

DOL NAME : ARYA ANGANE PRN : 121A3008 BATCH : E1

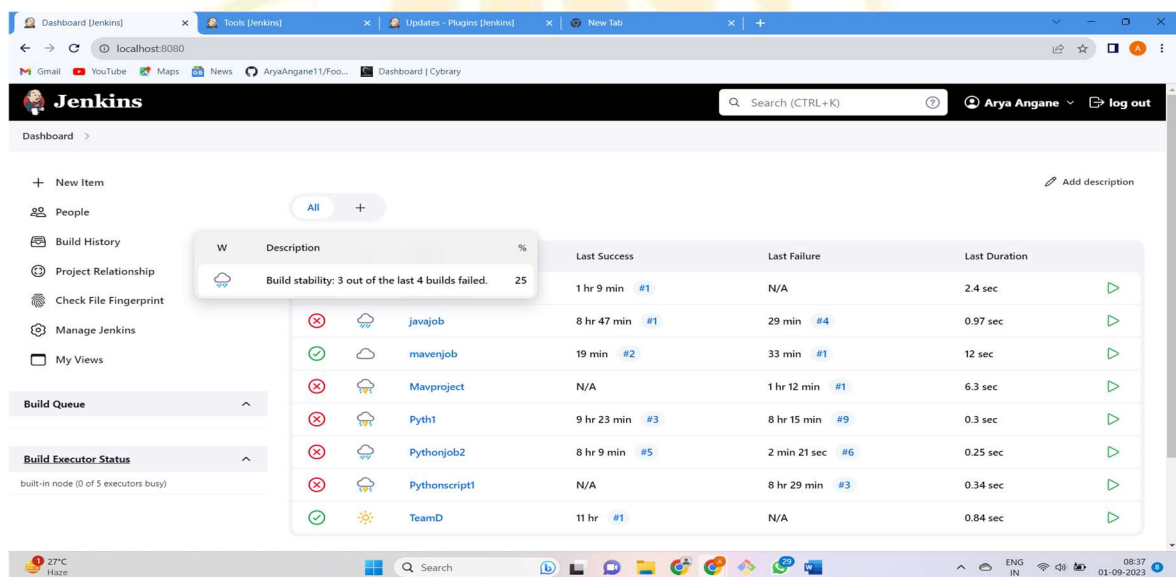
Experiment No: 5

Aim: - To set up and build a Java, Maven /Ant and Python jobs in Jenkins.

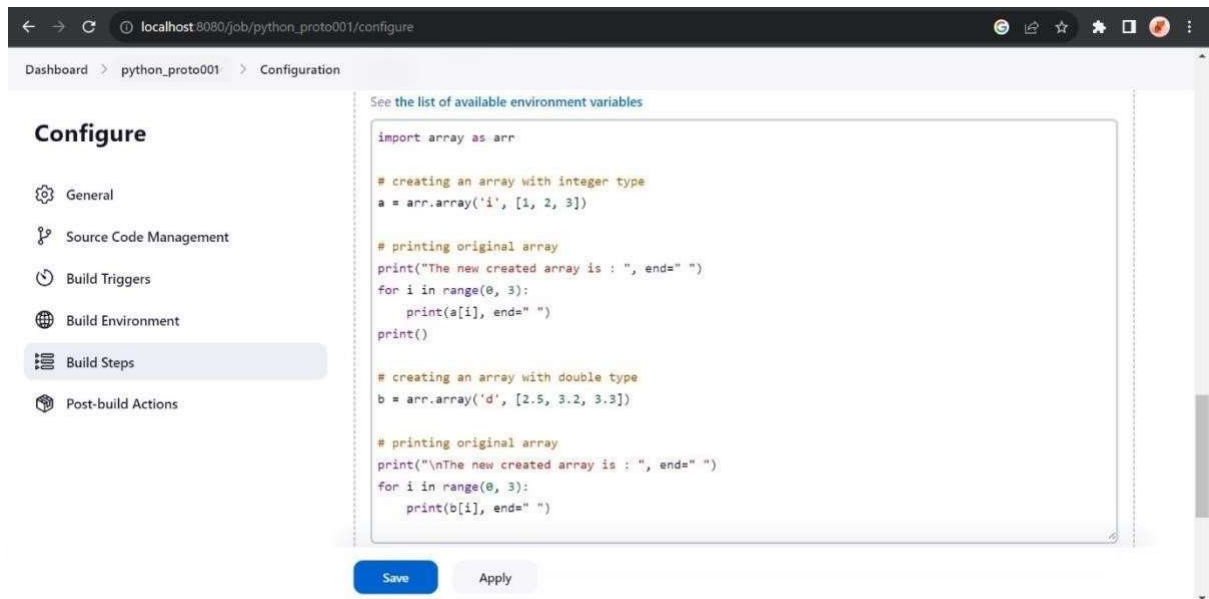
Theory: - Jenkins is an open-source automation tool written in Java with plugins built for continuous integration. Jenkins is used to build and test your software projects continuously making it easier for developers to integrate changes to the project, and making it easier for users to obtain a fresh build. It also allows you to continuously deliver your software by integrating with a large number of testing and deployment technologies. With Jenkins, organizations can accelerate the software development process through automation. Jenkins integrates development life-cycle processes of all kinds, including build, document, test, package, stage, deploy, static analysis, and much more.

