



Finally it runs the in it phase as been selected

STEP6:
Type the command gradle build

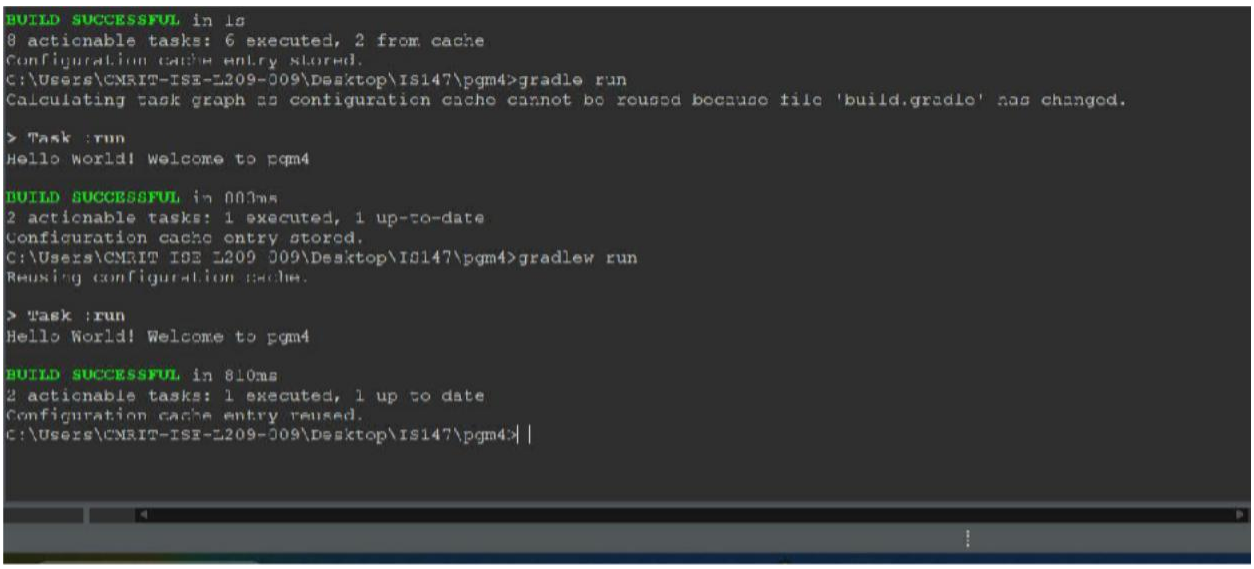


Now to get exact program output of our java file Locate to build gradle File from your local repository and Copy paste the code as in below shown in red color

```
        plugins
    { id("java-library")
    id("maven-publish")
    id("application")
}

application{
    mainClass.set("com.pgm4.test.App")//Use.set()for properties
}

repositories
    { mavenCentral()
    //Uncommentif youneed topublish locally
    //mavenLocal()
}

dependencies{
    testImplementation("junit:junit:4.13.2")//UseKotlinsyntaxfor dependencies
}

group = "com.pgm4.test"version="0.0.1-
SNAPSHOT"
description="pgm4"
java.sourceCompatibility=JavaVersion.VERSION_11//Considerupgrading

publishing
    { publications { create<MavenPublication>("maven")
        { from(components["java"])
      }
    }
}

tasks.withType<JavaCompile>().configureEach
    {options.encoding="UTF-8"
}
```

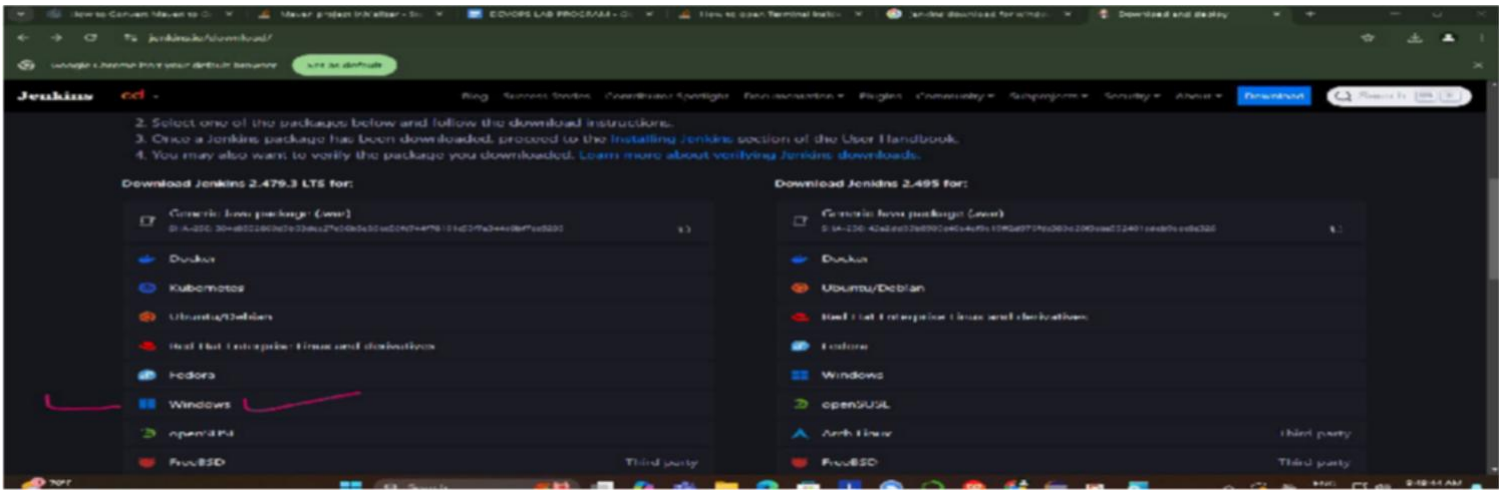
tasks.withType<Javadoc>().configureEach
{options.encoding="UTF-8"
}

AFTERDOINGALLCHANGESFINAL STEP
Toruncommands
gradlecleanbuild gradle
run
YouwillgetOutput as
HelloWorld!Welcometopgm4

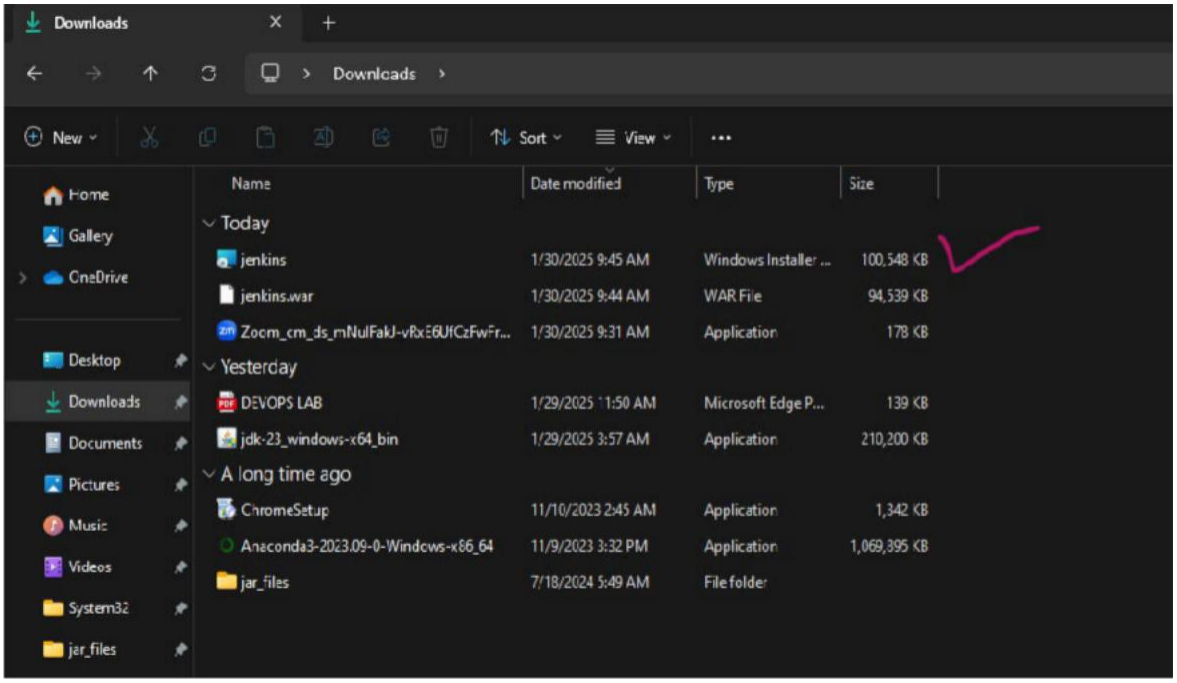


PROGRAM5: Introduction to Jenkins: What is Jenkins? Installing Jenkinson Localor Cloud Environment, Configuring Jenkins for First Use .

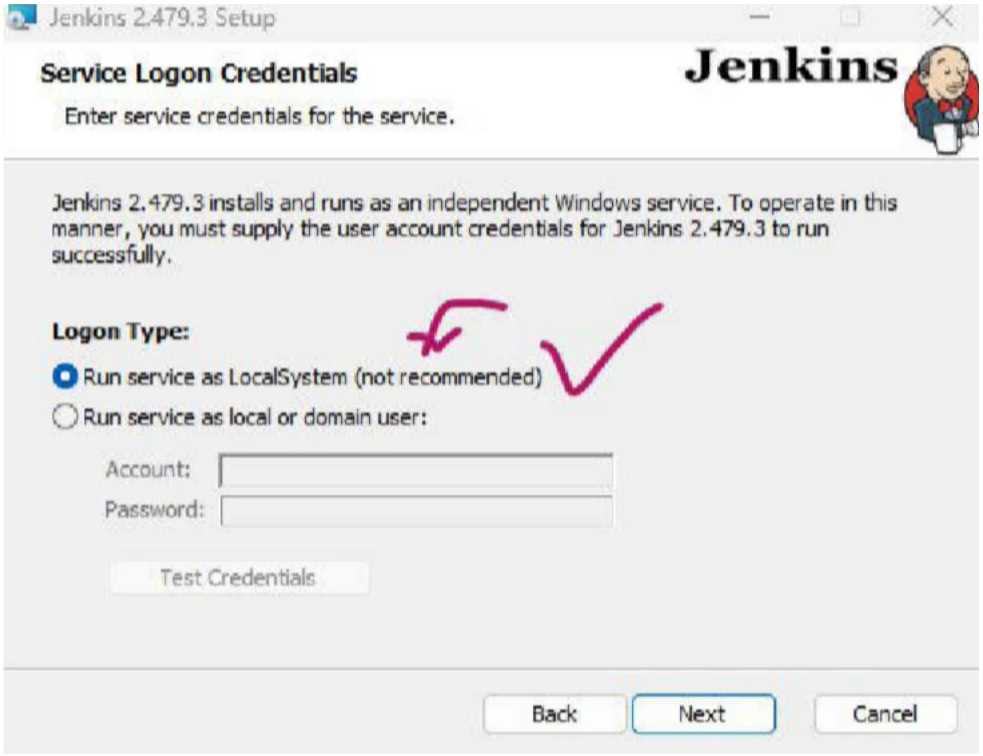
STEP1:Type Jenkins download for windows
https://www.jenkins.io/download/



STEP2:After clicking on Windows Jenkins MSI Install erexe file be installed

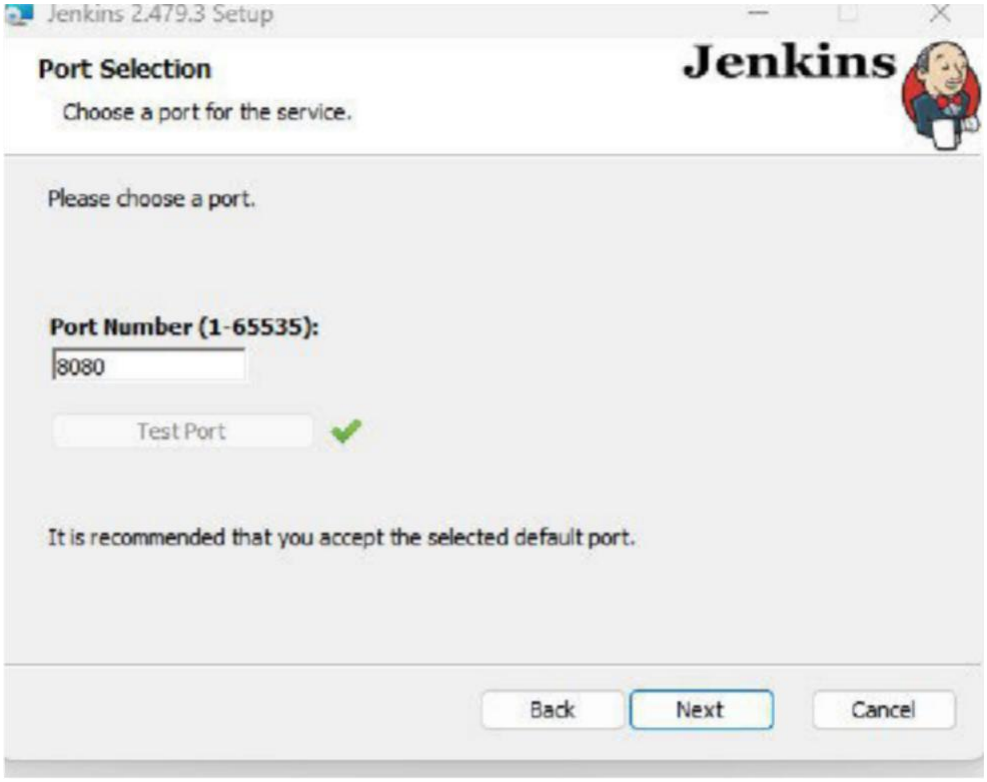


STEP3: Goto Jenkins MSI Installer click on it uopt for "Run service as Local System"



XXNeed not to ProvideAccount & Password XX

STEP4: Choose a port as 8080 and test the port and click Next



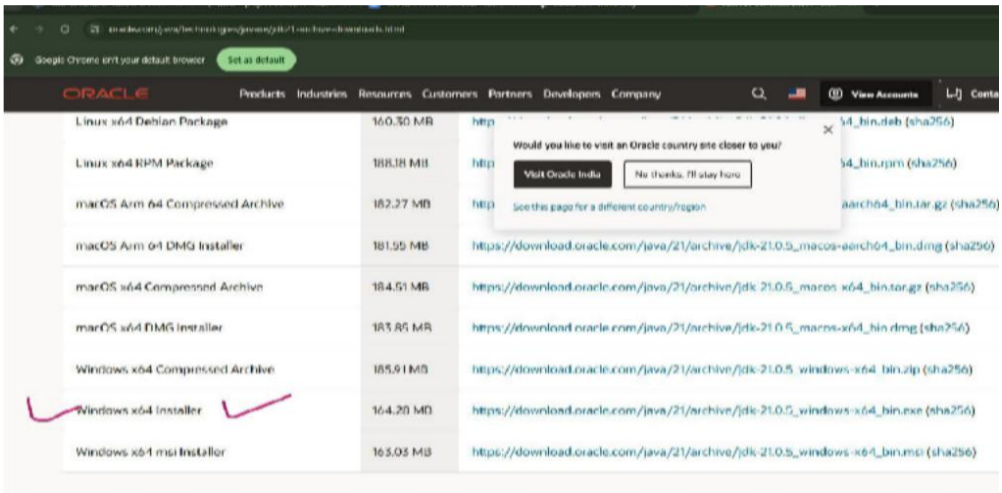
STEP5:It takes current jdk version that's available for safer side once goto cmd prompt and type command

java–version

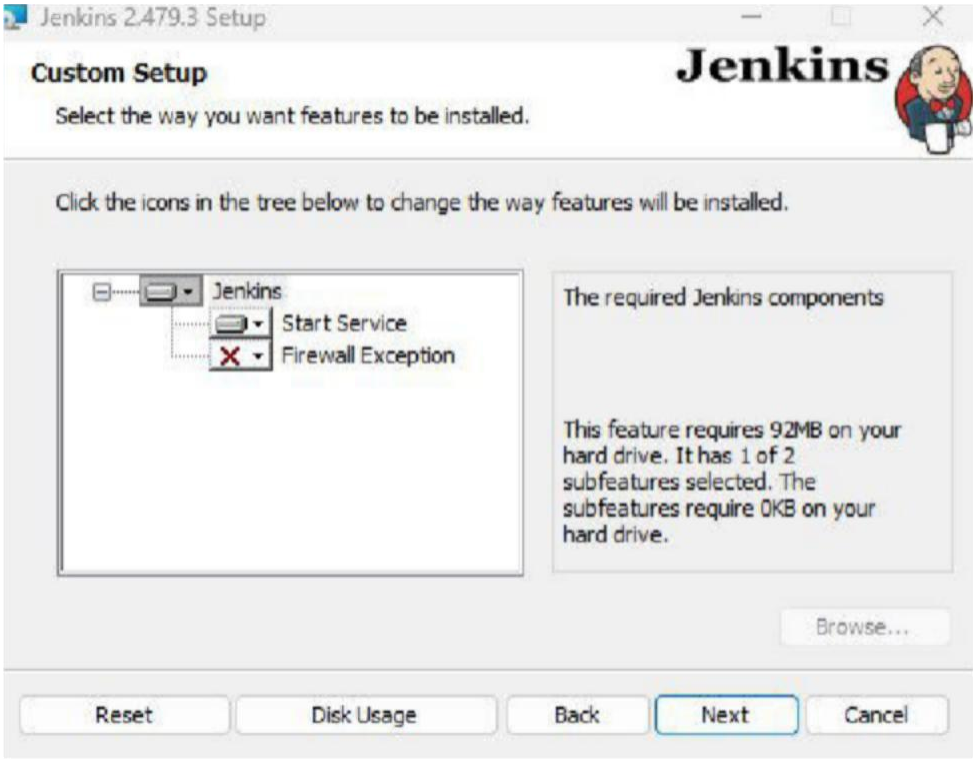
If matching click next

If failed to accept download jdk version 17 to 21 any of it

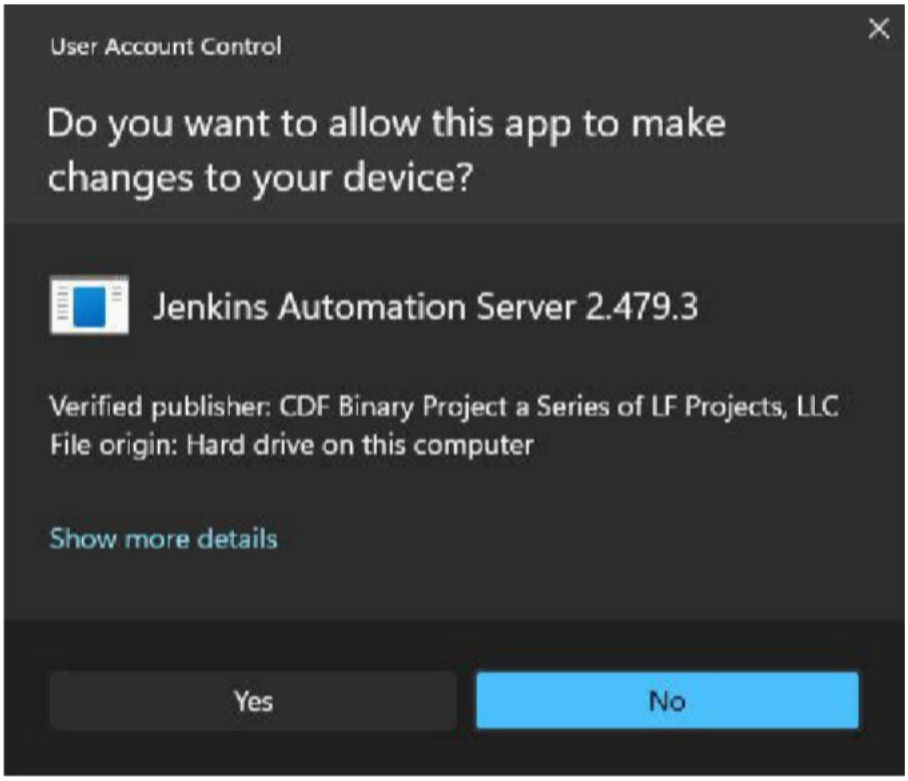
https://www.oracle.com/java/technologies/javase/jdk21-archive-downloads.html



STEP6: Click NEXT after doing above step you will getscreenasinBelowagaincontinuetoclickNext



