

Continuous Integration & Continuous Delivery

(CSDV3109)

Submitted to: **Dr**. **Hitesh Kumar Sharma**

Name: **Pulkit** **Heda**

SAP ID: **500095602**

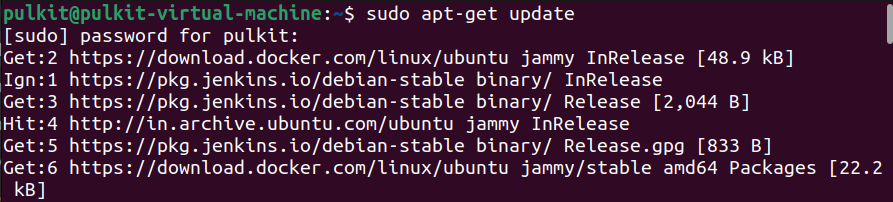
Batch: **BTech** **CSE** **DevOps**-**B3**

INDEX:

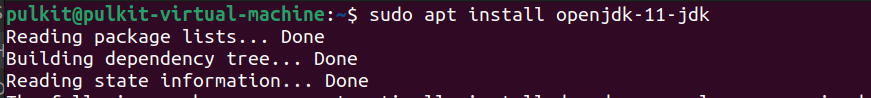
|  |  |  |  |
| --- | --- | --- | --- |
| Sno. | Experiment | Date | Sign |
| 1. | Installation Of Jenkins |  |  |
| 2. | Setting Up Jenkins |  |  |
| 3. | GitHub Actions |  |  |
| 4. | GitHub Actions with Docker |  |  |
| 5. | Master and Slaves |  |  |
| 6. |  |  |  |
| 7. |  |  |  |

