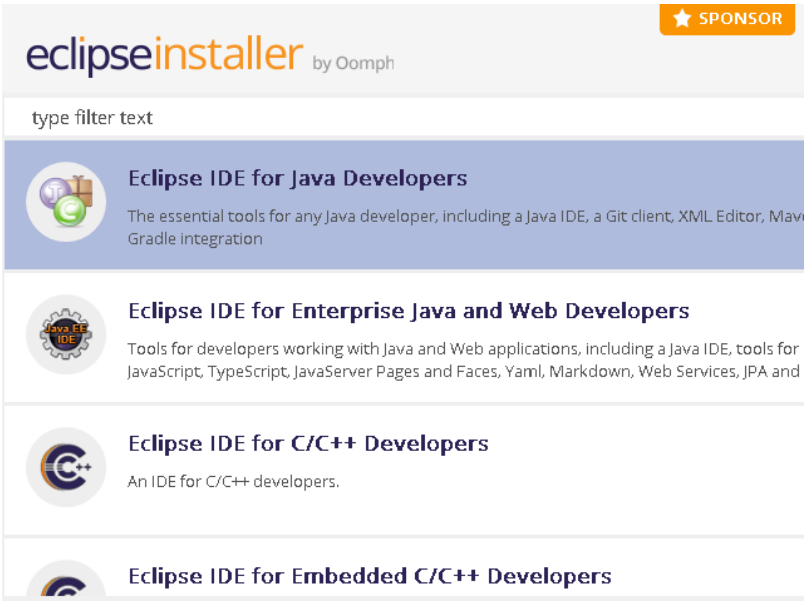
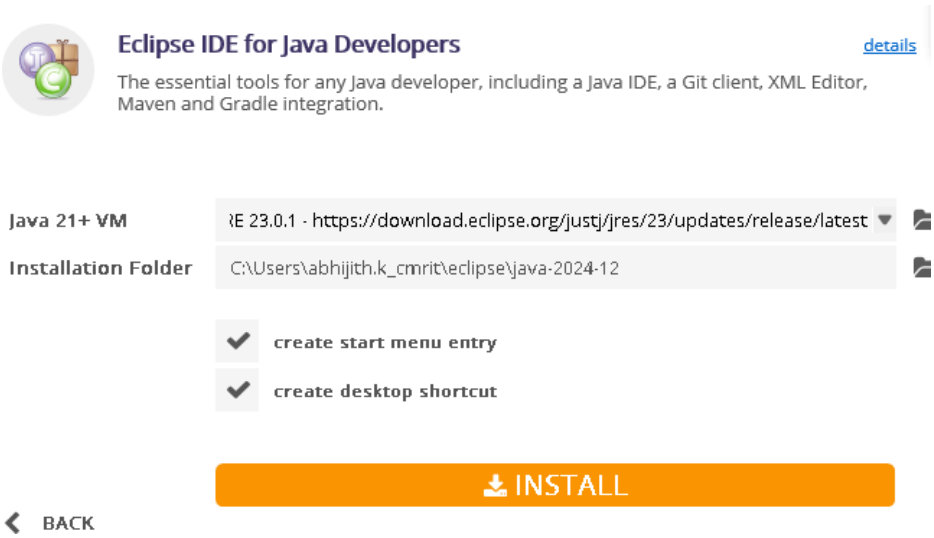
PROGRAM-1
Introduction to Maven and Gradle: Overview of Build Automation Tools, Key Differences Between Maven and Gradle, Installation and Setup

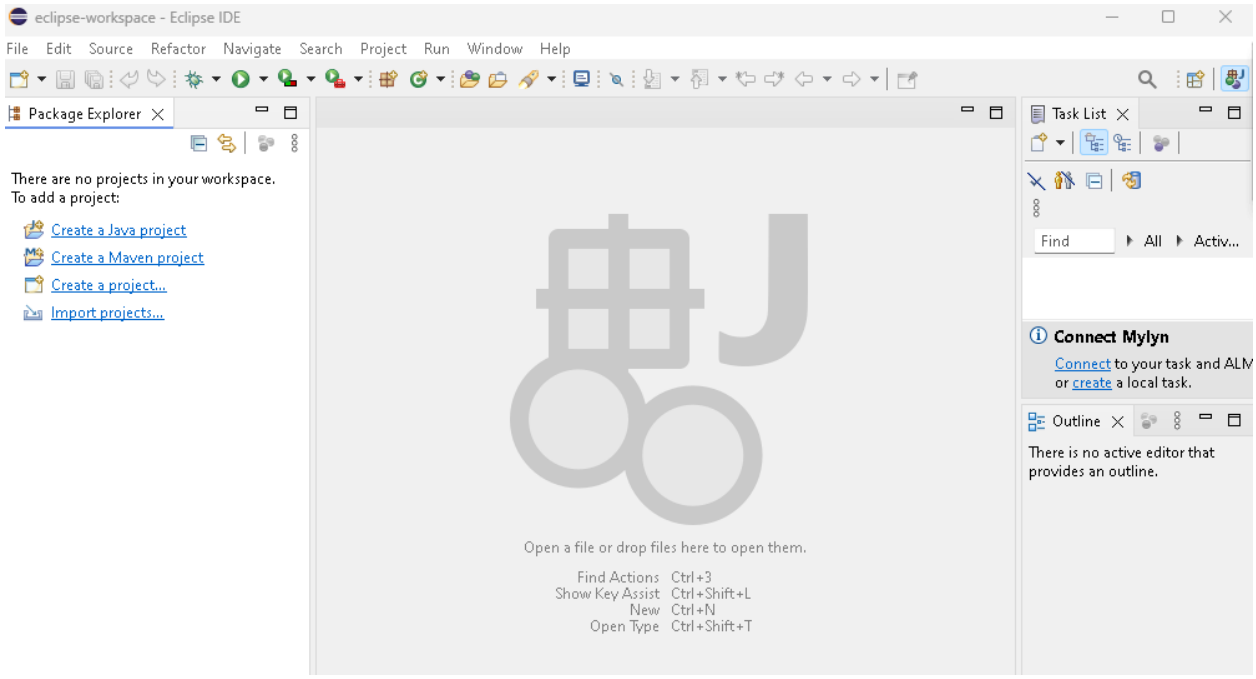
STEP1:Install Eclipse using this link
https://www.eclipse.org/downloads/
ENTERPRISE JAVA AND WEB DEVELOPERS



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

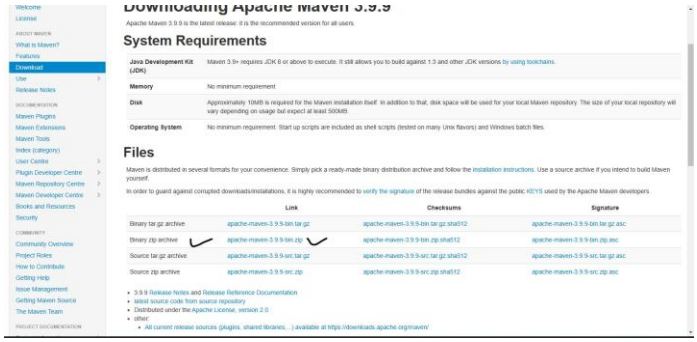


STEP5:Lets see Procedure to install MAVEN & GRADLE

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) Setup a PATH in environmental settings

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

e) After environment variable is set

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"
```

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

1. Create a new directory C:\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-8.12.1 to your newly created C:\Gradle folder.

Alternatively you can unpack the Gradle distribution ZIP into C:\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.**

```
------------------------------------------------------------

Gradle 8.12.1

------------------------------------------------------------

Build time:    2025-01-24 12:55:12 UTC
Revision:      0b1ee1ff81d1f4a26574ff4a362ac9180852b140

Kotlin:        2.0.21
Groovy:        3.0.22
Ant:           Apache Ant(TM) version 1.10.15 compiled on August 25 2024
Launcher JVM:  21.0.5 (Oracle Corporation 21.0.5+9-LTS-239)
Daemon JVM:    C:\Program Files\Java\jdk-21 (no JDK specified, using current Java home)
OS:            Windows 11 10.0 amd64
```

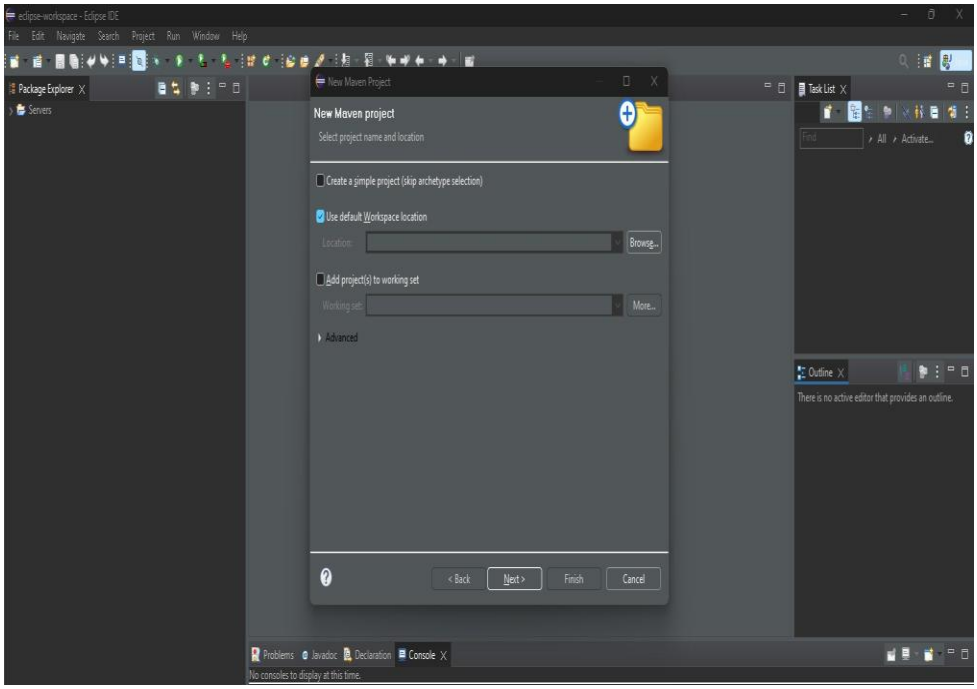
PROGRAM-2

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

STEP1:OPEN ECLIPSE THEN follow this navigation

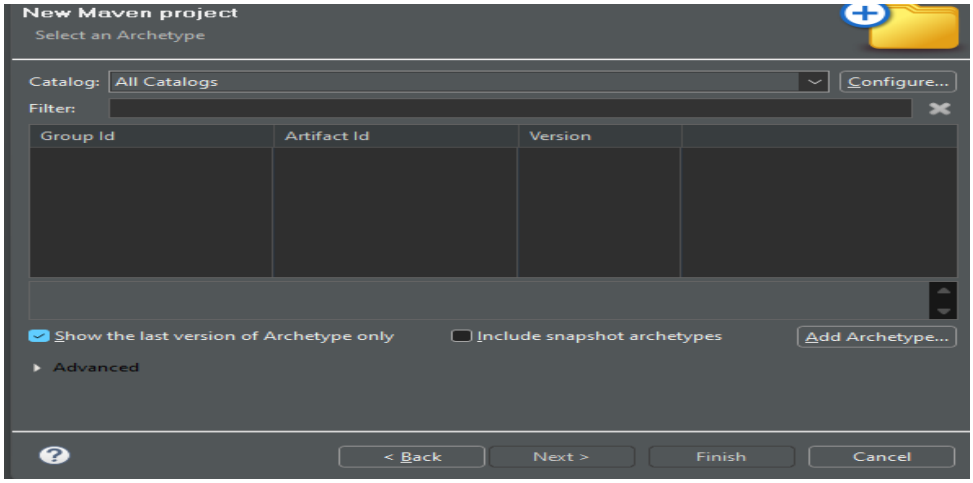
FIle —-----> New ---------->Maven Project

After that Screen be as in below

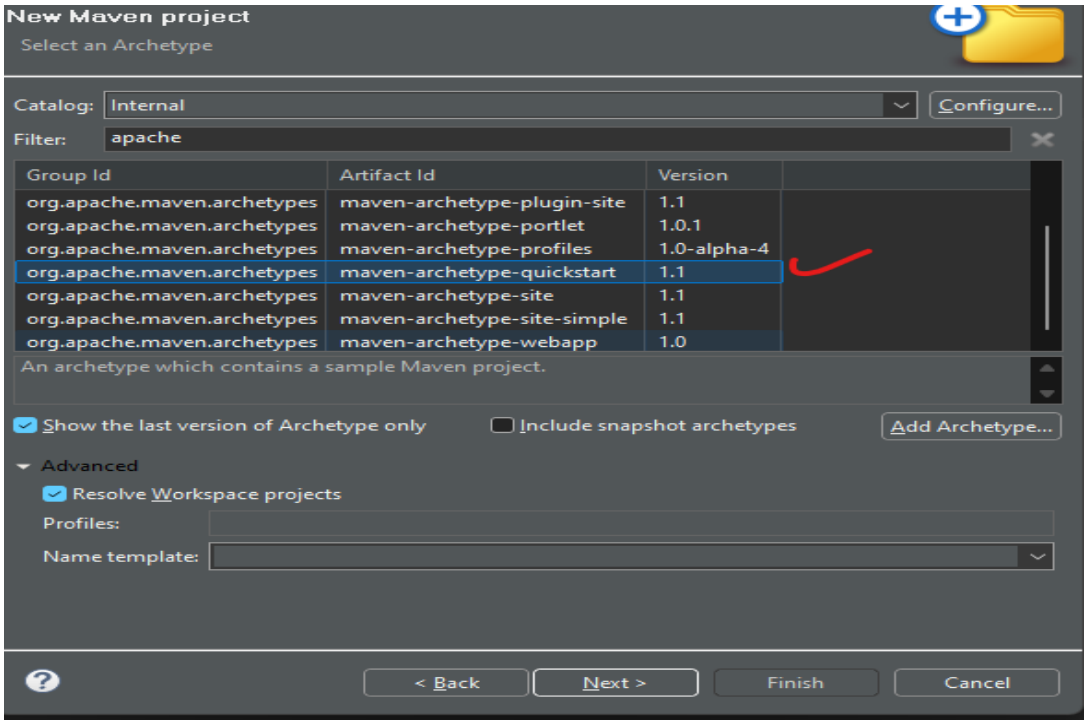


STEP2:Make sure Use default Workspace Location is selected, then click Next

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 a simple maven JAR based application. So, we will choose the “maven-archetype-quickstart” artifact to create the project.



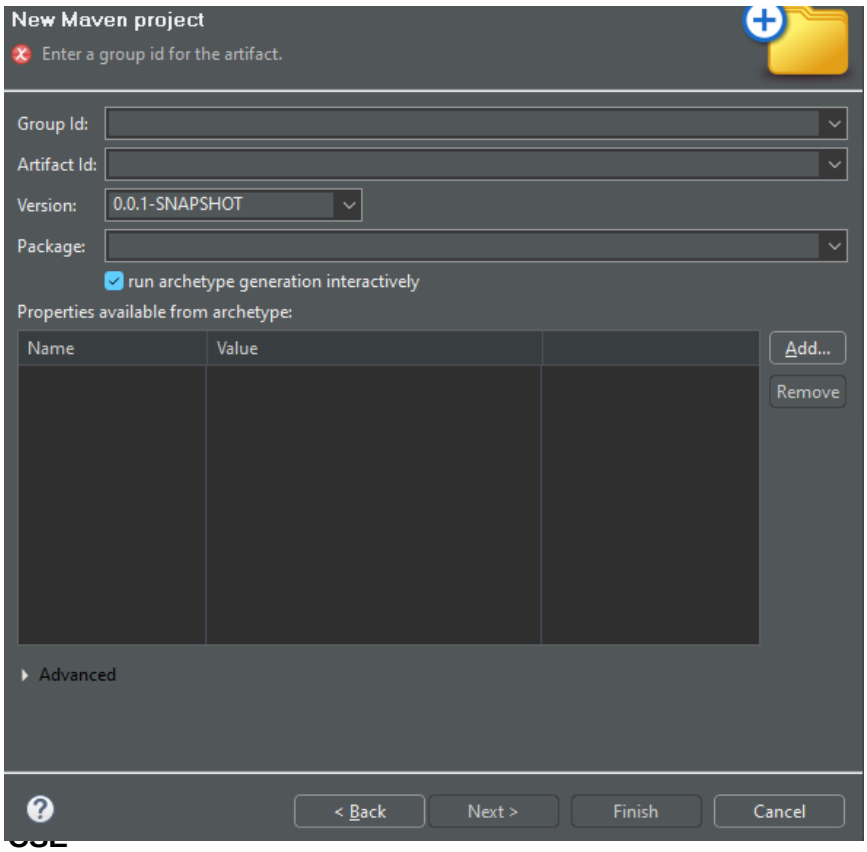
STEP4: Enter

Group Id:com.program2.maven

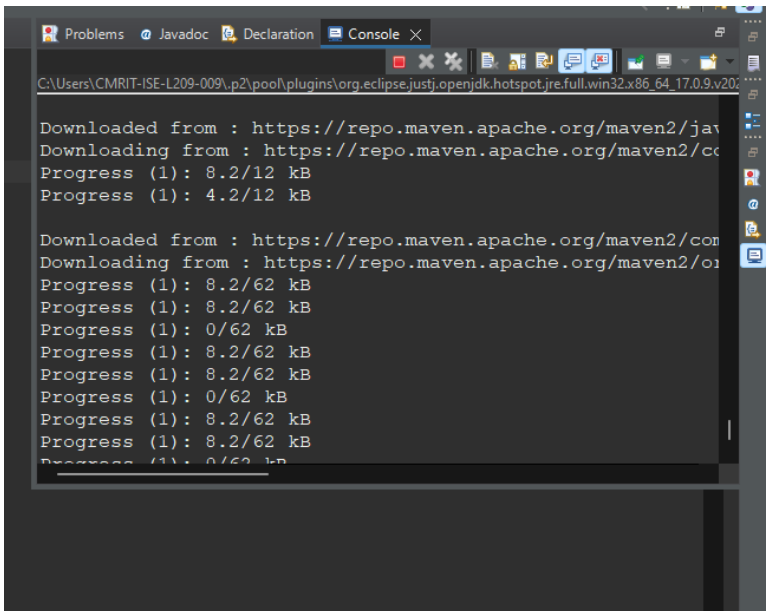
Artifact Id:program2-example-jar

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



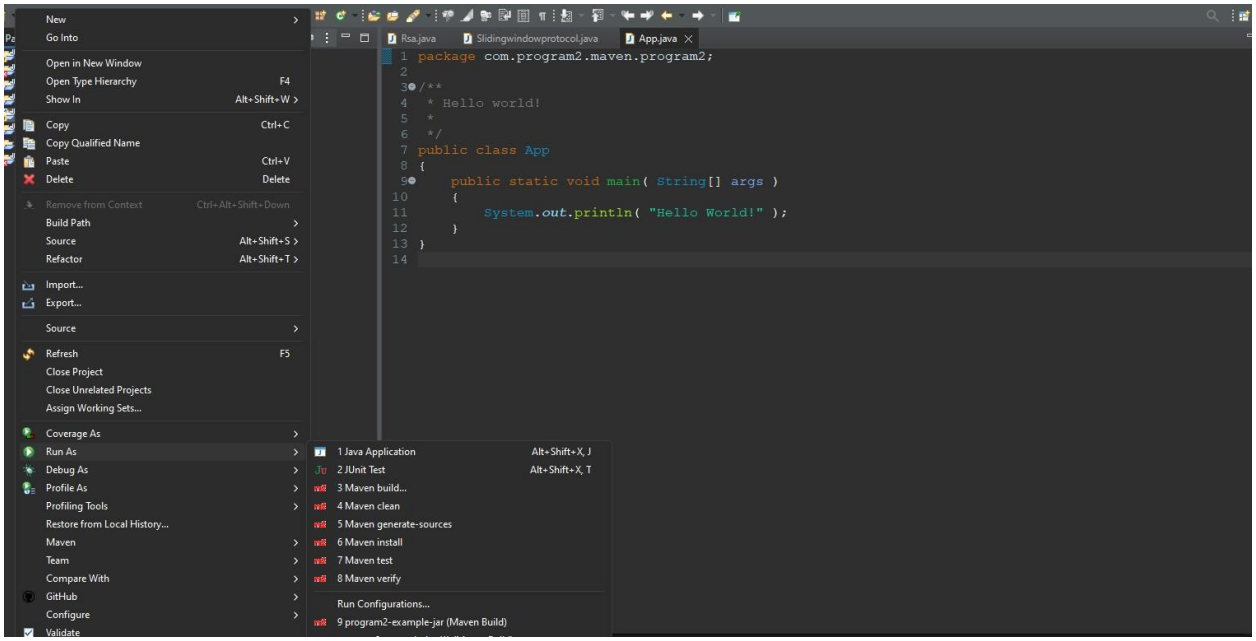
It asks for Configuration confirmation just click Y

```
Confirm properties configuration:
groupId: com.program2.maven
artifactId: program2-example-jar
version: 0.0.1-SNAPSHOT
package: com.program2.maven.program2
```

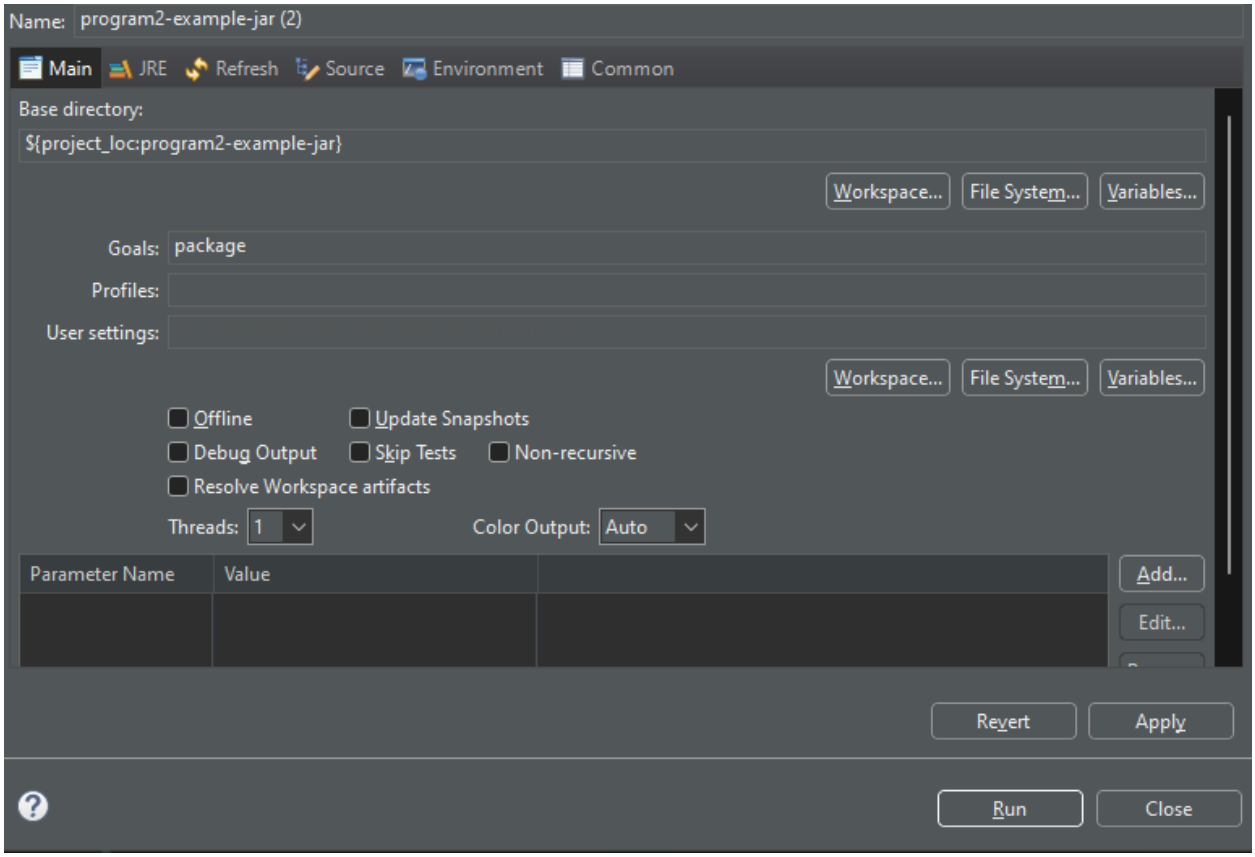
The RESULT be as in below