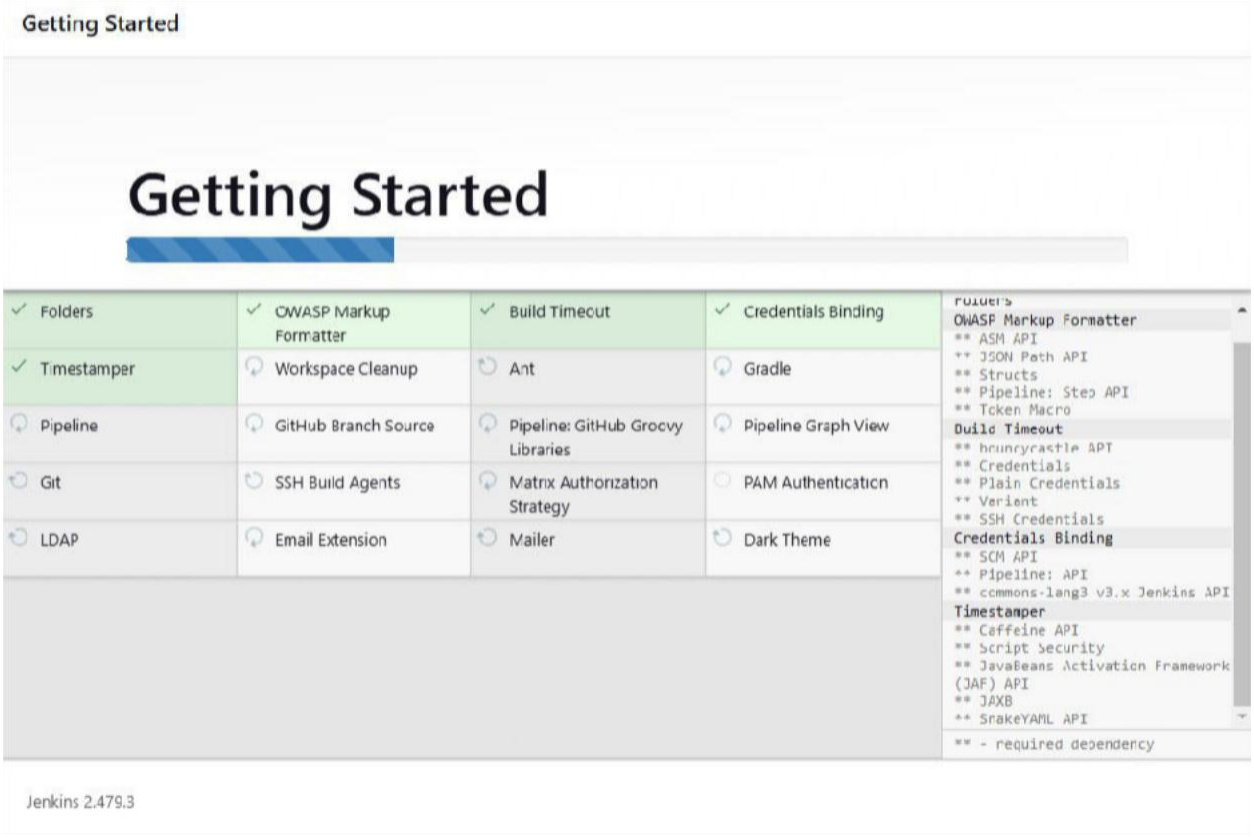
STEP7:The pop up allow for mat comes for Java Automation Server allow it click install ---------- >Finally Click Finish



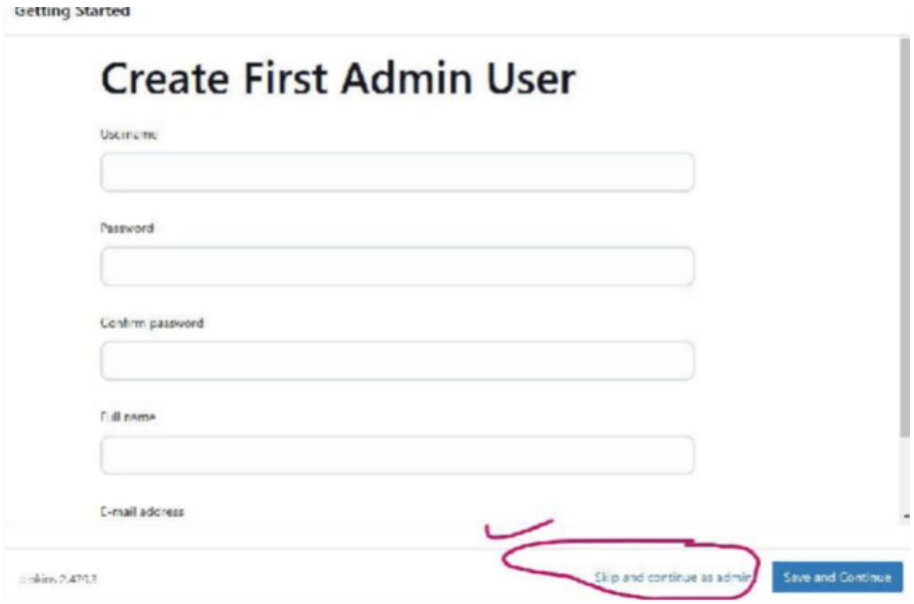
confi Now By default Our Jenkins run at
http://localhost:8080/

Final step very important Once  **It will ask for Administrator Passowords you should locate as in directory mention copy paste as in mention in file location:** C:\ProgramData\Jenkins\.jenkins\secrets\initialAdminPassword**and paste password as in mentioned belows administration password**

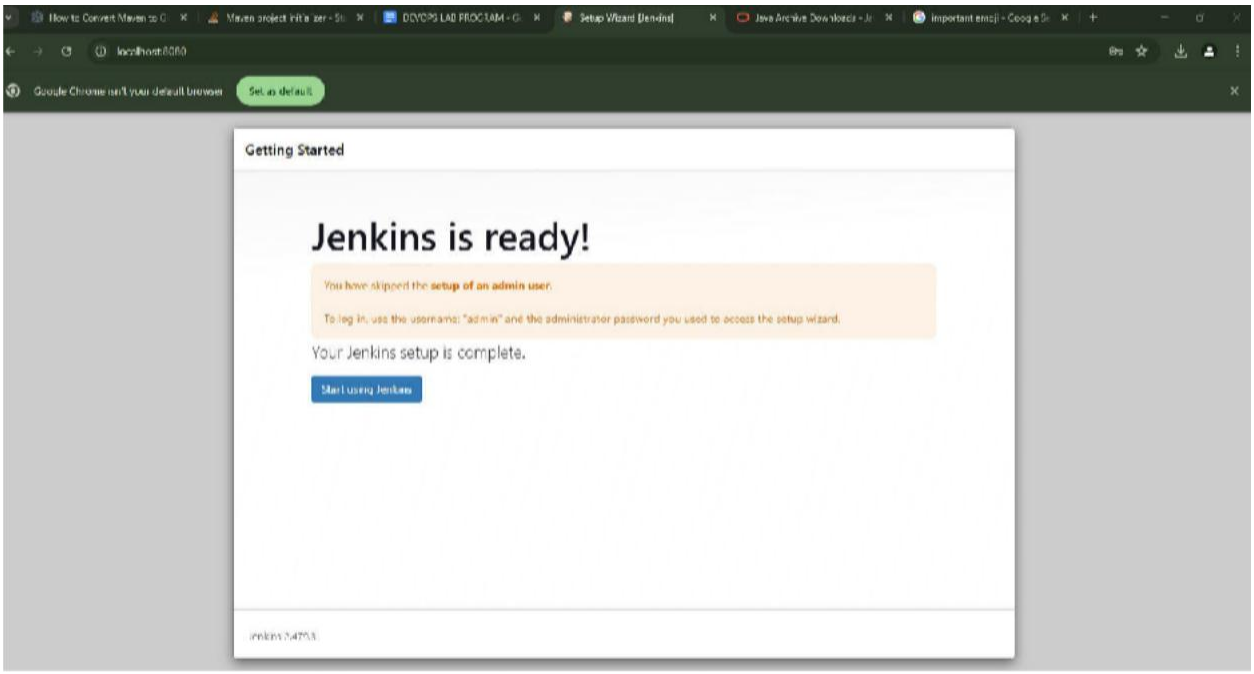
Then select Install Suggested Plugins it starts to install as in shown below



Then it asks for minimum registration You can skip and continue as admin



Final screen be:



PROGRAM6: Continuous Integration with Jenkins: Setting Up a CI Pipeline, Integrating Jenkins with Maven/Gradle, Running Automated Builds and Tests

How Is Jenkins Used for Continuous Integration?

Continuous Integration(CI) is a software development practice where developers integrate code into a shared repository frequently, usually several times a day.
Jenkins is an open-source automation server that facilitates CI by automating the build, testing, and deployment processes.

With Jenkins, developers can easily detect and fix integration issues early, improving collaboration and accelerating software delivery. By continuously integrating code, teams can maintain a high level of code quality, reduce development time, and minimize the risk of release failures.

Continuous Integration Features in Jenkins

Continuous integration involves the automatic building and testing of code whenever changes are committed to the version control system. Jenkins provides several features that facilitate CI, including:

- **Version control system integration:Jenkins integrates with various version control systems (VCS) such as Git, Subversion, and Mercurial. This allows Jenkins to monitor repositories for changes, trigger builds, and incorporate updates automatically.**
- **Build automation:Jenkins supports build automation using build tools like Maven, Gradle, and Ant. It can compile, package, and deploy code, ensuring that the latest changes are continuously integrated into the software project.**
- **Automated testing: Jenkins can execute automated tests for each build, using testing frameworks like JUnit, TestNG, and Selenium.This ensures that any issues introduced during development are quickly detected and reported, allowing developers to address them promptly.**
- **Pipeline as code: Jenkins Pipeline allows users to define their entire CI/CD pipeline as code using a domain-specific language called "Groovy." This makes the pipeline easily versionable, shareable, and more maintainable.**
- **Distributed builds: Jenkins supports distributed builds across multiple build agents, which allows for faster and more efficient build processes by distributing the workload across multiple machines.**
- **Plugins and extensibility: Jenkins offers a vast ecosystem of plugins that extend its functionality, allowing users to customize and adapt Jenkins to their specific needs. Plugins are available for various tasks, such as integrating with different VCS, build tools, notification systems, and more.**
- **Notifications and reporting: Jenkins can send notifications through various channels like email, Slack, or other messaging systems to keep the team informed about build status, test results, and potential issues. It also generates reports and visualizations for various metrics, such as test results, code coverage, and build trends.**
- **Access control and security:Jenkins provides fine-grained access control and user management, allowing administrators to control who can access specific projects, pipelines, or configuration settings. It also supports integration with LDAP and Active Directory for centralized user management.**
- **REST API: Jenkins exposes a RESTAPI that enables users to interact with Jenkins programmatically, allowing for integration with external tools, automation, and custom applications.**

Benefits and Drawbacks of Using Jenkins for CI

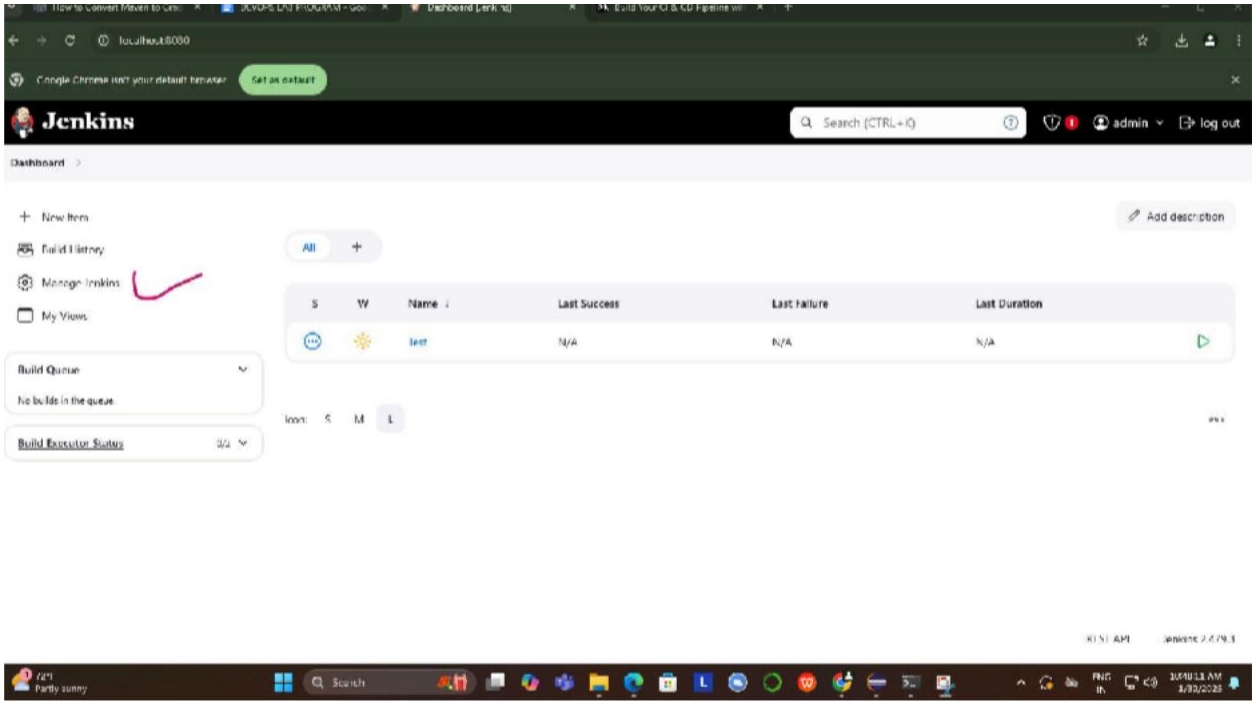
Jenkins CI offers numerous benefits that can streamline software development processes and improve overall efficiency:

- **Shorter development cycles: By automating repetitive tasks such as building, testing and deployment, Jenkins CI reduces the time developers spend on manual tasks, enabling them to focus on writing code and addressing critical issues. This accelerates the development cycle and speeds up time-to-market.**
- **Fast code integration: Jenkins CI facilitates frequent code integration into a shared repository, making it easier to detect and fix integration issues early on. This prevents the accumulation of integration problems, leading to more stable and reliable software.**
- **Short feedback loops: The automation provided by Jenkins CI allows developers to receive immediate feedback on the success or failure of their code changes. Rapid feedback helps in identifying problems early, ensuring that they can be addressed before they become more difficult and time-consuming to resolve.**
- **Automated workflows: Jenkins CI can be configured to trigger automated workflows based on specific events, such as code commits or pull requests. This enables a seamless and efficient flow of work, helping teams maintain a high level of productivity and consistency.**

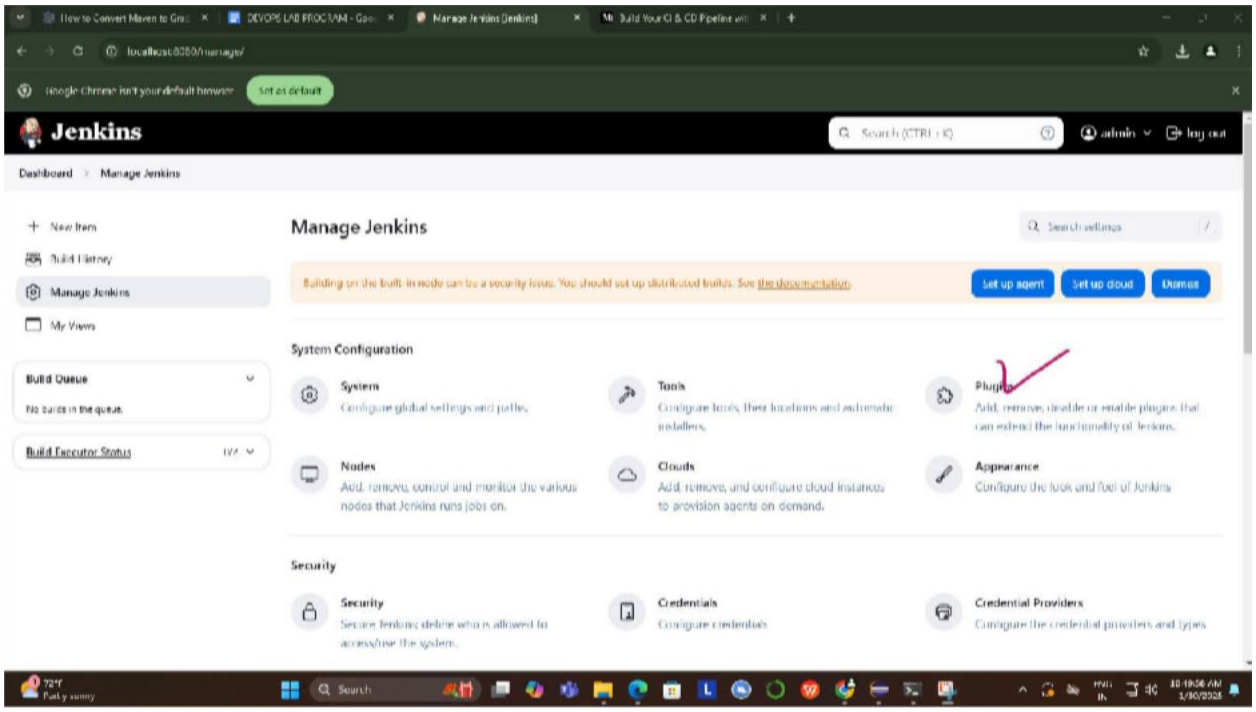
However, there are potential concerns associated with using Jenkins CI:

- **Expense: Although Jenkins itself is an open-source tool, the resources and infrastructure required to run and maintain it can be costly, especially for larger projects or organizations. Costs may include hardware, cloud services, or additional plugins and integrations needed for specific use cases.**
- **Maintenance: Jenkins CI requires regular maintenance to ensure its optimal performance, including updating plugins, monitoring the system for potential issues, and troubleshooting any problems that arise. This maintenance can be time-consuming and may require dedicated personnel with expertise in Jenkins and the underlying technologies.**
- **Not cloud native: Jenkins was designed before the advent of cloud computing, which means it doesn't naturally lend itself to cloud-based environments. To make Jenkins work in a cloud environment, substantial customization and additional tooling may be needed.**

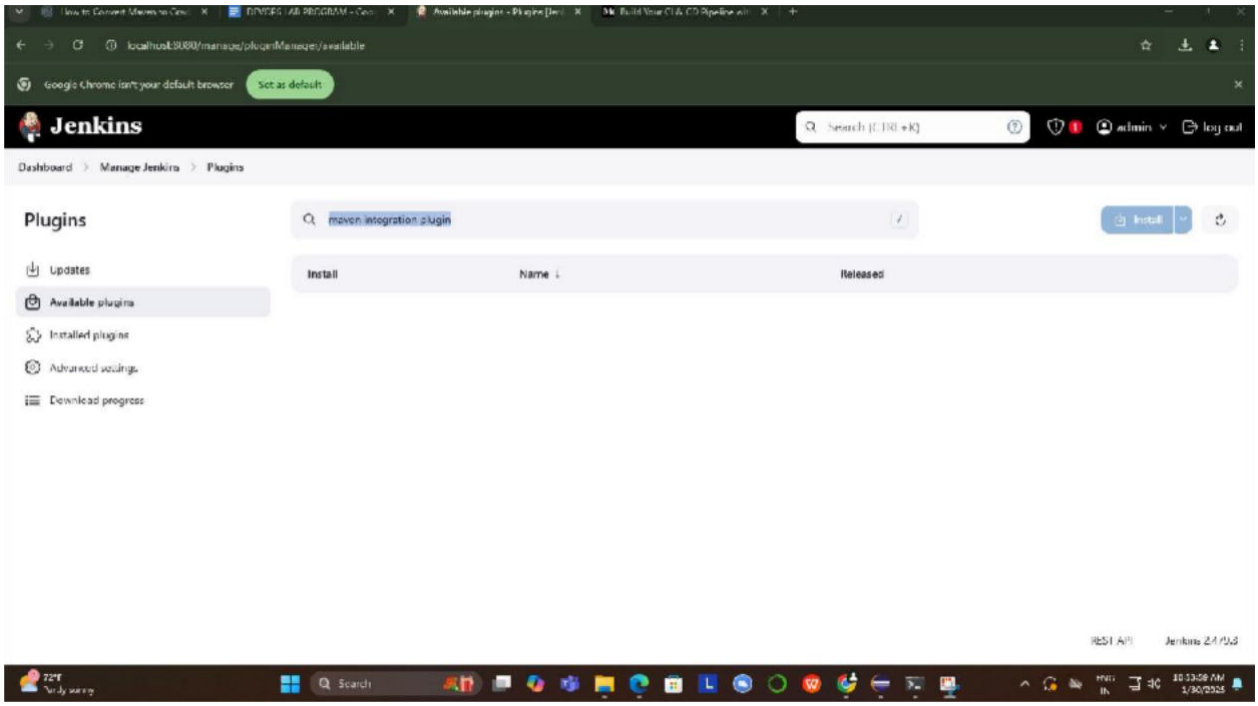
STEP1:Now coming to our Program to set CI Pipeline for Maven Go to Jenkins Dashboard and Click on Manage Jenkins