***Experiment-1***

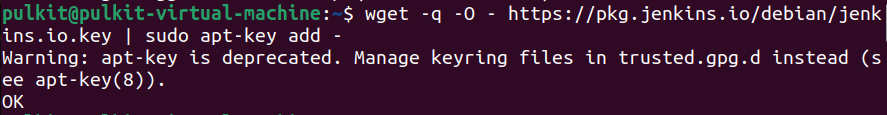
1. Install all the updates

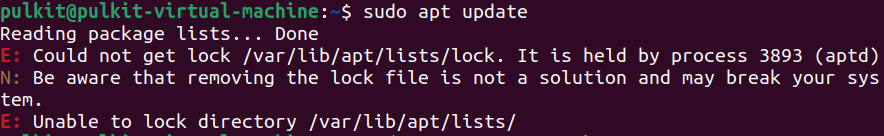


1. Install Java version 11

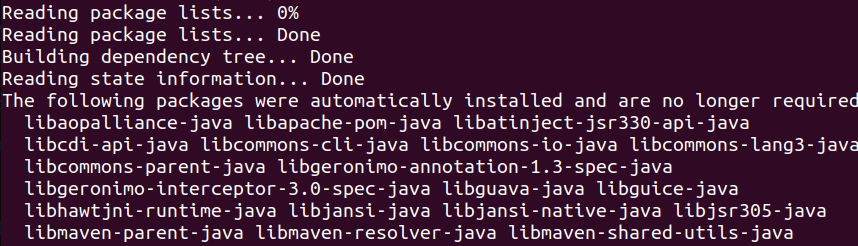


1. Install Jenkins