```
package: com.program2.maven.program2
 Y: y
[INFO] ----------------------------------------------------------------------------
[INFO] Using following parameters for creating project from Old (1.x) Archetype: maven-archetype-quickstart:1.1
[INFO] ----------------------------------------------------------------------------
[INFO] Parameter: basedir, Value: C:\Users\CMRIT-ISE-L209-009\Desktop\IS147
[INFO] Parameter: package, Value: com.program2.maven.program2
[INFO] Parameter: groupId, Value: com.program2.maven
[INFO] Parameter: artifactId, Value: program2-example-jar
[INFO] Parameter: packageName, Value: com.program2.maven.program2
[INFO] Parameter: version, Value: 0.0.1-SNAPSHOT
[INFO] project created from Old (1.x) Archetype in dir: C:\Users\CMRIT-ISE-L209-009\Desktop\IS147\program2-example-
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  03:37 min
[INFO] Finished at: 2025-01-29T10:31:45+05:30
[INFO] ------------------------------------------------------------------------
```

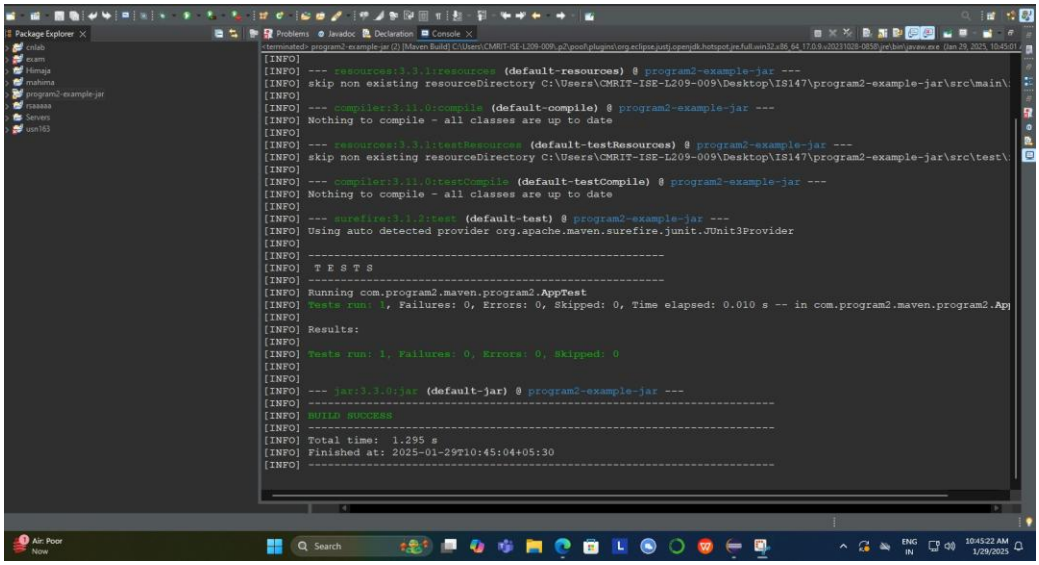
STEP5:Now its time to build Maven Project
Go to Maven Project ---------- >Right Click on the Project and select Maven Build



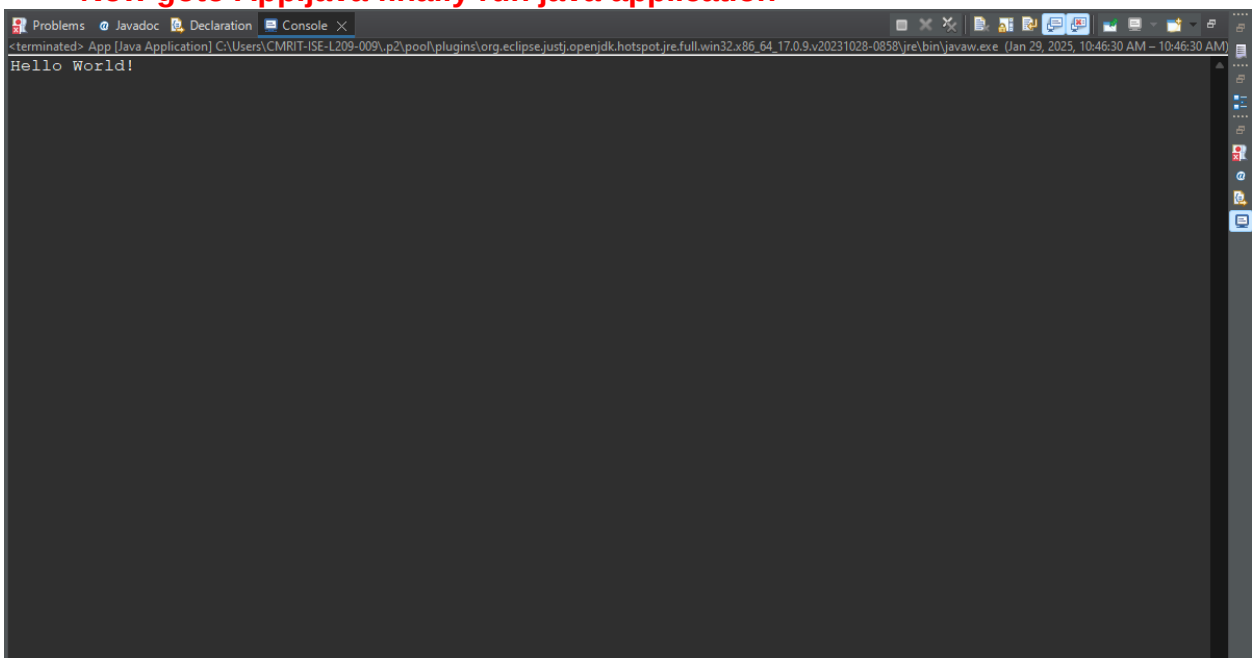
After the above procedure is done
Select Goal as package



And Click Run
The result be as in below



Now goto App.java finally run java application



Description

What is groupId in maven ?

groupId identifies a particular project uniquely across all projects, so we should follow a naming 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 groupId 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.java rewind.spring, com.javarewind.struts .. etc.

What is artifactId in maven ?

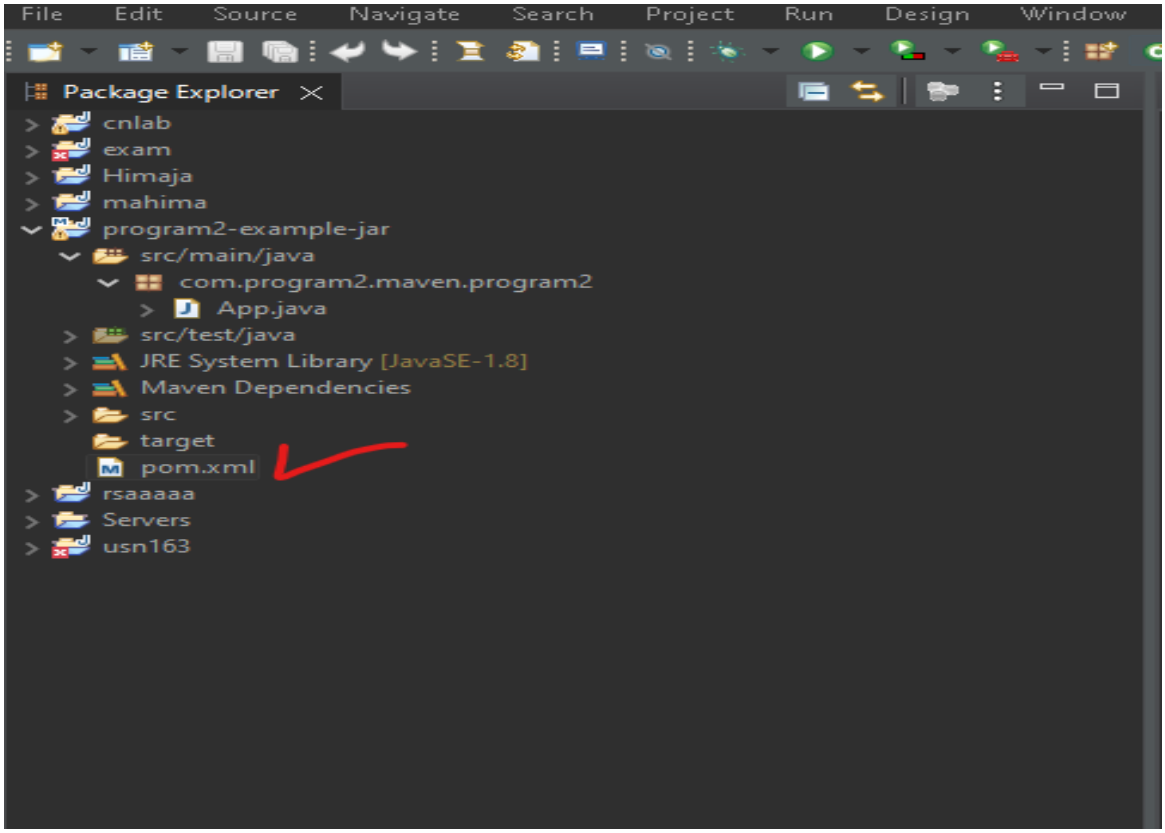
artifactId is the name of war file without version, if you are creating it by yourself you are free to took any name of your choice in lower case and without any strange symbol. But if this is a third party jar than we have to take the name of the jar as suggested by its distribution.

What is the archetype in maven ?

Archetype is a Maven project templating toolkit which tells the maven the type of project we are going to create. Archetype enables the maven 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 a new project from the provided archetype or updates the actual project if using a partial archetype. Maven provides a number of predefined archtypes, see more details from Maven Documentation.

HOW POM.XML LOOKS IS AS IN SCREEN BELOW




```
  http://maven.apache.org/xsd/maven-4.0.0.xsd (xsi:schemaLocation with catalog)
1 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instanc
2   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
3   <modelVersion>4.0.0</modelVersion>
4
5   <groupId>com.program2.maven</groupId>
6   <artifactId>program2-example-jar</artifactId>
7   <version>0.0.1-SNAPSHOT</version>
8   <packaging>jar</packaging>
9
10  <name>program2-example-jar</name>
11  <url>http://maven.apache.org</url>
12
13  <properties>
14    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
15  </properties>
16
17  <dependencies>
18    <dependency>
19      <groupId>junit</groupId>
20      <artifactId>junit</artifactId>
21      <version>3.8.1</version>
22      <scope>test</scope>
23    </dependency>
24  </dependencies>
25 </project>
26
```

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 cmd prompt
Goto Command Prompt

Then first make a new directory the command is
mkdir pgm3

For changing to a current directory the command is
cd pgm3

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

```
C:\Users\CMRIT-ISE-L209-009\gradletest>gradle init
Starting a Gradle Daemon (subsequent builds will be faster)

Select type of build to generate:
  1: Application
  2: Library
  3: Gradle plugin
  4: Basic (build structure only)
Enter selection (default: Application) [1..4] 1
```

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

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

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:
gradlew run

It will take atleast 3-5 minutes 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

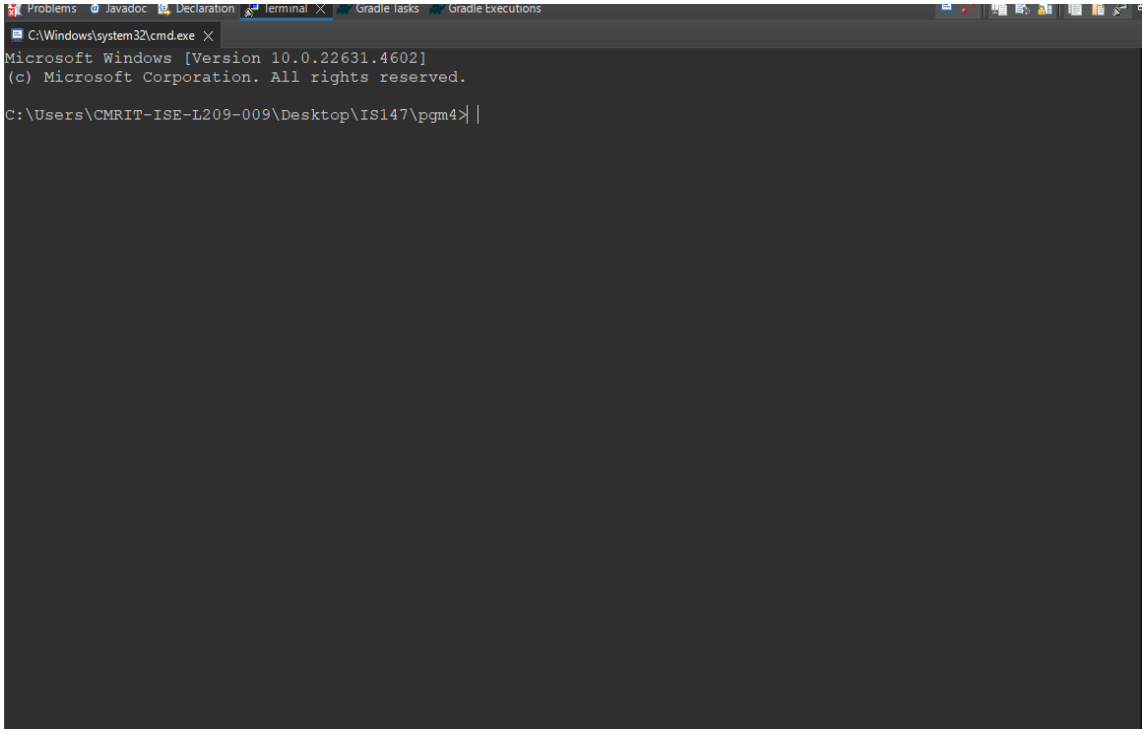
```
C:\Users\CMRIT-ISE-L209-009\gradletest>gradlew run
Calculating task graph as no cached configuration is available for tasks: run

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

```
C:\Users\CMRIT-ISE-L209-009\gradletest>tree
Folder PATH listing for volume Windows
Volume serial number is CA13-EB08
C:.
├---.gradle
|   ├---8.12.1
|   |   ├---checksums
|   |   ├---executionHistory
|   |   ├---expanded
|   |   ├---fileChanges
|   |   ├---fileHashes
|   |   └---vcsMetadata
|   ├---buildOutputCleanup
|   ├---configuration-cache
|   |   ├---87330068-729b-48ed-8a38-57771bbaae67
|   |   └---8qybe7c3ykh3sf9t2sllkie4w
|   └---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
```

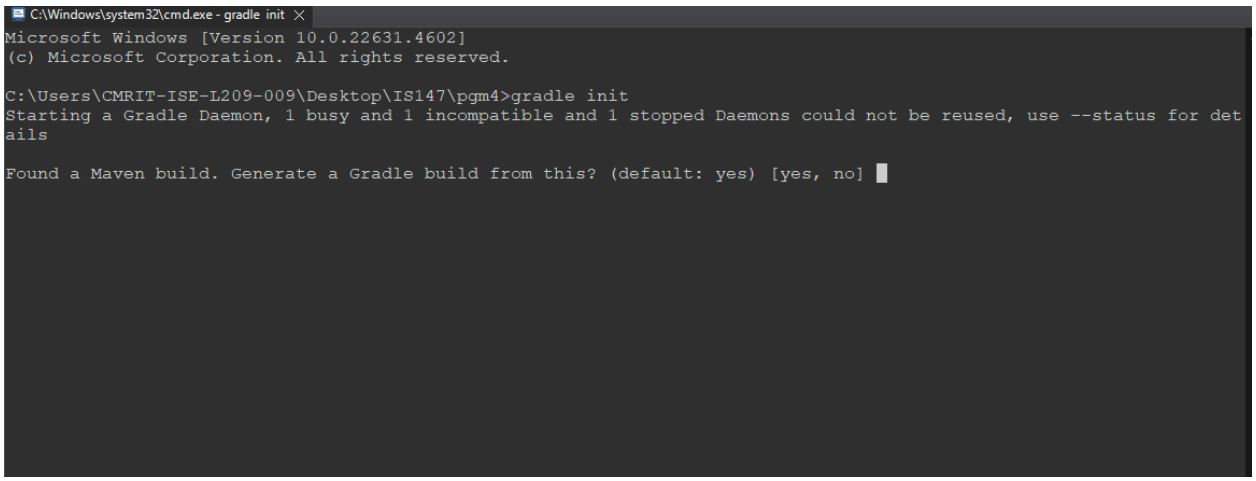
PROGRAM4:Practical Exercise: Build and Run a Java Application with Maven, Migrate the Same Application to Gradle

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

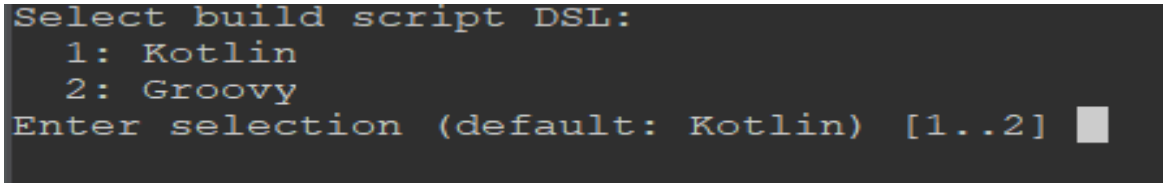
STEP2:Then to migrate to gradle use shortcut Key Ctrl+Alt+Shif+T To get Terminal screen as in below:



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

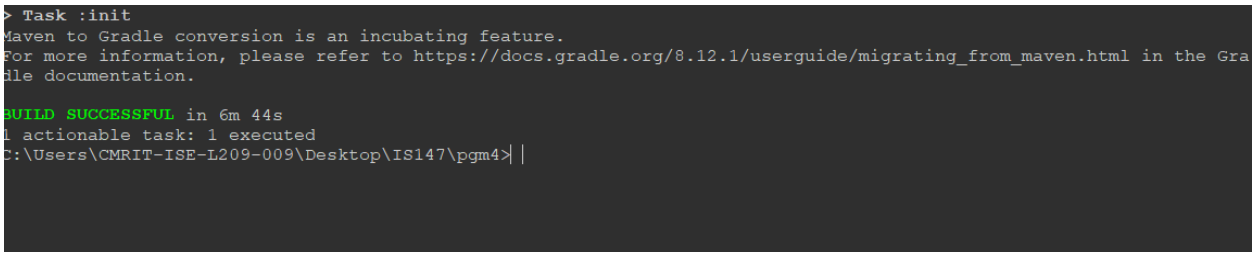


STEP4: After the above command is validated to yes it prompts to select Domain Specific Language as in screen below select 2 (as we have done for Kotlin)

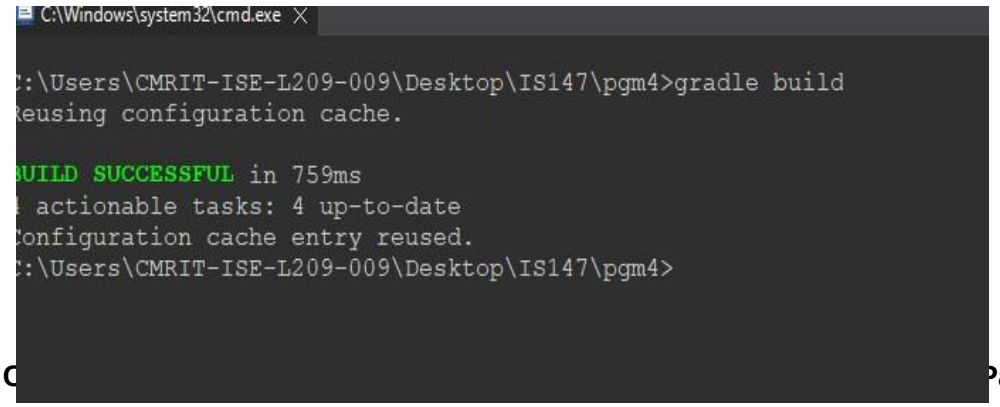


STEP5:After selecting Groovy it asks for validating prompt for API Generator just validate as yes