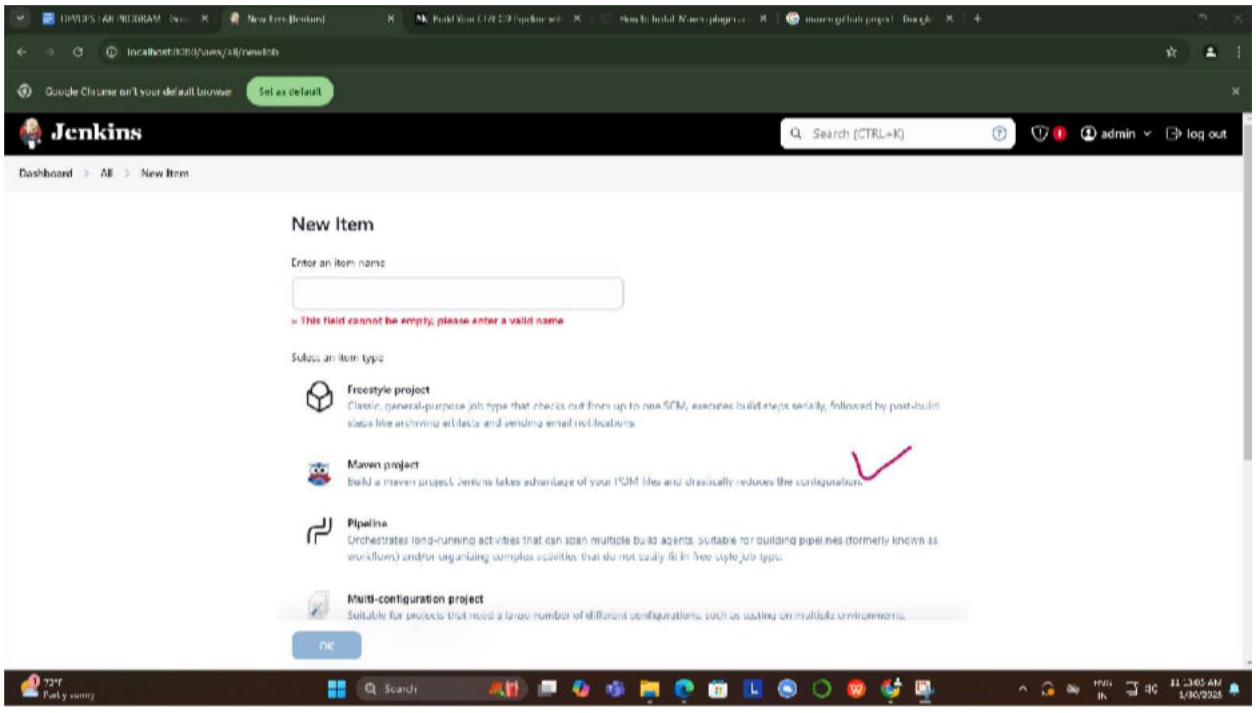
STEP2:Select Plugins



STEP3:Search for Maven Integration Plugin in Available Plugins and Install

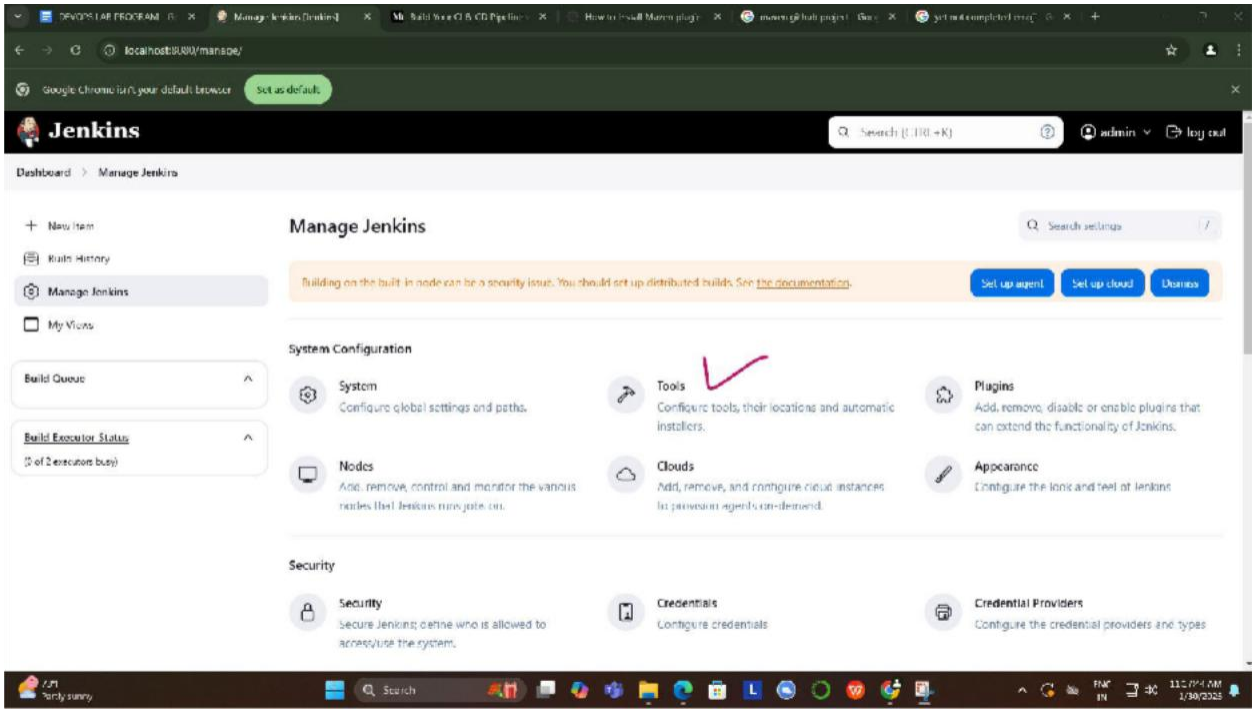


STEP4:After Maven Integration Plugin is Installed We able to see Maven Project as New Item



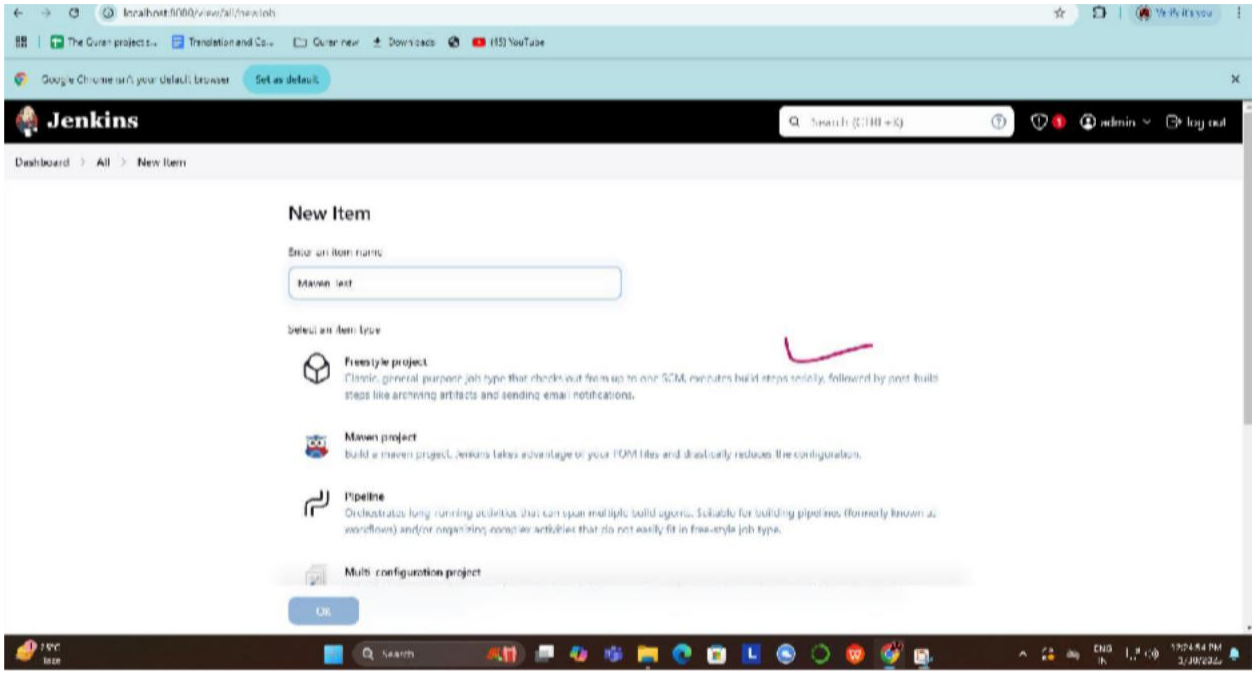
STEP5:YET not completed ⌛ **we have to configure the Location to properly Build and Run Maven Project**

So again click on Manage Jenkins and select Tools

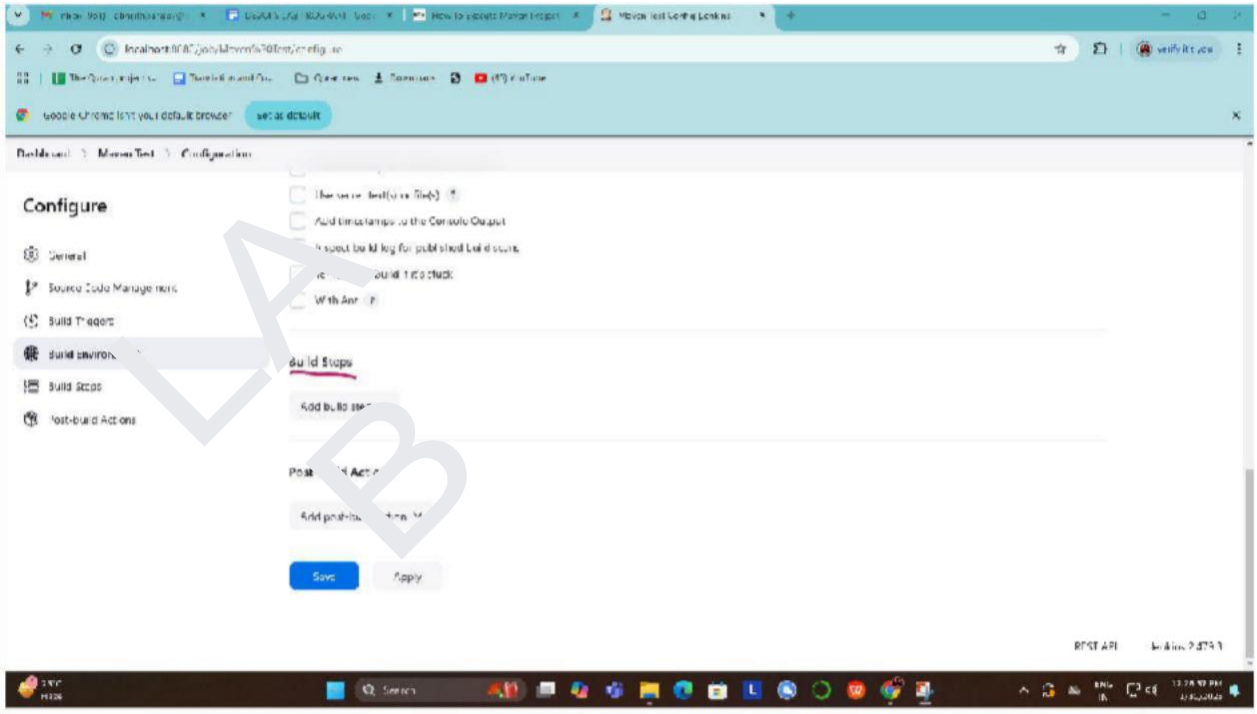


STEP6:Now lets not select Maven Project as new Item as we already have Maven project in local systems lets see how we can run the Maven Project with POM.XML

a) **Click on New Item**
b) **Provide Item Name and select Free style Project**



c) **Scroll down to‘Build‘ option. Click on ‘Add Build Step‘ and choose the value ‘Invoke top-level Maven targets‘ from the drop down list.**



LAB

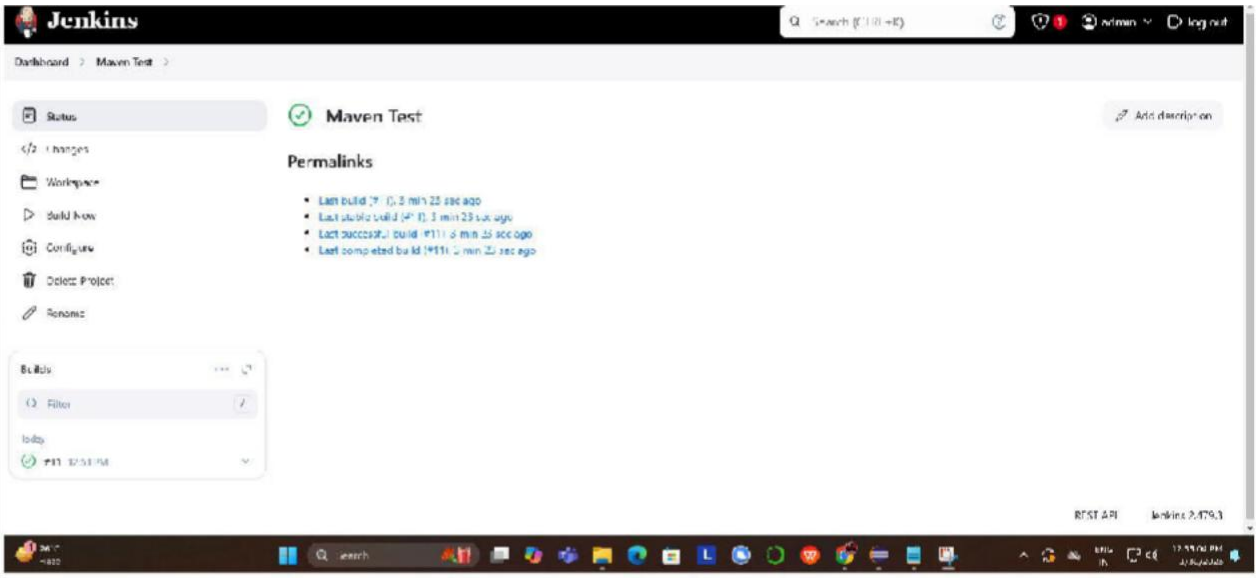
d) After selecting Invoke top-level Maven targets opt for proper environment version as in set in previous steps in my case its MAVEN_HOME

e) Enter Goal as
clean install

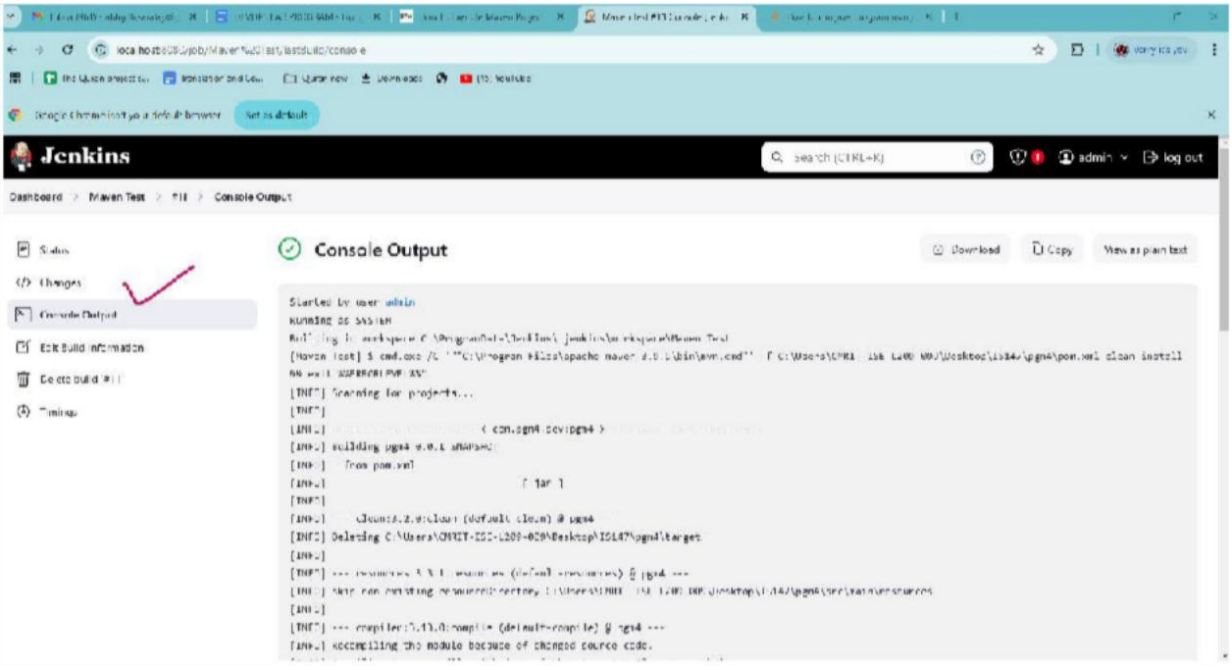
f) Before you save and apply just below Goal there is Advance option add pom.xml path

Goto pom.xml of your particular pgm and take path in my case its C:\Users\CMRIT-ISE-L209-009\Desktop\IS147\pgm4\pom.xml

After all Steps is over click on Build button the output be as in below



To see either click on build texts and goto console output or you can goto dashboard and opt to see build scripts.



PROGRAM7: Configuration Management with Ansible: Basics of Ansible: Inventory, Playbooks and Modules, Automating Server Configurations with Playbooks, Hands-On: Writing and Running a Basic Playbook.

How Do I Install Ansible on Ubuntu?

Installing Ansible on Ubuntu requires setting up an Ansible control node and connecting it to one or more Ansible hosts. The following steps describe how to perform the necessary configuration and test the new Ansible installation.

STEP1: Configure Ansible Control Node

The Ansible control node is a system used to connect to and manage Ansible host servers. Proceed with the steps below to setup the control node on the main server:

1.Create an administrator-leveluser for the control node. Use the add user command:

sudo adduser[username]

2.When prompted, define a strongaccount password.

Optionally, provide more details about the user by answering questions. Press Enter to skip a question.

3. Use the following user mod command to assign super user privileges to the account:

sudo usermod -aG sudo [username]

A membership in the sudo group allows the user to utilize the sudo command to perform administrative tasks.

4. Switch to the newly created user on the control node:

sudo su [username]

Note: The Ansible control node can be a dedicated server, alocalmachine, ora virtual machine running Ubuntu.

STEP2: Set upan SSH Key pair

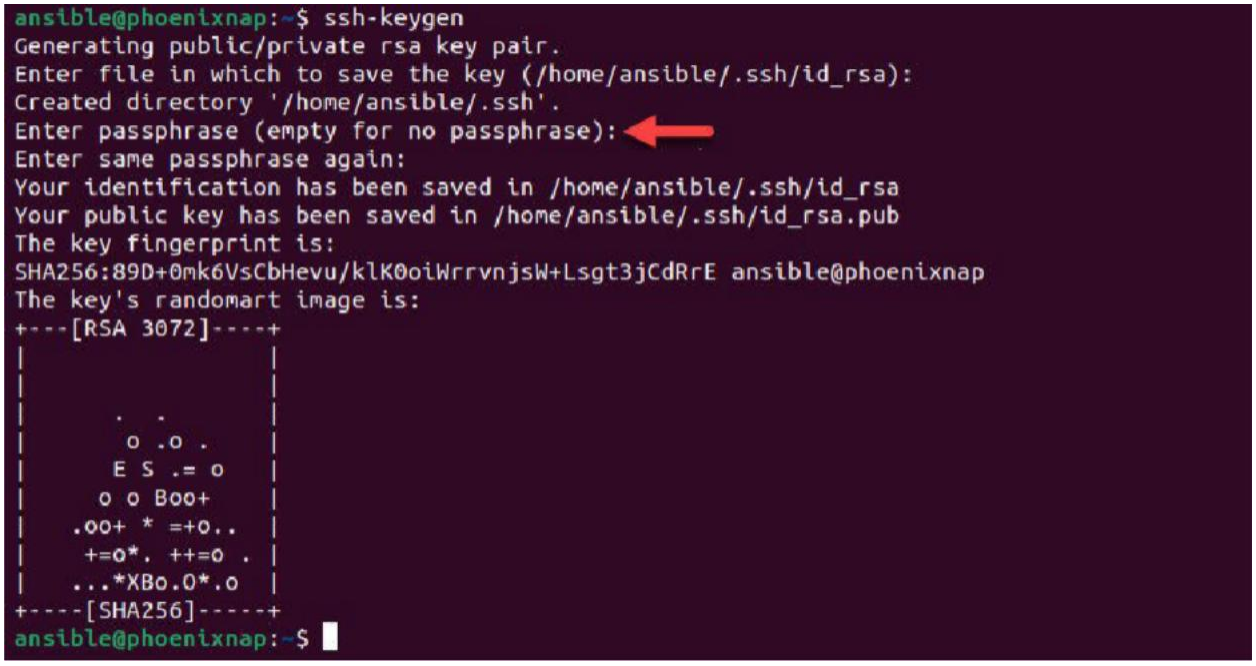
The Ansible control node uses SSH to connect to hosts. Generate an SSH key pair for the Ansible user by executing the following steps:

1. Enter the command below using the Ansible control node command line: ssh-keygen

Note: If an SSH key pair with the same name already exists, SSH displays a warning asking the user to decide whether to over write it. Over writing makes the previous SSH key pair unusable, so ensure the old keys are no longer needed before confirming.

2. When prompted, provide a pass phrase. While adding a strong pass phrase is recommended, pressing Enter allows the user to skip the pass phrase creation.

The system generates the public/private key pair and prints the random art image.



STEP3: Configure an Ansible Host

Ansible hosts are remote servers managed by the Ansible control node. Each host must have the control node's SSH public key in to authorized_keys directory. Apply the steps below for each new

1.Use the following ssh-copy-id command on the control node to copy the public

ssh-copy-id[username]@[remote-host]

Replace [username] with an existing administrative user on the host system and [remote-host] with the remote host domain or IP address. For example, to copy the key to the user ansible on the host with the local IP address 192.168.0.81, type:

To know IP type command

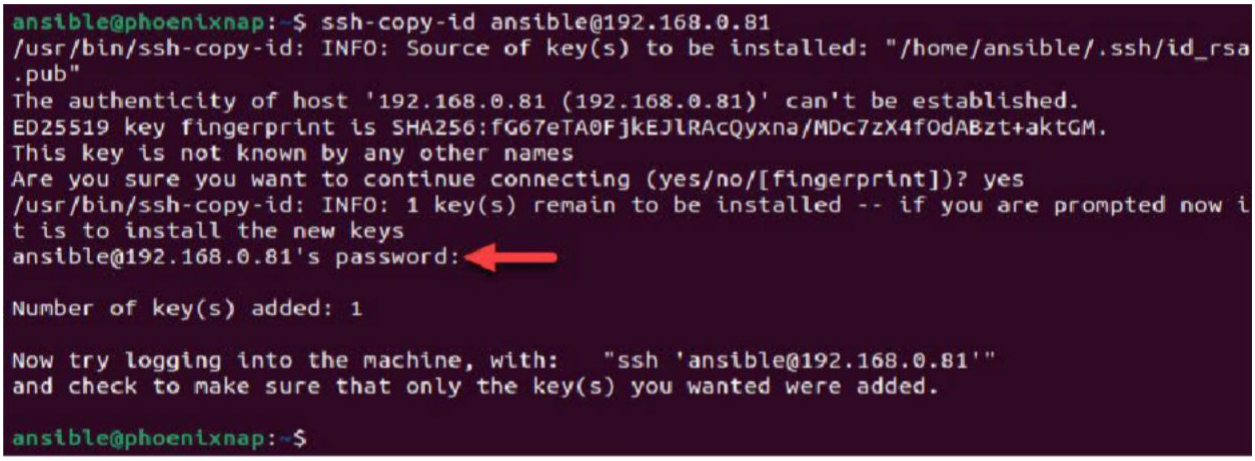
cat/etc/resolv.conforhostname-i

ssh ansible@192.168.0.81

sudoaptinstallansible-y

2.Type yes and hit Enter when asked whether to continue connecting to an

3.Enter the remote host account



Theutilityuploadsthepublickeytotheremotehostacco

STEP4:Install Ansible

Use the APT package manager to install the Ansible package on the control node system:

1.Ensurethepackageindexisuptodat

sudo apt

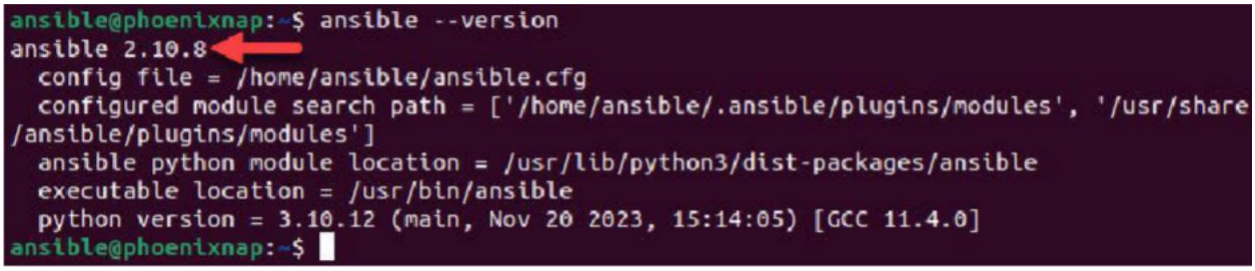
2.InstallAnsibleonUbuntuwiththefollowingcomma

STEP5: Verify the Installation

Check that Ansible was successfully installed on your Ubuntu system using the ansible command:

ansible--version

The output displays the Ansible version number, the location of the configuration



STEP6: Set up the Inventory File

Once Ansible is installed on the control node, set up an inventory file to allow Ansible to communicate with remote hosts. The inventory file contains all the

Note: Foranin-depth overview of creating files on remote hosts, refer to our article How to Create a File In Ansible.