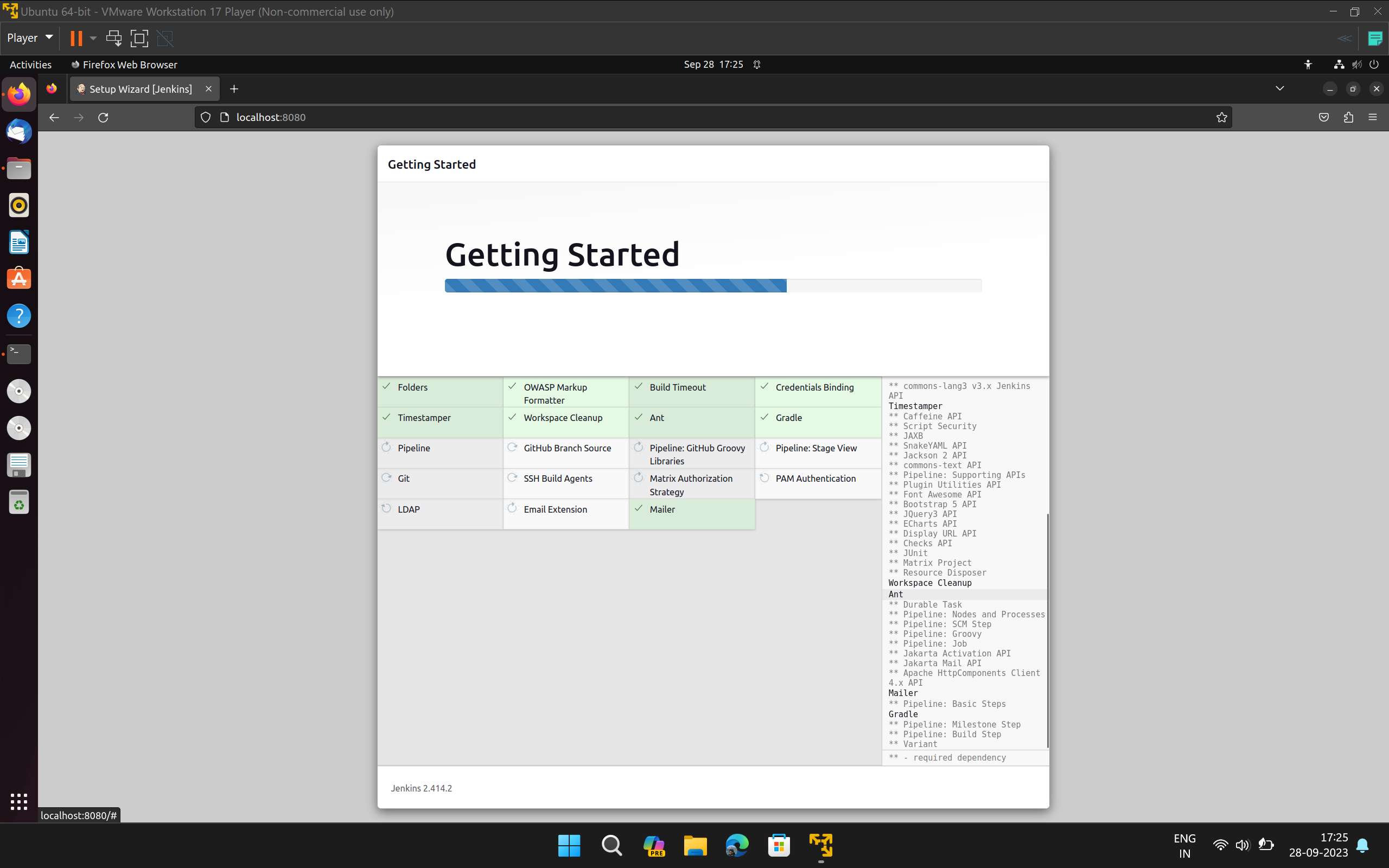


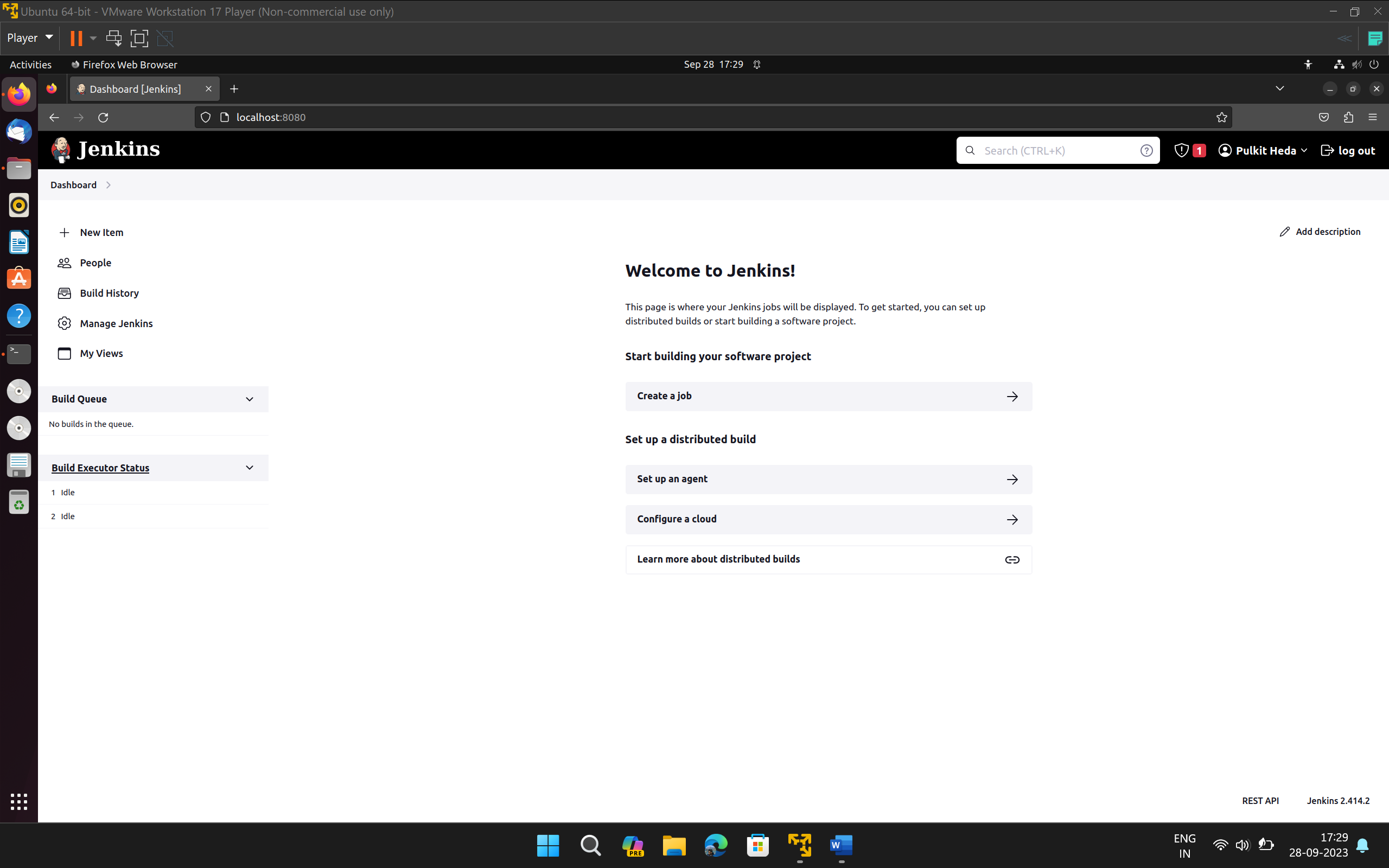




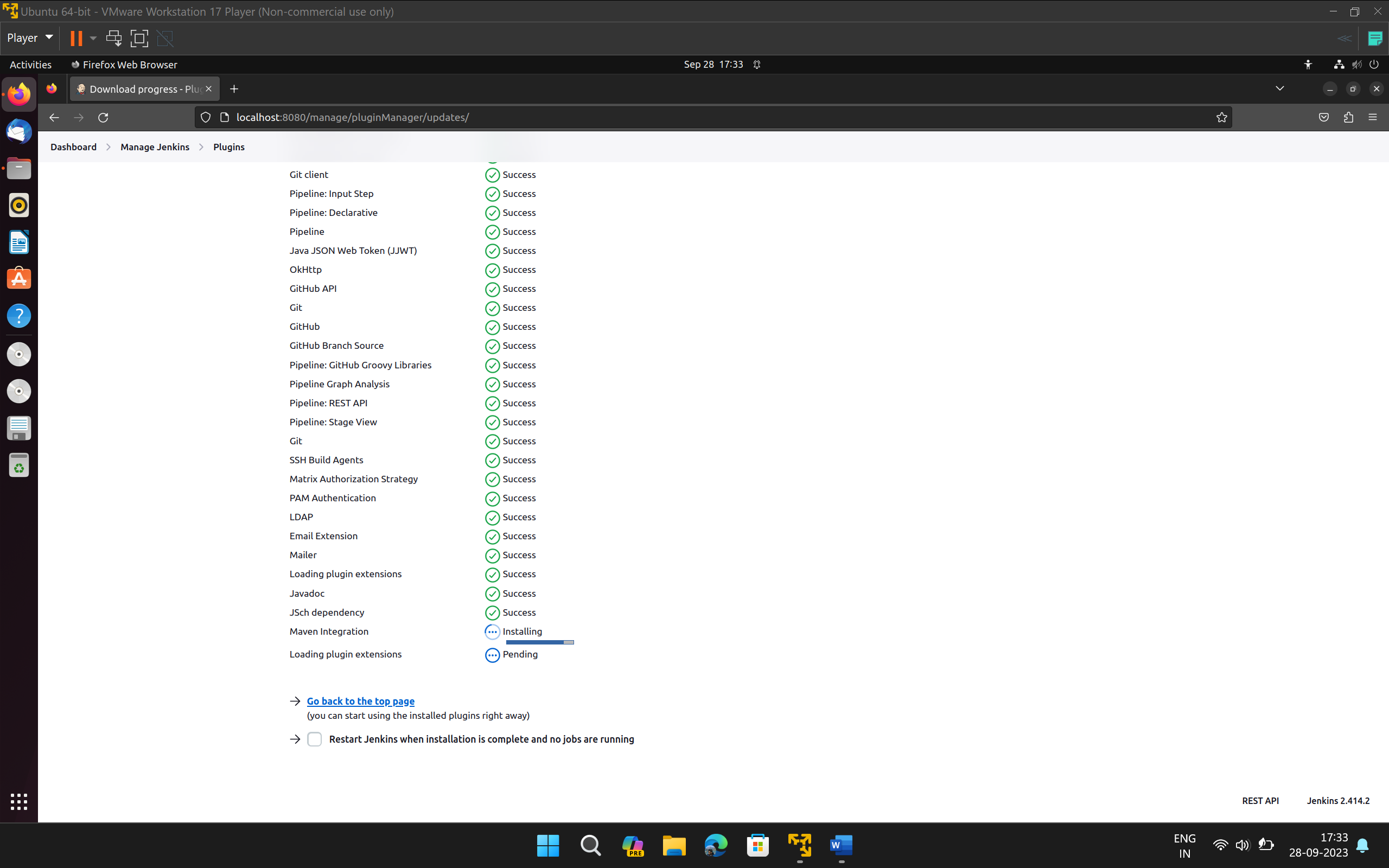


1. Jenkins Plugins installation and Homepage



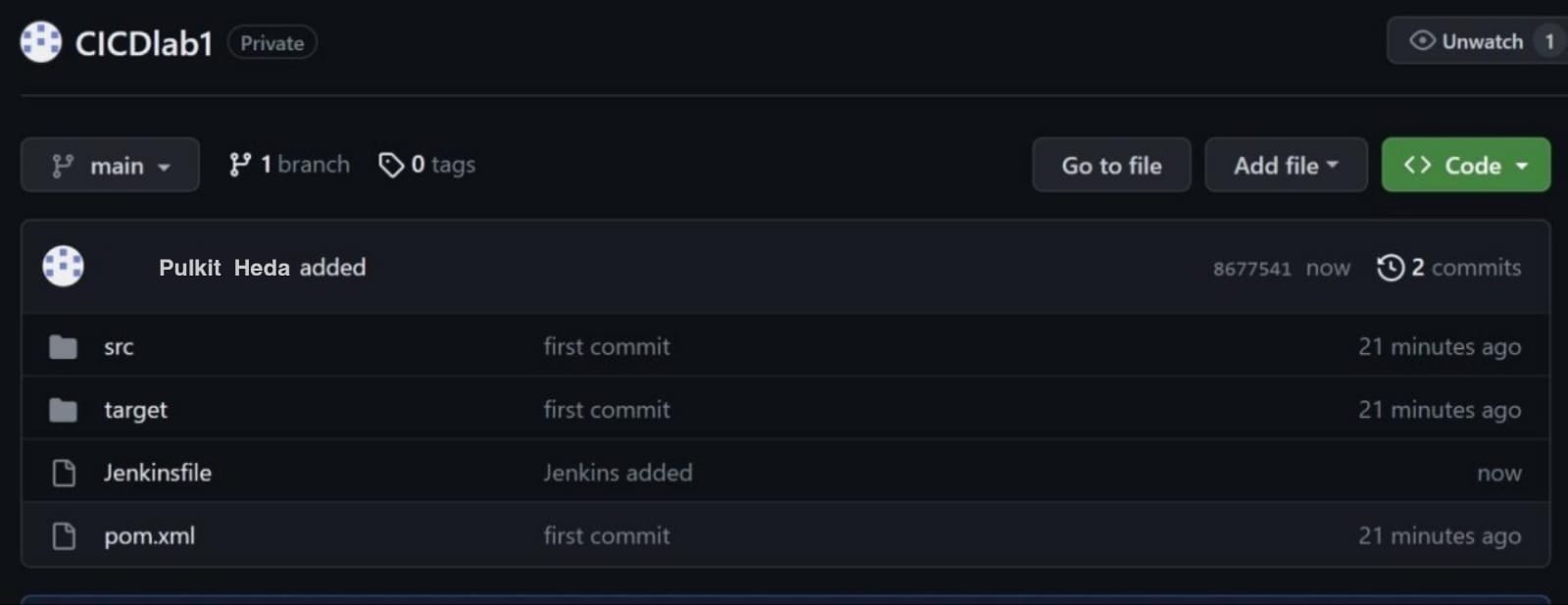


1. Install Maven Plugin.