Jenkins achieves Continuous Integration with the help of plugins. Plugins allow the integration of Various DevOps stages. If you want to integrate a particular tool, you need to install the plugins for that tool. For example Git, Maven 2 project, Amazon EC2, HTML publisher etc.

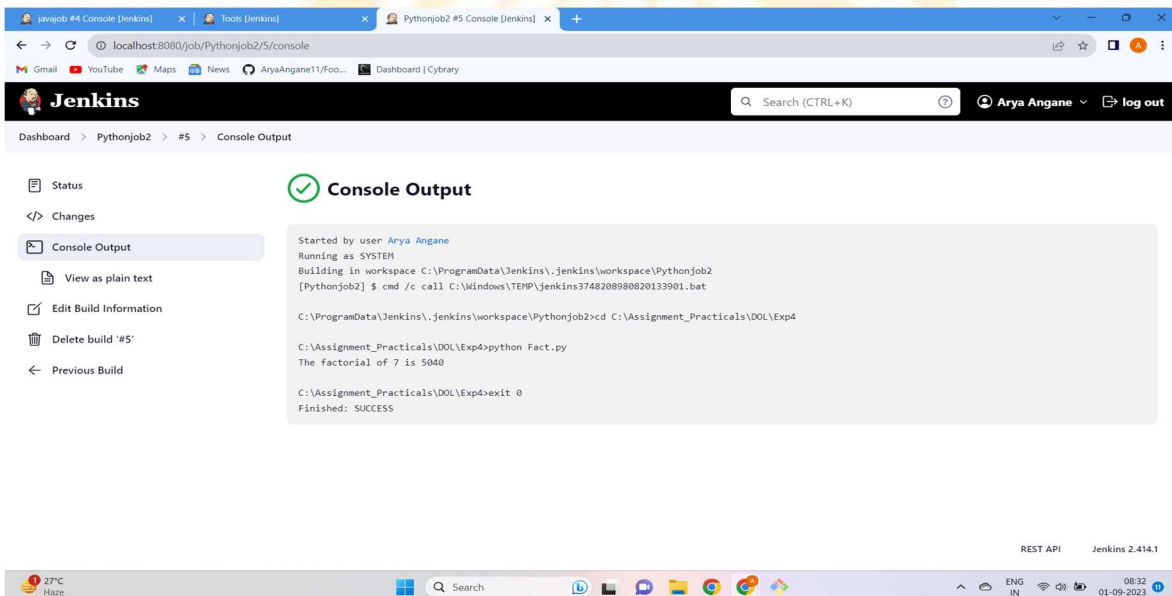
Install Jenkins: Download and install Jenkins on your server or machine.