Finally it runs the init phase as been selected



STEP6:
Type the command
gradle build

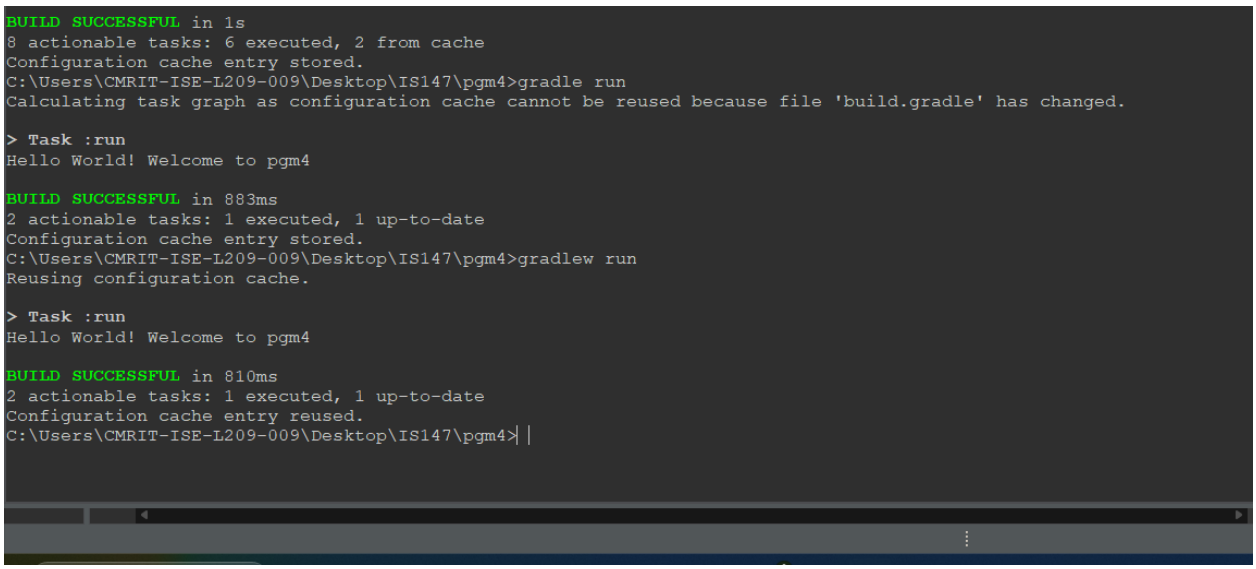


Now to get exact program output of our java file Locate to build gradle File from ur local repository and Copy paste the code as in below shown in red colorplugins { id("java-library") id("maven-publish") id("application")

```
}

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

repositories {
mavenCentral()
// Uncomment if you need to publish locally
// mavenLocal()
}

dependencies {
testImplementation("junit:junit:4.13.2") // Use Kotlin syntax for dependencies
}

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

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

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

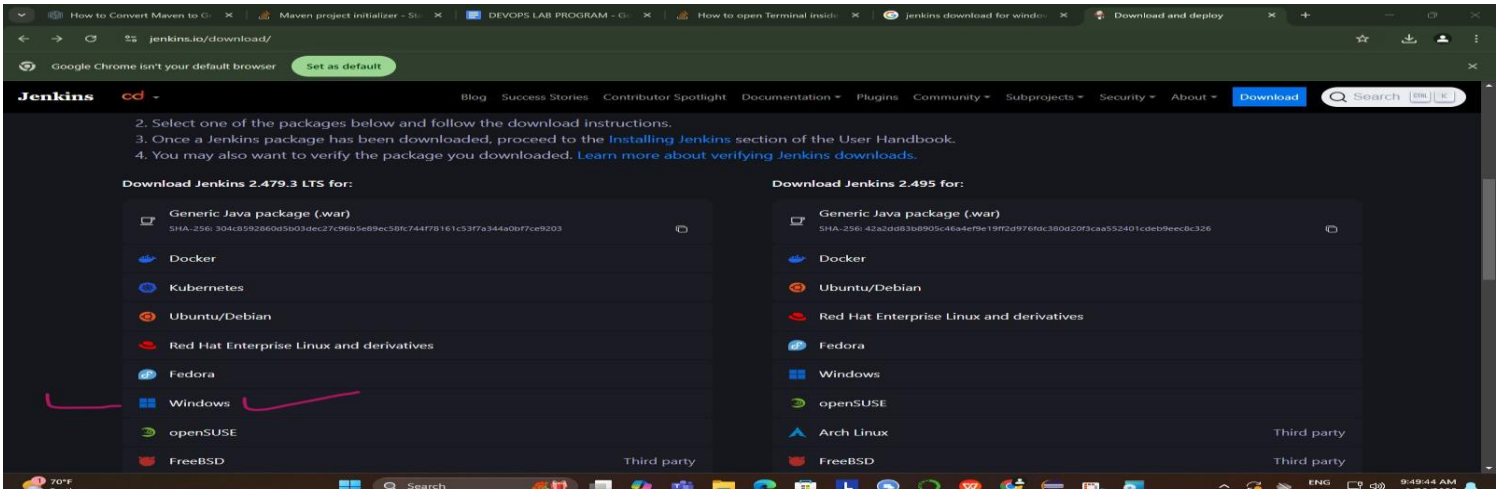
```
tasks.withType<Javadoc>().configureEa
    ch { options.encoding = "UTF-8"
    }
```

AFTER DOING ALL CHANGES FINAL STEP
To run commands gradle
clean build gradle run
You will get Output as
Hello World! Welcome to pgm4

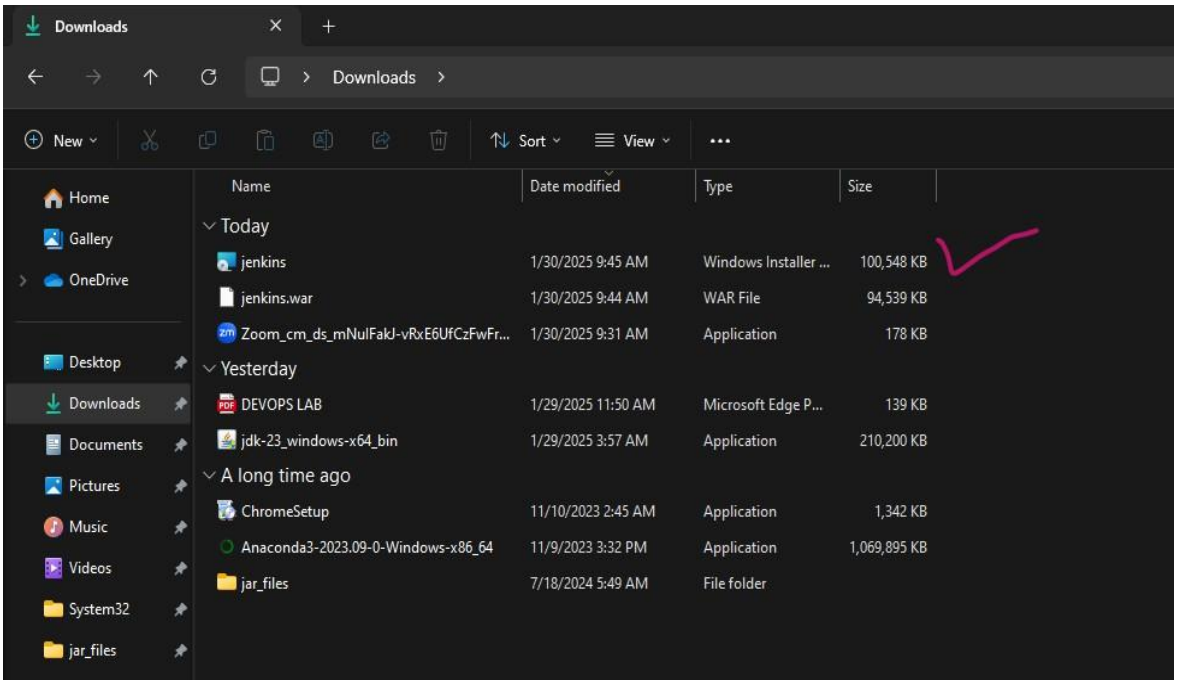


PROGRAM5:Introduction to Jenkins: What is Jenkins?, Installing Jenkins on Local or Cloud Environment, Configuring Jenkins for First Use .

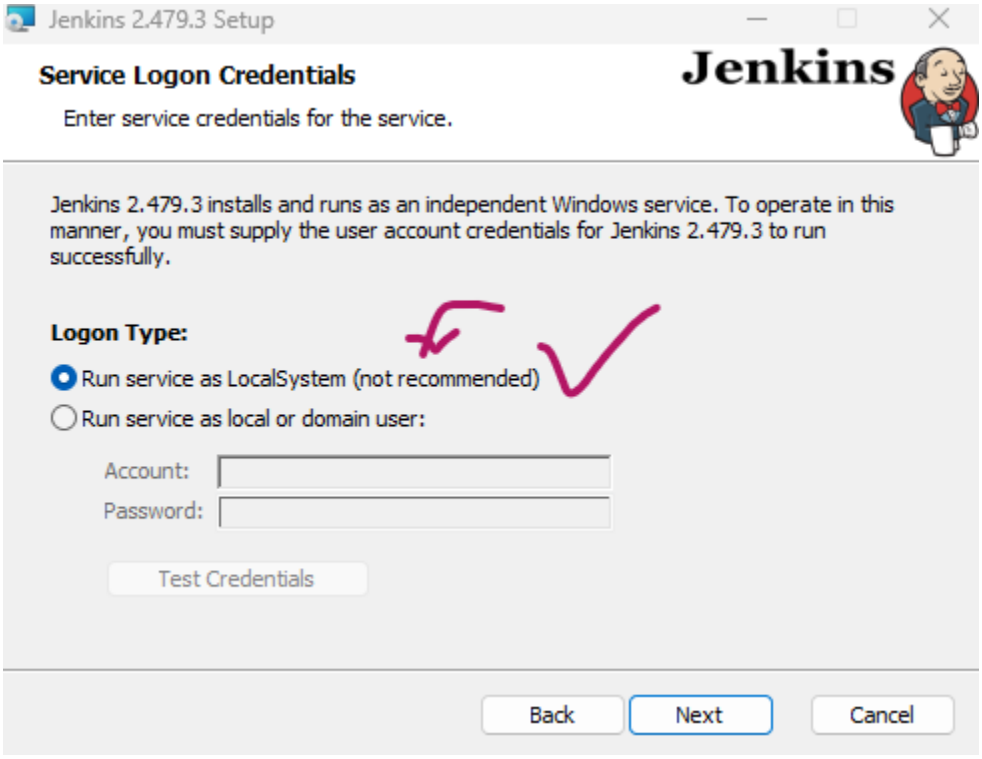
STEP1: Type jenkins download for windows
https://www.jenkins.io/download/



STEP2: After clicking on Windows Jenkins MSI Installer exe file be installed

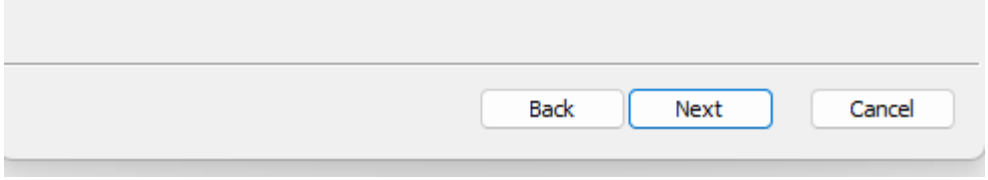


STEP3: Goto Jenkins MSI Installer click on it u opt for "Run service as Local System"



XX Need not to Provide Account & Password XX

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



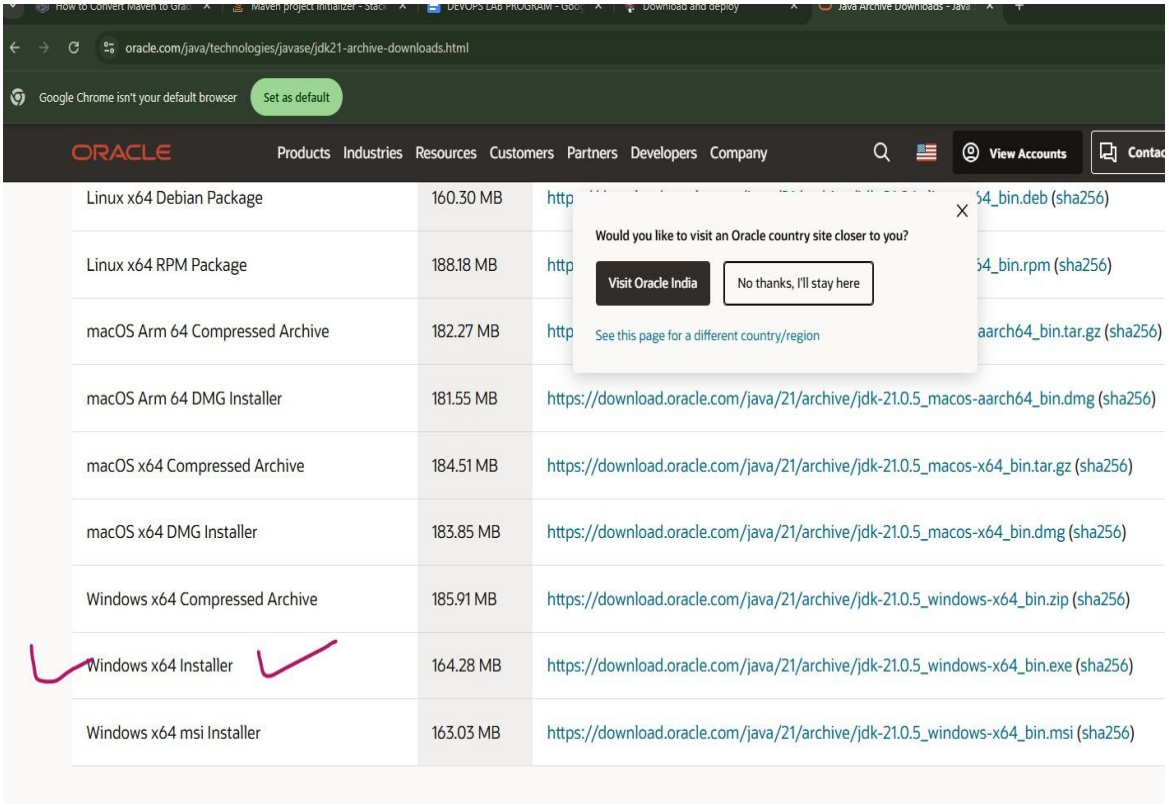
STEP5:It takes current jdk version thats available for safer side once goto cmd prompt and type command

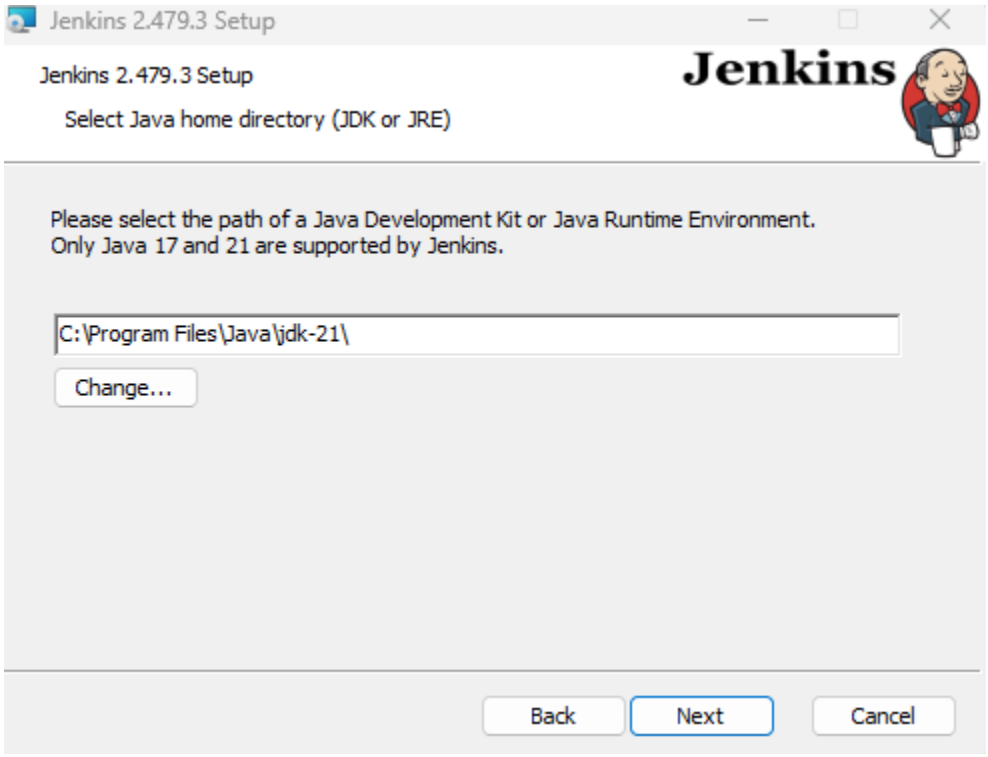
java –version

If matching click next

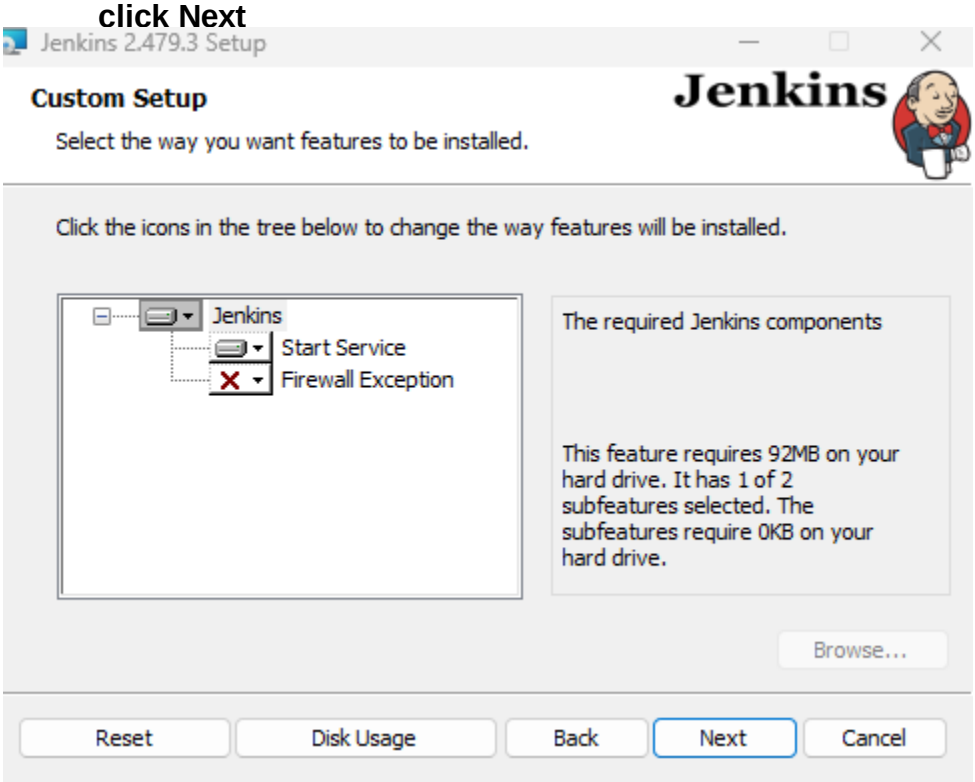
If failed to accept download jdk version 17 to 21 any of it

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

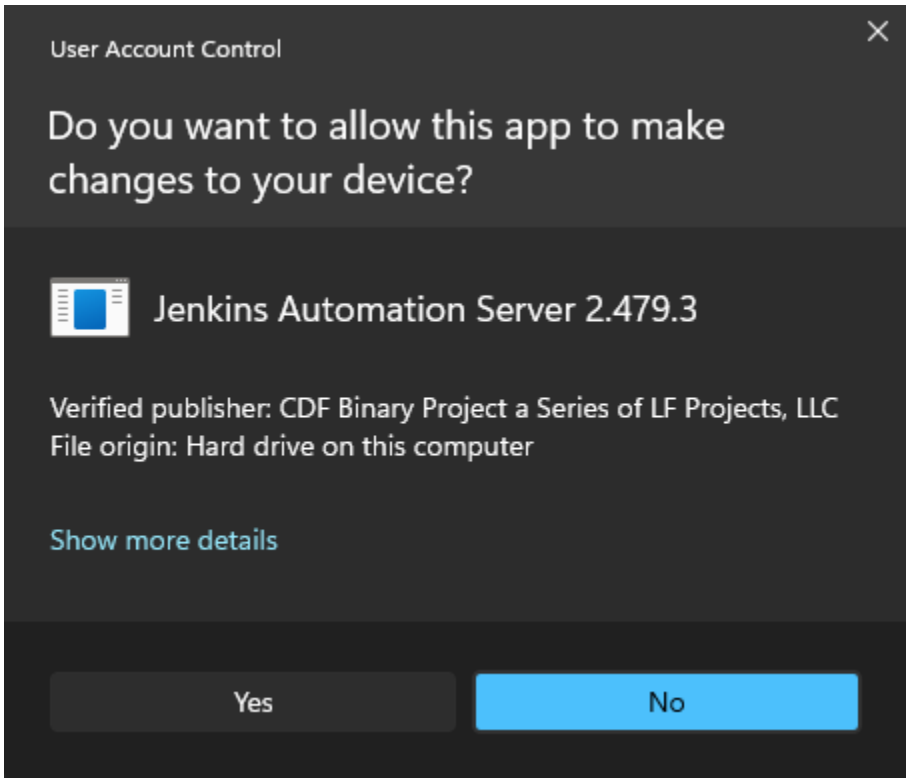




STEP6: Click NEXT after doing above step you will get screen as in Below again continue to click Next



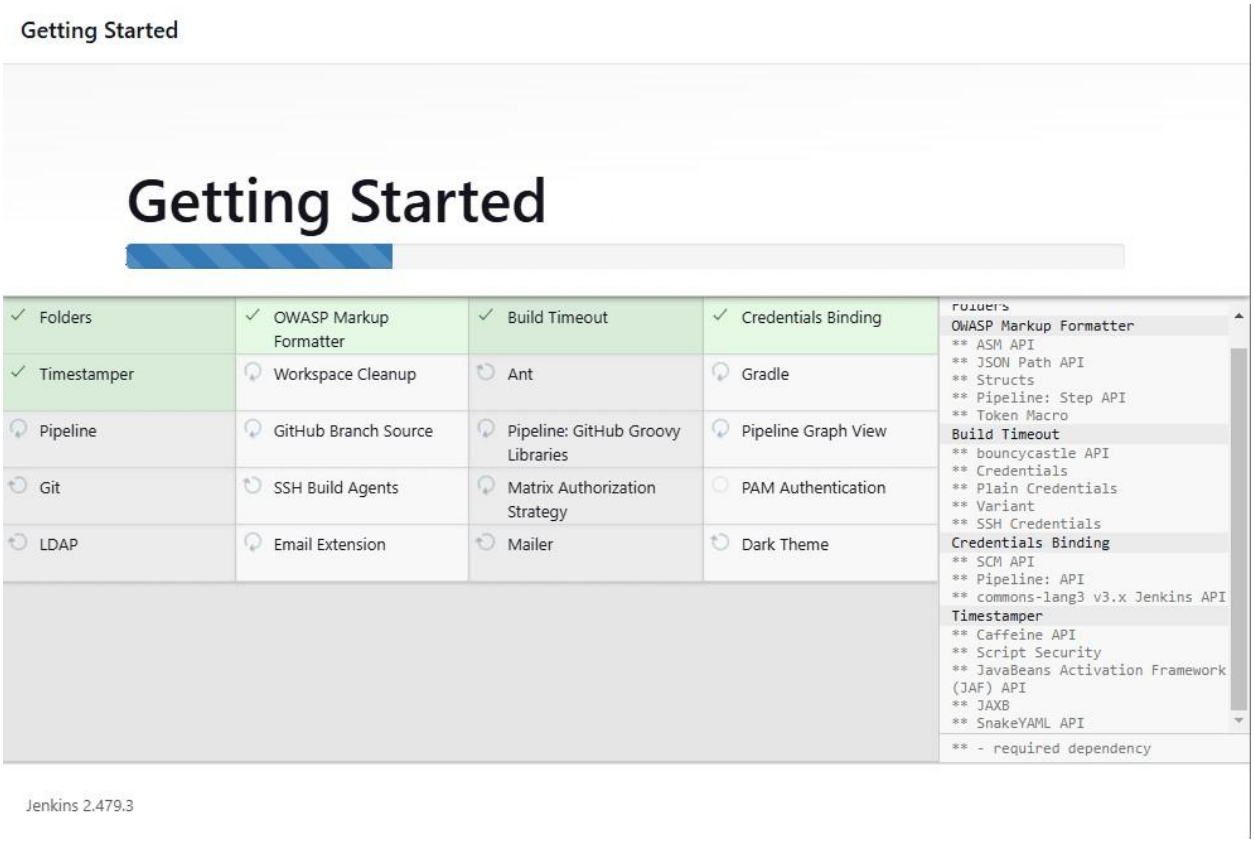
STEP7: The popup allow format comes for Java Automation Server allow it click install > Finally
Click Finish



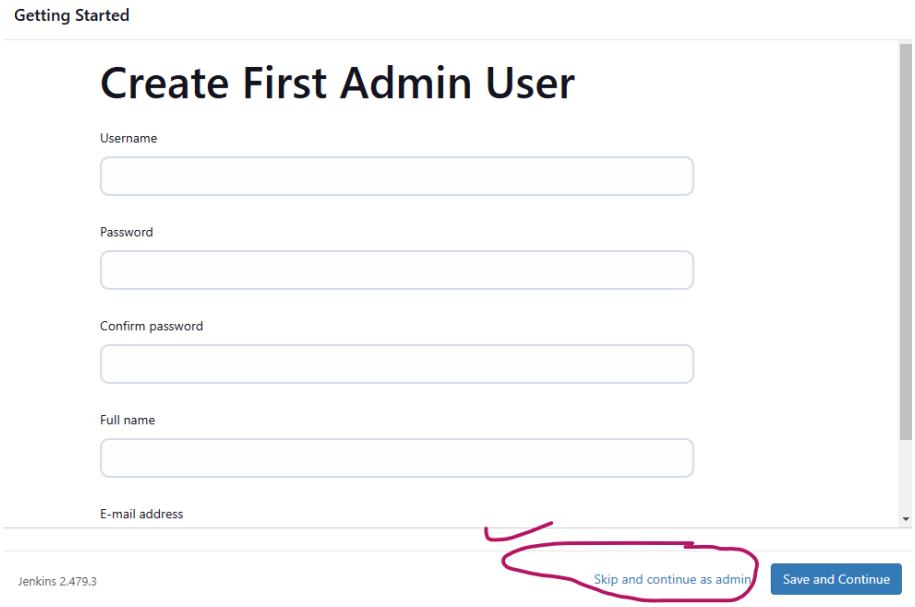
confi Now By default Our Jenkins run at
http://localhost:8080/

Final step very important Once  **It will ask for Administrator Passowrd so u 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 passowrd**

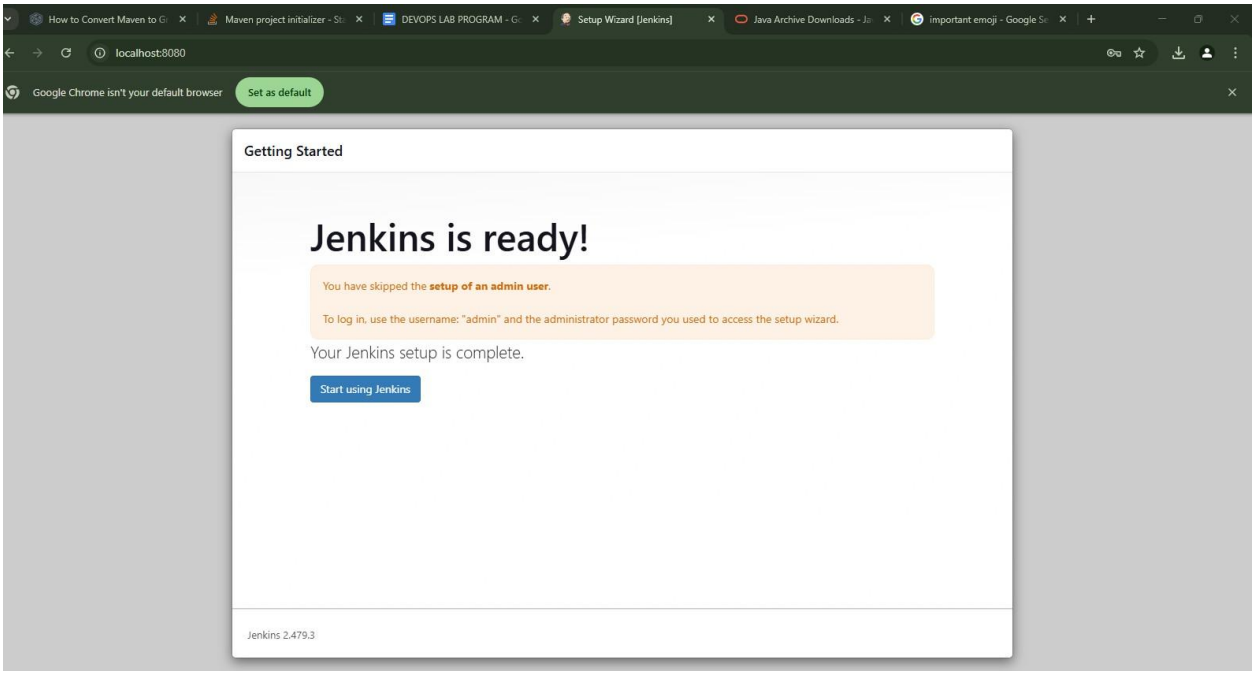
Then select Install Suggested Plugins it starts to install as in shown below



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.

VTU LAB

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 REST API that enables users to interact with Jenkins programmatically, allowing for integration with external tools, automation, and custom applications.**

VTU PROGRAMS

Benefits and Drawbacks of Using Jenkins for CI

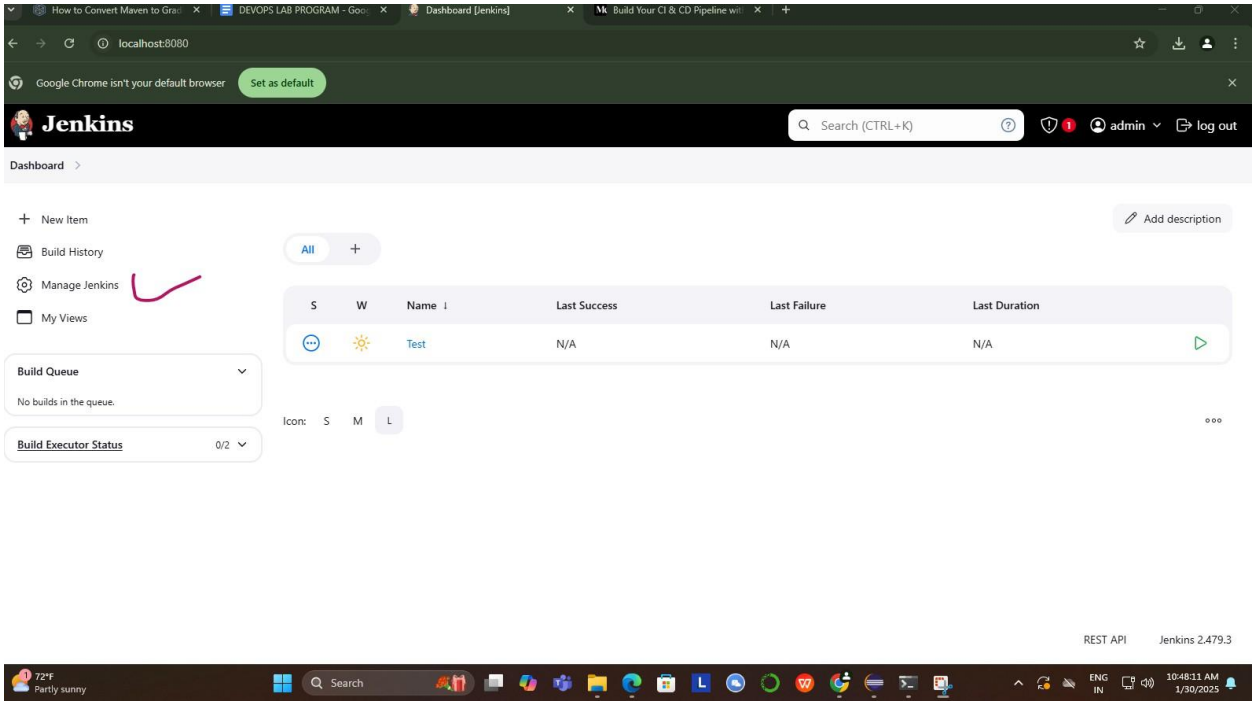
Jenkins CI offers numerous benefits that can streamline software development processes and improve overall efficiency:

- **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.**

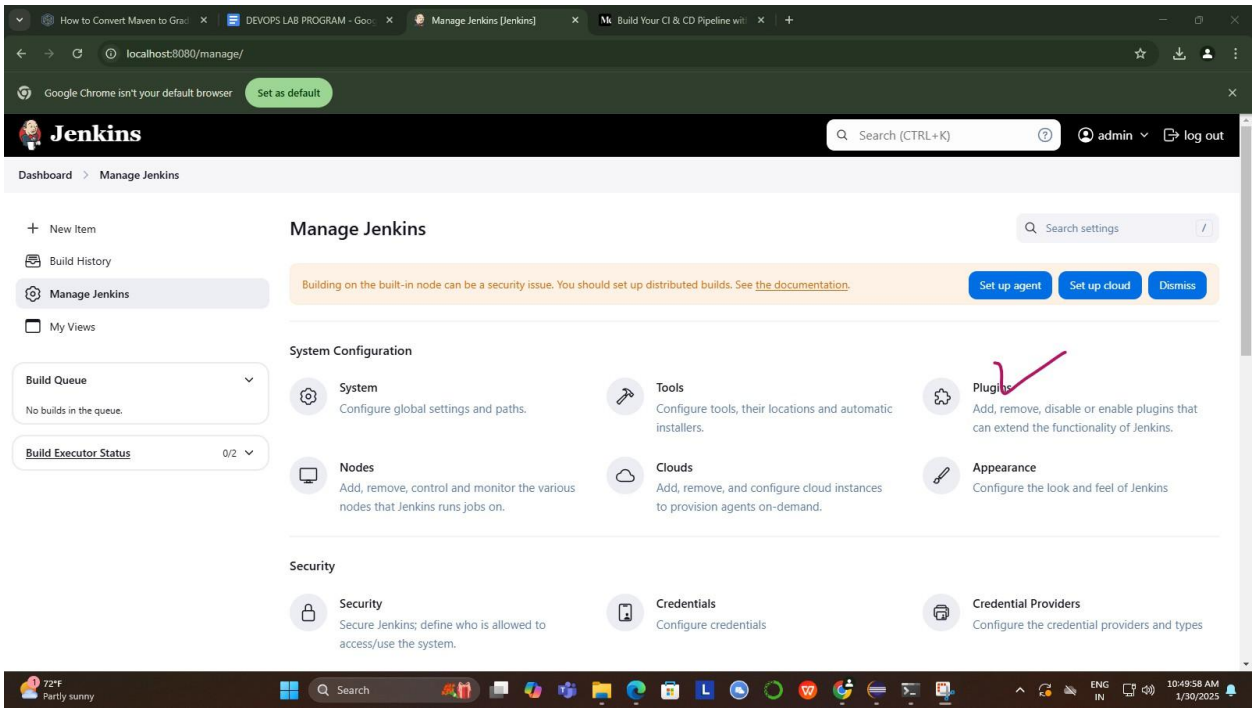
However, there are potential concerns associated with using Jenkins CI:

- **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.**

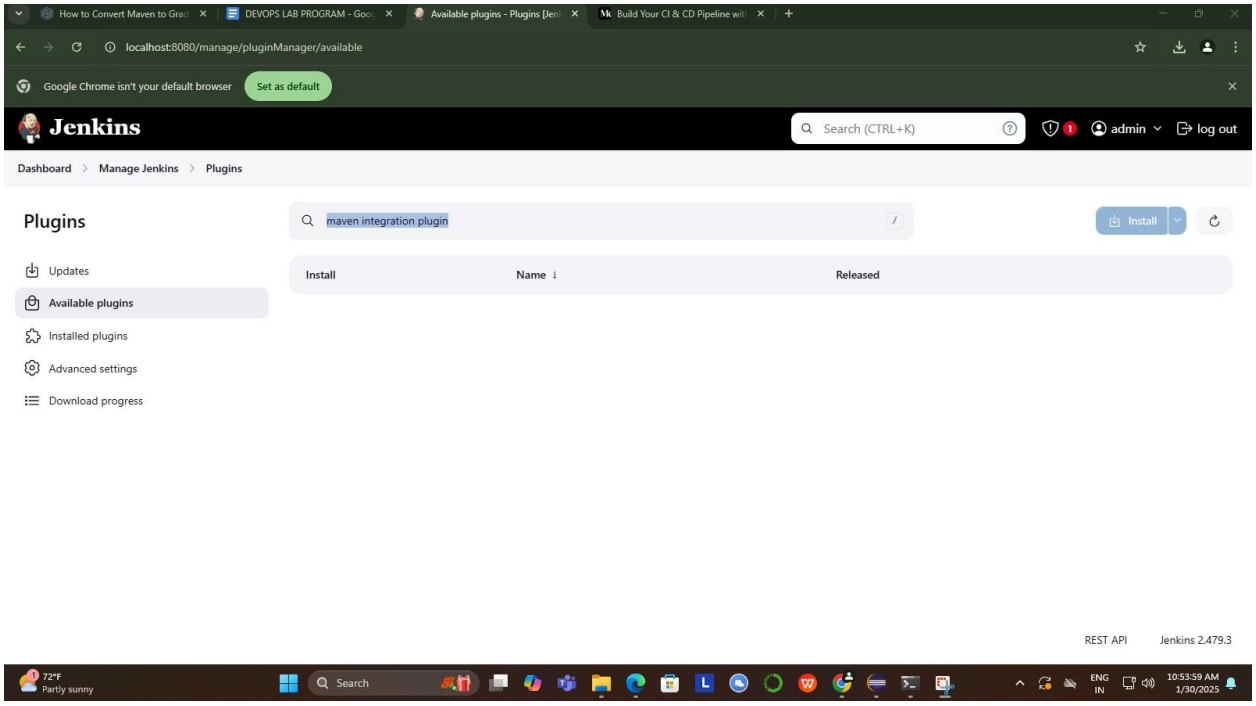
STEP1: Now coming to our Program to set CI Pipeline for Maven
Go to Jenkins Dashboard and Click on Manage Jenkins



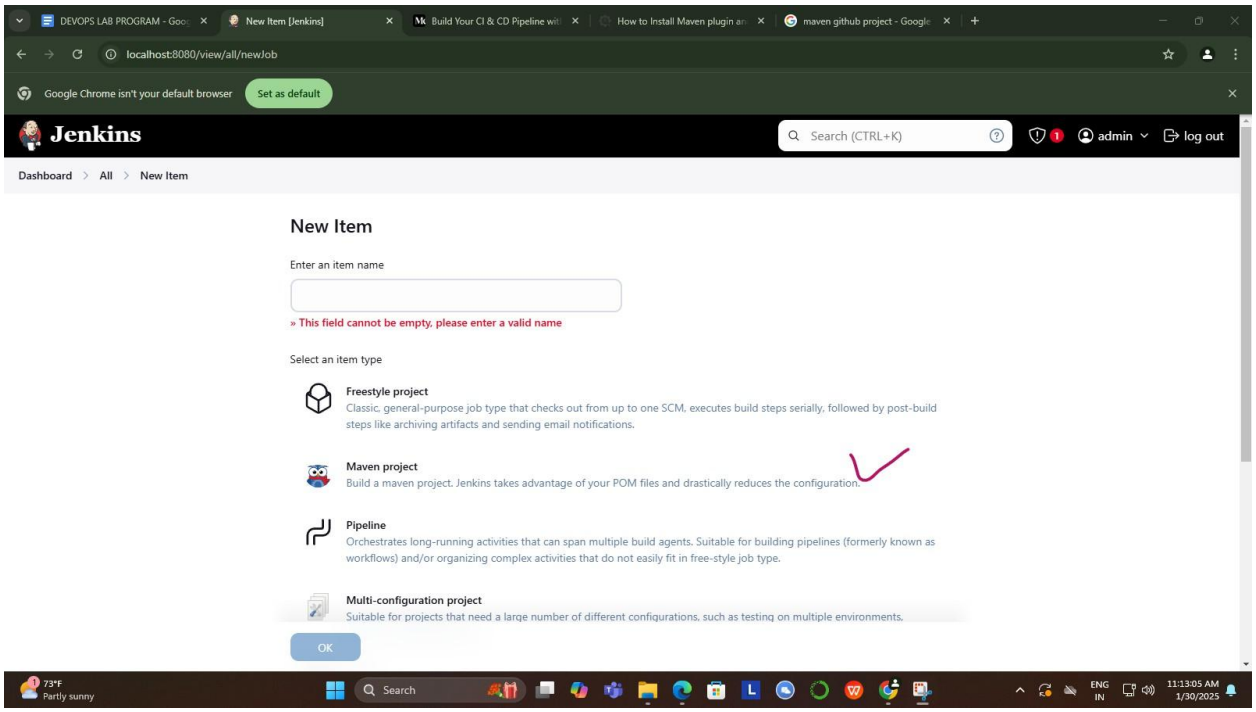
STEP2: Select Plugins



STEP3: Search for Maven IntegrationPlugin in Available Plugins and Install

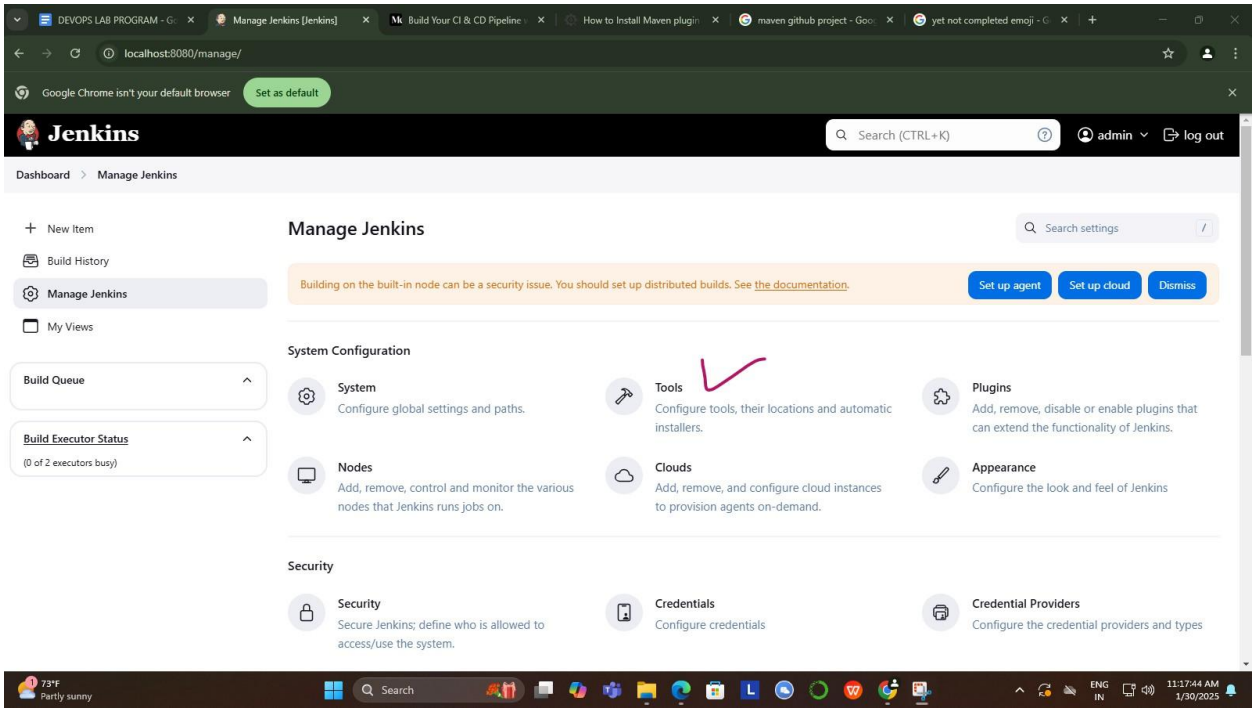


STEP4: After Maven Integration Plugin is Installed We able to see Maven Project as New Item



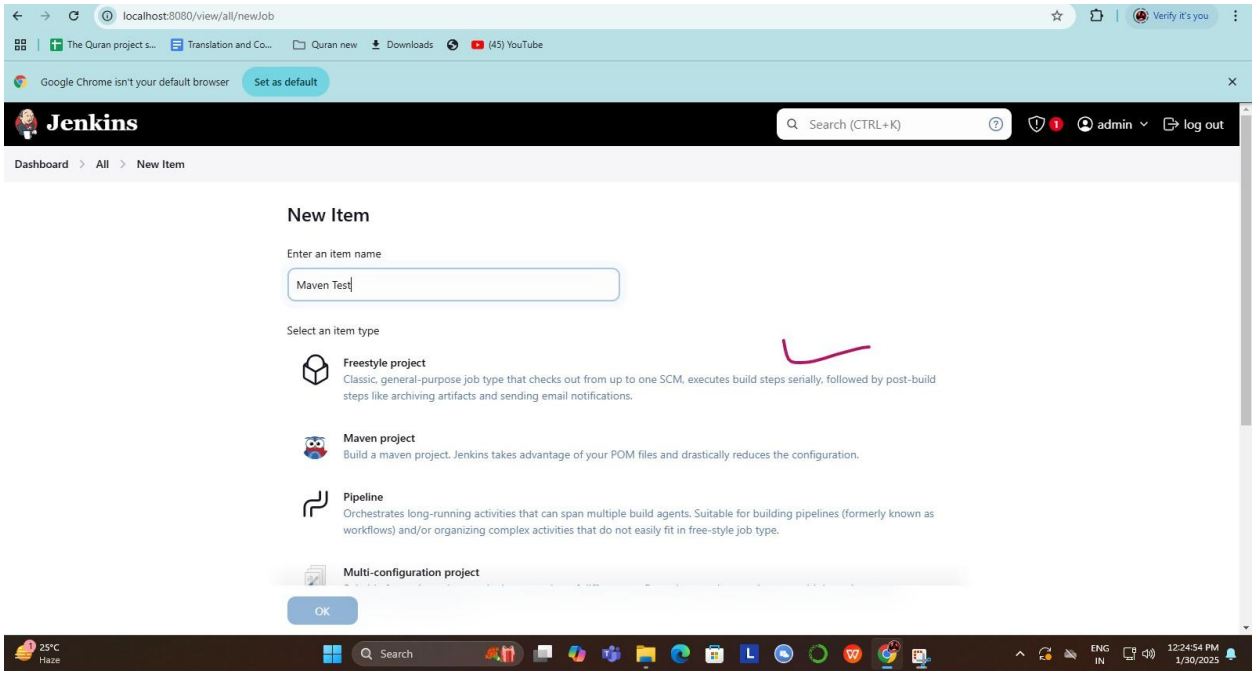
STEP5: YET not completed  **we have to configure the Location to properly Build and Run Maven Project**

So again click on Manage Jenkins and select Tools

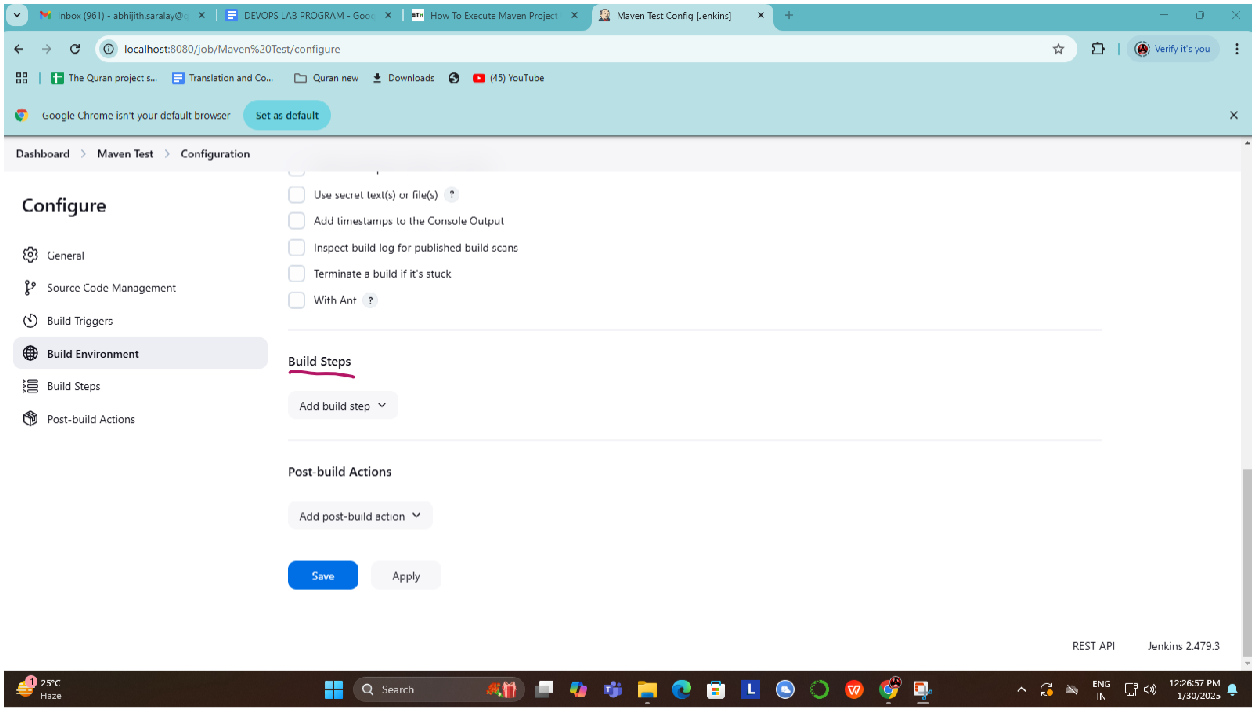


STEP6: Now lets not select Maven Project as new Item as we already have Maven project in local systems lets see how we can run the Maven Project with POM.XML

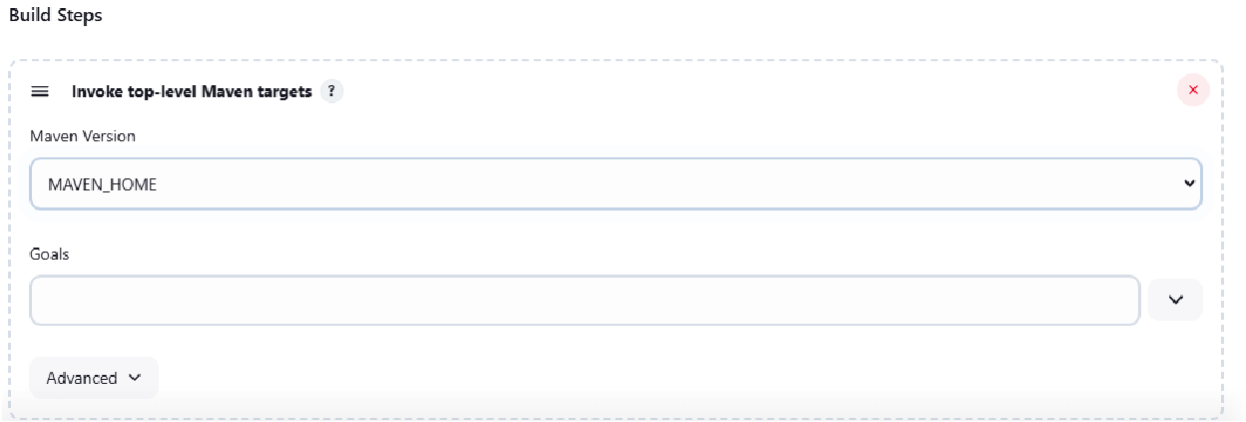
a) **Click on New Item**
b) **Provide Item Name and select Freestyle Project**



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.**

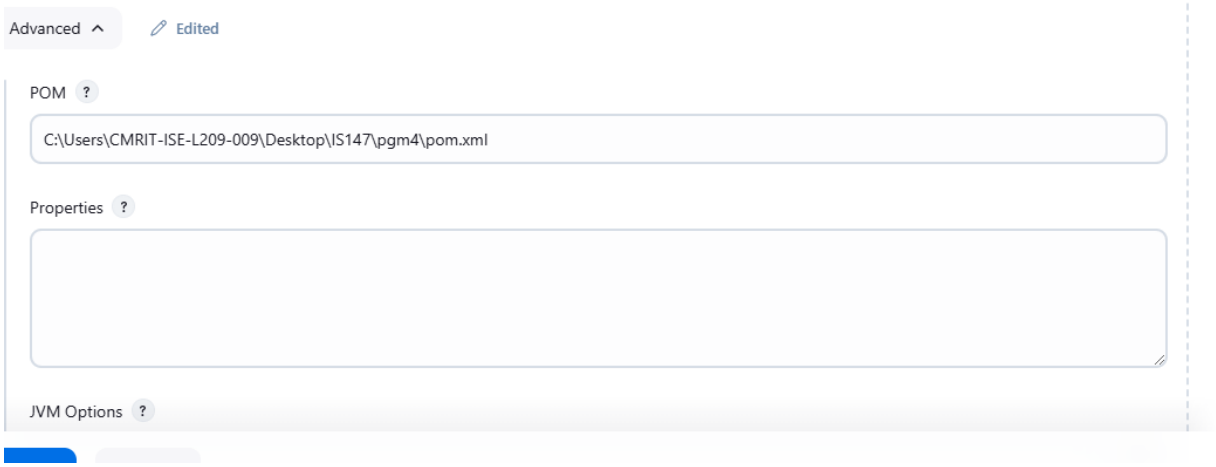


d) After selecting Invoke top-level Maven targets opt for proper environment version as in set in previous steps in my case its MAVEN_HOME



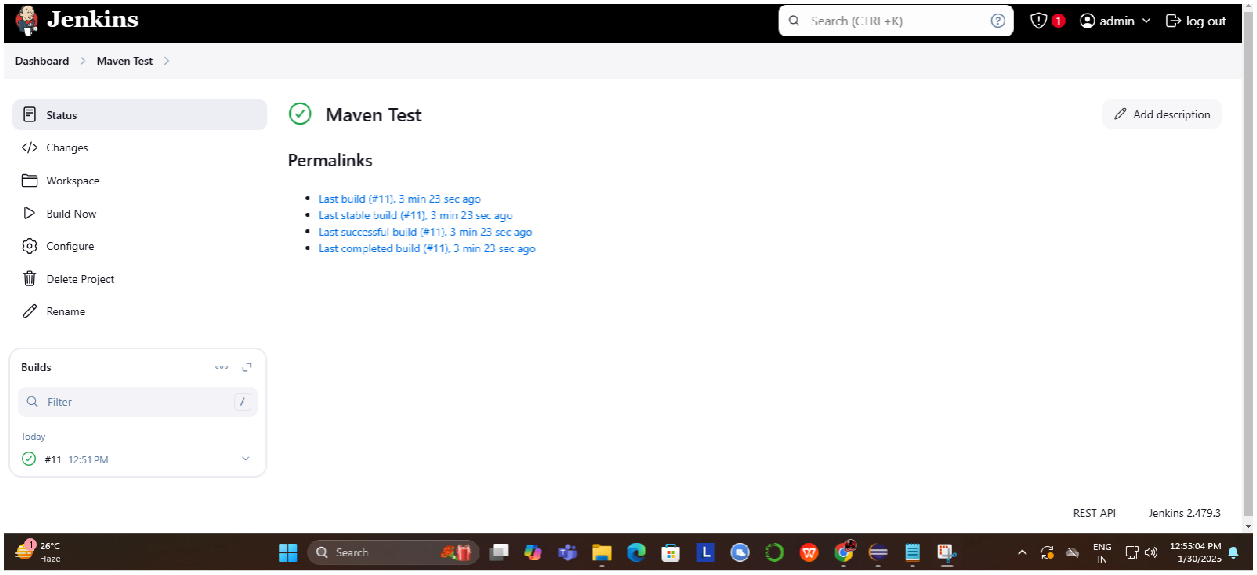
e) Enter Goal as
clean install

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

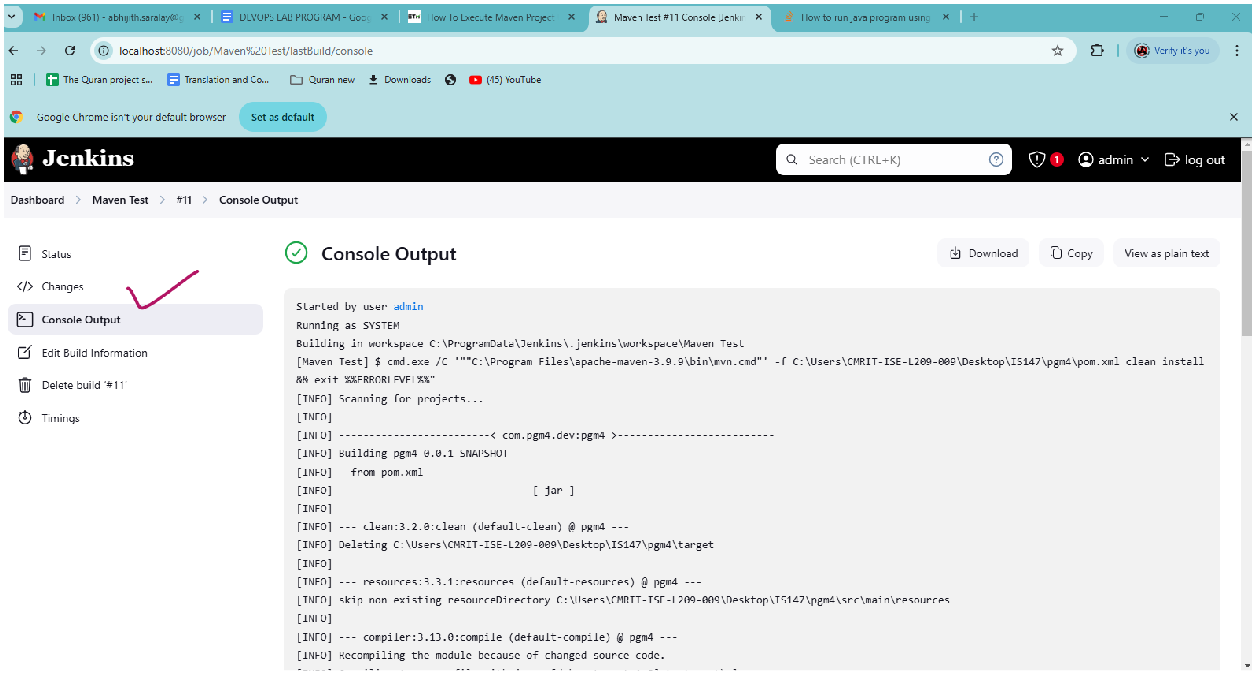


Goto pom.xml of your particular pgm and take path in my case its
C:\Users\CMRIT-ISE-L209-009\Desktop\IS147\pgm4\pom.xml

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 u can goto dashboard and opt to see build scripts.



PROGRAM 7: 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.

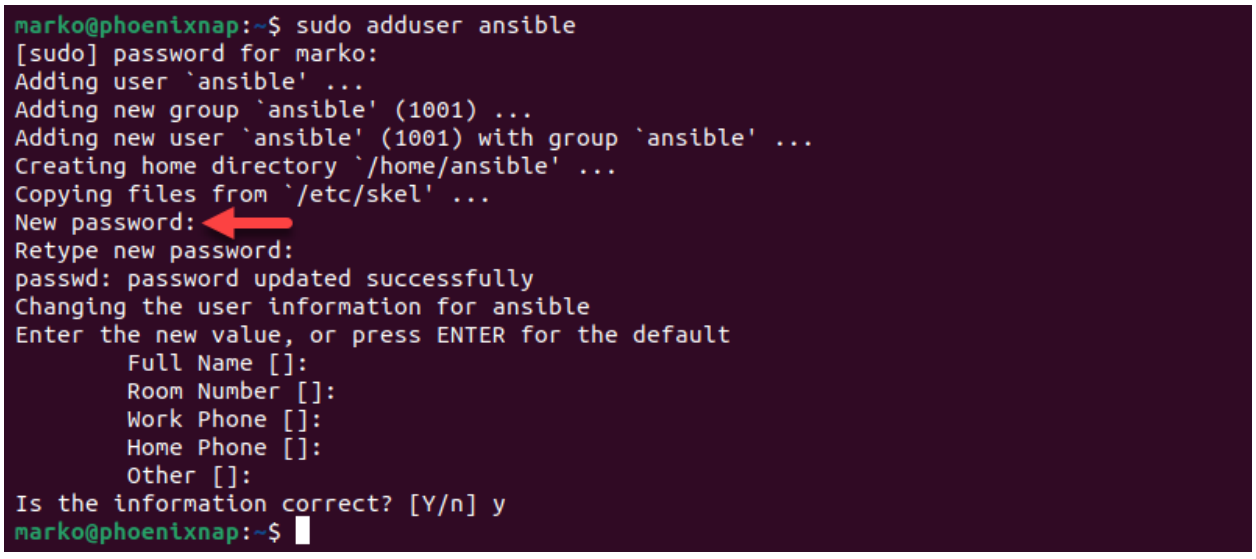
STEP 1: Configure Ansible Control Node

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

1. Create an administrator-level user for the control node. Use the adduser command:

sudo adduser [username]

2. When prompted, define a strong account password.



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

3. Use the following usermod command to assign superuser 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, a local machine, or a virtual machine running Ubuntu.**

STEP 2: Set up an 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:

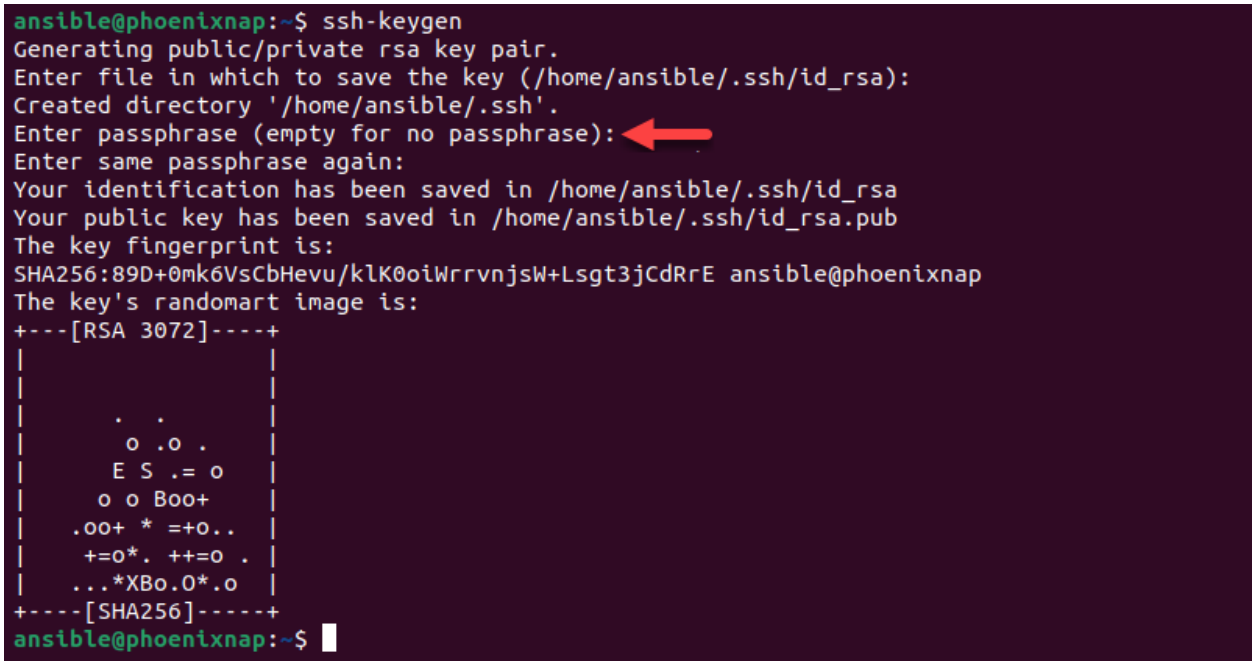
1. Enter the command below using the Ansible control node command line:

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 overwrite it. Overwriting makes the previous SSH key pair unusable, so ensure the old keys are no longer needed before confirming.**

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

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



STEP 3: 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 into authorized_keys directory. Apply the steps below for each new Ansible host:

1. Use the following ssh-copy-id command on the control node to copy the public key to a host:

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.conf or hostname -i
```