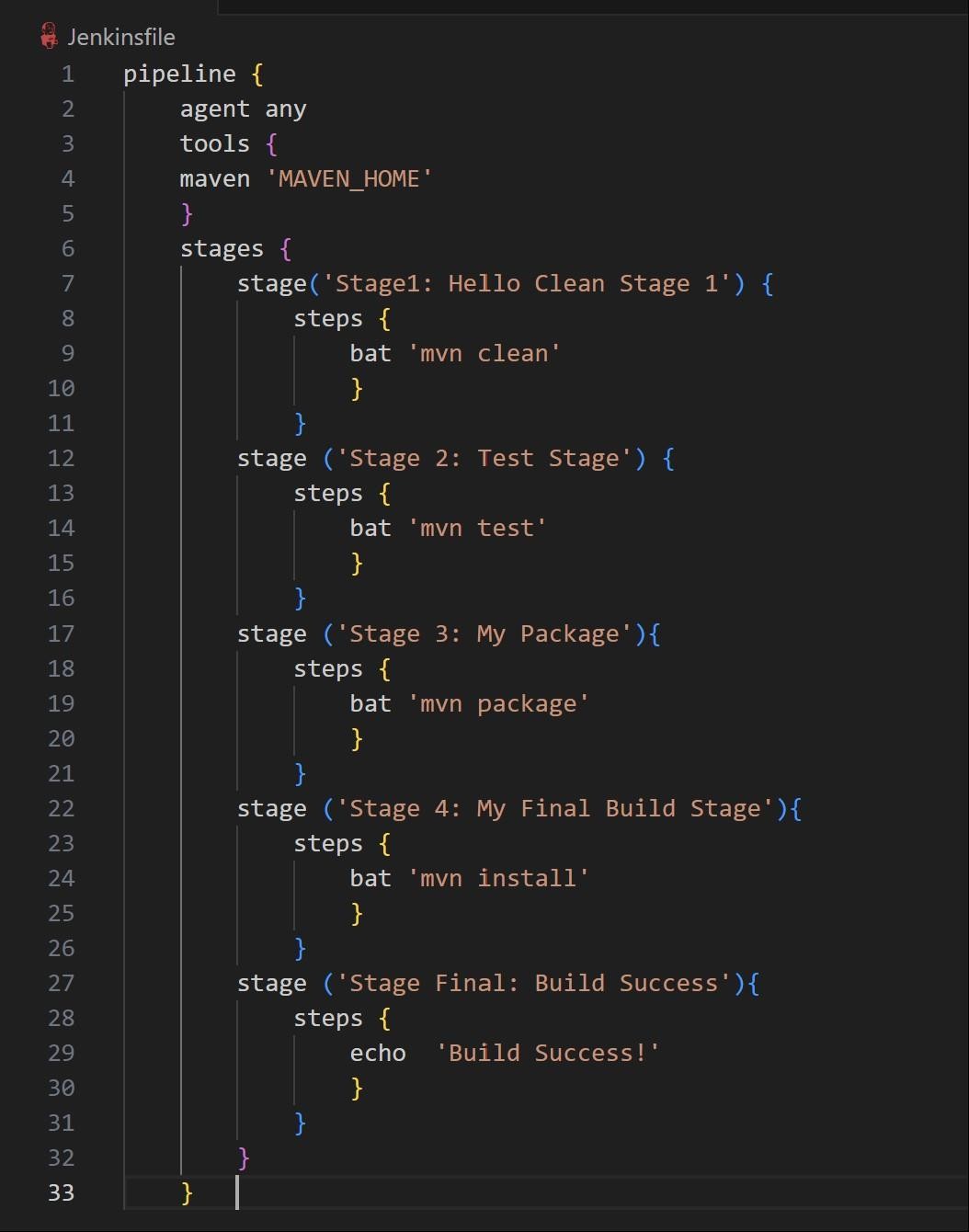


***Experiment-2***

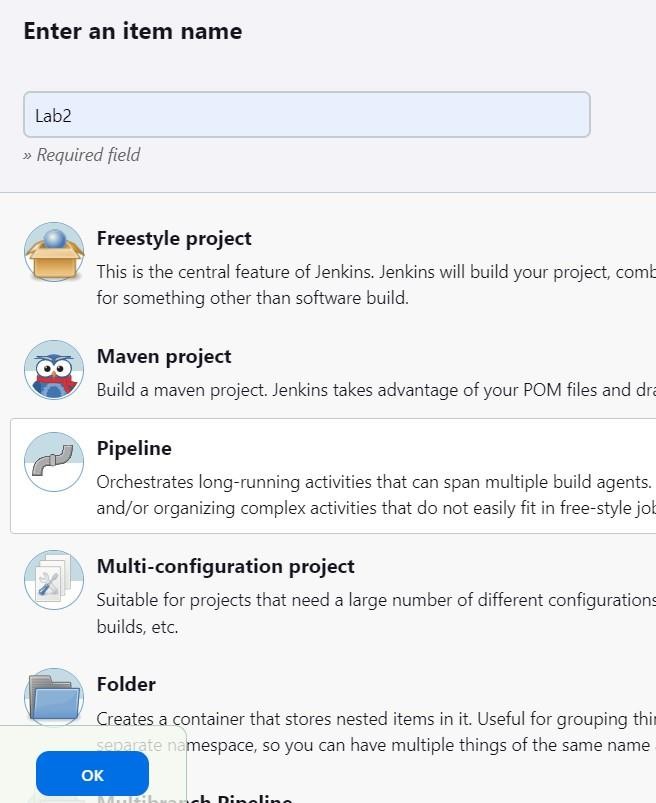
1. Create a simple maven Project (the same one used in Experiment -1) and push the project to GitHub.



1. Create a Jenkins file and push it to GitHub repo (in the root dir) Note: the following Jenkins file includes stages associated with maven project



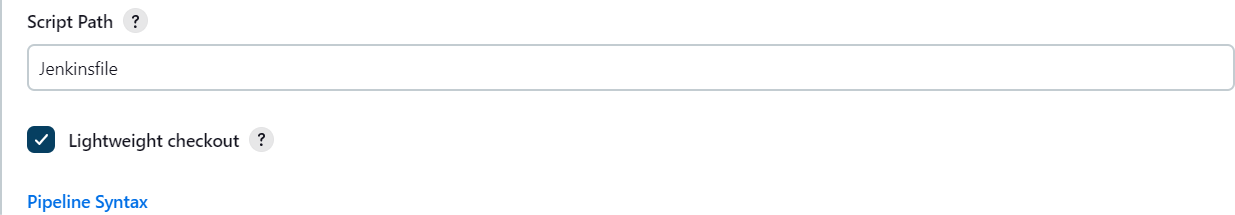
1. Create a Jenkins Pipeline CONFIGURATION:



1. Select pipeline Script from SCM (git) and provide GitHub repo URL.



1. Provide the Jenkins file path (path in the repo) in the Script Path.

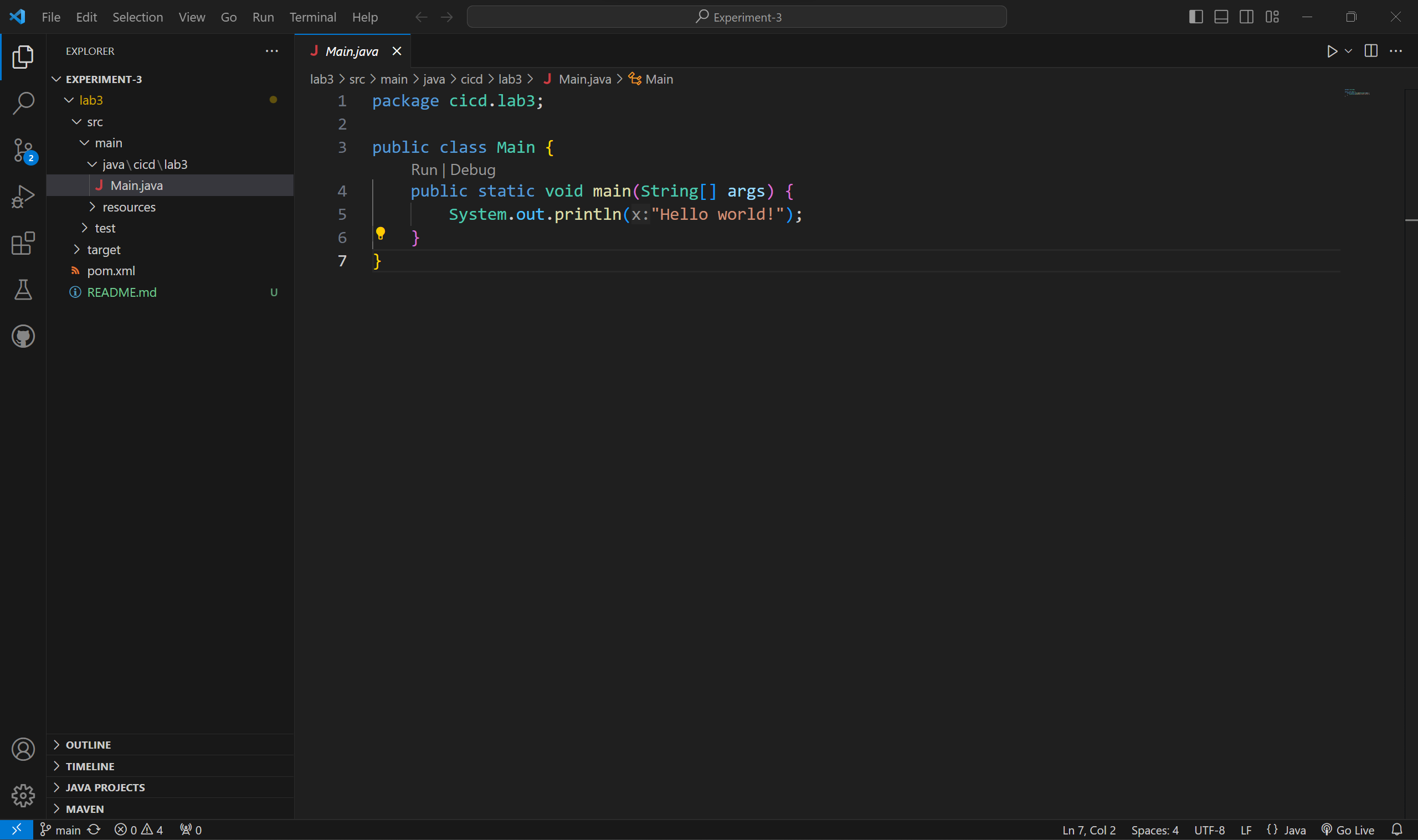


1. Now if we make any changes to the GitHub repo and push