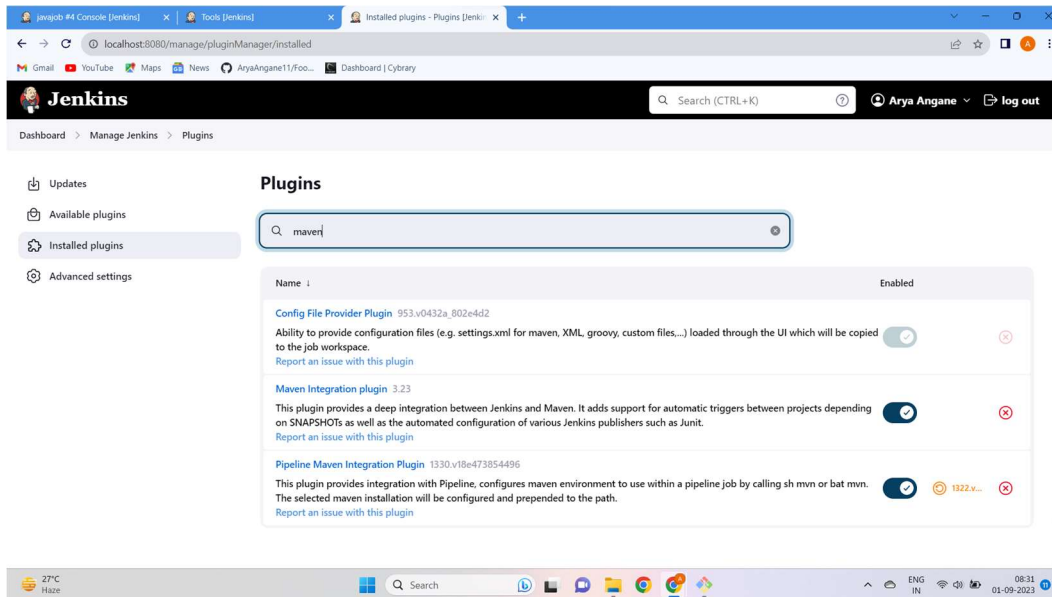
Writing a python script in the job: Inside the configure section, set the python script to be executed.



Executing the python script in Jenkins:



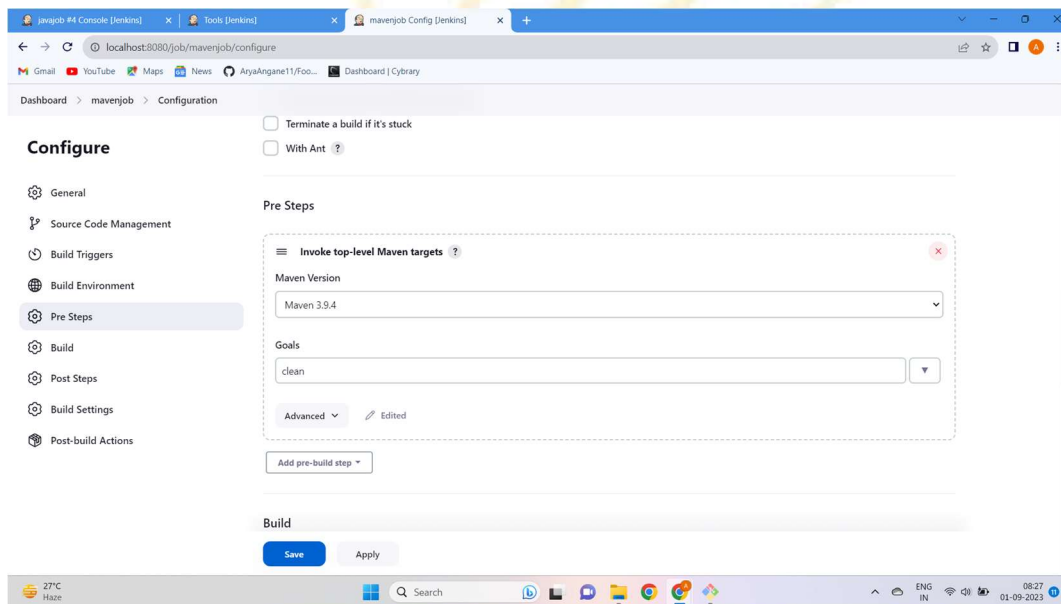
Install Maven Plugin: Once Jenkins is up and running, navigate to the "Manage Jenkins" > "Manage Plugins" section.