ssh ansible@192.168.0.81

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

3. Enter the remote host account password.

```
ansible@phoenixnap:~$ ssh-copy-id ansible@192.168.0.81
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/ansible/.ssh/id_rsa
.pub"
The authenticity of host '192.168.0.81 (192.168.0.81)' can't be established.
ED25519 key fingerprint is SHA256:fG67eTA0FjkEJlRAcQyxna/MDc7zX4fOdABzt+aktGM.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now i
t is to install the new keys
ansible@192.168.0.81's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'ansible@192.168.0.81'"
and check to make sure that only the key(s) you wanted were added.

ansible@phoenixnap:~$
```

The utility uploads the public key to the remote host account.

STEP4 : Install Ansible

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

1. Ensure the package index is up to date

sudo apt update

2. Install Ansible on Ubuntu with the following command:

sudo apt install ansible -y

STEP 5: 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 file, the path to the executable, and other information.

```
ansible@phoenixnap:~$ ansible --version
ansible 2.10.8
  config file = /home/ansible/ansible.cfg
  configured module search path = ['/home/ansible/.ansible/plugins/modules', '/usr/share
/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  executable location = /usr/bin/ansible
  python version = 3.10.12 (main, Nov 20 2023, 15:14:05) [GCC 11.4.0]
ansible@phoenixnap:~$
```

STEP 6: 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 information about the remote hosts managed through the Ansible control node.

: For an in-depth overview of creating files on remote hosts, refer to our ar Note ticle 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

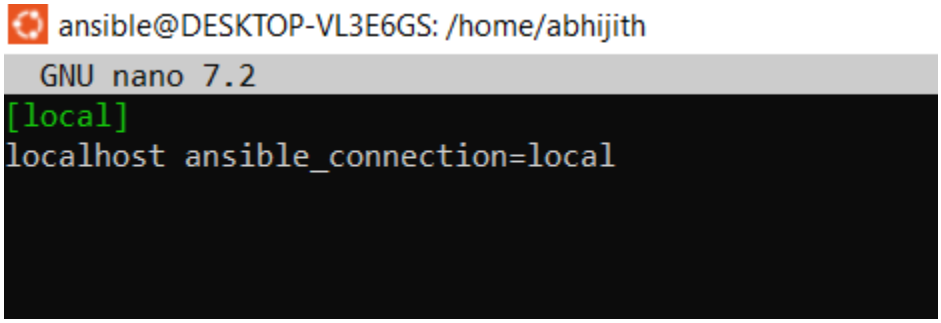
2. Use a text editor such as Nano to create a file named hosts:

sudo nano /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:

STEP 7: 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:

sudo ansible all -m ping

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 connection.

```
ansible@DESKTOP-VL3E6GS:/home/abhijith$ sudo ansible all -m ping
localhost | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
ansible@DESKTOP-VL3E6GS:/home/abhijith$
```

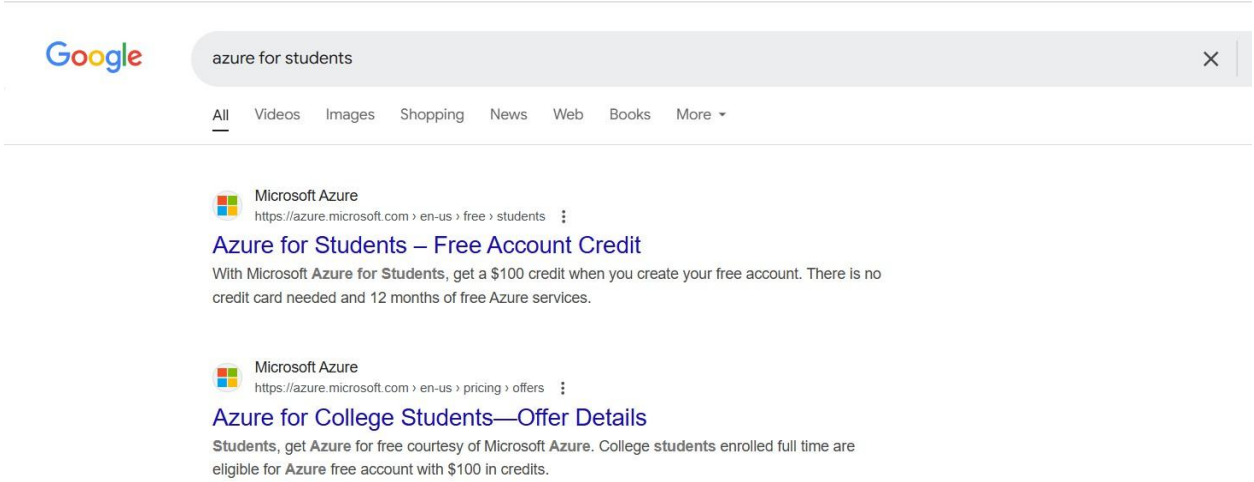
The Ansible control node is now set up to control the connected remote hosts.

Conclusion

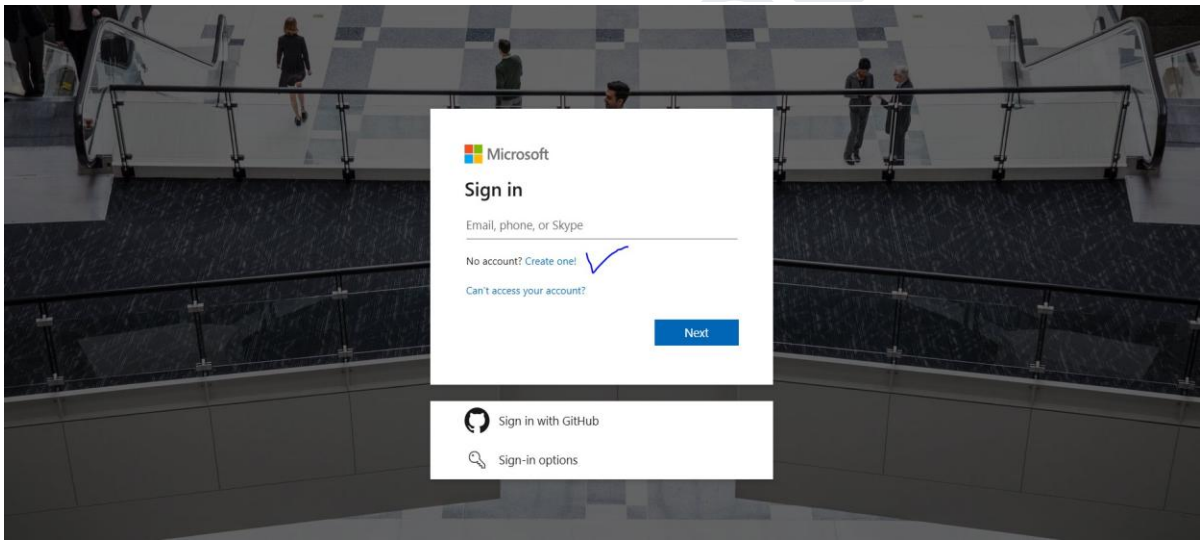
After following the steps in this guide, you have successfully installed Ansible on Ubuntu and can execute commands and playbooks on remote hosts. The guide provided instructions for setting up the Ansible control node and connecting it with the hosts via SSH.

PROGRAM 9:Introduction to Azure DevOps: Overview of Azure DevOps Services, Setting Up an Azure DevOps Account and Project.

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

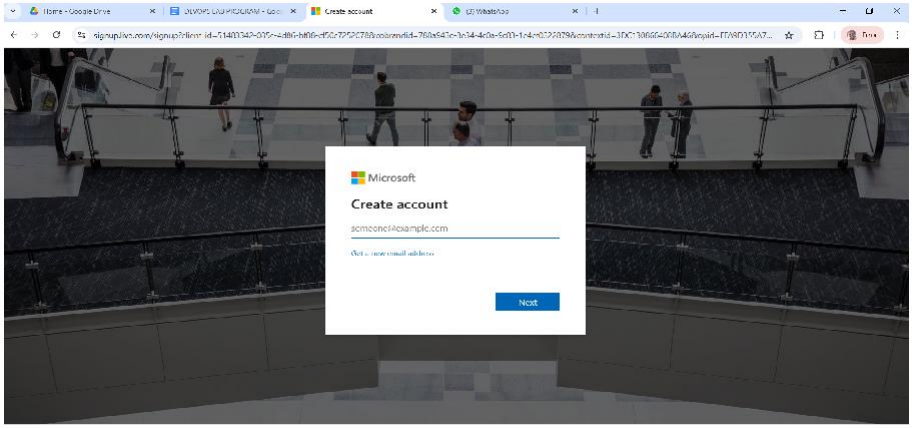


STEP2: Click on Start Free after that u get screen as in below

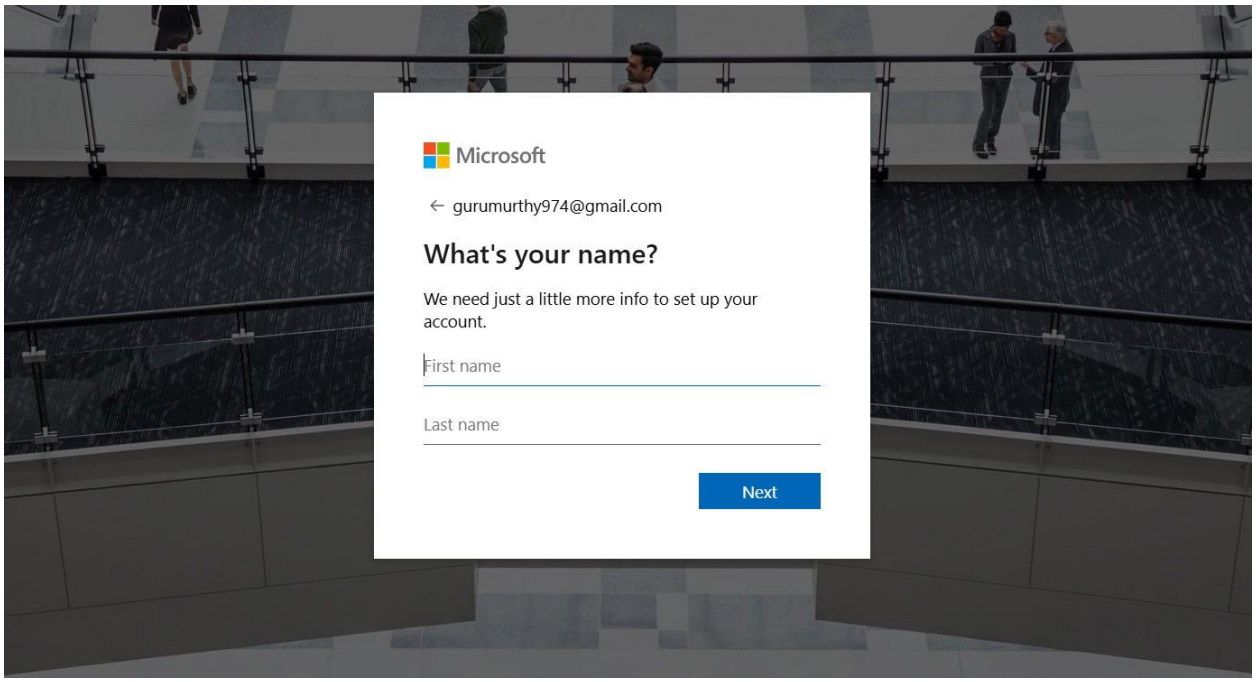


Click on Create one, If u have Github account u can Sign in using github account better way is to create one account

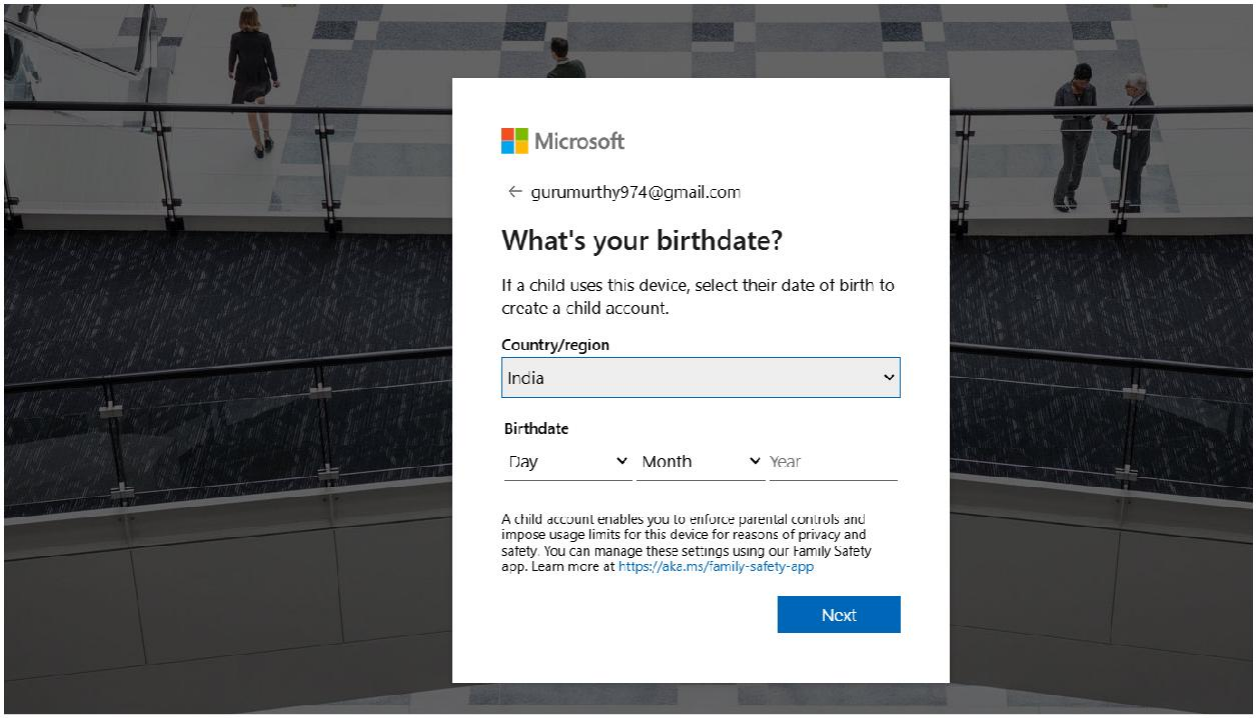
STEP3:Provide your email id at place of create account



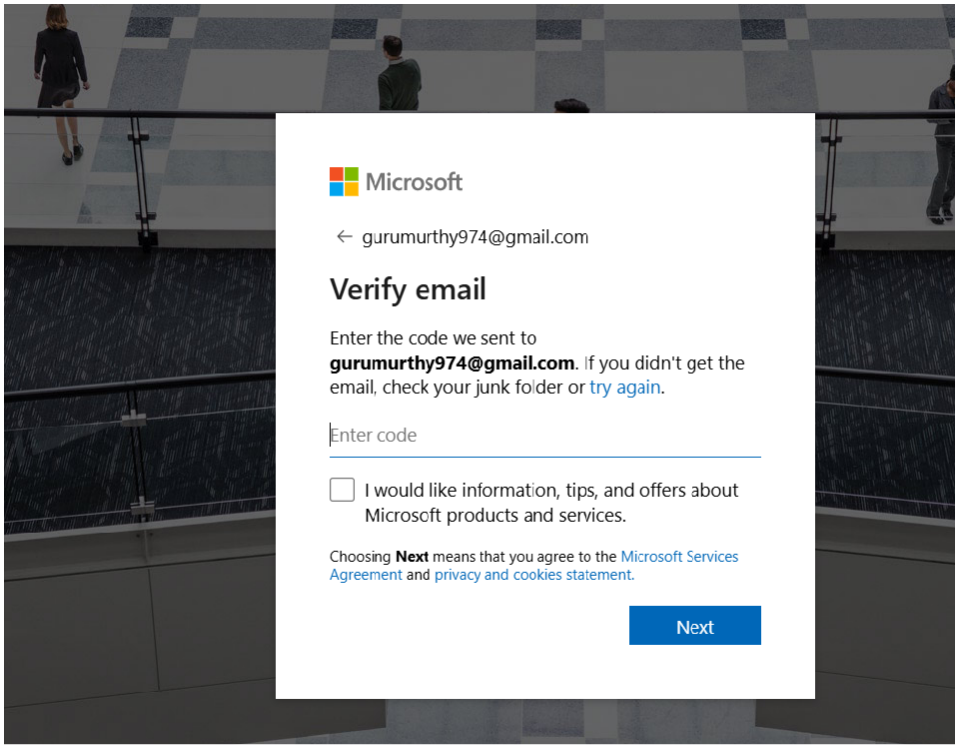
STEP4:After password is set provide your First name Last name



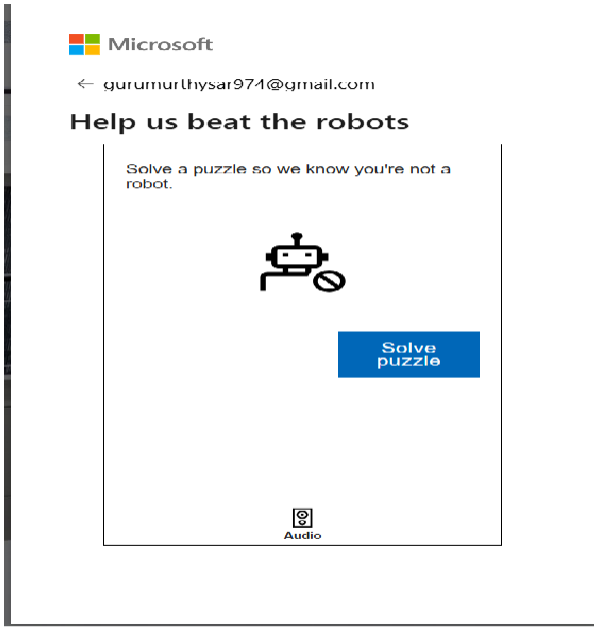
Then provide Country,Date of Birth



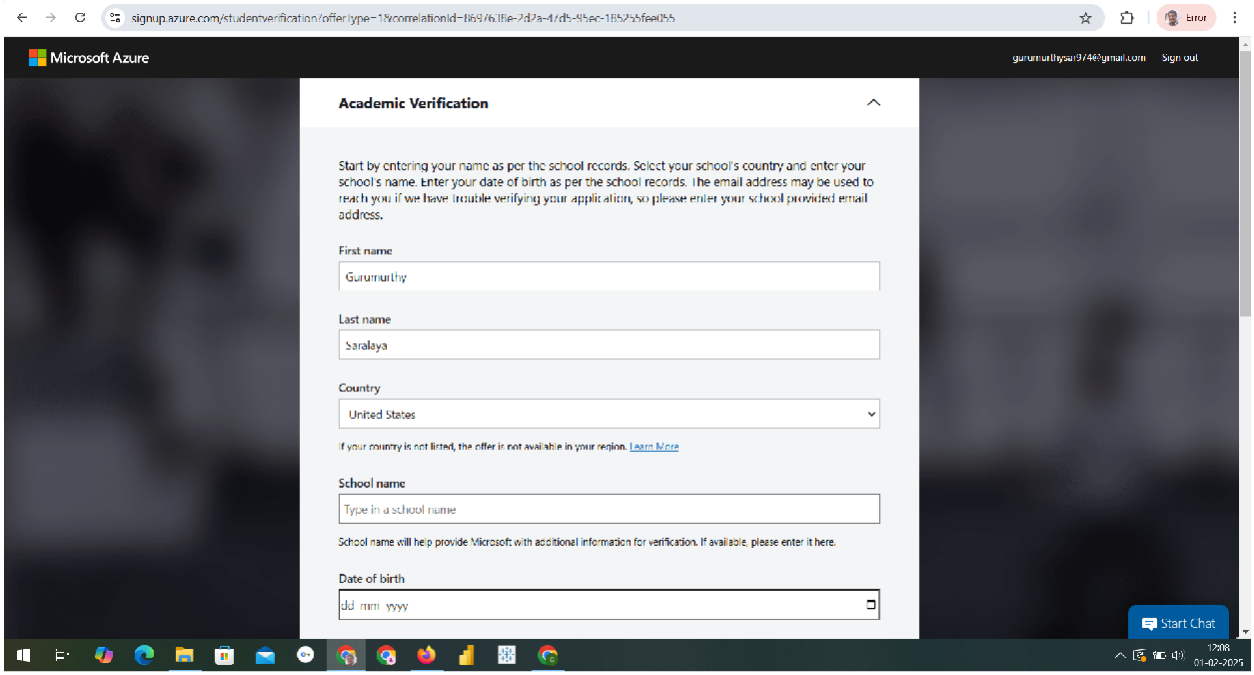
STEP5: Verification code be mailed to the mentioned once kindly type it



STEP6:After code is verified as u got in the mail referred
U be given an option to solve puzzlegame



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



After proper college email is given u get verification mail with link to mail u have provided

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.