Follow the steps below to create an inventory file on the control node:

1. Create the ansible subdirectory in the etc directory: sudo

mkdir -p /etc/ansible

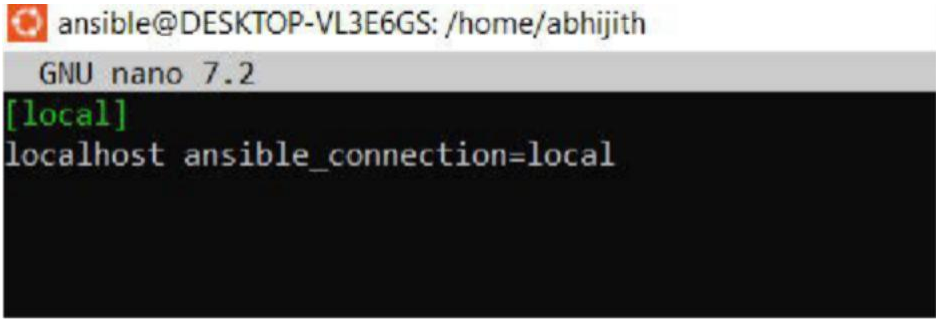
2. Use a text editor such as Nano to create a file named hosts:

sudonano /etc/ansible/hosts

3. Add localhost that the control node will manage. Use the following format: [local]

localhost ansible_connection=local

The [local] line allows for the creation of categories to organize local hosts.The following example adds a local host using its local IP address 192.168.0.81 and sorts it into the servers category:

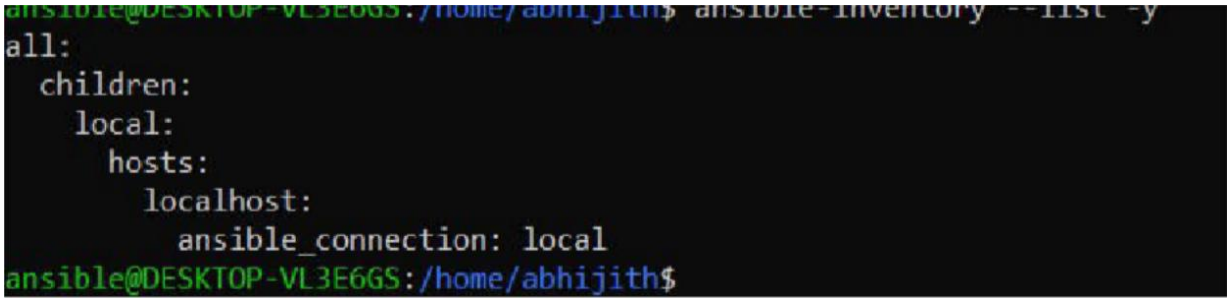


4. Save the file and exit.

5. Enter the command below to check the items in the inventory:

ansible-inventory --list -y

The output lists the hosts:



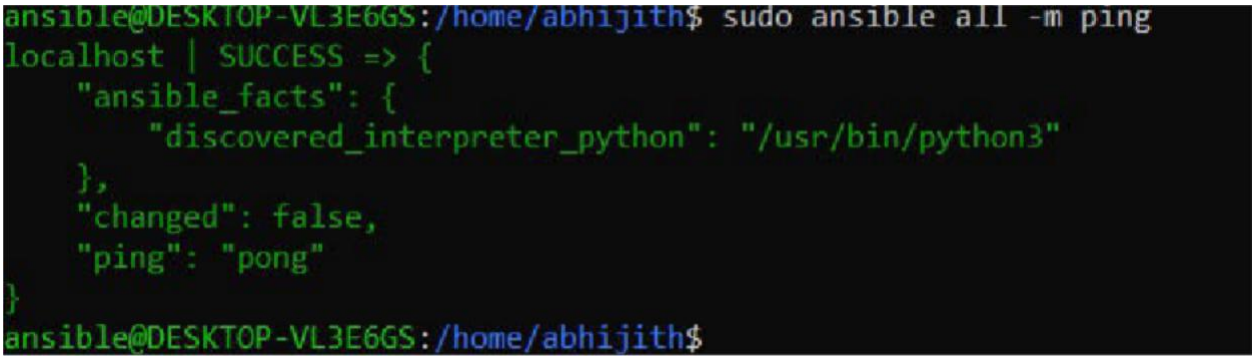
STEP7: Test the Connection

To ensure the Ansible control node can connect to the local hosts and run commands, use the following ansible command to ping the hosts from the control node:

sudoansibleall-mping

Note: When a user connects to the remote hosts for the first time, Ansible asks for confirmation that the hosts are authentic. To confirm the authenticity, enter yes when prompted.

The output confirms the successful



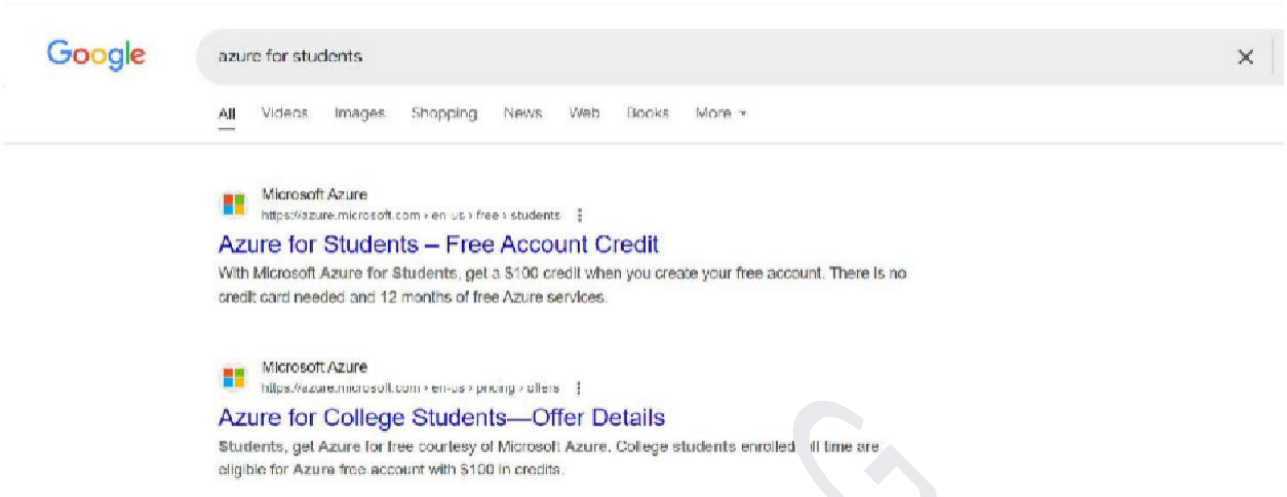
TheAnsiblecontrolnodeisnowsetuptocontroltheconnectedremotehost

Conclusio

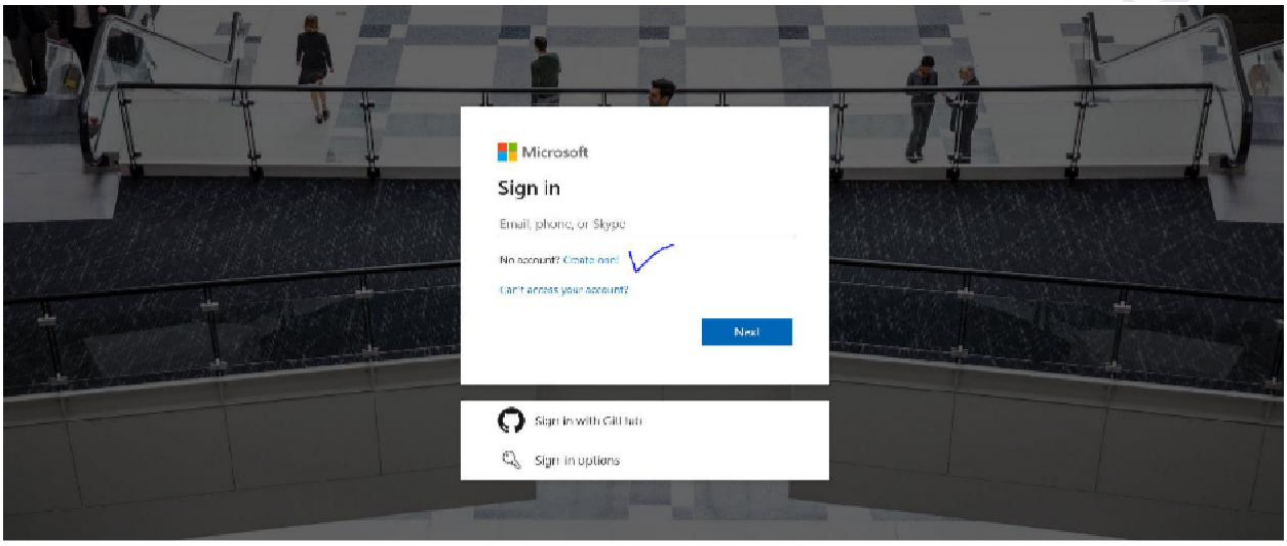
After following the steps in this guide, you have successfully installed Ansible on Ubuntu and can execute commands and playbooks on remote hosts. The guide provided instructions for setting up the Ansible control node and connecting it

PROGRAM9: Introduction to Azure DevOps: Overview of Azure DevOps Services, Setting Up an Azure DevOps Account and Project.

STEP1:Goto Google chrome and type azure for students
https://azure.microsoft.com/en-us/free/students

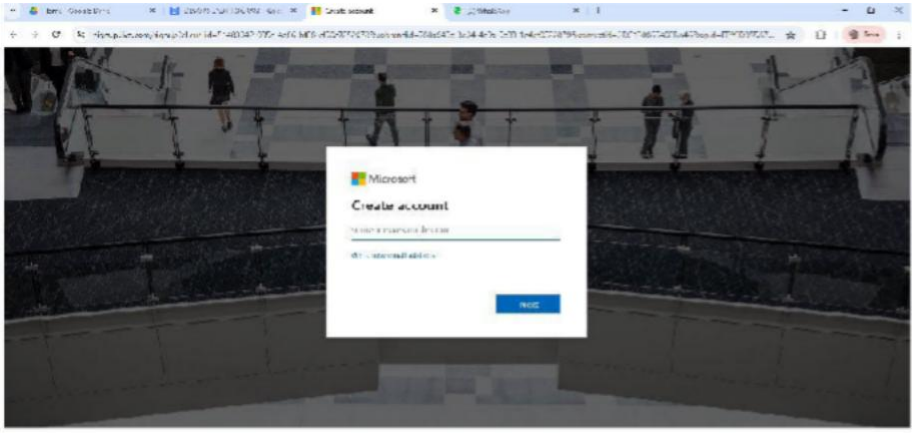


STEP2:Click on Start Freeafterthatugetscreenasin

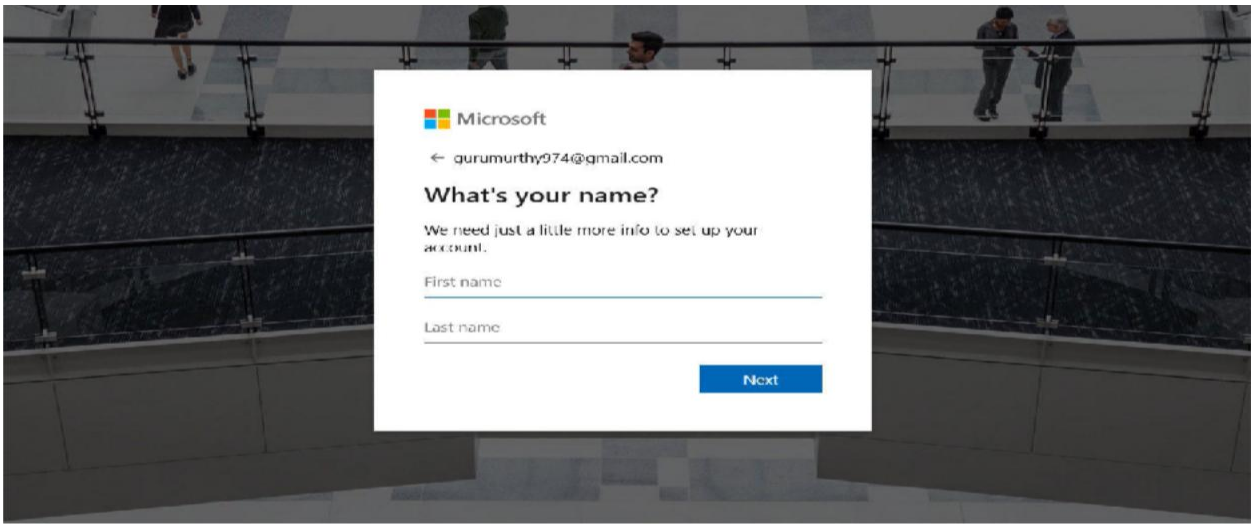


Click on Create one, If u have Github account you can Sign in using github account better way is to create one account

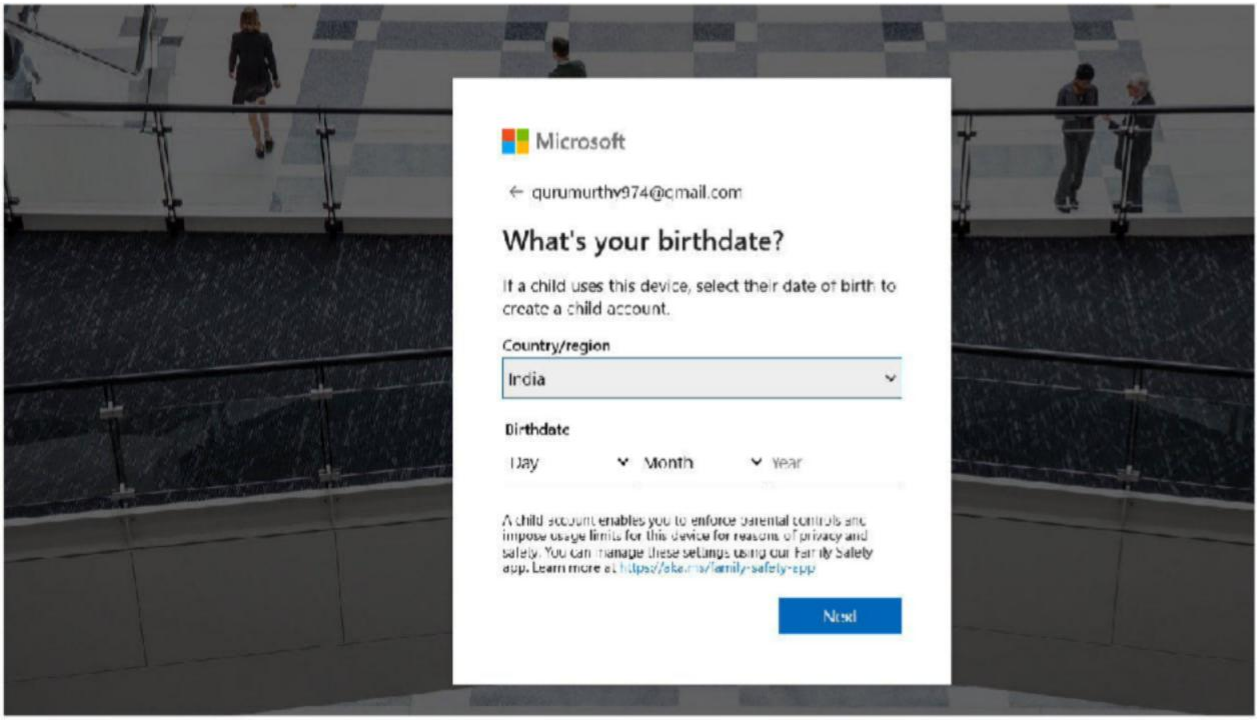
STEP3: Provide your email id at place of create account



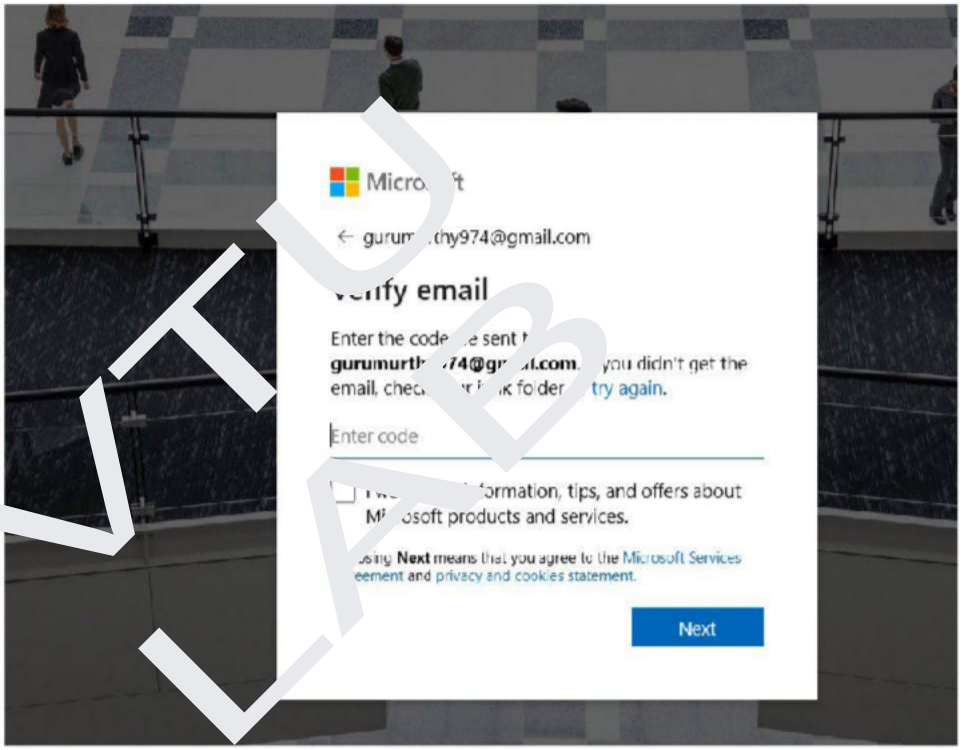
STEP4:After password is set provide your Firstname Lastname