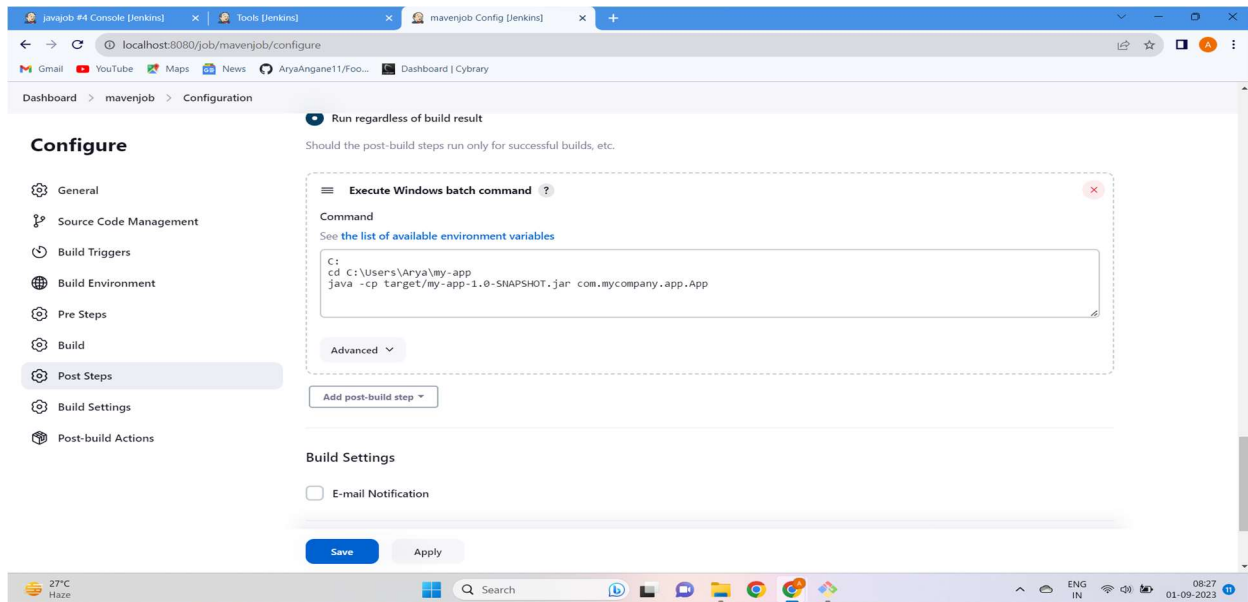
Create a Java project: Inside the local machine of user create an entire project which would be executed on maven.

```
Command Prompt
C:\Users\Arya\my-app>mvn package
[INFO] Scanning for projects...
[INFO]
[INFO] -----< com.mycompany.app:my-app >-----
[INFO] Building my-app 1.0-SNAPSHOT
[INFO] from pom.xml
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- resources:3.0.2:resources (default-resources) @ my-app ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory C:\Users\Arya\my-app\src\main\resources
[INFO]
[INFO] --- compiler:3.8.0:compile (default-compile) @ my-app ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- resources:3.0.2:testResources (default-testResources) @ my-app ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory C:\Users\Arya\my-app\src\test\resources
[INFO]
[INFO] --- compiler:3.8.0:testCompile (default-testCompile) @ my-app ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- surefire:2.22.1:test (default-test) @ my-app ---
[INFO]
[INFO] -----
[INFO] T E S T S
[INFO] -----
[INFO] Running com.mycompany.app.AppTest
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.015 s - in com.mycompany.app.AppTest
```