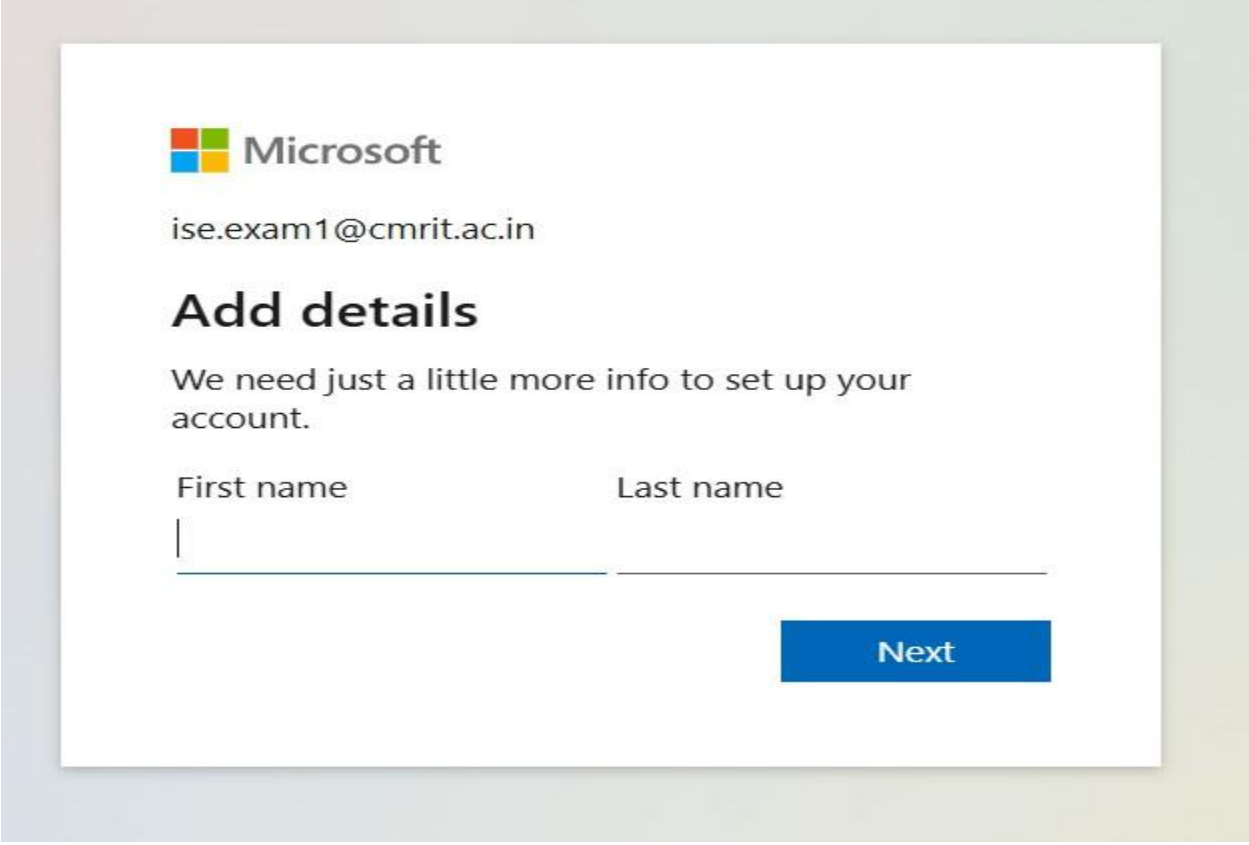
To complete your academic status verification, please click the link below. The link will automatically expire if not used within 5 days.
After clicking the link, your verification status will be confirmed and you will return to the site.

-Navigate to: https://verifyemail.microsoft.com/v1.0/tokenverification/verify?signature=YPQIdNoTloPJ8nJy8QXZhbyR5Y5wAf
wKVaNqhX5hgX6Yb0hOyqw5Aa5CsLqDGTWbHGIFSKjx37ROJFB7B0BoqKqDTCsJ3ADClXL9McGPZs97Ms2qbP54fx0A
GGTgdu7Y2SGVdpLBk98PiP33rVZf%2F2ES3z4lR8zM%2BjjuRvGxNuTK%2B4lM2tfFKm2kSsbp%2Bi754BLlZSJW4216NcWohhks8I%
2Fdd2qk7fgLurMrUhi8MaiPwQjLzyBmWbd9bjRY4PtX%2FfhwpOZTPGdROYFUpjsYuf6vwYojSylbtfBATxAI1se5jIzZsAL2qdHWV03
x8Yhm3gjN4TGYdxzizDLE0CGCn6h8GVjyls%2BQEIO96N0Rvqi%2FnlccNibsdcmtAN0u9ivHcdpoHRnwX22Q58JgggCzyefDnDbFgxuiteaHQn
%2BPkgKnOs26OHSdeVVEc%2B0MBsBAOjBTFiHeL7XEkYdahHXReTh8a77XxbRAQAdXGXIfW0caHpUHafaNKxiQBM28dR%
2BPgZB0n3kKFMiR20ltYQtxpSkOGa6DOssh38bumS2WrSQ%3D

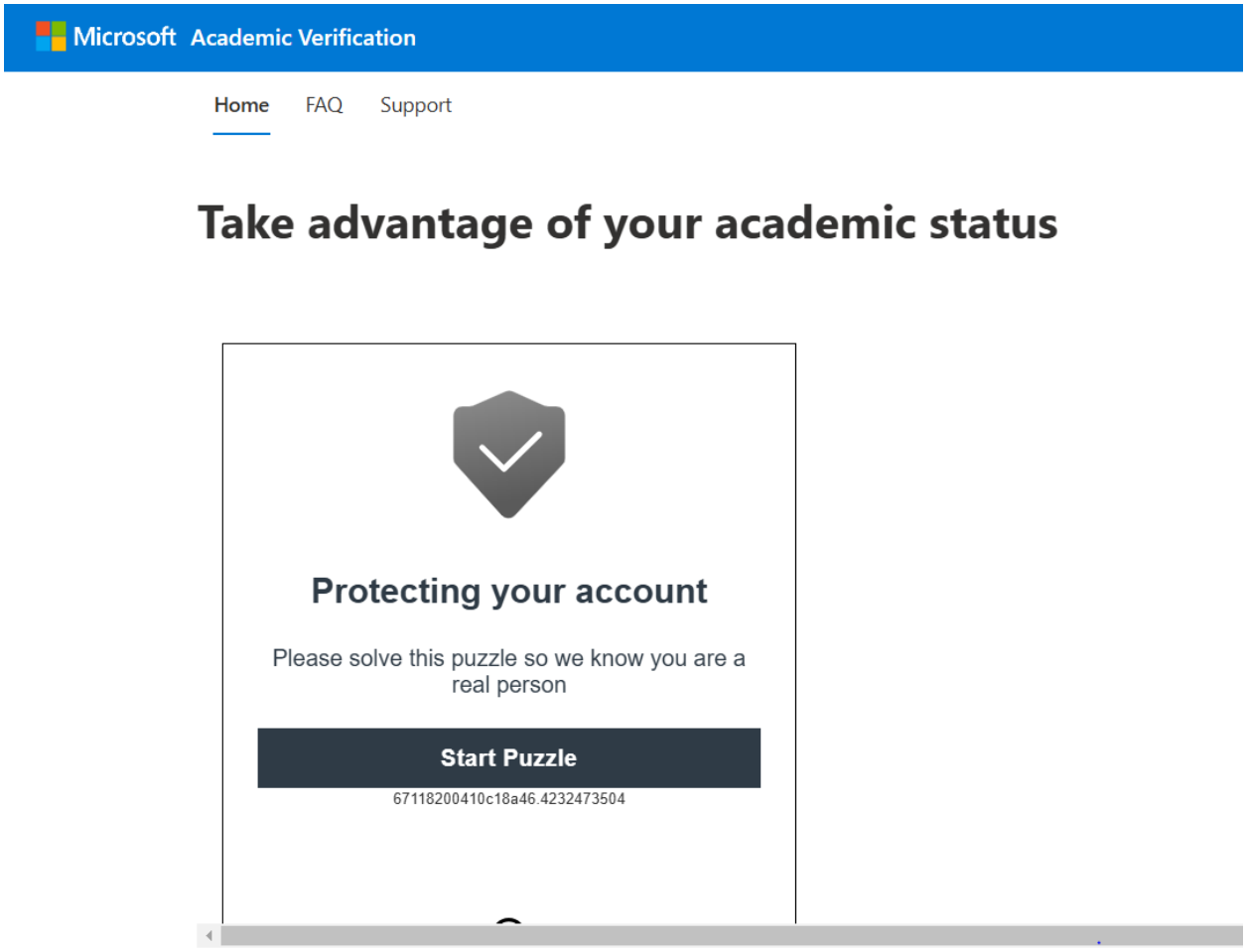
Thank You,
The **Microsoft Academic Verification Team**

Click on link sent

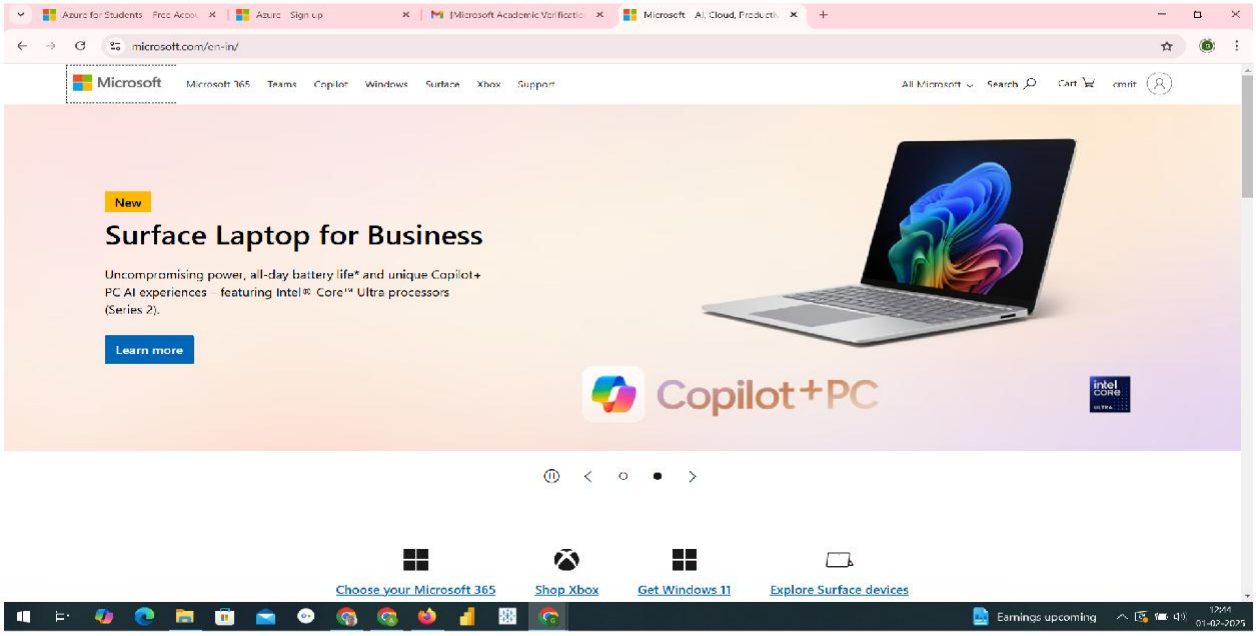
After u Click The screen be as in Below



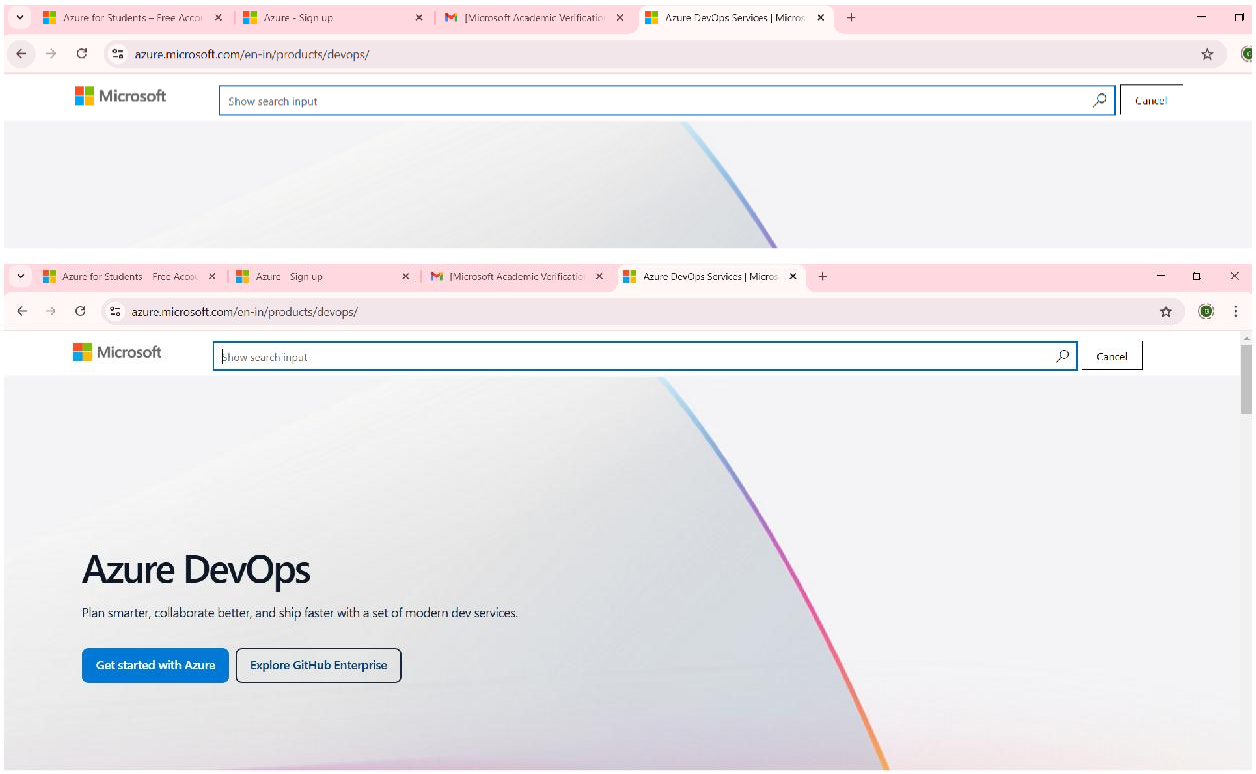
TO MAKE UR ACCOUNT SECURE IT AGAIN HAVE PUZZLE



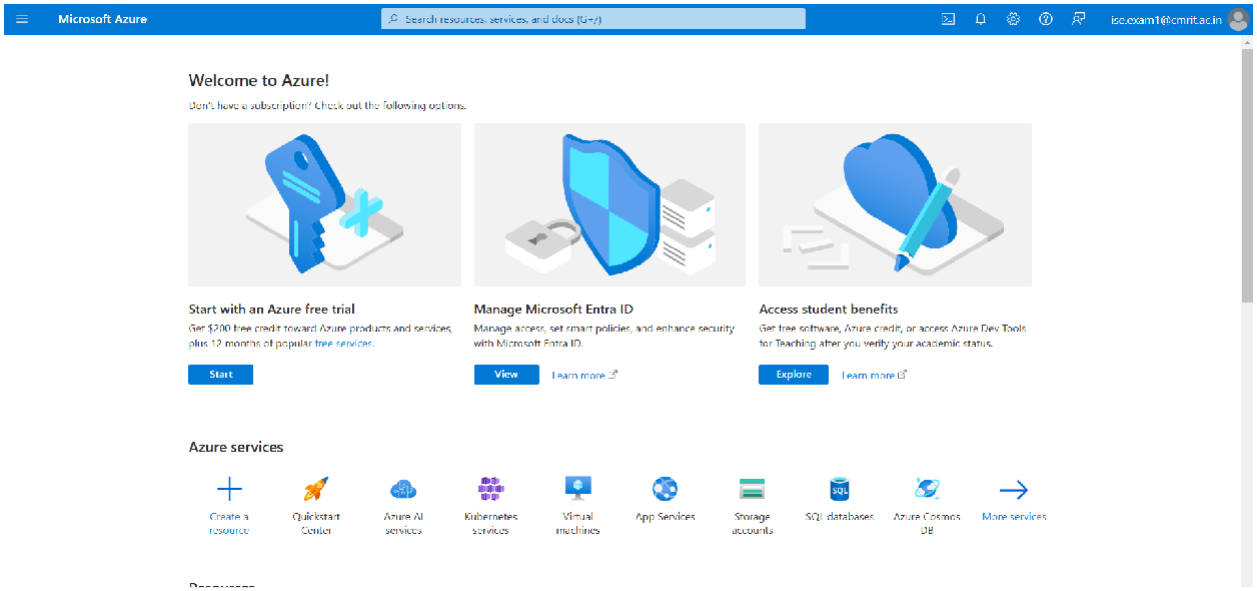
Finally ur screen be like this



Go To search and type Azure Devops



Click on Get started with Azure
After the click u get the screen of Get free need not to do anything just click on Signin You will get screen as in below

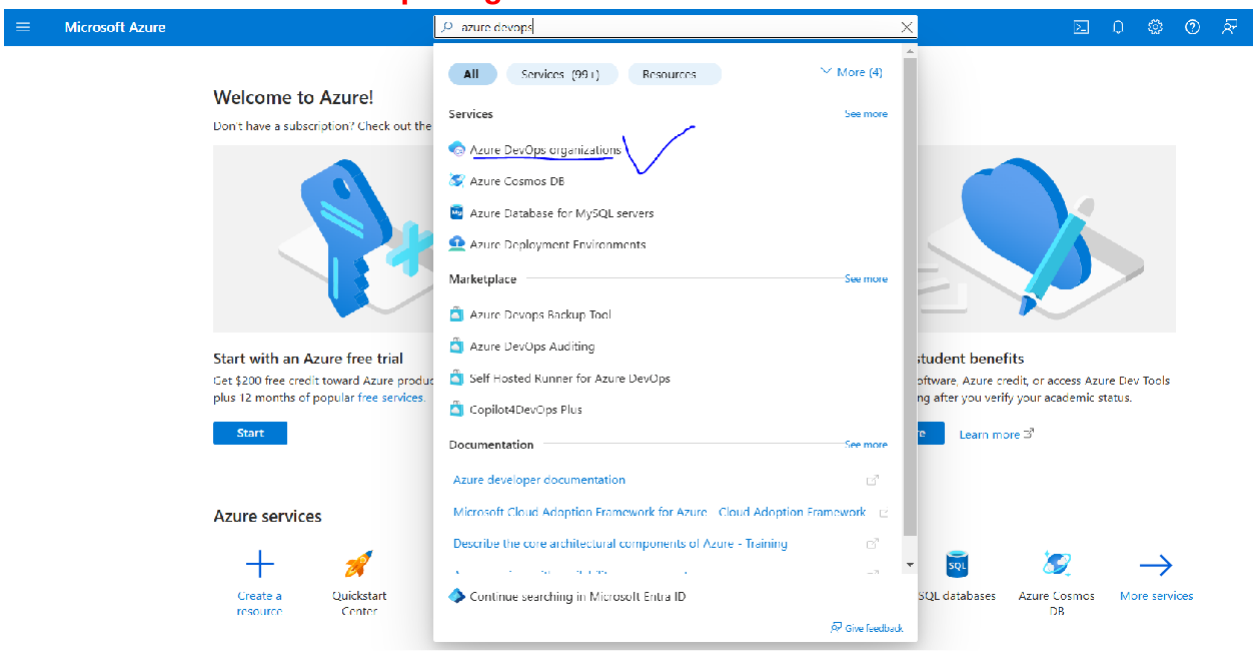


THIS IS THE HOME PAGE OF THE MICROSOFT AZURE where you can see n number of services now our target is Azure Devops

In top where have search for services

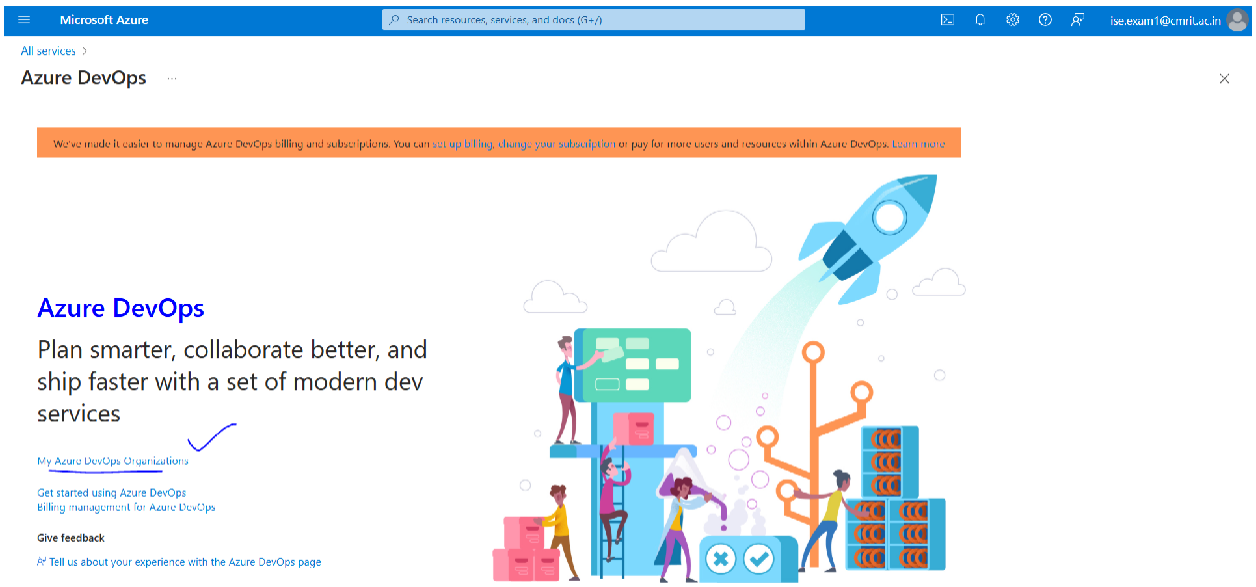
Type Azure Devops

STEP 8:Select Azure Devops Organization



STEP9:

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

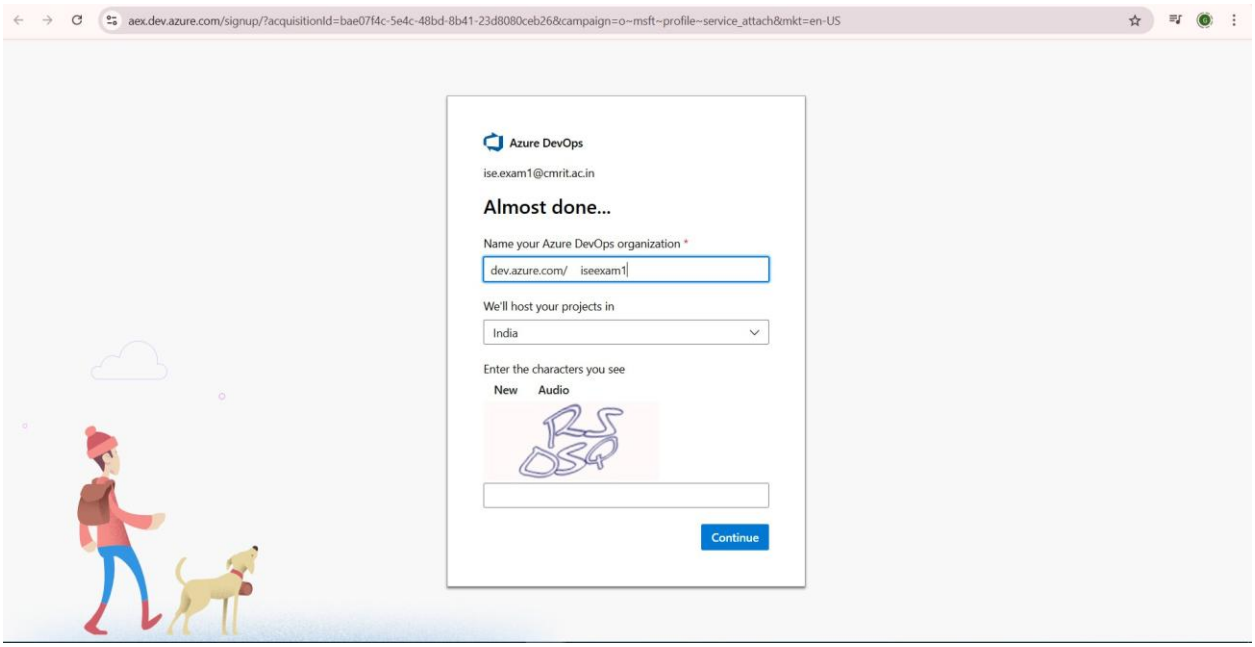


After above selection it once again reverifies name and email just click Continue
After it U get a Screen