Then provide Country,Date of Birth

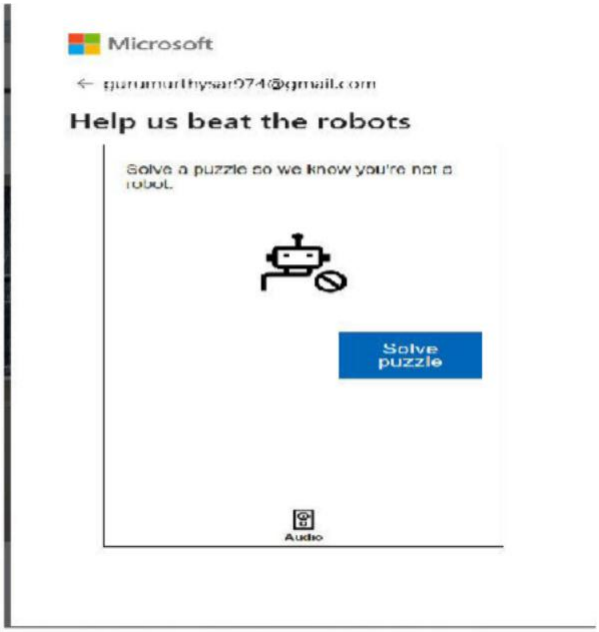


STEP5: Verification code be mailed to the mentione done kindly type it

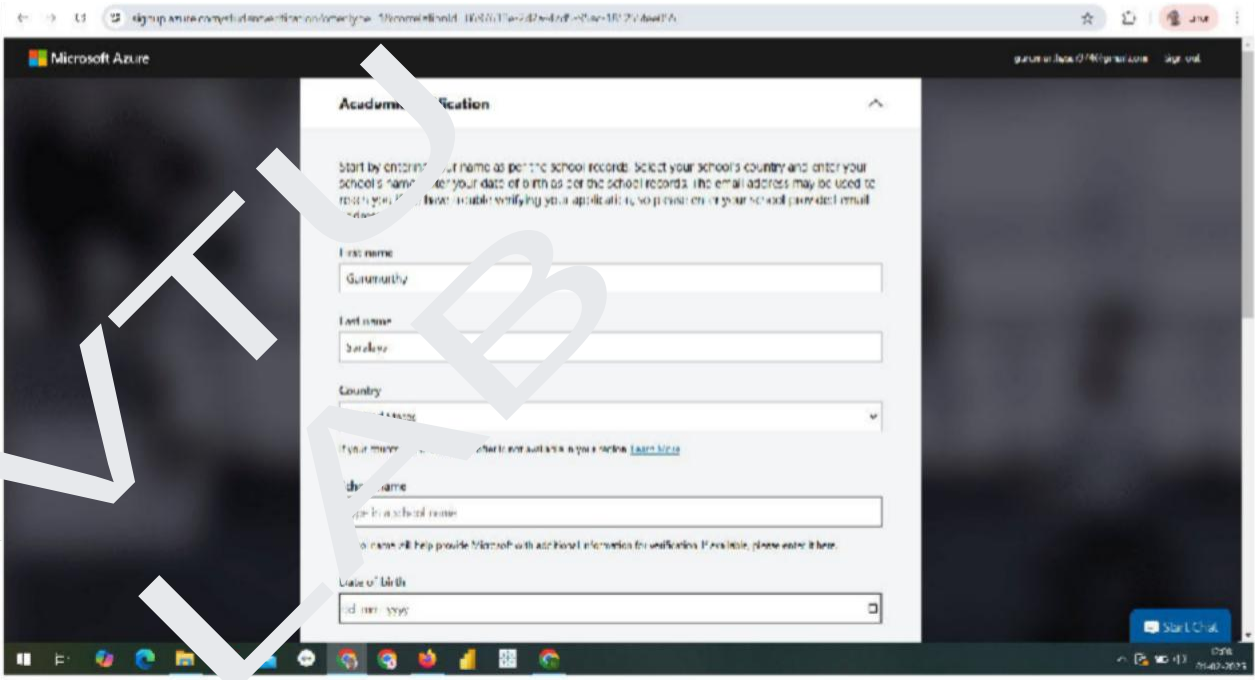


VTU LAB

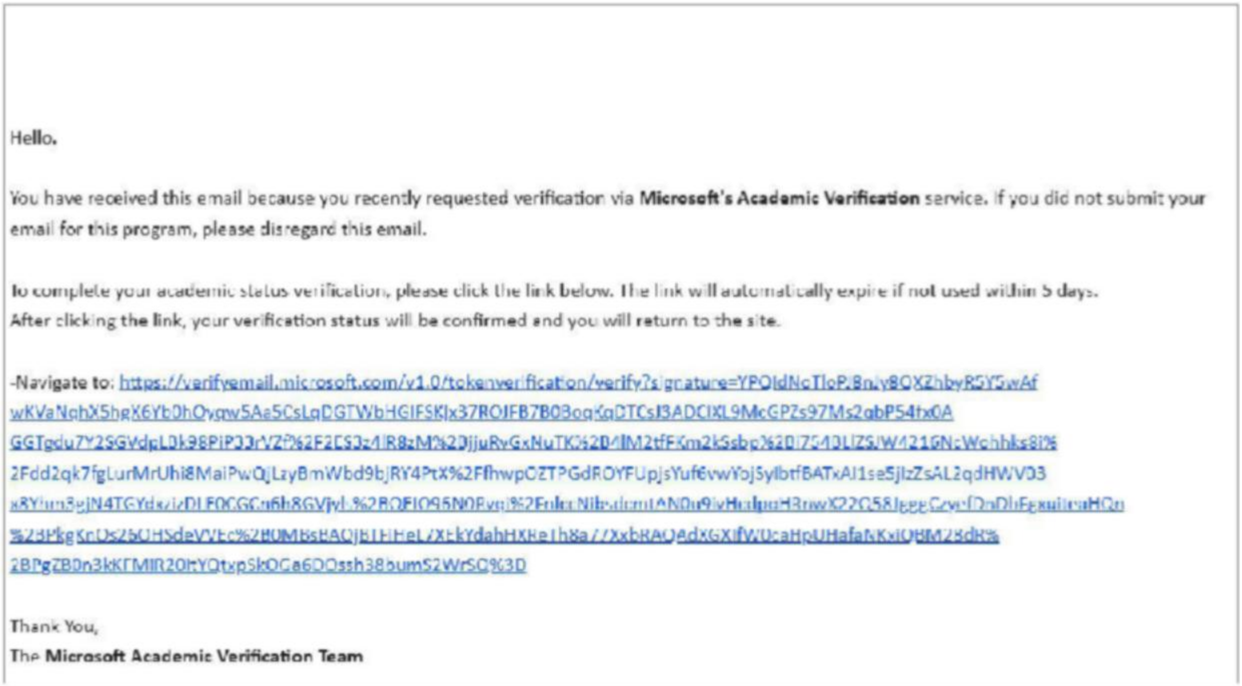
STEP6: After code is verified as you got in the mail referred U be given an option to solve puzzle game



STEP7:This step is the important once where u fill your Academic Details properly where you provide College email id as in provided by your Individual colleges later the verification code again comes



VTU LAB


Hello,

You have received this email because you recently requested verification via **Microsoft's Academic Verification** service. If you did not submit your email for this program, please disregard this email.

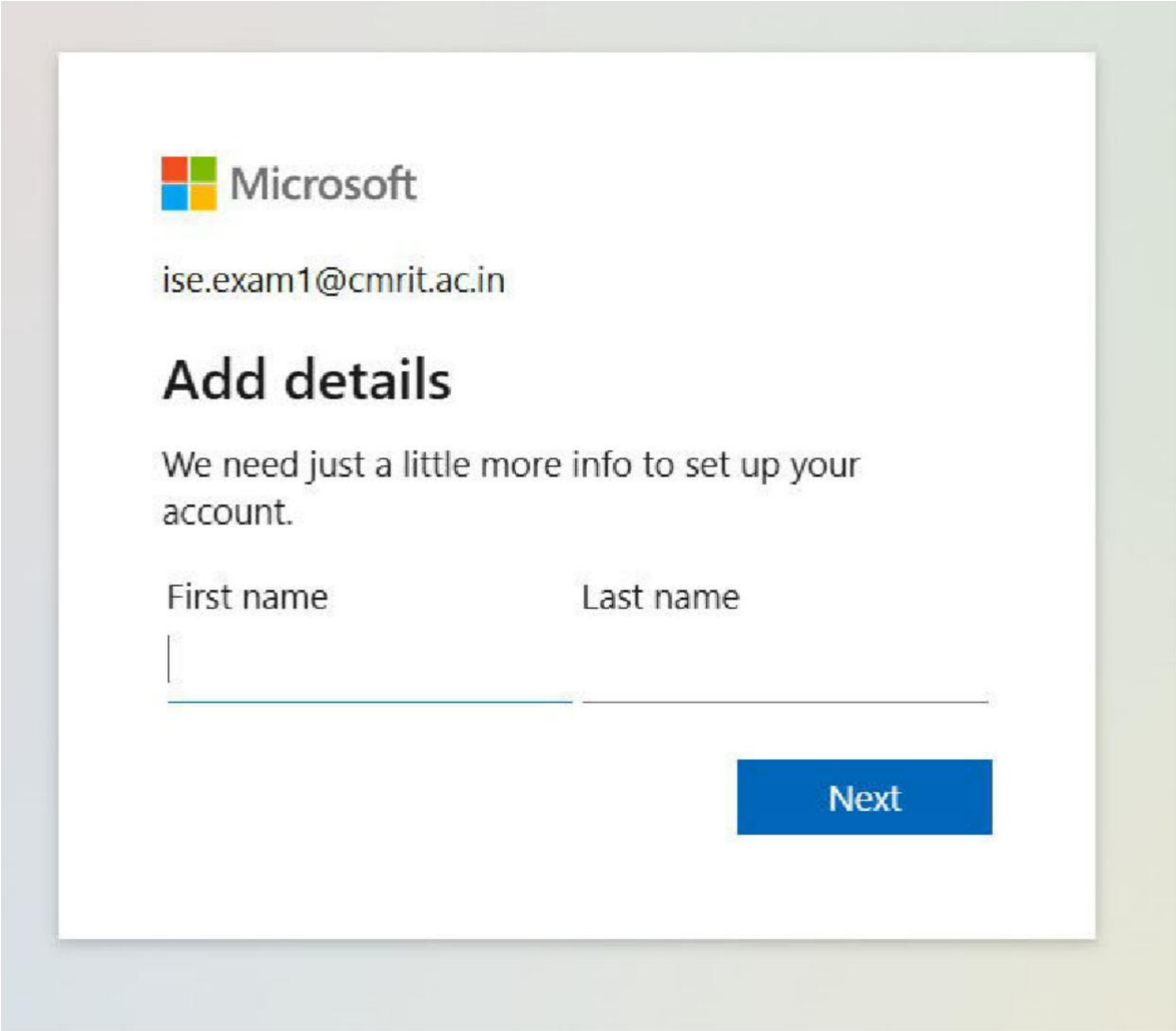
To complete your academic status verification, please click the link below. The link will automatically expire if not used within 5 days.
After clicking the link, your verification status will be confirmed and you will return to the site.

-Navigate to: https://verifyemail.microsoft.com/v1.0/tokenverification/verify?signature=YPOIdNcTIoPJ8nJy8QXZhbyRSY5wAf
wKVaNqhX5hgX6Yb0hOyqw5Aa5CsLqDGTWbHGIFSKJx37ROJFB7B0BoqKqDTCsJ3ADCIXL9McGPZs97Ms2qbP54tx0A
GGTgdu7Y2SGVdpLDk98PiP33rVZP%2F2ES3z4IR8zM%2DjjuRvGxNuTK%2D4IM2tfFKm2kSsbp%2DI7S4DLIZSJW421GNcWchhks8I%
2Fdd2qk7fgLurMrUhi8MaiPwQjLzyBmWbd9bjRY4PtX%2FfhwpOZTPGdROYFUpjsYuf6vwYoj5ylbtfBATxAJ1se5jlzZsAL2qdHWVD3
[illegible]
[illegible]
2BPgZB0n3kKTMIR20HYQtvpSkOGa6DOssh38bum52WrSQ%3D

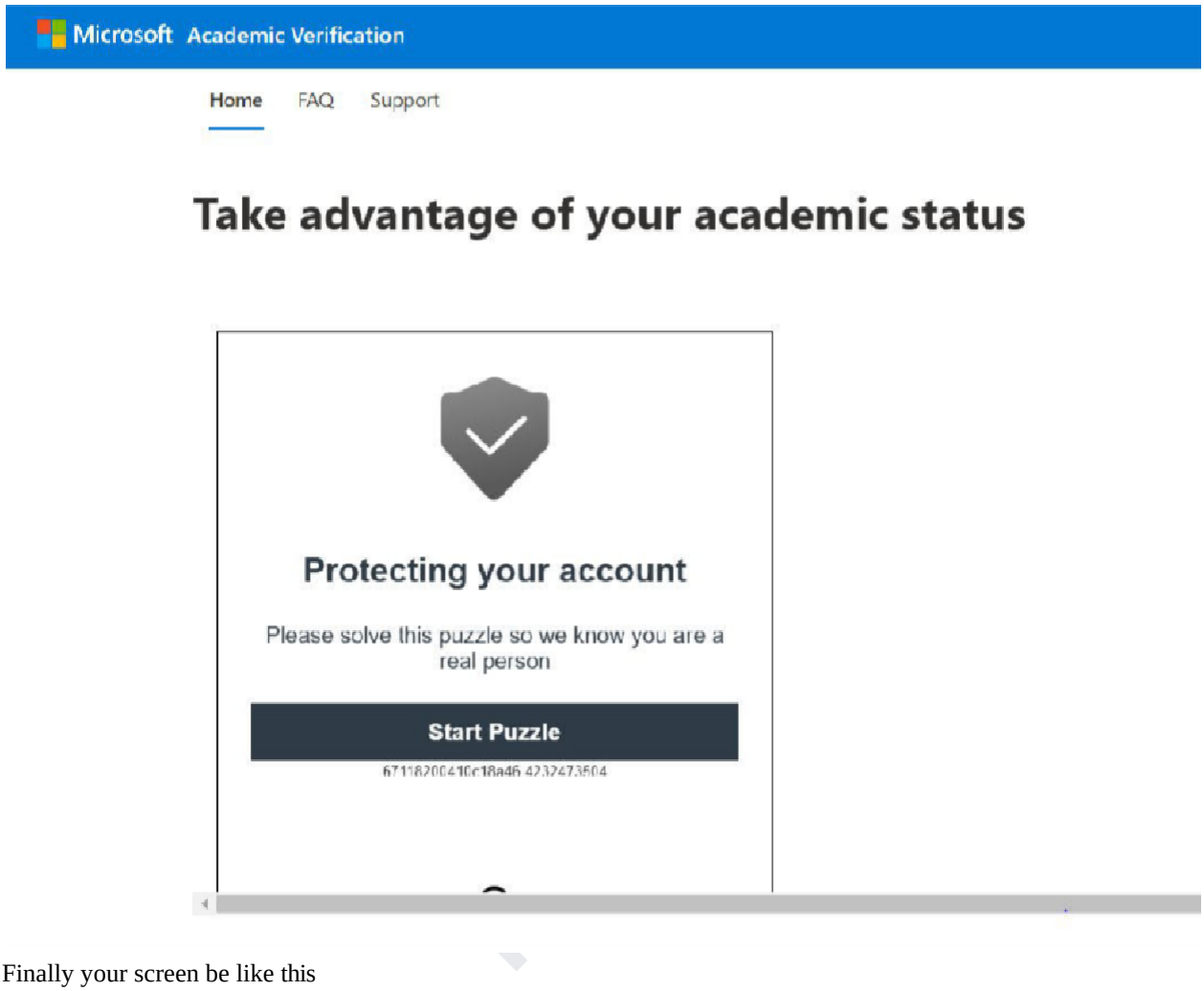
Thank You,
The **Microsoft Academic Verification Team**

Click on link sent

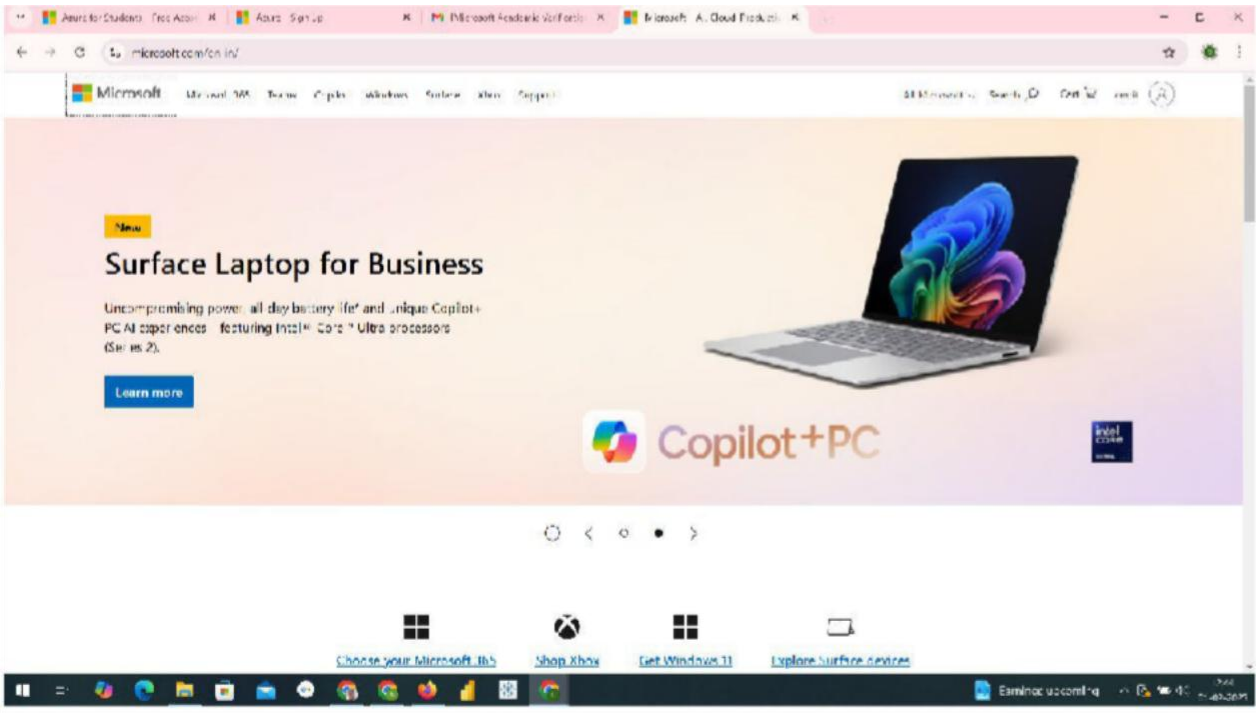
After you Click The screen be as in Below



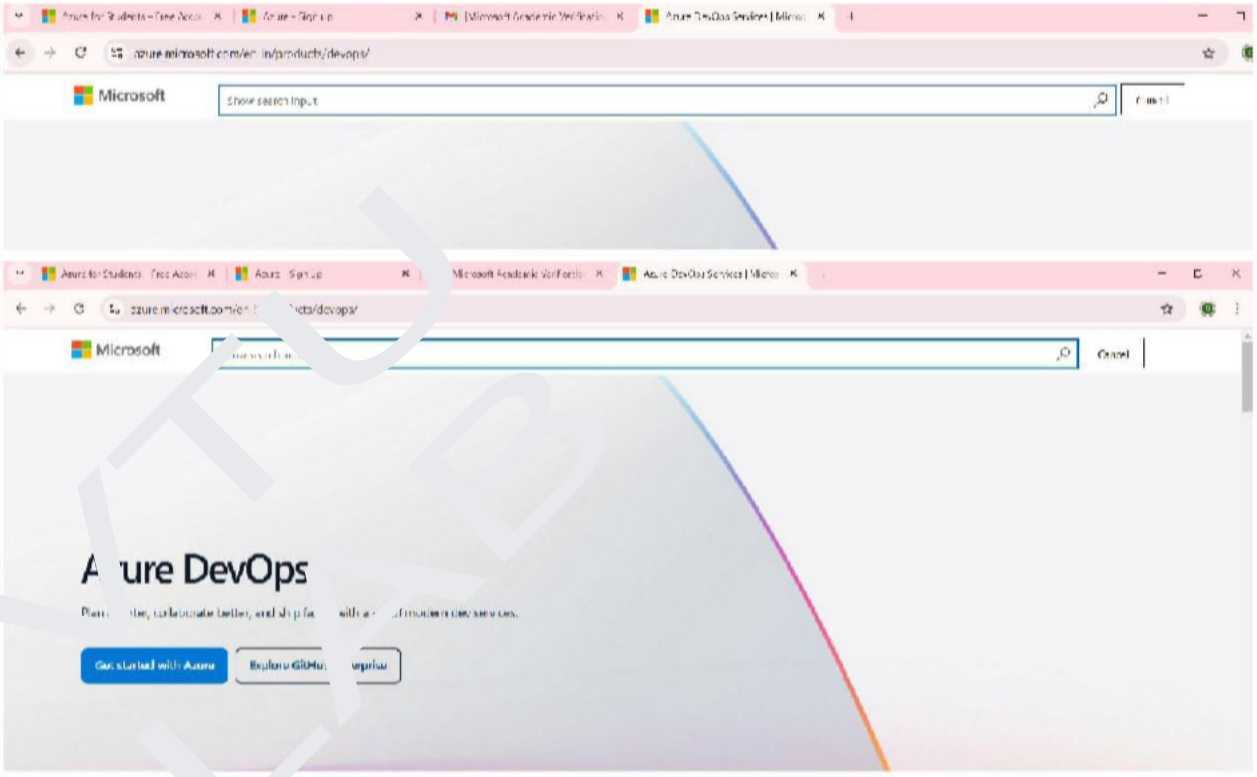
TO MAKE YOUR ACCOUNT SECURE IT AGAIN HAVE PUZZLE