it, it will trigger a new pipeline build.

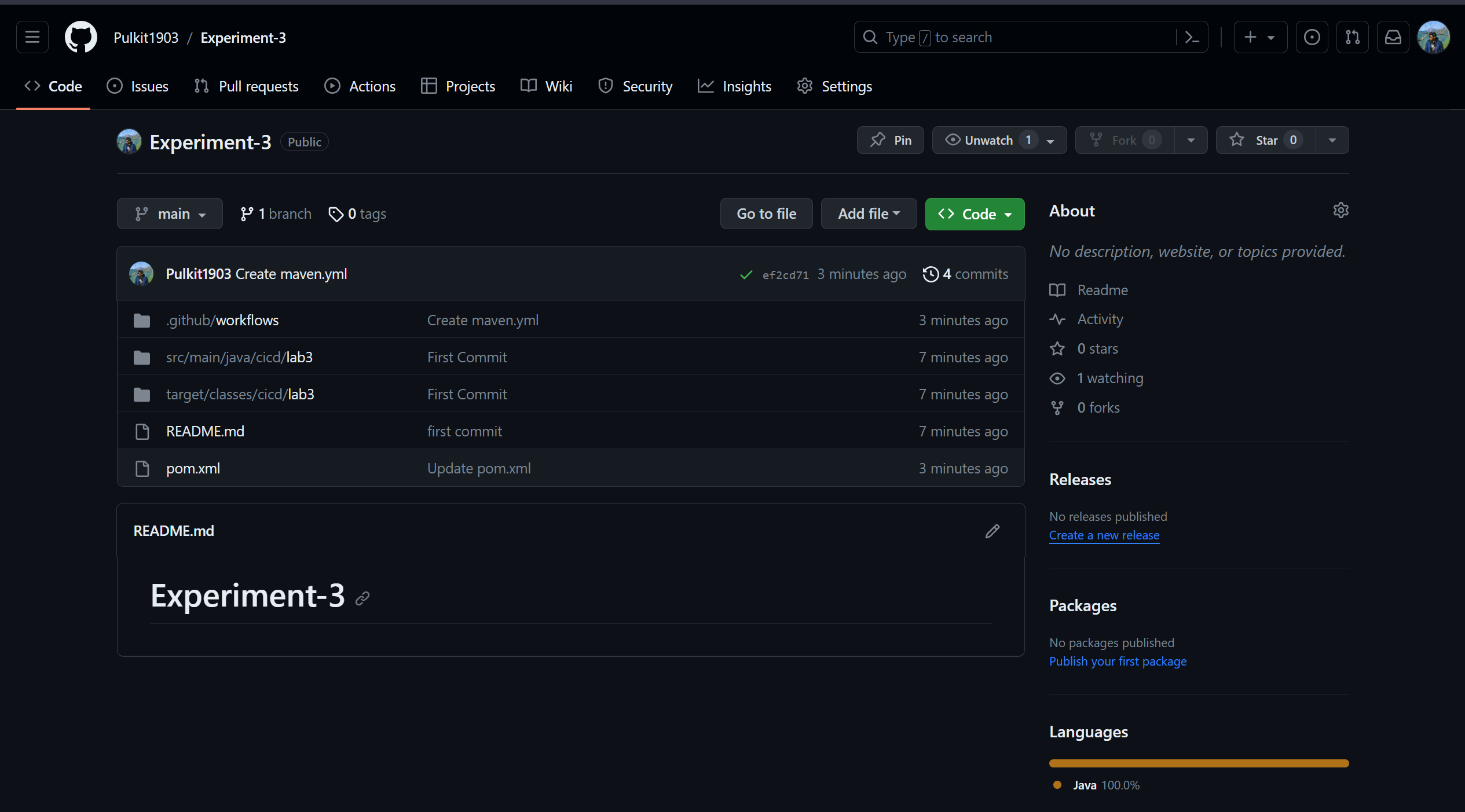


***Experiment-3***

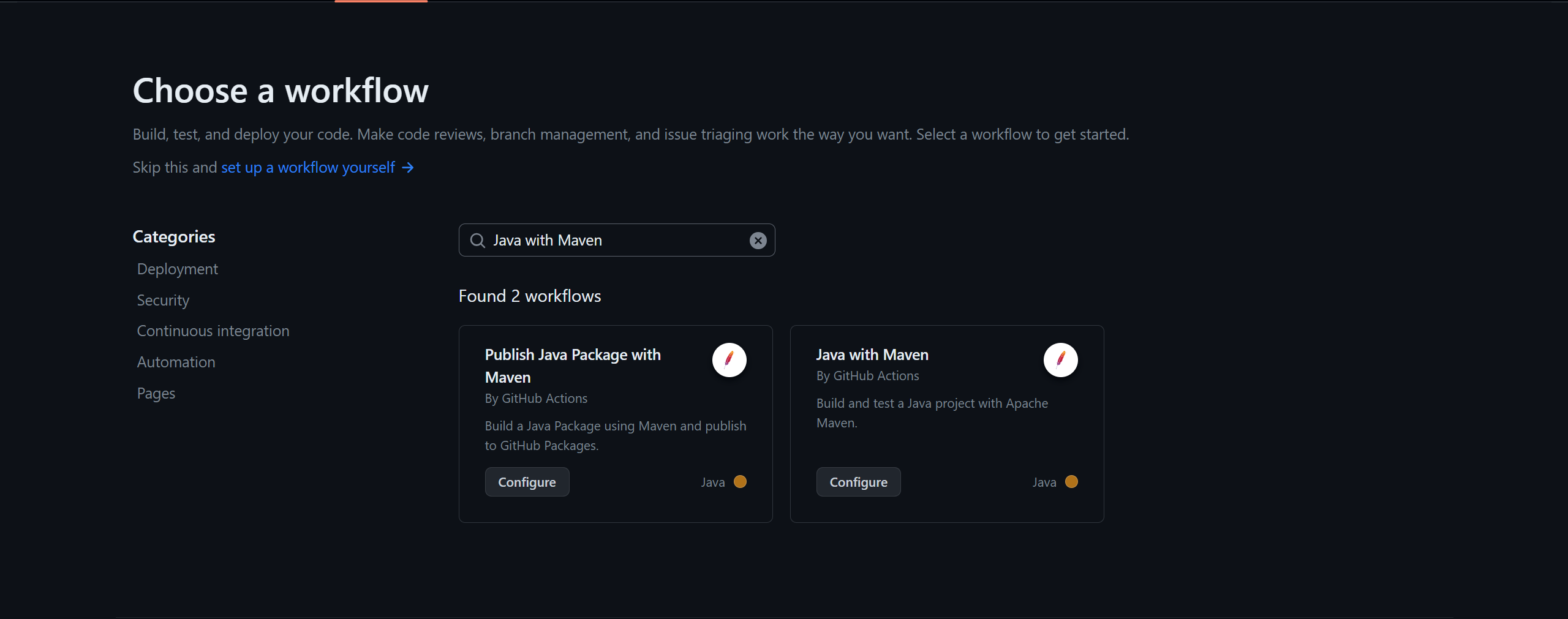
1. Create a Maven Project.



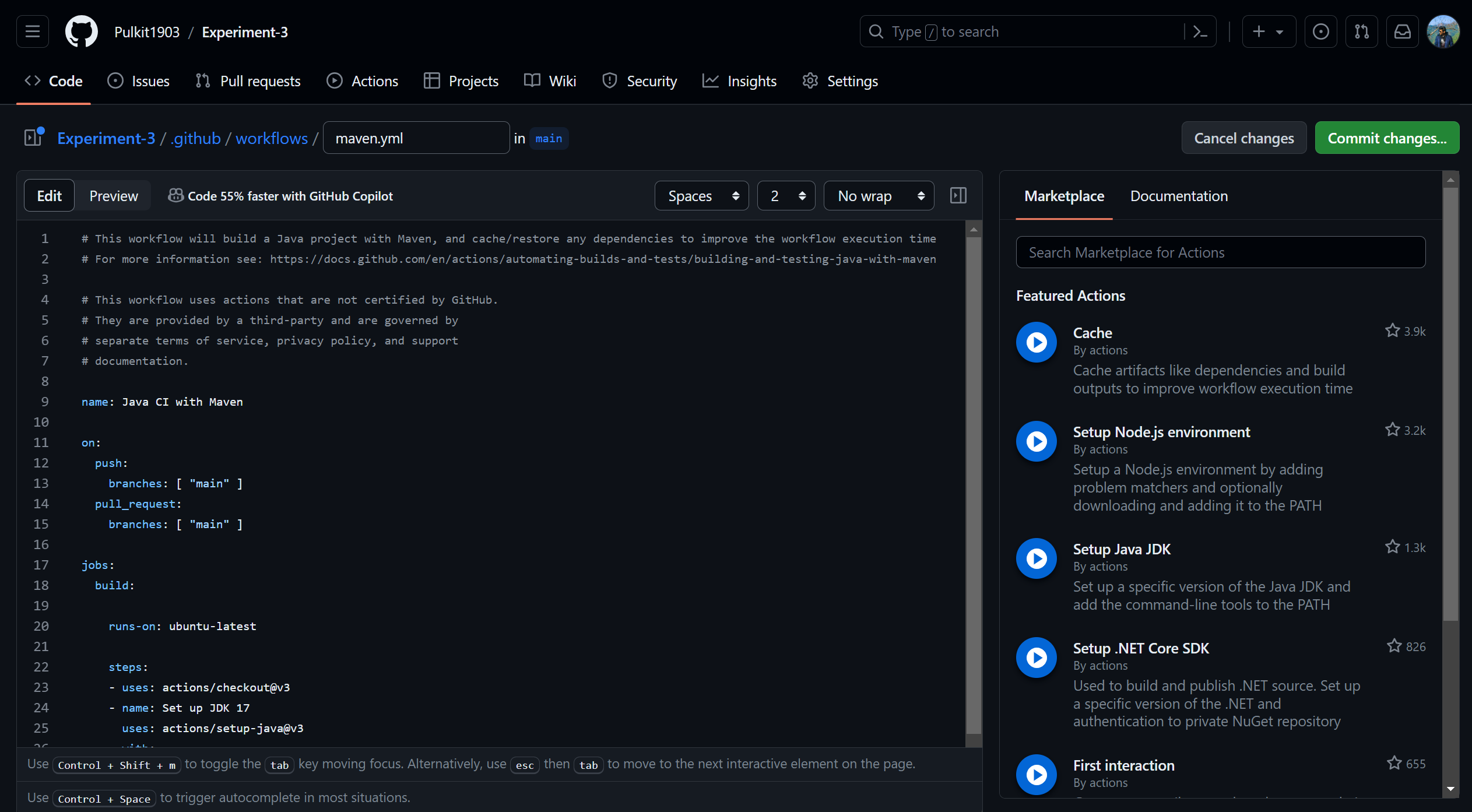
1. Push it to GitHub Repository.