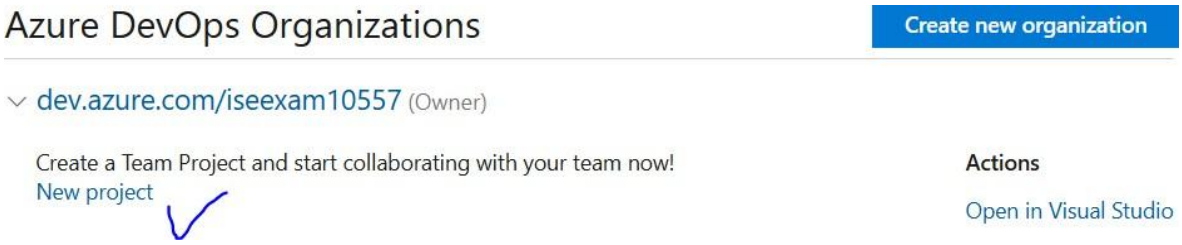
Get started with Azure DevOps

Plan better, code together, ship faster with Azure DevOps

Create new organization



You will be able to see Organization is Created



Finally After Creating a New Organization

U can create Project of ur choice as per requirement

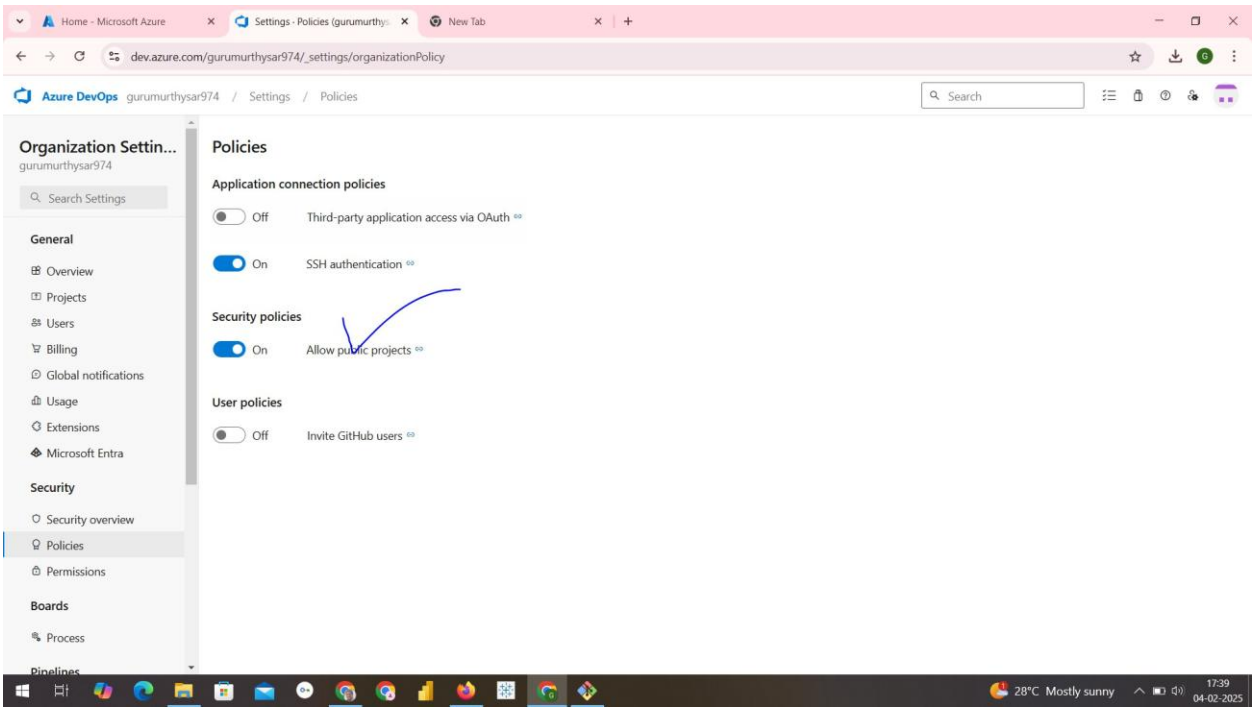
Every time u need not to signin u can bookmark or add the below link as shortcut

https://aex.dev.azure.com/

https://portal.azure.com/#home

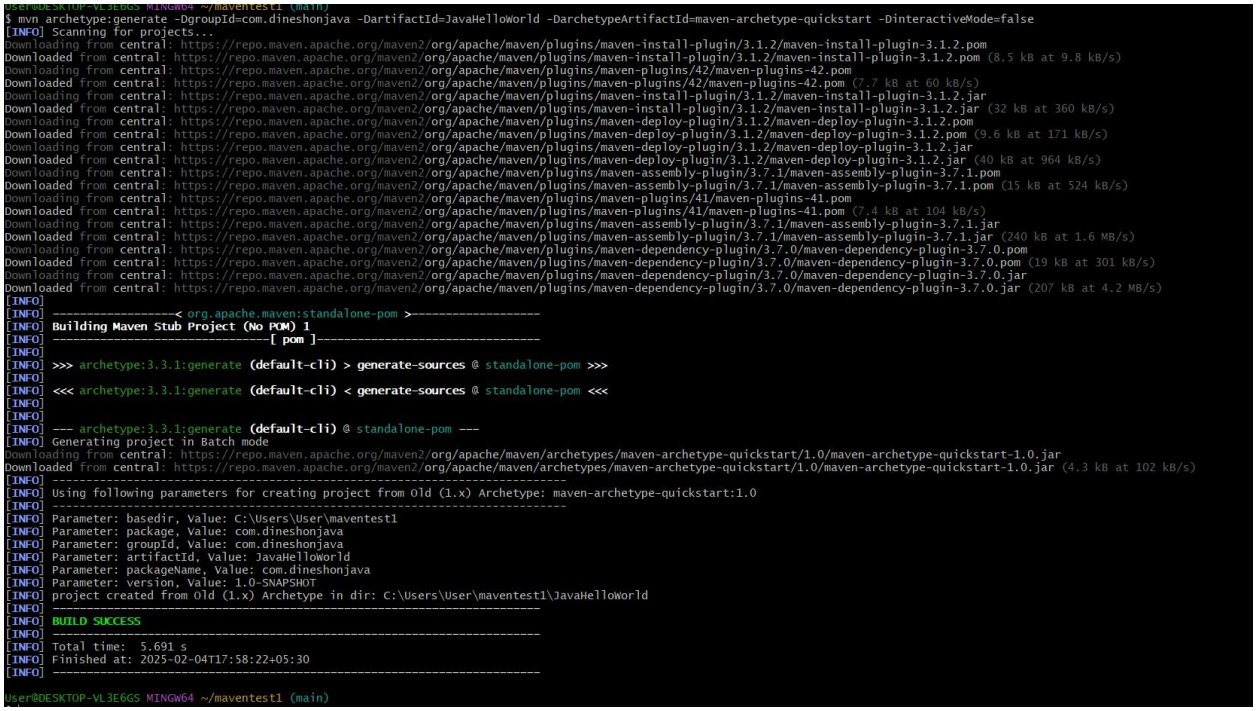
PROGRAM10: Creating Build Pipelines: Building a Maven/Gradle Project with Azure Pipelines, Integrating Code Repositories (e.g., GitHub, Azure Repos), Running Unit Tests and Generating Reports.

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



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

STEP3: to create simple hellow 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 repository in github as maventestazure**

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

git init

git add .

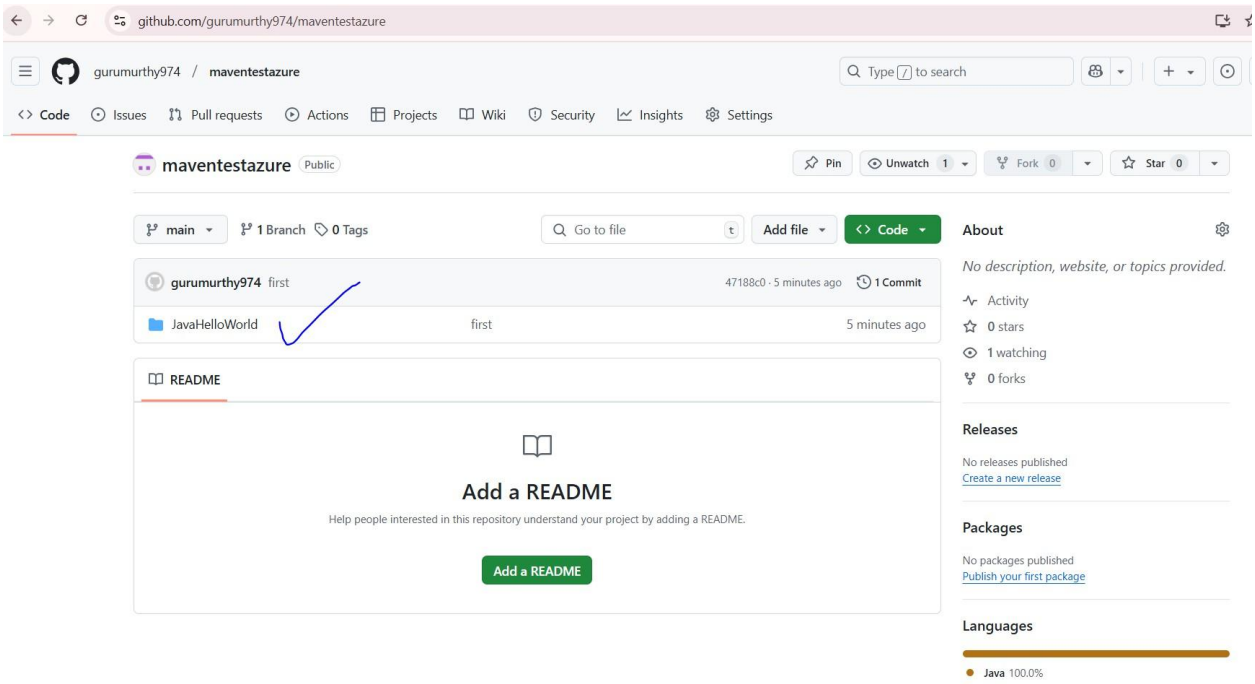
git commit -m “azure pipeline example”

git branch -M main

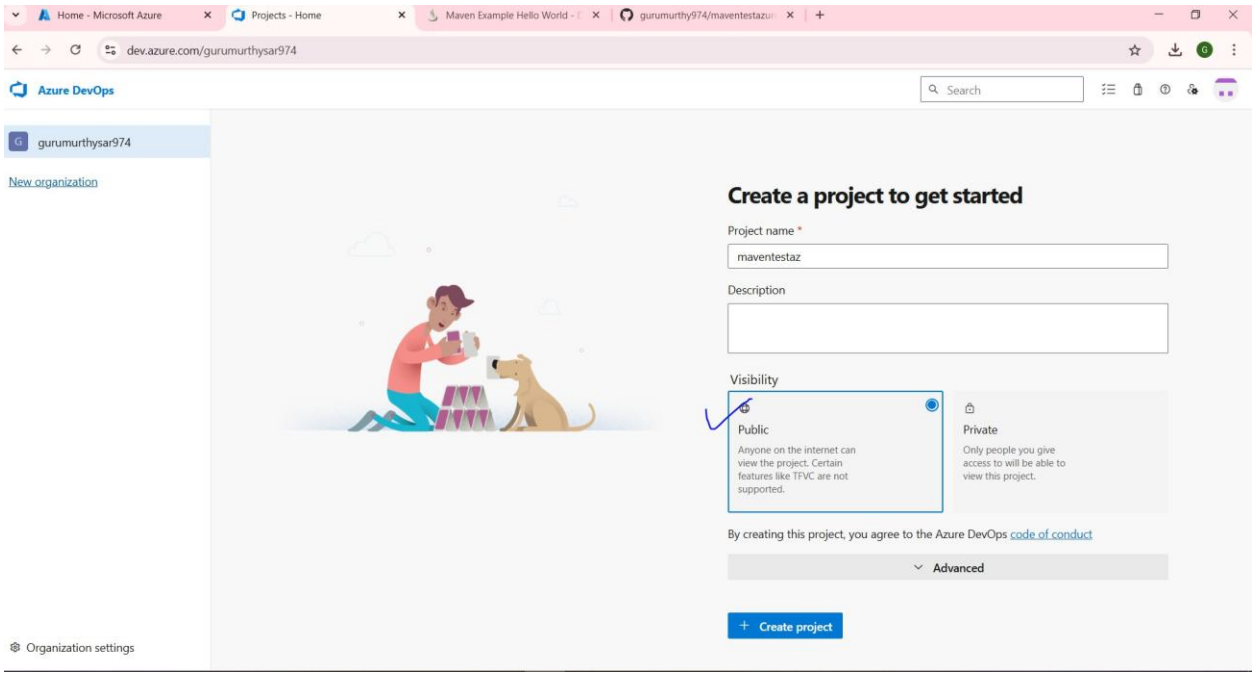
git remote add origin https://github.com/gurumurthy974/maventestazure.git

git push -u origin main

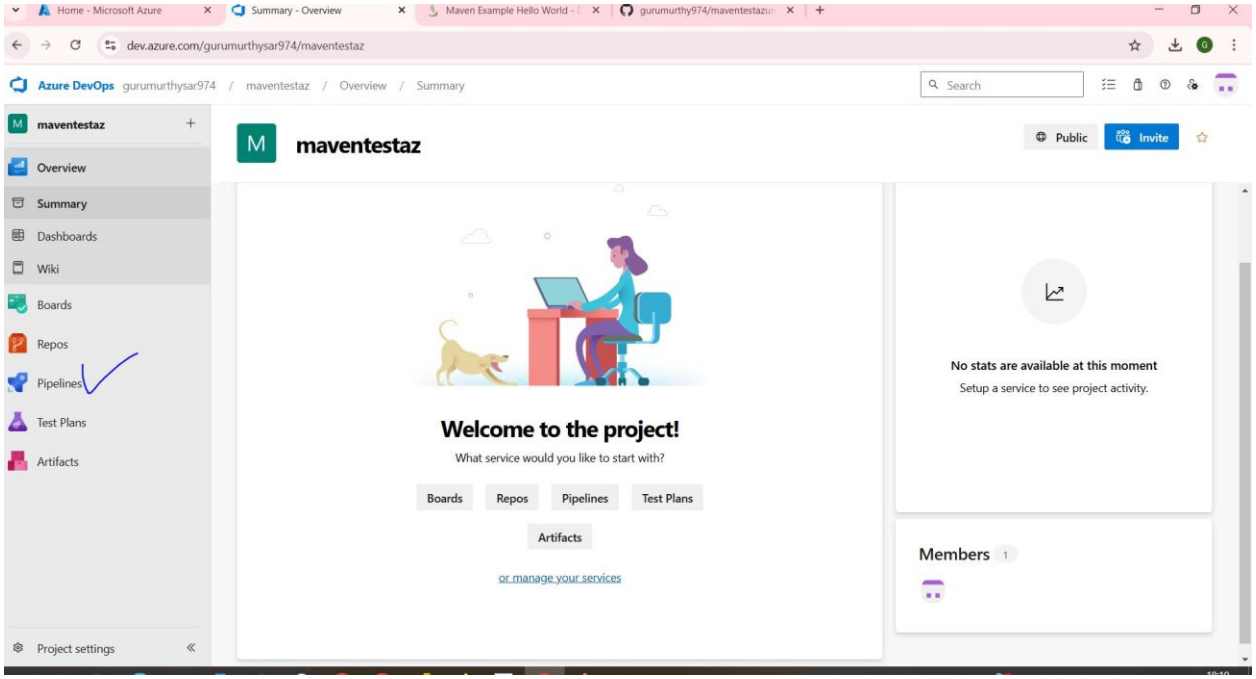
After completion of above command my repository looks



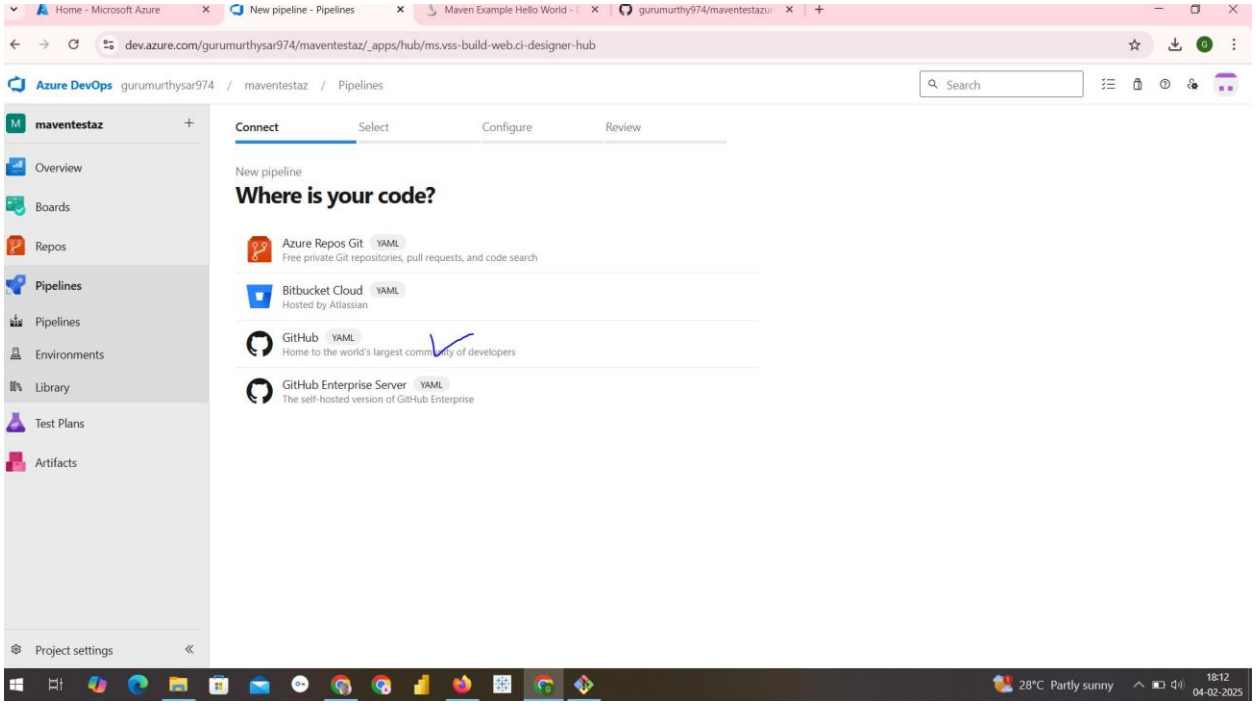
STEP4: Now goto Azure Devops Organization create Public Project



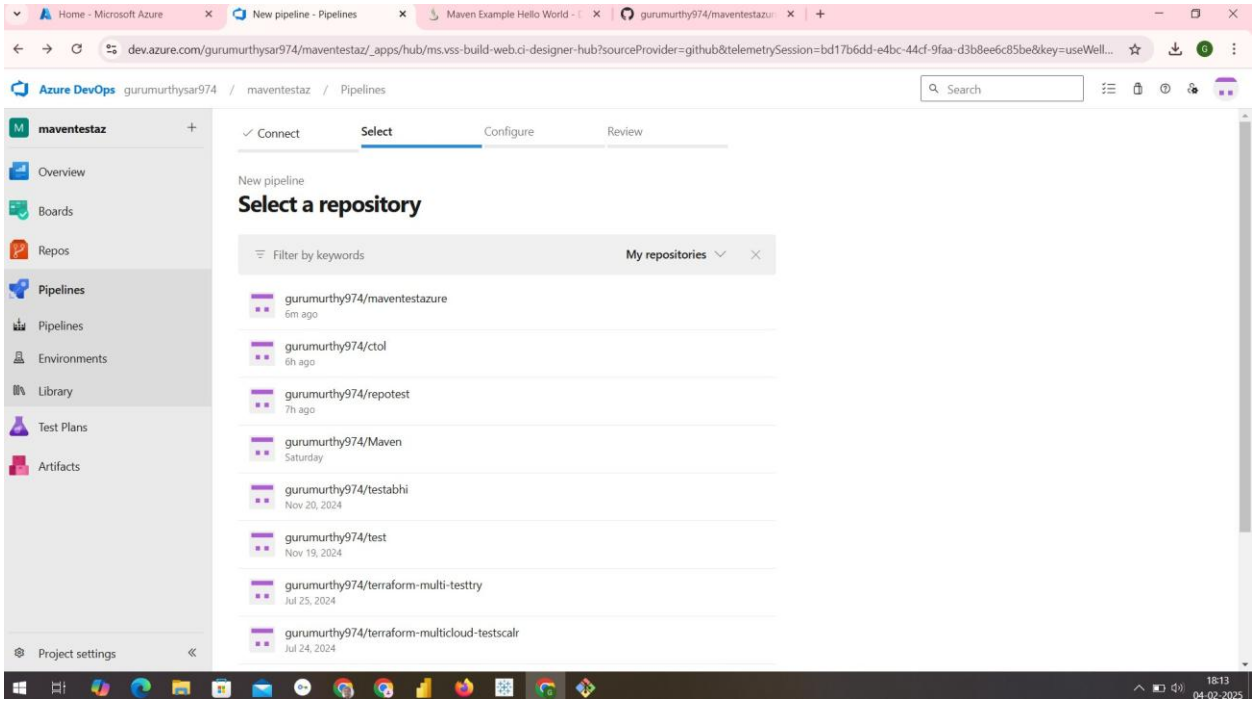
STEP5: SELECT PIPELINE and then click on create pipeline



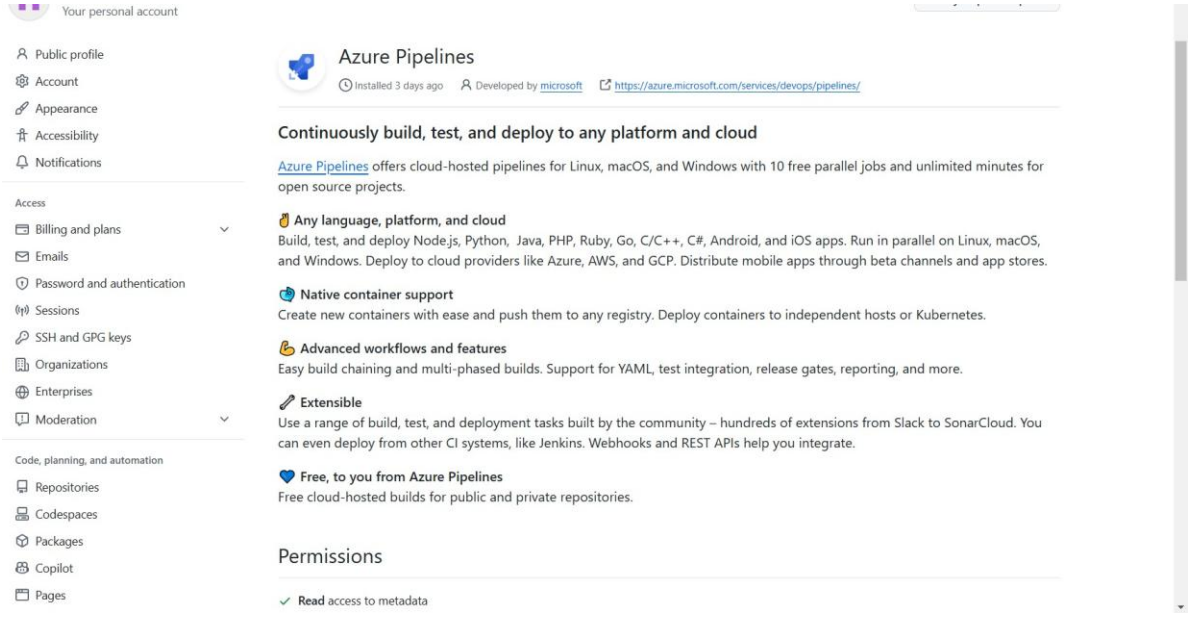
STEP6: After creating Pipeline select type of repo as Github



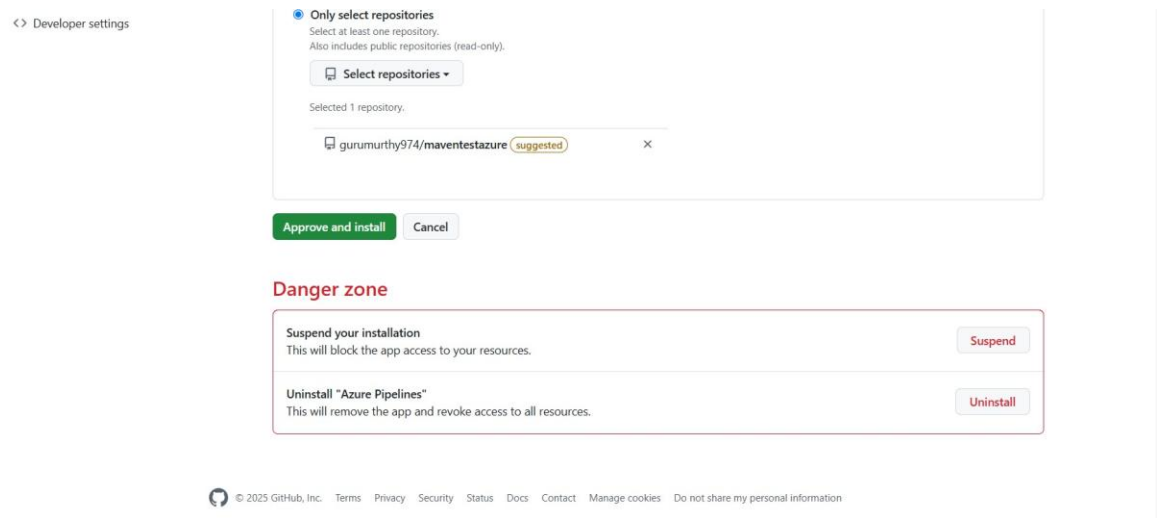
STEP7: It asks for minimum signin verification after that ur screen be as in below select required repository there to run maven project in my case its maventest123



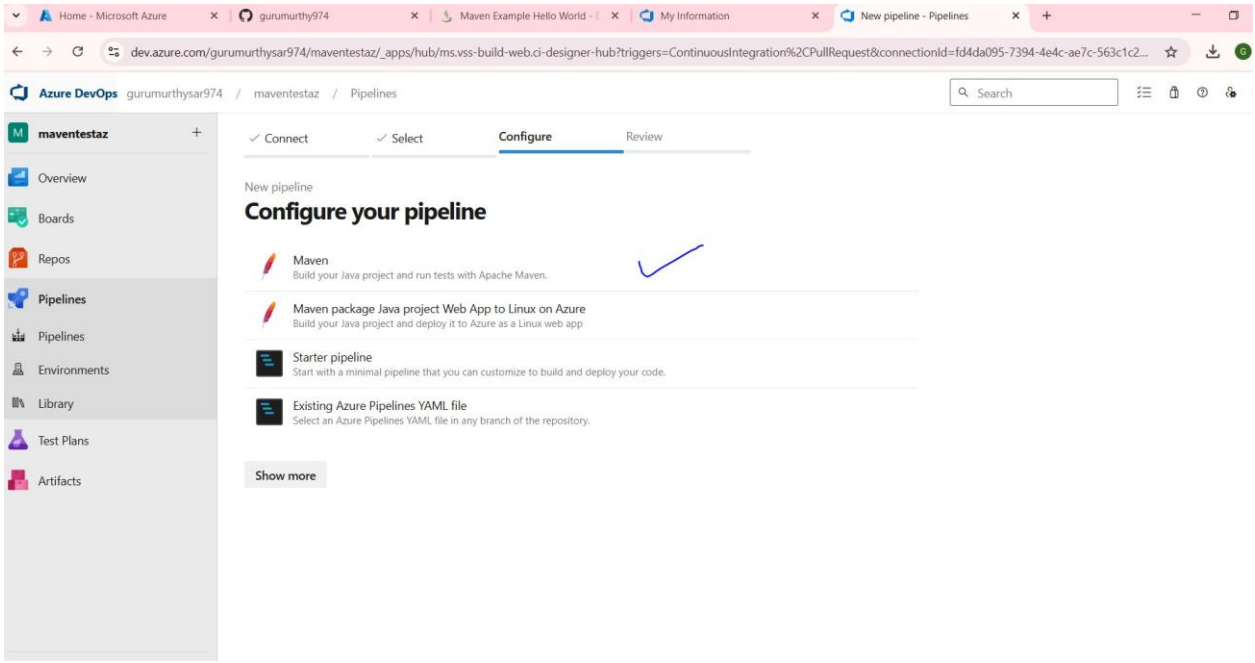
STEP8: AFTER REQUIRED REPO IS SELECTED the screen be as in below