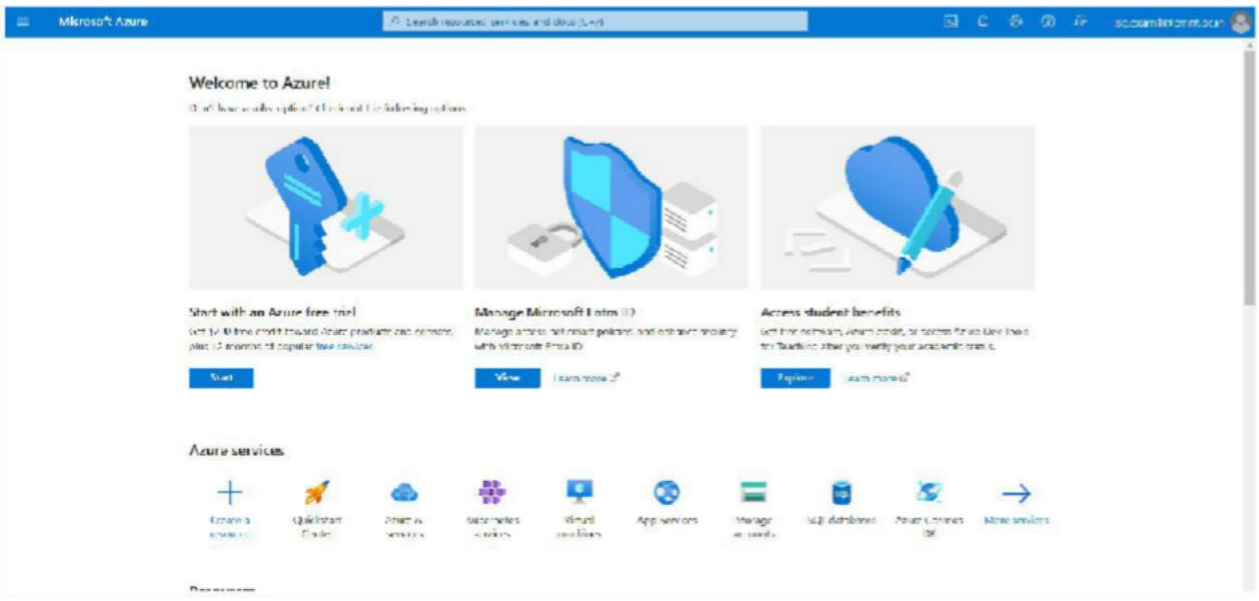


Finally your screen be like this



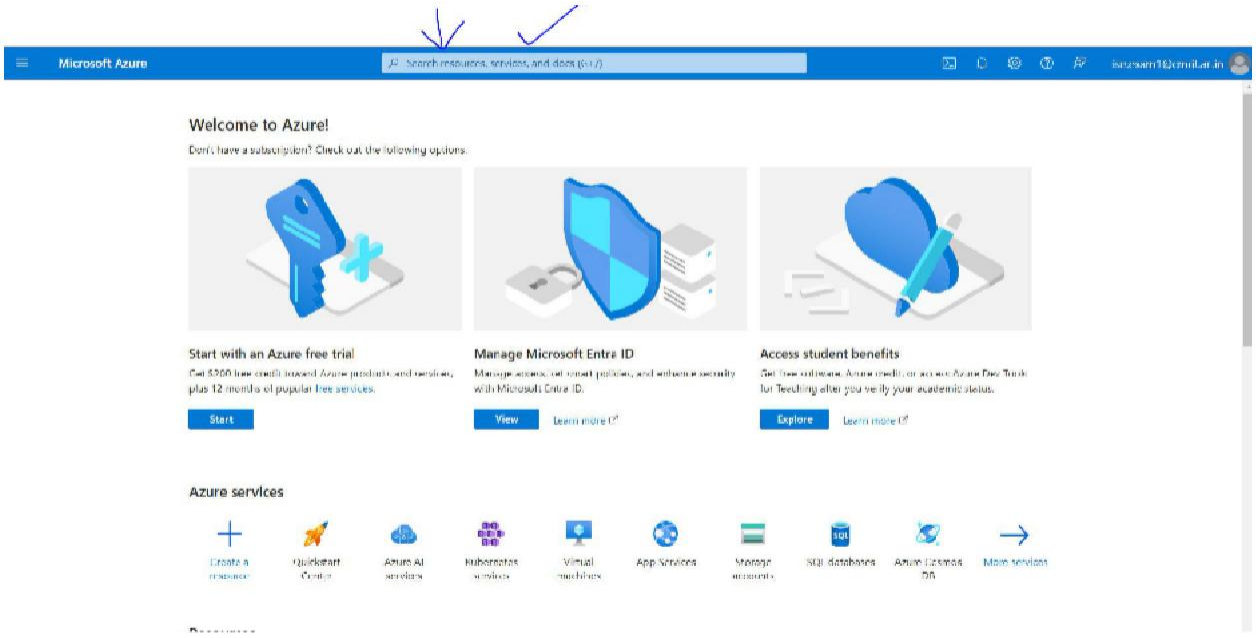
GoTosearchandtypeAzureDevops



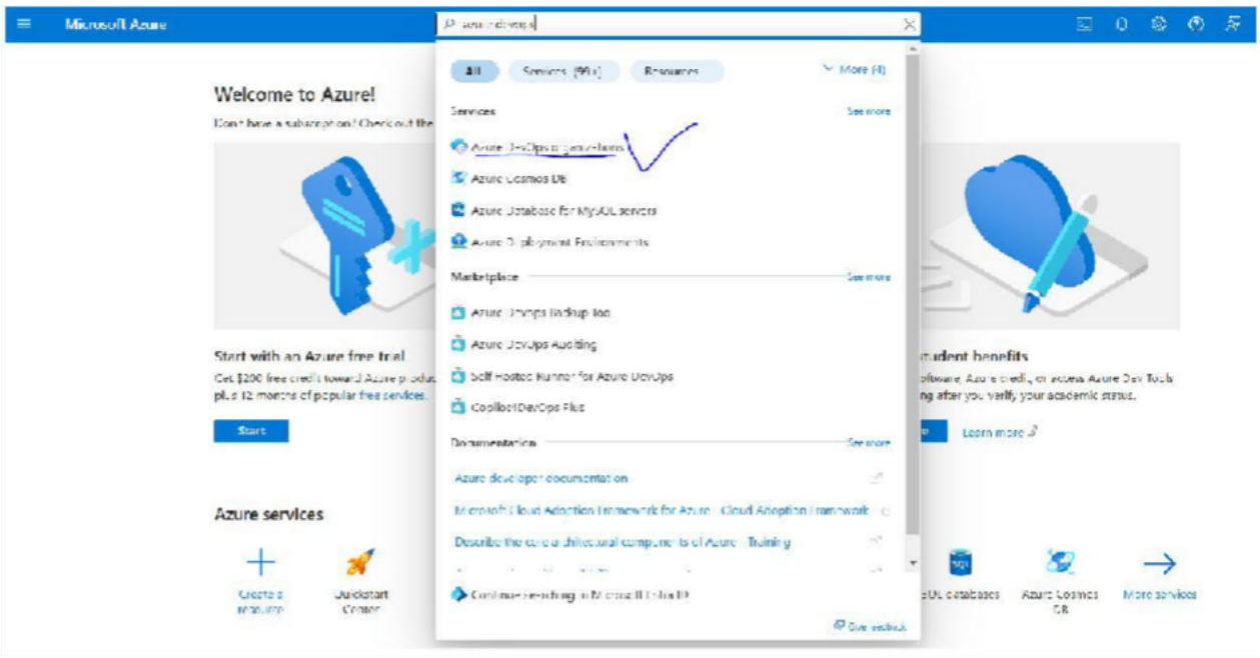


THIS IS THE HOME PAGE OF THE MICROSOFT AZURE where you can seen number of services now our target is Azure Devops In top where have

search for services

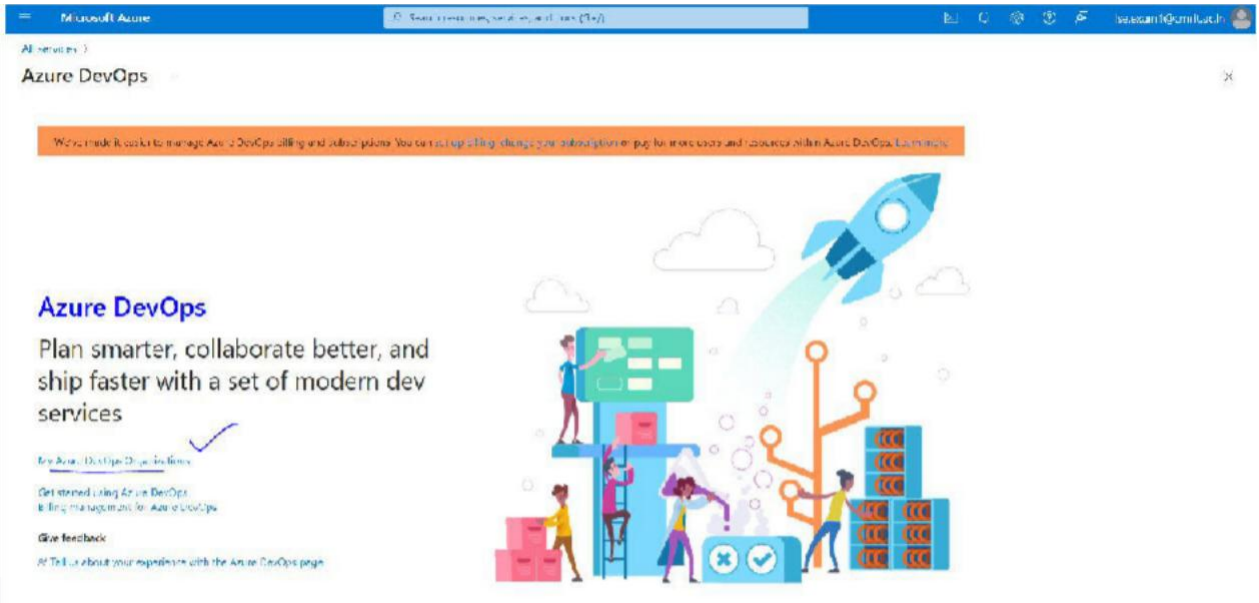


TypeAzureDevops



STEP9:

After your opting for Azure Devops Organizations you get a screen as in below now select My Azure DevOps Organizations

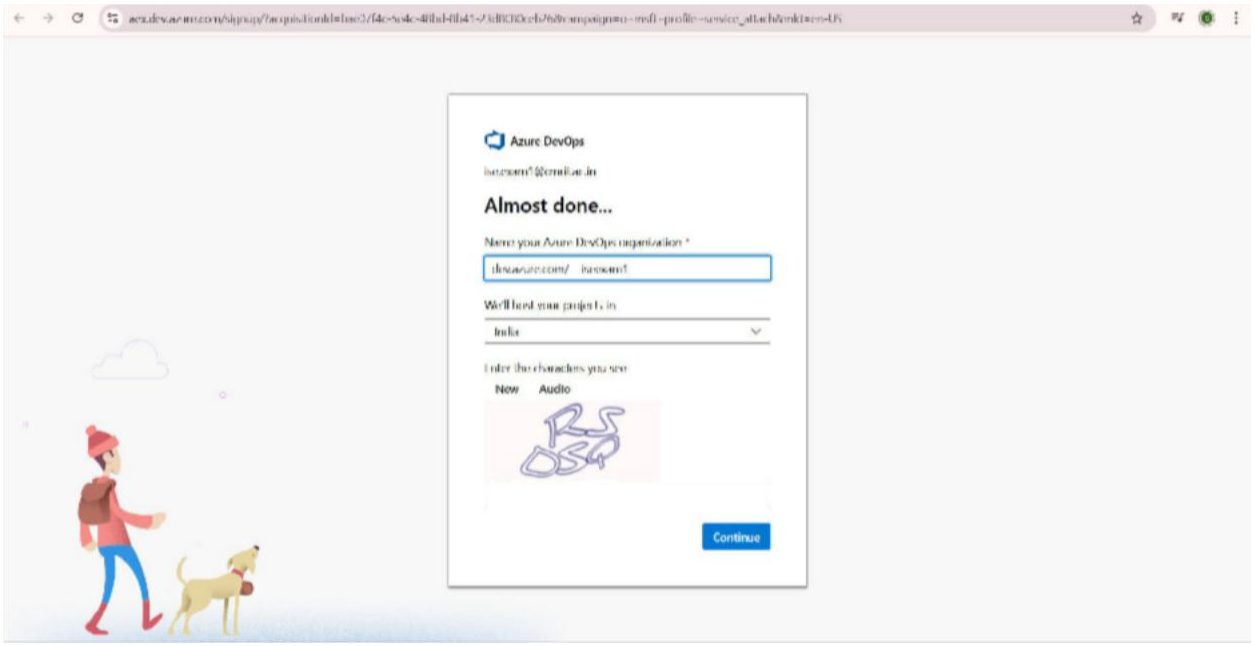


After above selection it once again reverifies name and email just click Continue After it you get a Screen

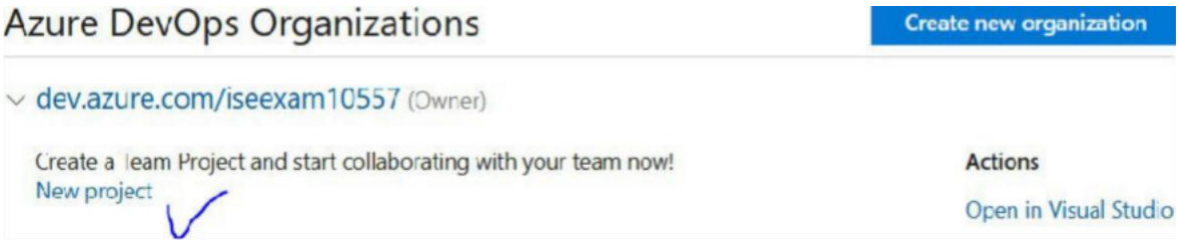
Get started with Azure DevOps

Plan better, code together, ship faster with Azure DevOps

Create new organization



YouwillbeabletoseeOrganizationisCreated



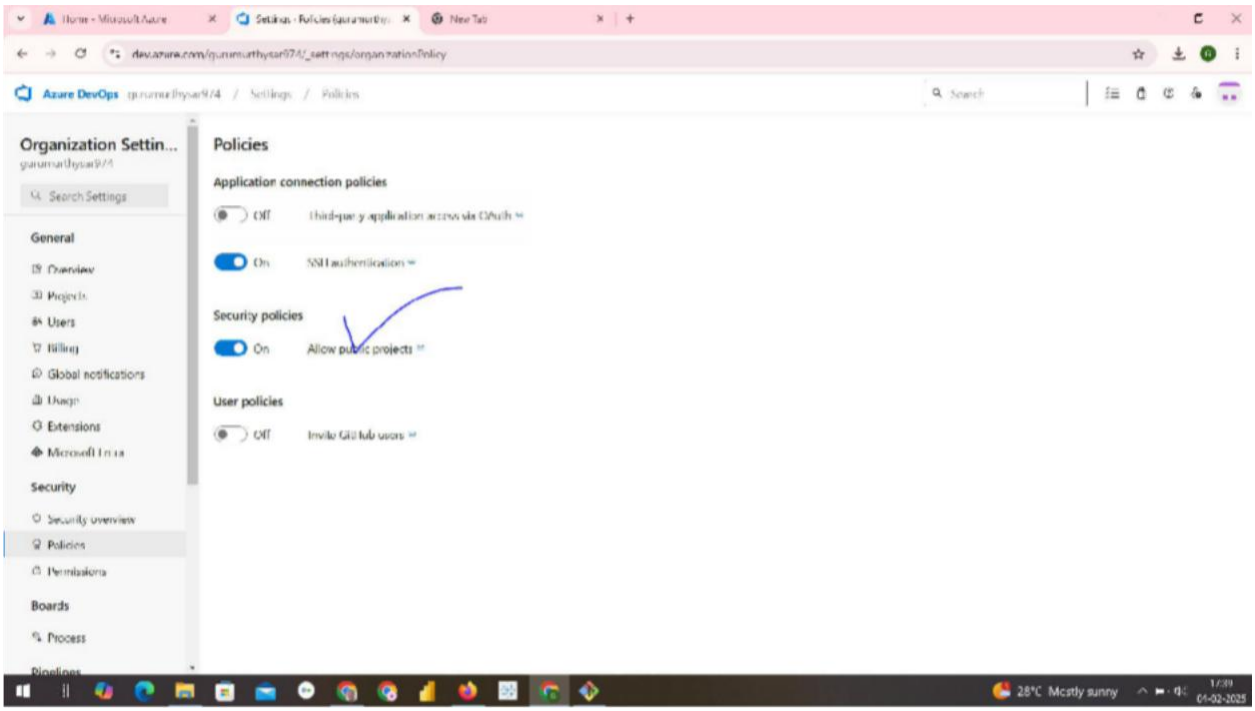
Finally After Creating a New Organization

U can create Project of your choice as per requirement

Everytime you need not to sign in u can book mark or add the below link as shortcut https://aex.dev.azure.com/ https://portal.azure.com/#home

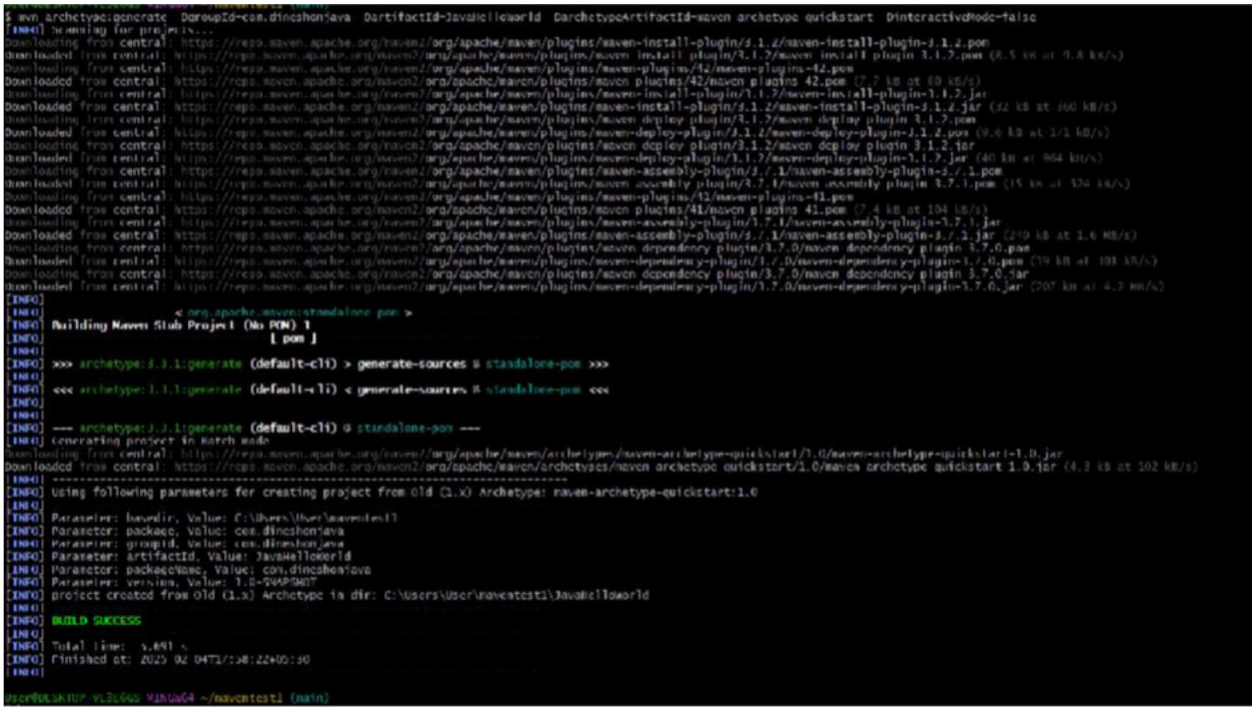
PROGRAM10: Creating Build Pipelines: Building a Maven/Gradle Project with Azure Pipelines, Integrating Code Repositories (e.g.,GitHub,AzureRepos), Running Unit Tests and Generating Reports.

STEP1: On creating organization goto Organization settings goto Policy And Allow Public Projects active



STEP2: GOTO GITBASH
TYPE COMMANDS AS IN BELOW
mkdirmaventest1 cd
maventest1

STEP3: To create simple hello world maven project type command as in below mvn archetype:generate -DgroupId=com.dineshonjava -DartifactId=Javateam
-DarchetypeArtifactId=maven-archetype-quickstart-DinteractiveMode=false



STEP3: To add files from local to github Follow the procedure

a) **First create a repositorying it hub as maven test azure**

b) **Then come to gitbash and type**

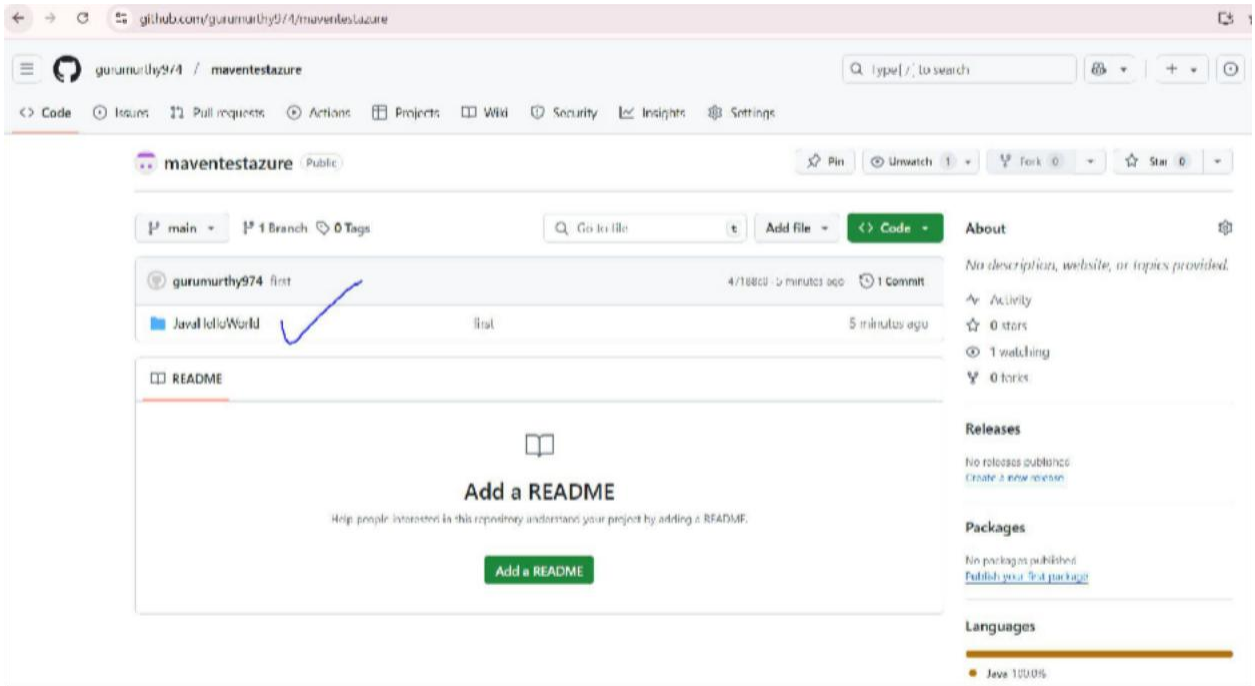
git init

gitadd .