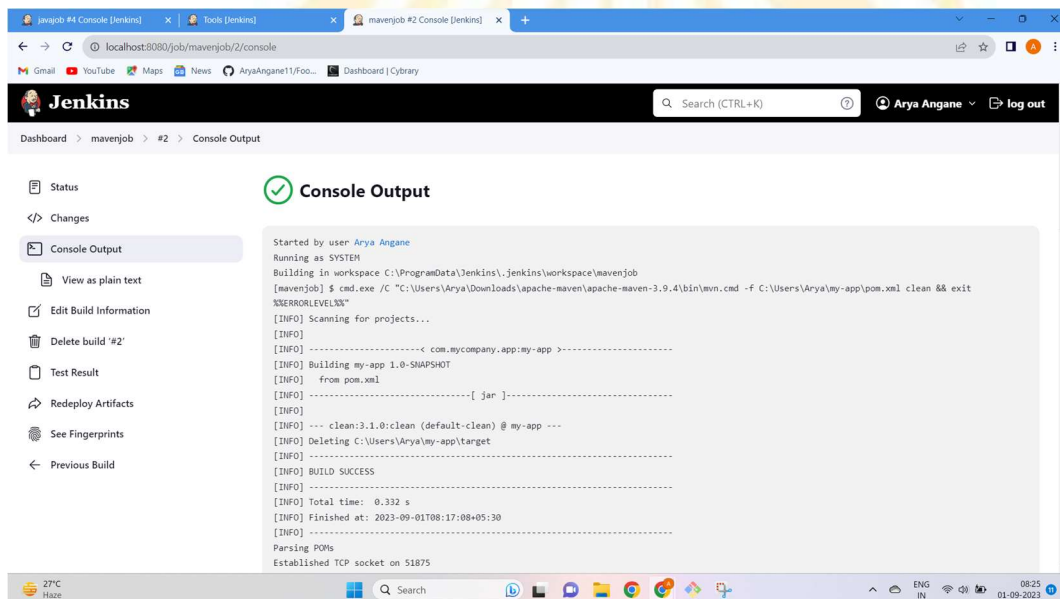
Create a New Jenkins Job: From the Jenkins dashboard, click on "New Item" to create a new job.