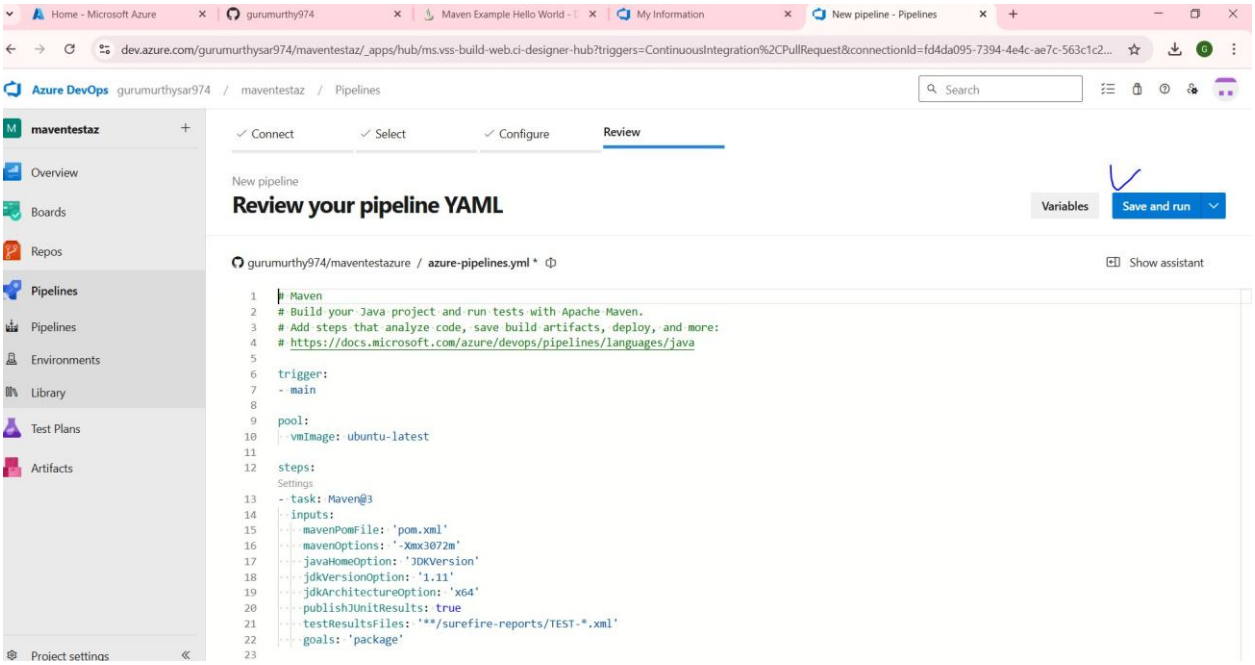
Drag the screen down check once again the selected repository is correct or not then click on Approve and Install



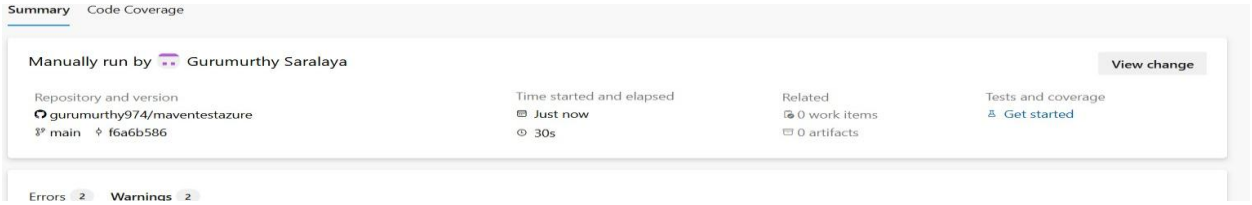
STEP9: It again verifies signin verification of microsoft account
You be able to see starter pipeline select for Maven

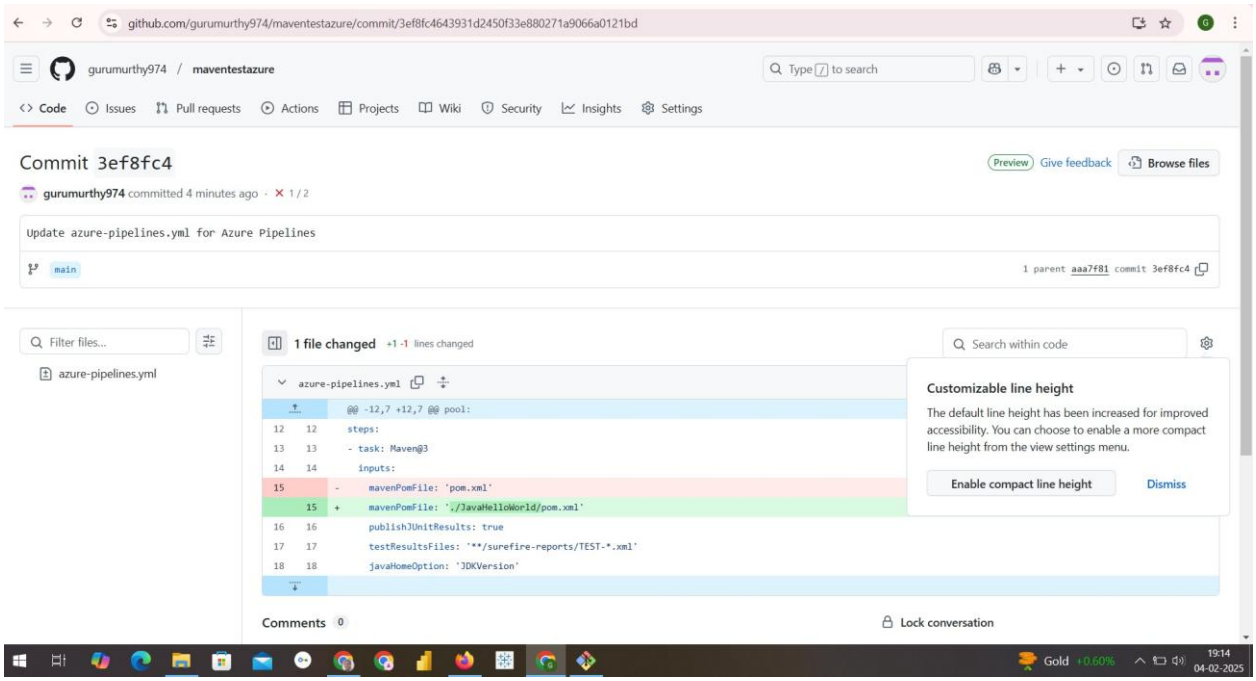


After selecting maven it asks for save and run just click on it



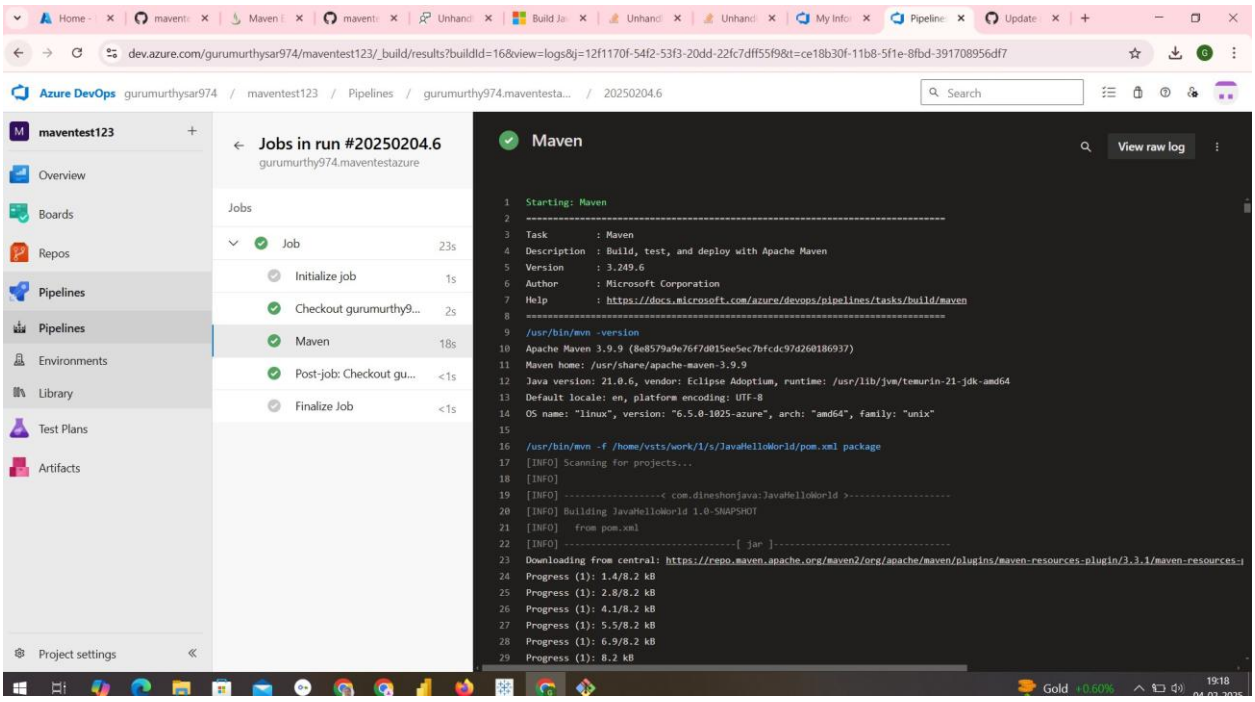
Finally You be able to see tasks running its failed bec we shld mention proper path For pom.xml





The commits will also be visible in github

WE can download and also see individual Raw Log Reports

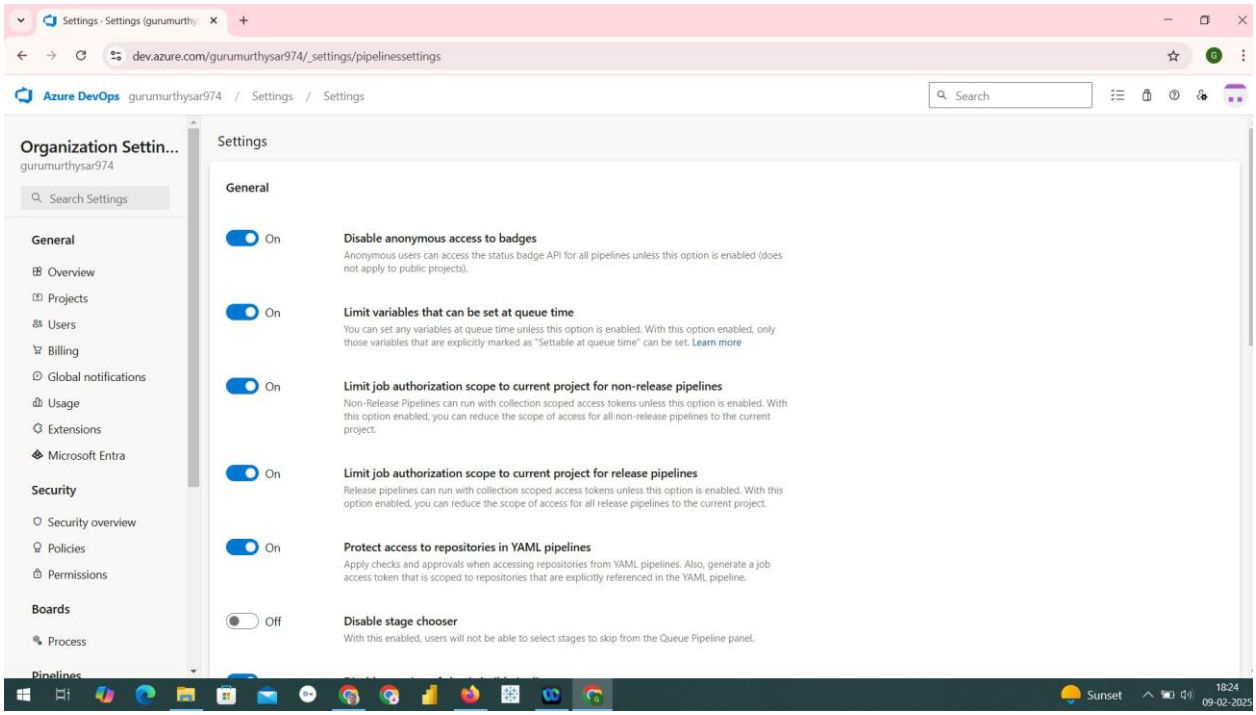


If pipeline permission error comes please complete the registration form of Parallel Jobs after 2 working days(48hrs) u be able to run pipelines for private projects same happens to be for

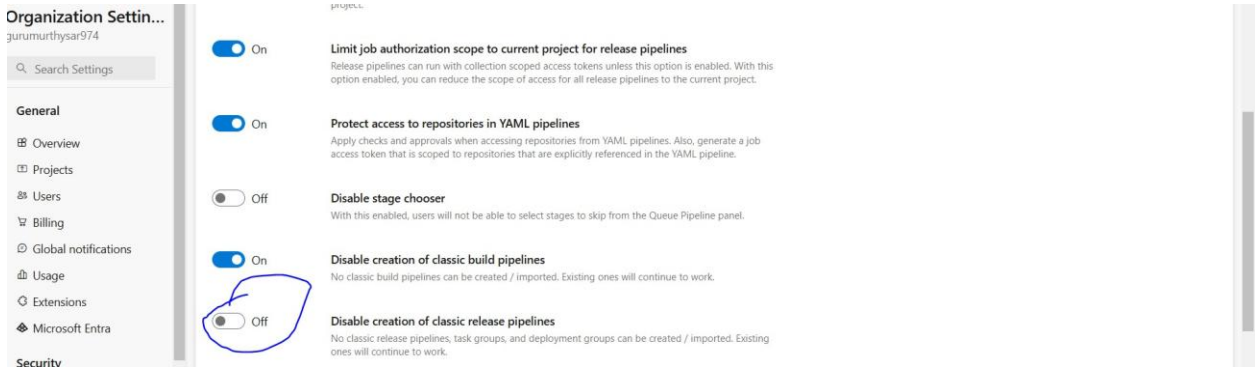
public Projects.

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.

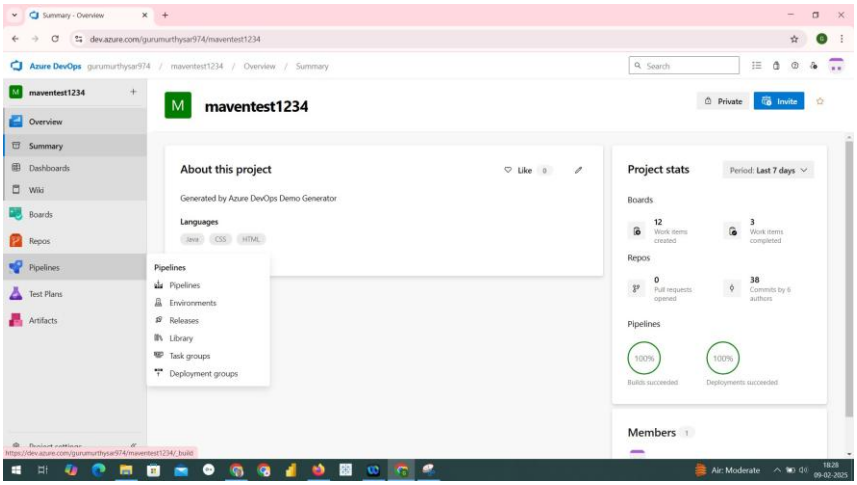
STEP1: Click on Organization setting and click on Pipeline Settings You get screen as in below



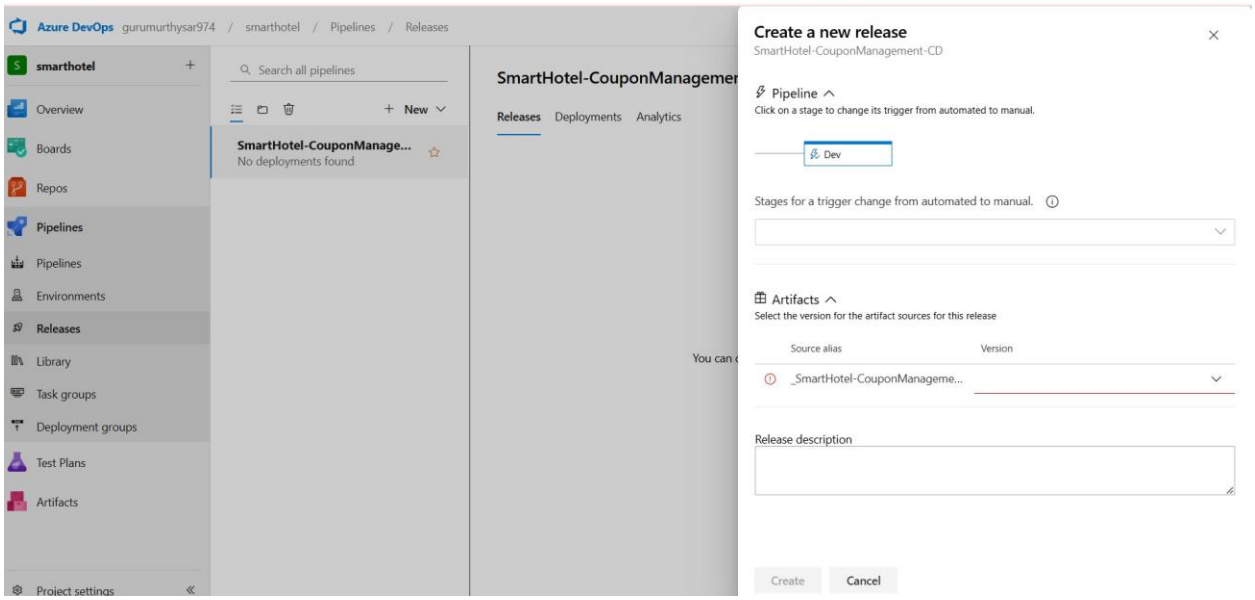
STEP2: Off the Disable creation of classic pipeline



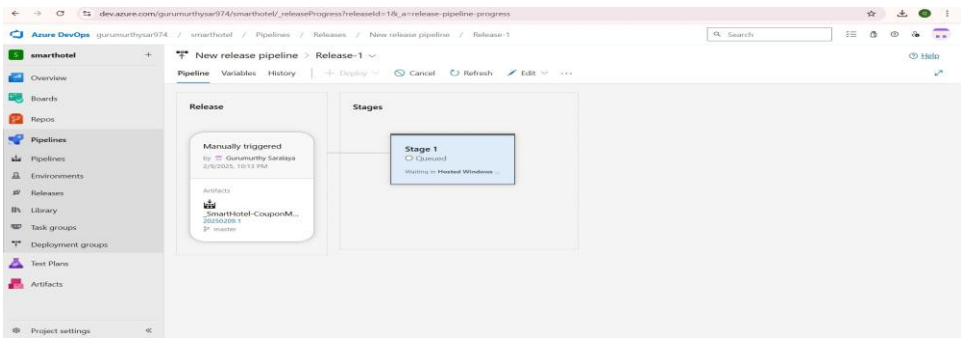
STEP3: NOw you be able to see the visibility of Release for any pipeline creation as in screen below.

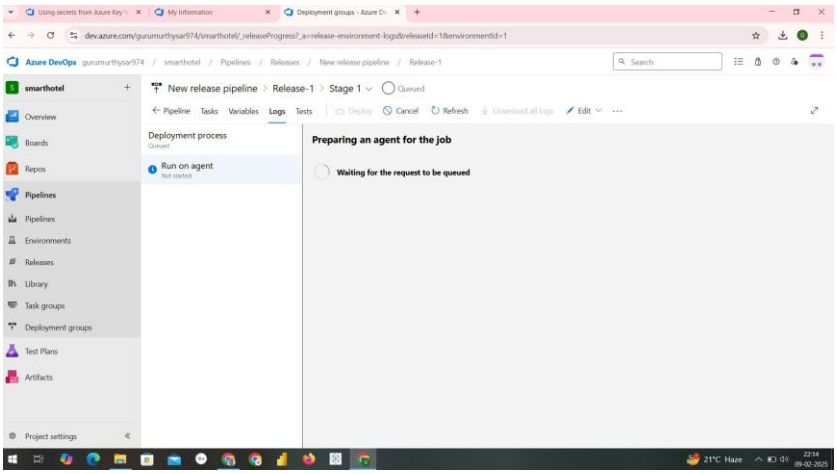


STEP4: YOu can run simple test plans

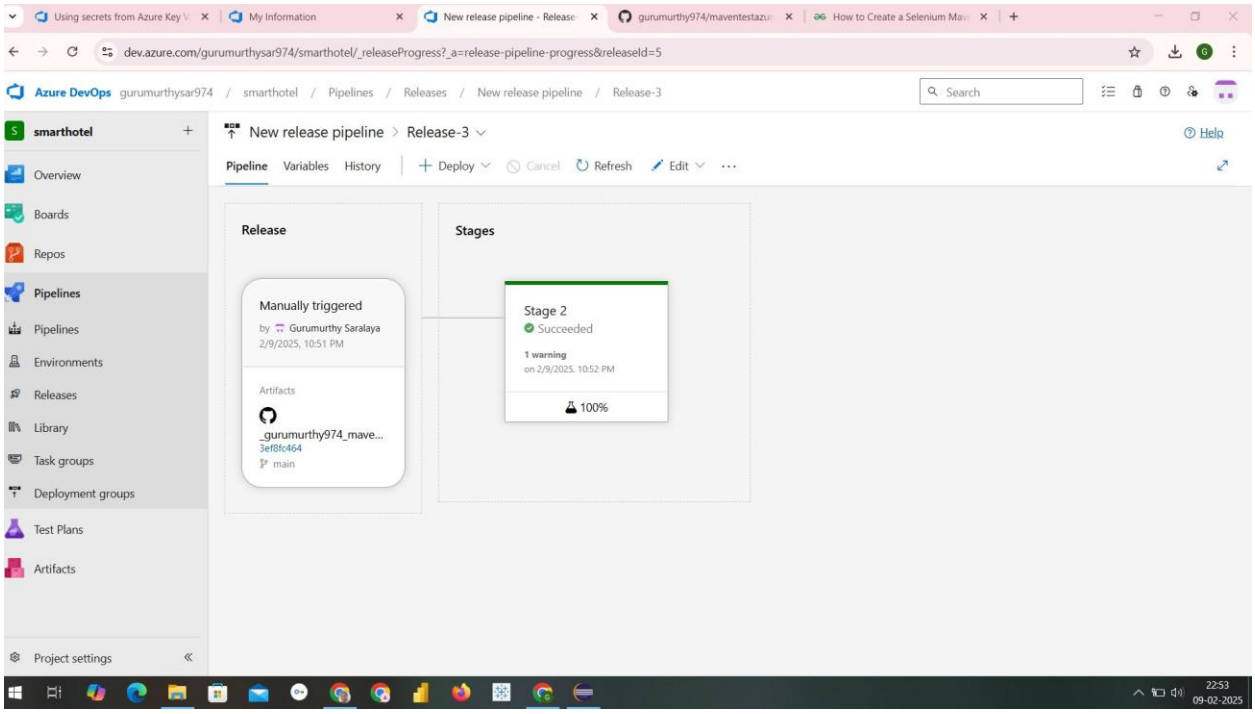


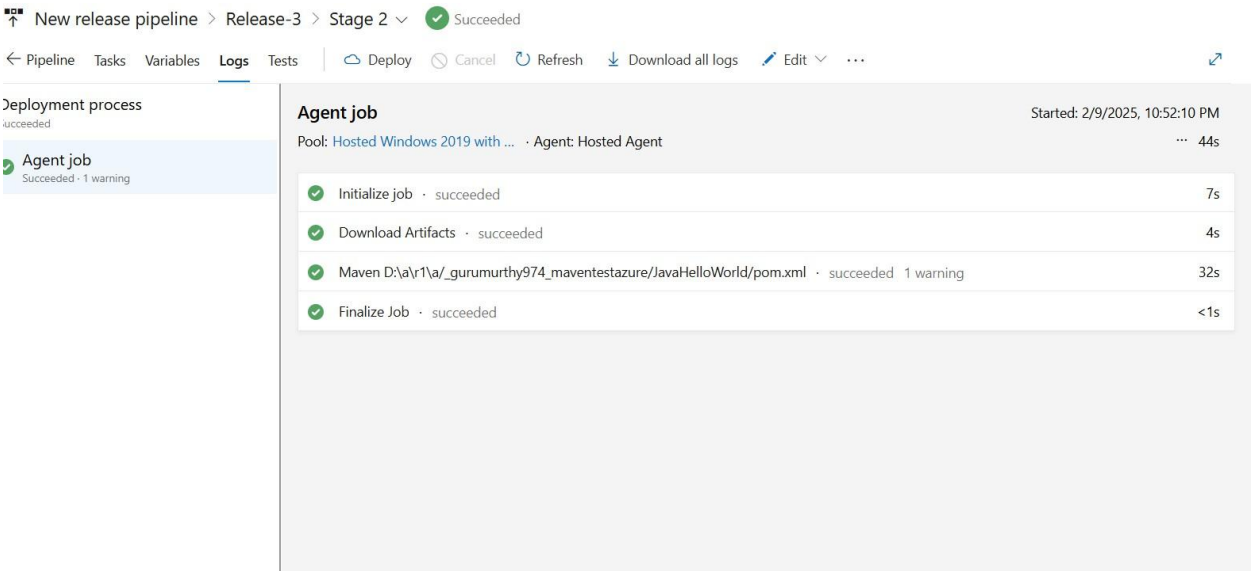
We can build tasks and run them




New release pipeline
Release-1
Stage 1
Queued
Deployment process
Run on agent
Preparing an agent for the job
Waiting for the request to be queued

W
e can


dev.azure.com/gurumurthysar974/smarthotel/_releaseProgress?_a=release-pipeline-progress&releaseId=5
Azure DevOps
gurumurthysar974
smarthotel
Pipelines
Releases
New release pipeline
Release-3
Search
New release pipeline
Release-3
Help
Pipeline
Variables
History
Deploy
Cancel
Refresh
Edit
Release
Stages
Manually triggered
by Gurumurthy Saralaya
2/9/2025, 10:51 PM
Artifacts
_gurumurthy974_mave...
3ef8fc464
main
Stage 2
Succeeded
1 warning
on 2/9/2025, 10:52 PM
100%
Overview
Boards
Repos
Pipelines
Environments
Releases
Library
Task groups
Deployment groups
Test Plans
Artifacts
Project settings
22:53
09-02-2025


New release pipeline > Release-3 > Stage 2 Succeeded
Pipeline Tasks Variables Logs Tests
Deploy Cancel Refresh Download all logs Edit
Deployment process
Succeeded
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 7s
Download Artifacts · succeeded 4s
Maven D:\a\r1\a/_gurumurthy974_maventestazure/JavaHelloWorld/pom.xml · succeeded 1 warning 32s
Finalize Job · succeeded <1s