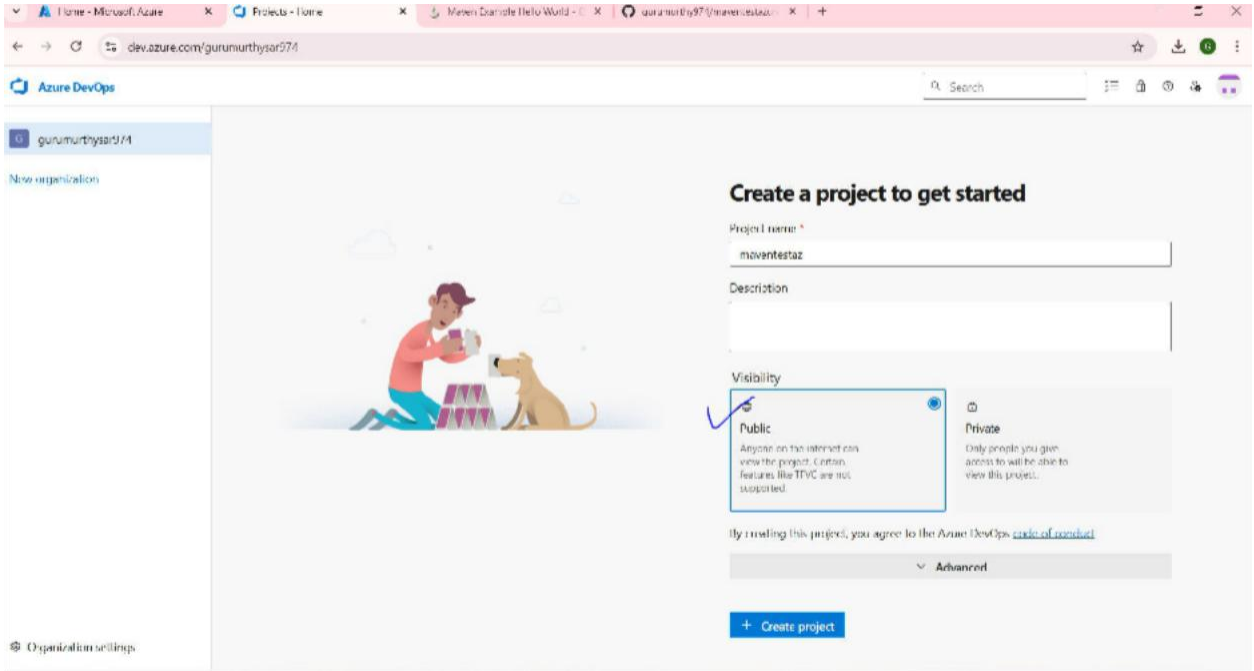
gitcommit-m“azurepipelineexample” git branch -M main

gitremoteaddoriginhttps://github.com/gurumurthy974/maventestazure.gitgit push -u origin main

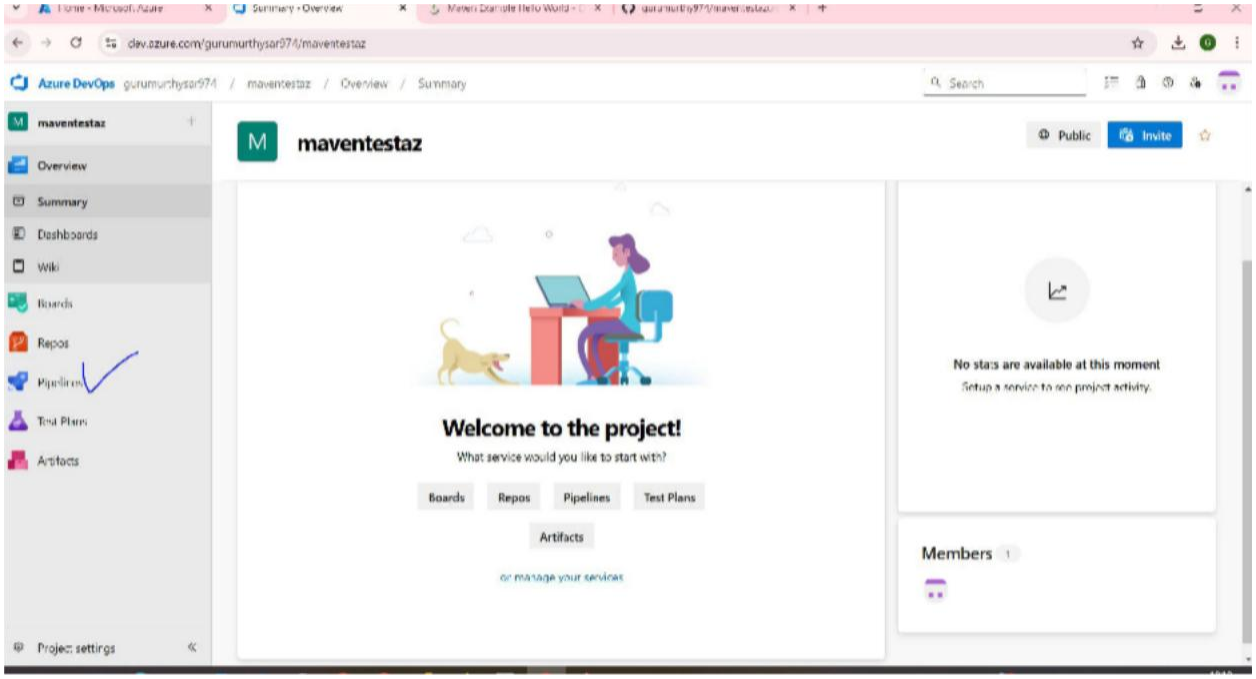
After completion of above command my repository looks



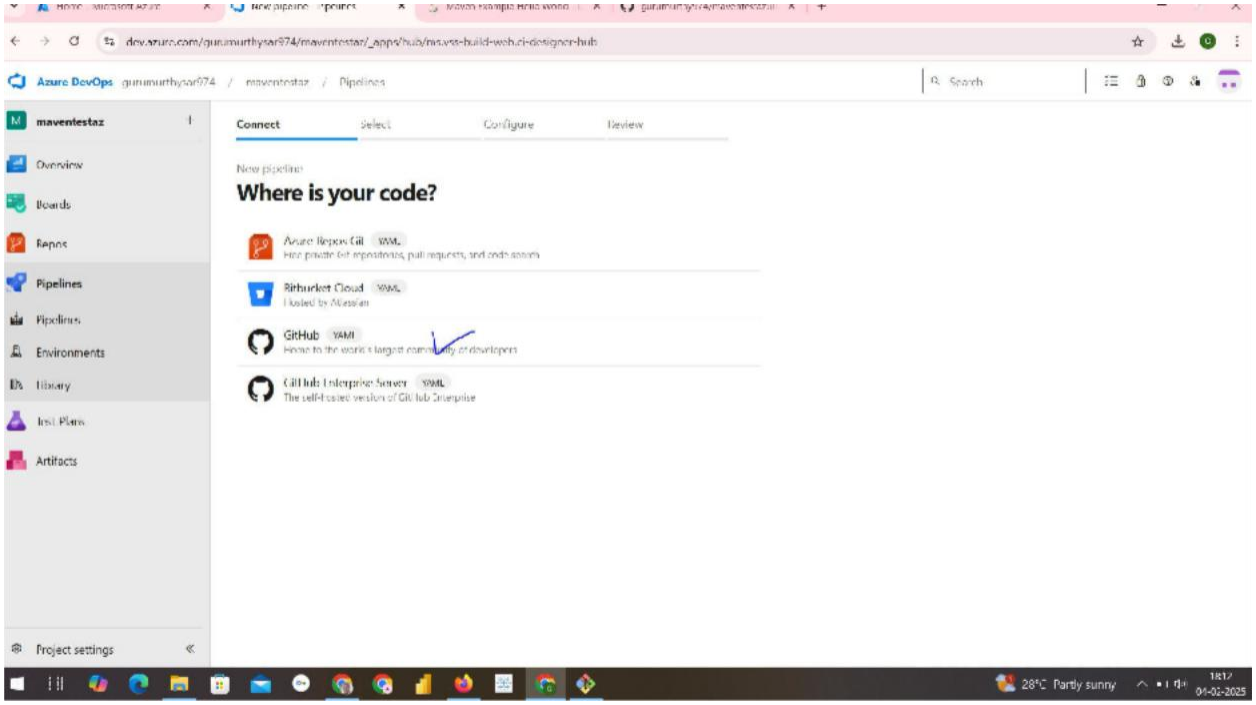
STEP4: Now goto Azure Devops Organization create Public Project



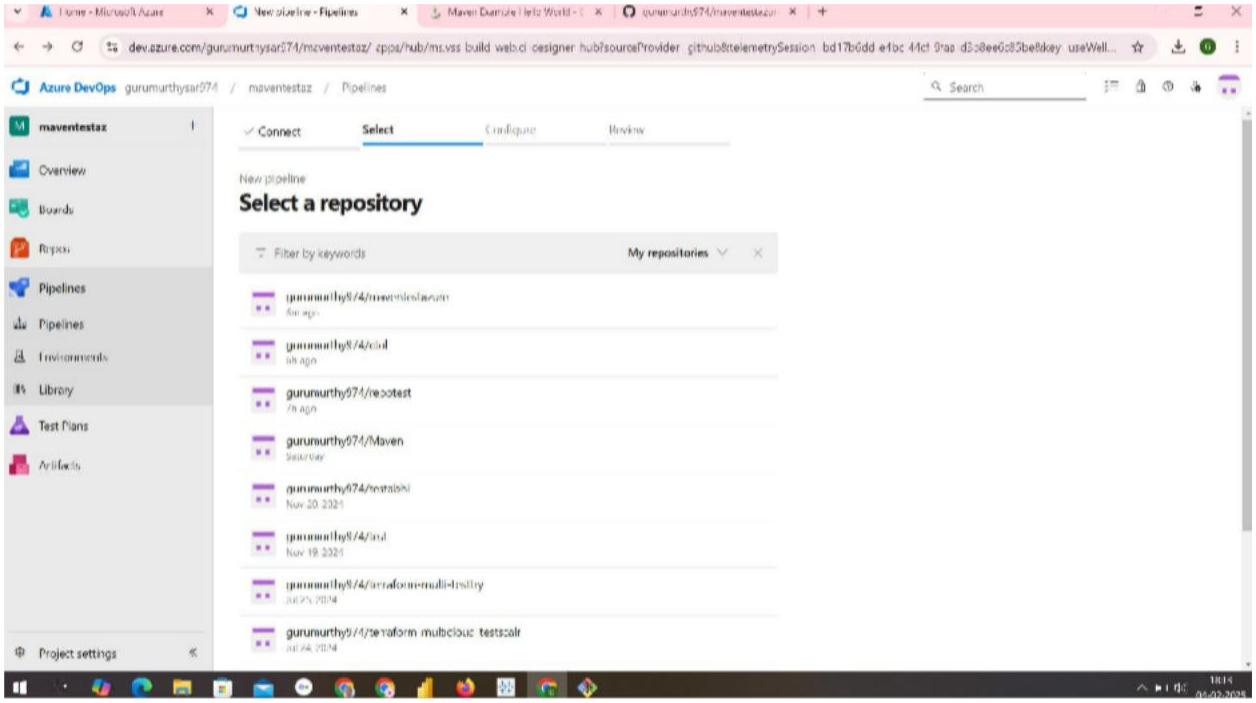
STEP5: SELECT PIPELINE and then click on create pipeline



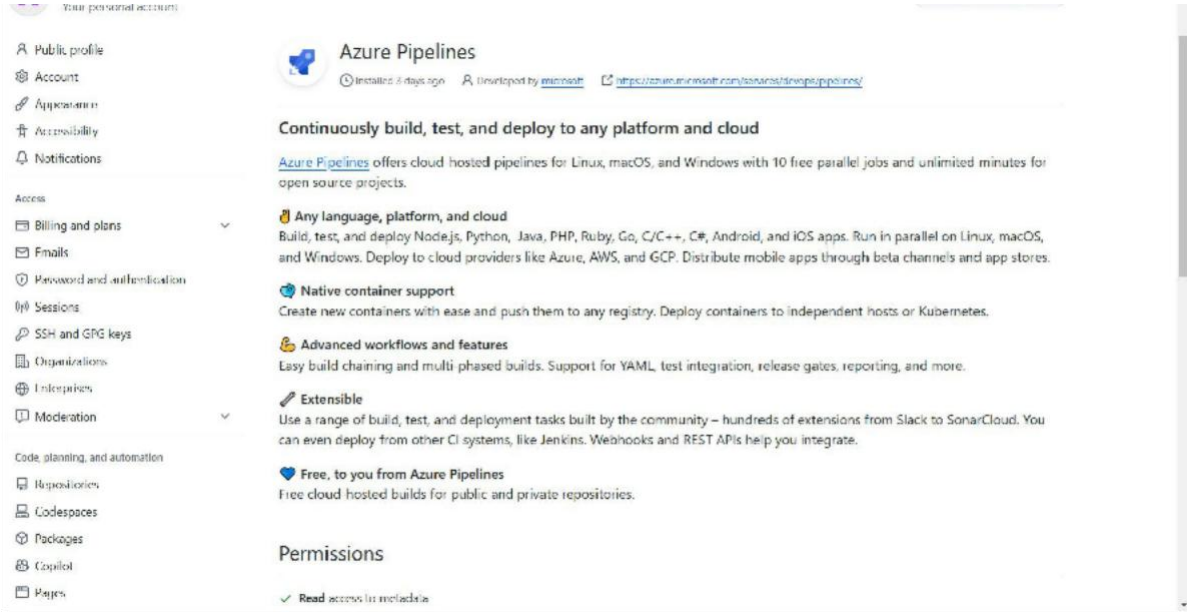
STEP6: After creating Pipeline select type of repoas Github



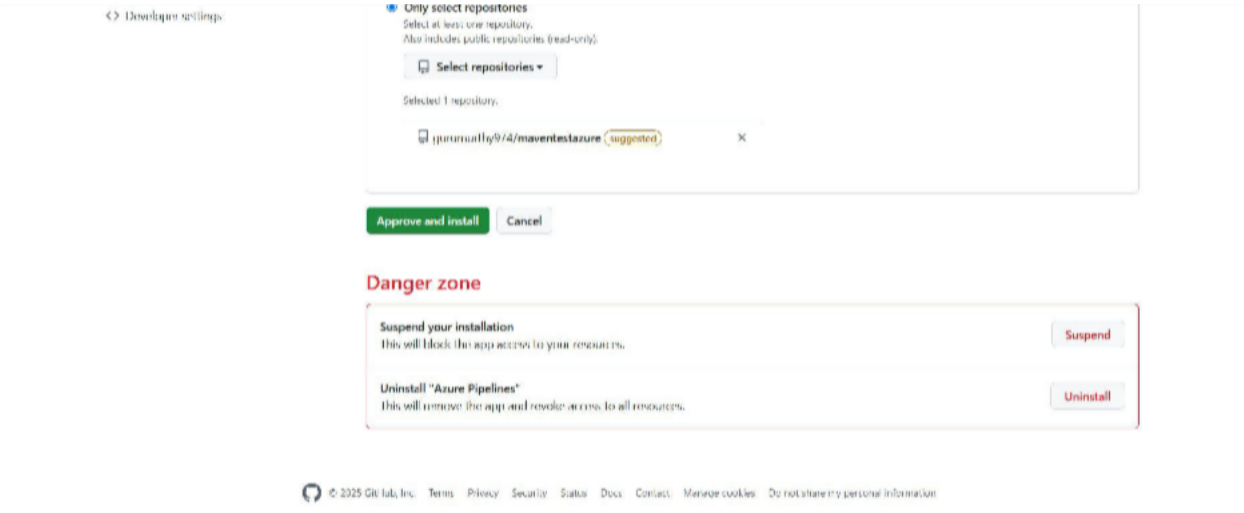
STEP7: It asks for minimum sign in verification after that your screen be as in below select required repository there to run maven project in my case its maventest123



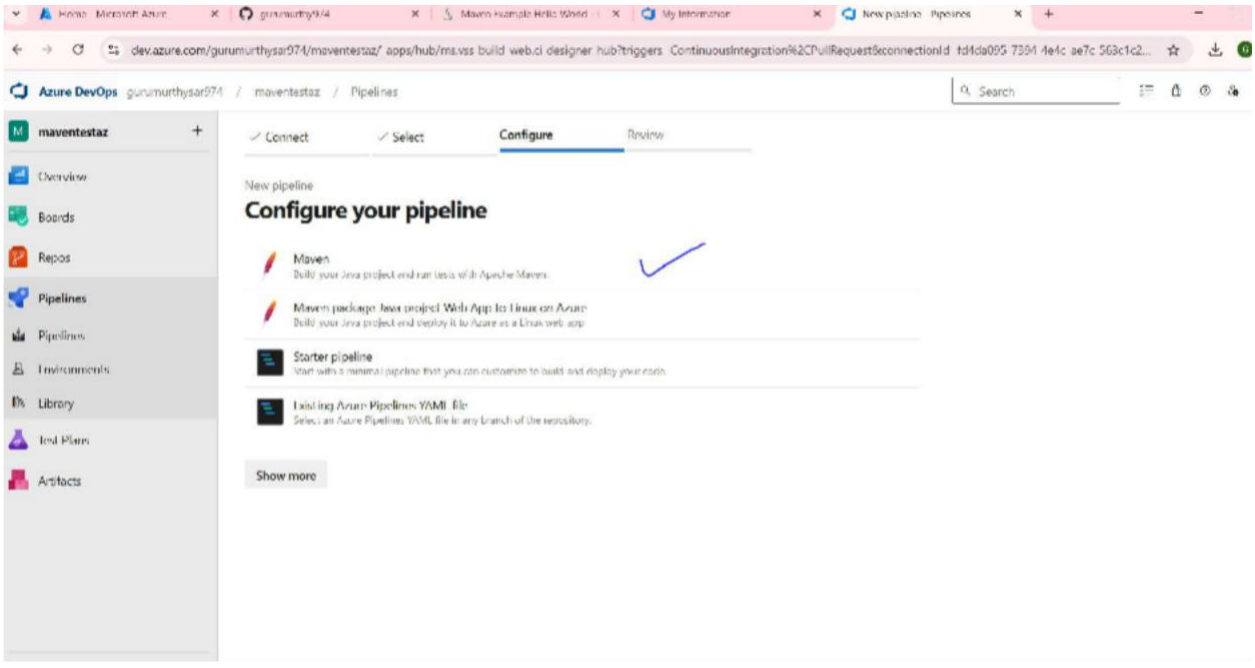
STEP8:AFTER REQUIREDREPO ISSELECTED thescreen beas in below



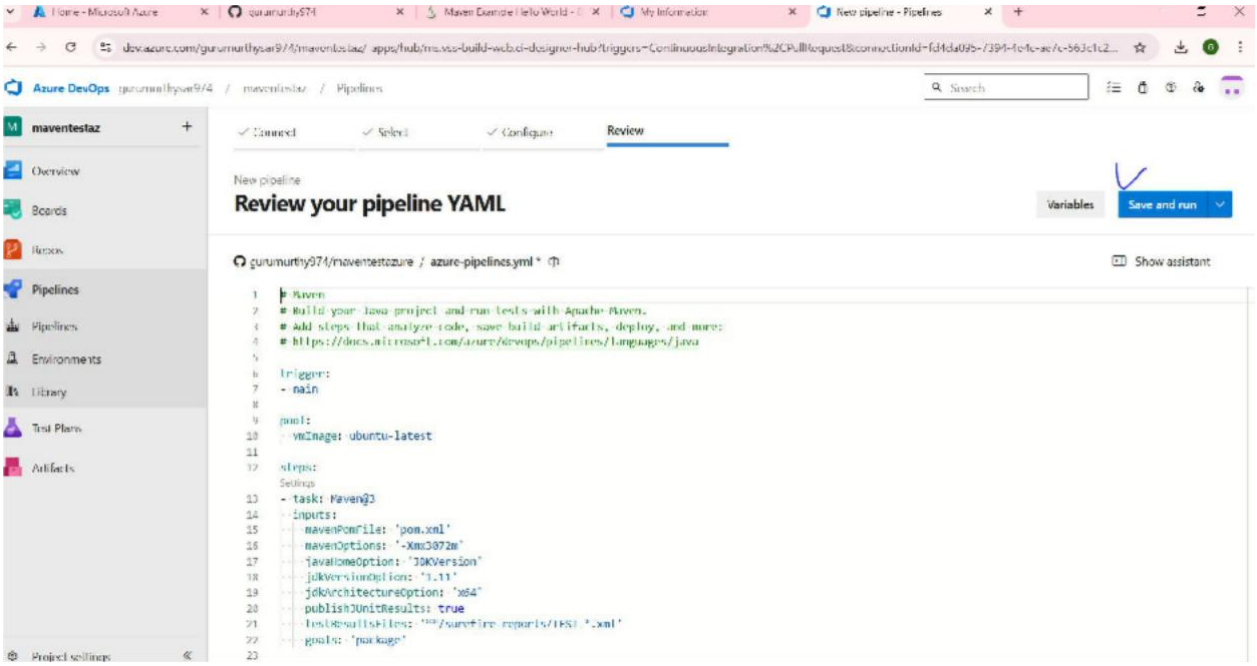
Drag the screen down check once again the selected repository is correct or not then click on Approve and Install