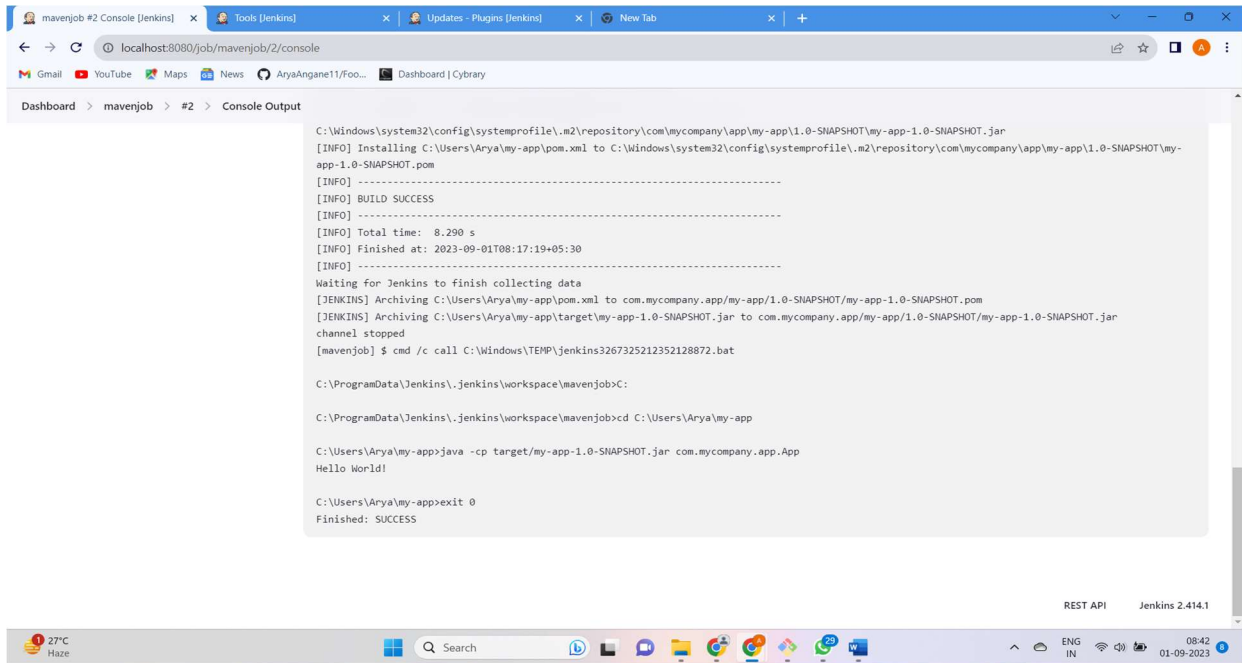
Configure Build Steps: Under the "Build" section, click on "Add build step" and select "Invoke top-level Maven targets."



Save and Run: Save your job configuration and manually trigger the build by clicking on "Build Now" on the job's dashboard. Jenkins will now clone your repository, run the specified Maven goals, and execute the build steps.



View Build Results: Once the build is completed, you can view the build results and console output to see if the Maven build was successful.



The screenshot shows a web browser window displaying the Jenkins console output for a build named 'mavenjob #2'. The browser's address bar shows 'localhost:8080/job/mavenjob/2/console'. The console output is as follows:

```
C:\Windows\system32\config\systemprofile\.m2\repository\com\mycompany\app\my-app\1.0-SNAPSHOT\my-app-1.0-SNAPSHOT.jar
[INFO] Installing C:\Users\Arya\my-app\pom.xml to C:\Windows\system32\config\systemprofile\.m2\repository\com\mycompany\app\my-app\1.0-SNAPSHOT\my-app-1.0-SNAPSHOT.pom
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 8.290 s
[INFO] Finished at: 2023-09-01T08:17:19+05:30
[INFO] -----
Waiting for Jenkins to finish collecting data
[JENKINS] Archiving C:\Users\Arya\my-app\pom.xml to com.mycompany.app\my-app\1.0-SNAPSHOT\my-app-1.0-SNAPSHOT.pom
[JENKINS] Archiving C:\Users\Arya\my-app\target\my-app-1.0-SNAPSHOT.jar to com.mycompany.app\my-app\1.0-SNAPSHOT\my-app-1.0-SNAPSHOT.jar
channel stopped
[mavenjob] $ cmd /c call C:\Windows\TEMP\jenkins3267325212352128872.bat

C:\ProgramData\Jenkins\jenkins\workspace\mavenjob>C:

C:\ProgramData\Jenkins\jenkins\workspace\mavenjob>cd C:\Users\Arya\my-app

C:\Users\Arya\my-app>java -cp target\my-app-1.0-SNAPSHOT.jar com.mycompany.app.App
Hello World!

C:\Users\Arya\my-app>exit 0
Finished: SUCCESS
```

The bottom of the screenshot shows a Windows taskbar with a search bar, task icons, and system tray information including temperature (27°C), time (08:42), and date (01-09-2023).

Conclusion: In conclusion, setting up and building Java, Maven/Ant, and Python jobs in Jenkins involves a series of steps that streamline the continuous integration and delivery process for software projects. By integrating Jenkins with these technologies, development teams can automate the build, test, and deployment phases, leading to more efficient and reliable software development.