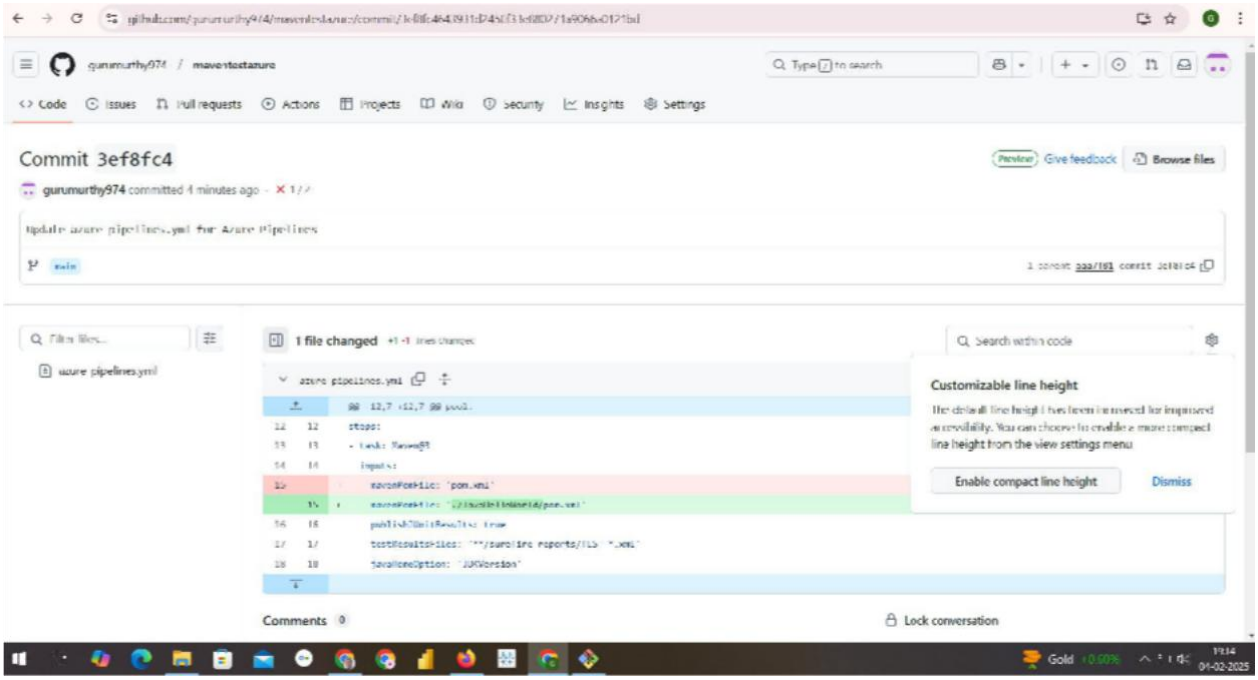
STEP9: It again verifies signin verification of Microsoft account You be able to see starter pipeline select for Maven



After selecting maven it asks for save and run just click on it

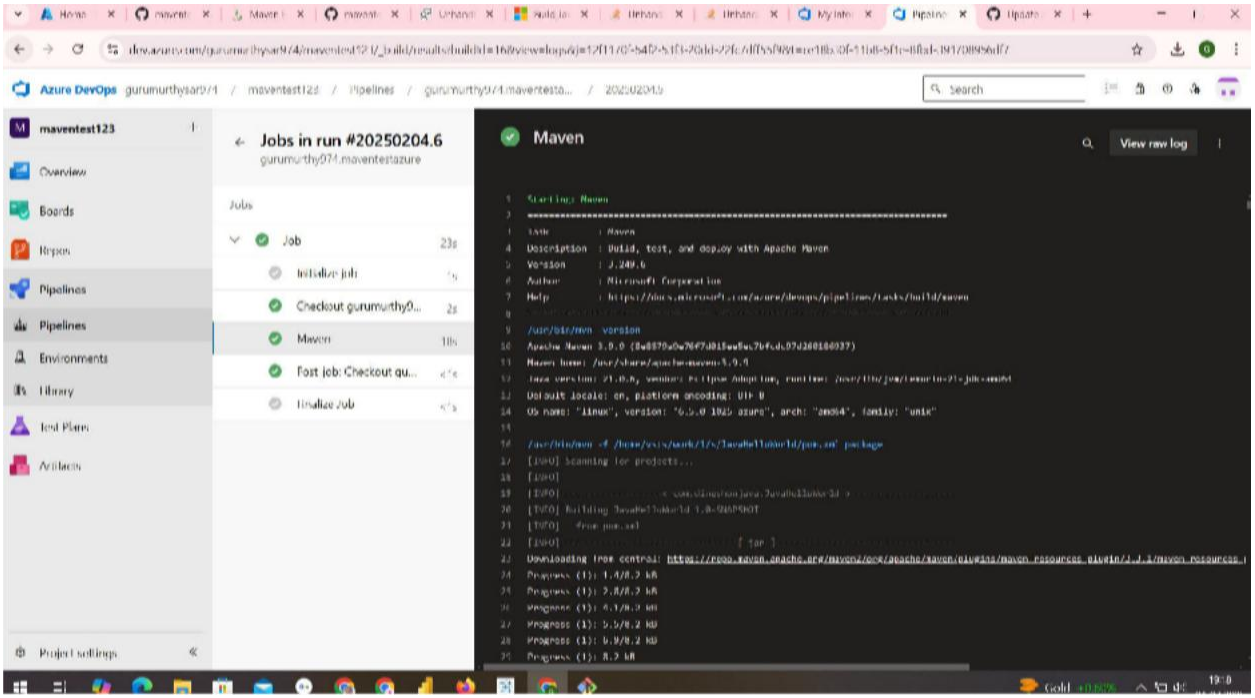


Finally You be able to see tasks running its failed be c .we should mention proper path For pom.xml



The commits will also be visible in github

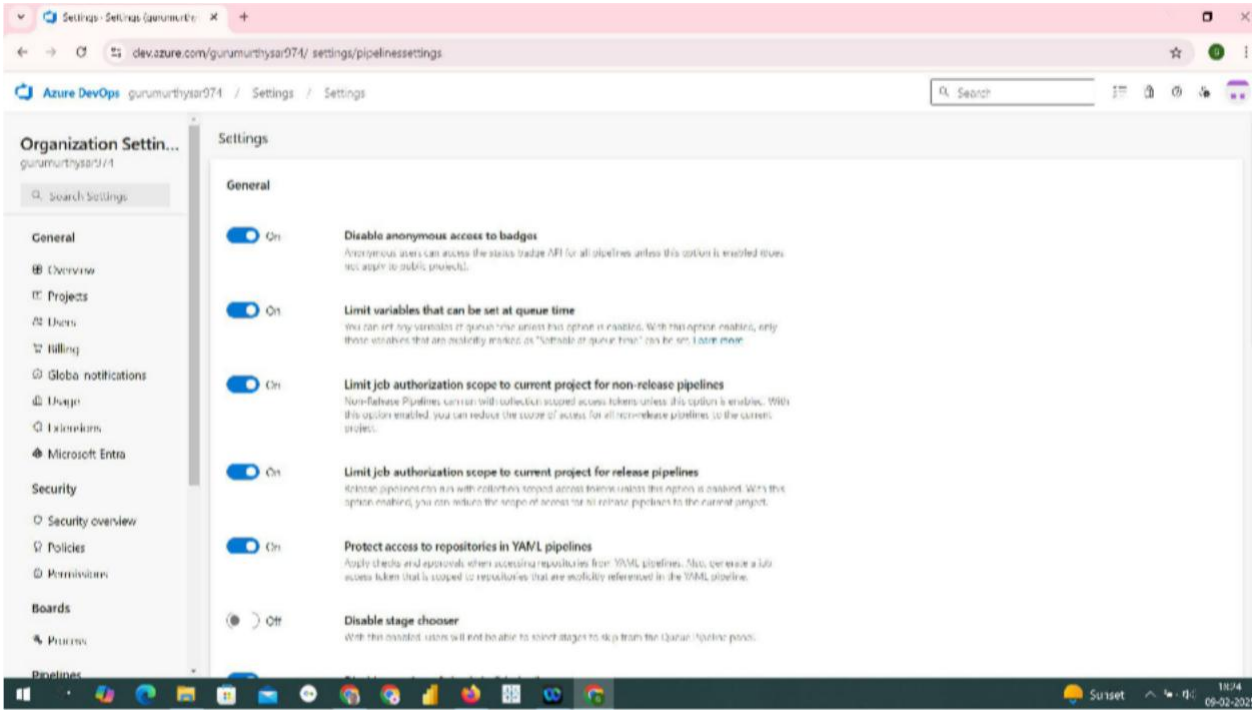
WE can download and also see individual Raw Log Reports



If pipeline permission errorcomes please complete the registration form of Parallel Jobs after 2 working days(48hrs) you be able to run pipelines for private projects same happens to be for public Projects.

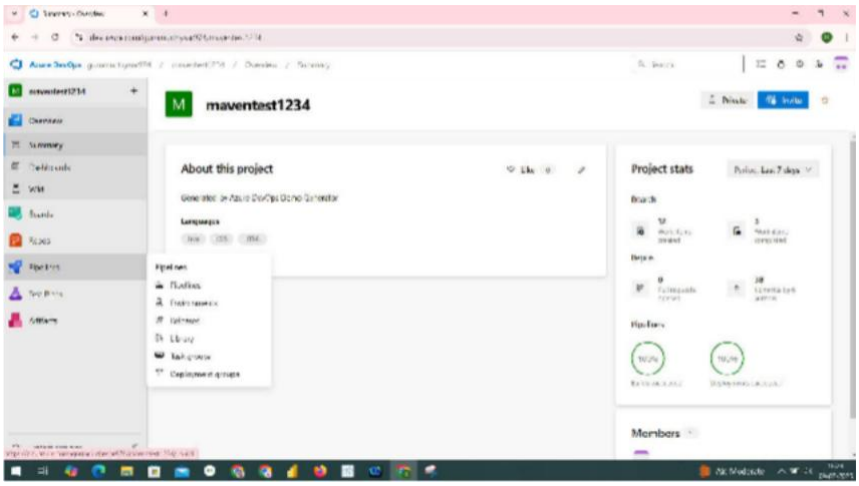
Program11: Creating Release Pipelines: Deploying Applications to Azure App Services, Managing Secrets and Configuration with Azure Key Vault, Hands-On: Continuous Deployment with Azure Pipelines.

STEP1: Click on Organization setting and click on Pipeline Settings You get screen as in below

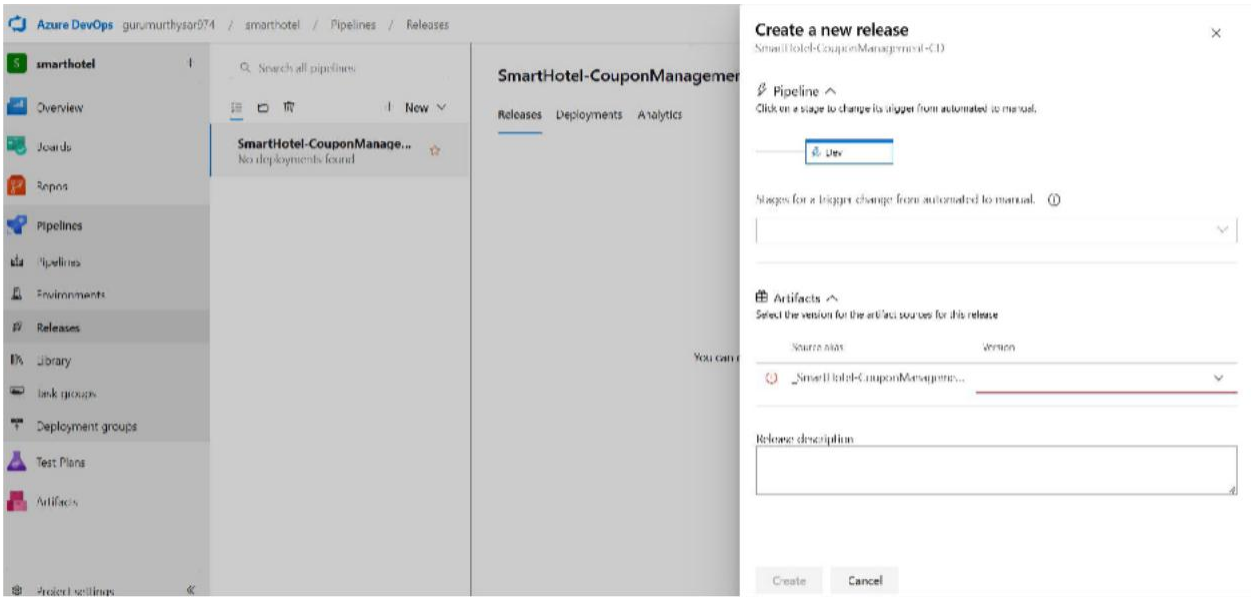


STEP2: Off the Disable creation of classic pipeline

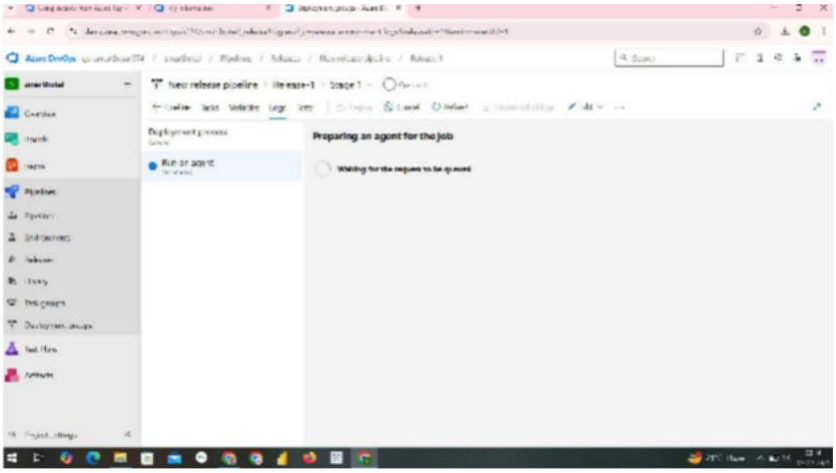
STEP3: Now you be able to see the visibility of Release for any pipeline creation as in screen below.

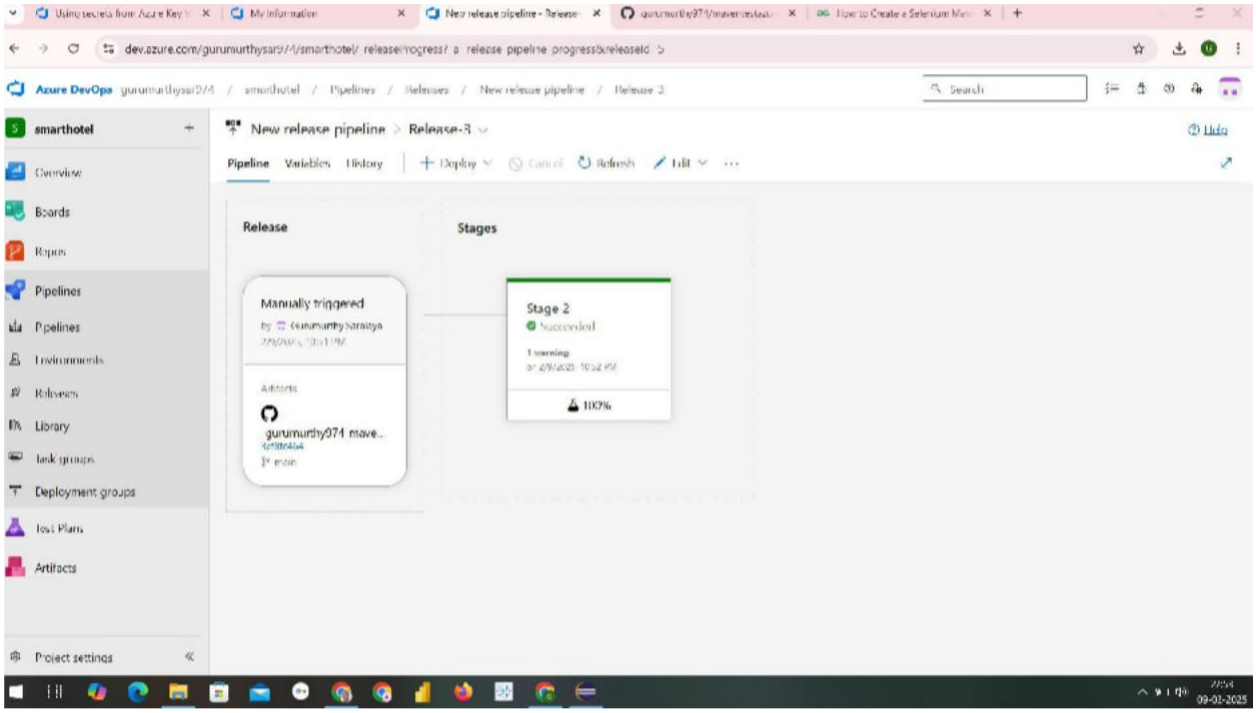


STEP4: You can run simple test plans

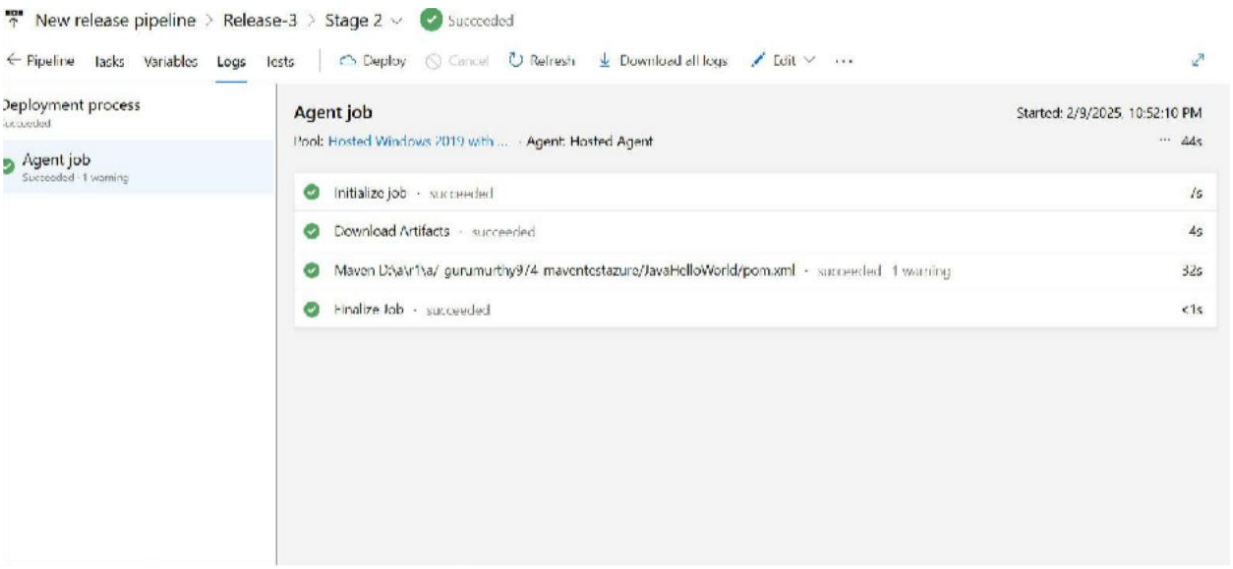


We can build tasks and run them





We can see our Agent Job releases logs


New release pipeline > Release-3 > Stage 2
Succeeded
Pipeline
Tasks
Variables
Logs
Tests
Deploy
Cancel
Refresh
Download all logs
Edit
Deployment process
Agent job
Succeeded · 1 warning
Agent job
Started: 2/9/2025 10:52:10 PM
Pool: Hosted Windows 2019 with ... · Agent: Hosted Agent
44s
Initialize job · succeeded
/s
Download Artifacts · succeeded
4s
Maven D:\a\r1\a/ gurumurthy974 maventeststazure/JavaHelloWorld/pom.xml · succeeded · 1 warning
32s
Finalize Job · succeeded
<1s

Program 12: Practical Exercise and Wrap-up for Build and Deploy a Complete DevOps Pipeline.

Title: Build and Deploy a Complete DevOps Pipeline

This program exercise helps you apply all major DevOps tools in one integrated workflow.

Objective:

To build and deploy a complete CI/CD pipeline using tools like Git, Jenkins, Docker, and Kubernetes.

Tools Required:

Git (for version control)

Jenkins (for CI/CD automation)

Docker (for containerization)

Kubernetes or Docker Compose (for container orchestration)

GitHub (as the remote repository)

A sample web app (Node.js, Python Flask, or any simple web app)

Steps to Implement the Pipeline:

1. Source Code Management (Git & GitHub):

Create a GitHub repository and push a simple web app (e.g., Python Flask with app.py and requirements.txt).

Example repo structure:

- app.py
- requirements.txt
- Dockerfile
- Jenkinsfile

2. Create Docker Image:

Write a Dockerfile:

```
FROM python:3.9
WORKDIR /app
COPY . /app
RUN pip install -r requirements.txt
CMD ["python", "app.py"]
```

3. Jenkins Setup for CI/CD:

Create a Jenkins Pipeline (Jenkinsfile):

```
pipeline {
    agent any
    stages {
        stage('Clone') {
            steps {
                git 'https://github.com/your-repo/devops-app'
            }
        }
        stage('Build Docker Image') {
            steps {
                script {
                    docker.build('myapp:latest')
                }
            }
        }
        stage('Run Docker Container') {
            steps {
                script {
                    docker.image('myapp:latest').run('-p 5000:5000')
                }
            }
```

```
        }
    }
}
```

4. Deployment:

Option 1: Deploy to local Docker or remote server

Option 2 (Optional): Use Kubernetes with deployment.yaml and service.yaml to deploy the Docker image

Output/Result:

A fully automated pipeline that:

1. Fetches code from GitHub

2. Builds the Docker image

3. Runs the app inside a container

4. Optionally deploys it to a cloud or Kubernetes cluster

Wrap-Up Points:

You've learned how to:

Set up CI/CD with Jenkins

Use Git/GitHub for version control

Create Docker containers

Automate deployment