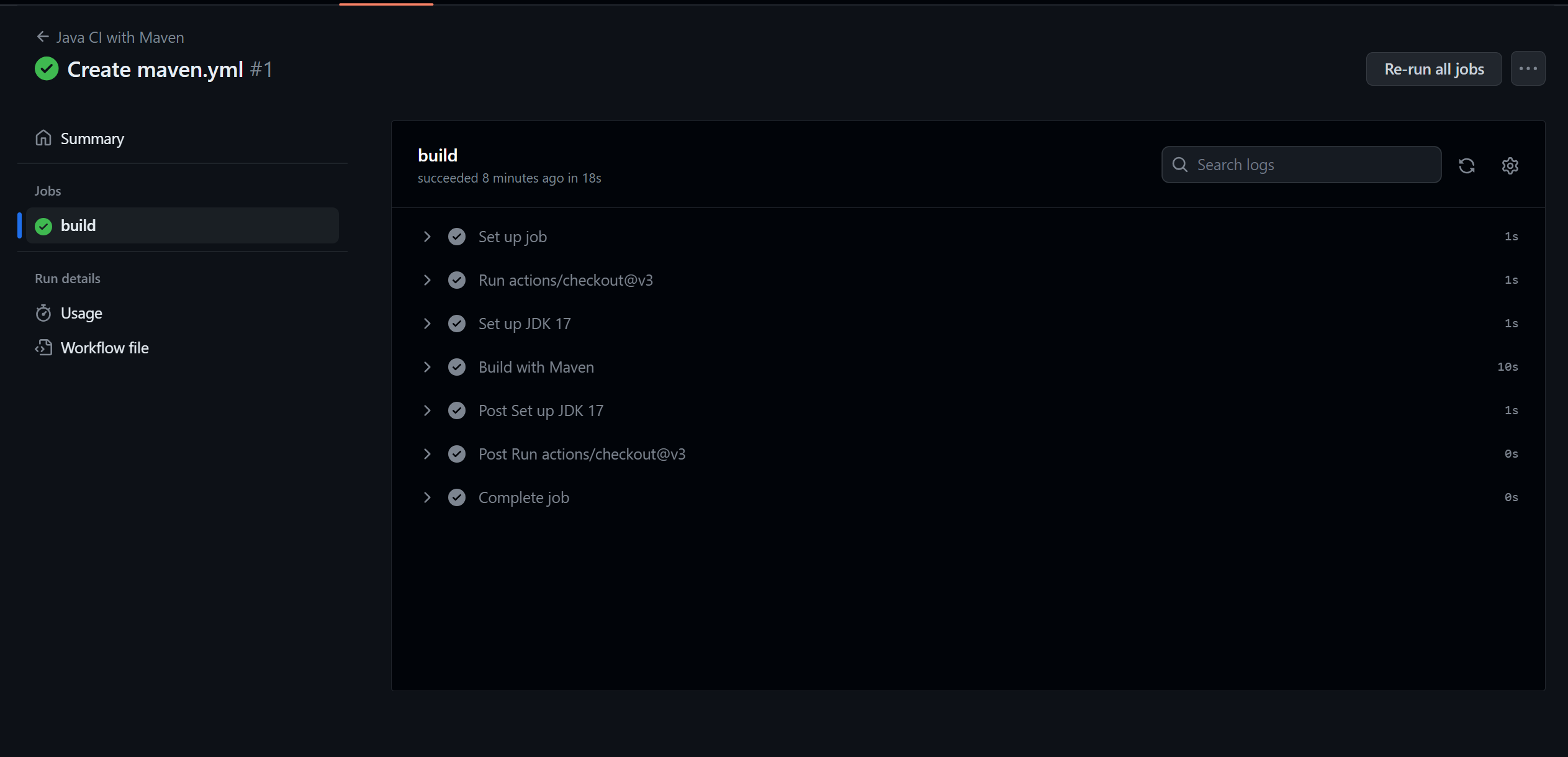
1. Go to Actions Tab and choose “JAVA WITH MAVEN.”



1. Edit maven.xml and click on Commit Changes.

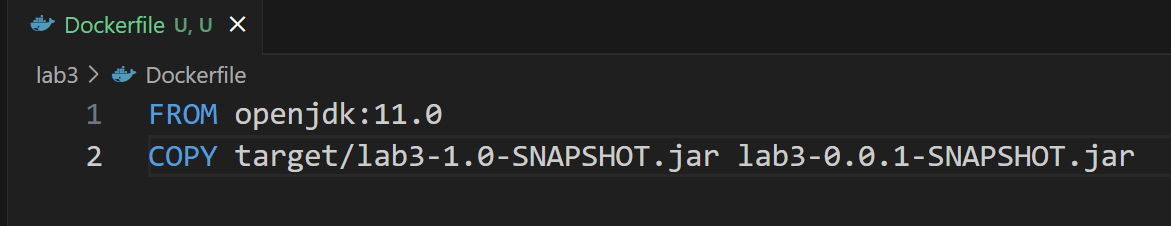


1. Navigate to Actions Tab and your Job will be completed.

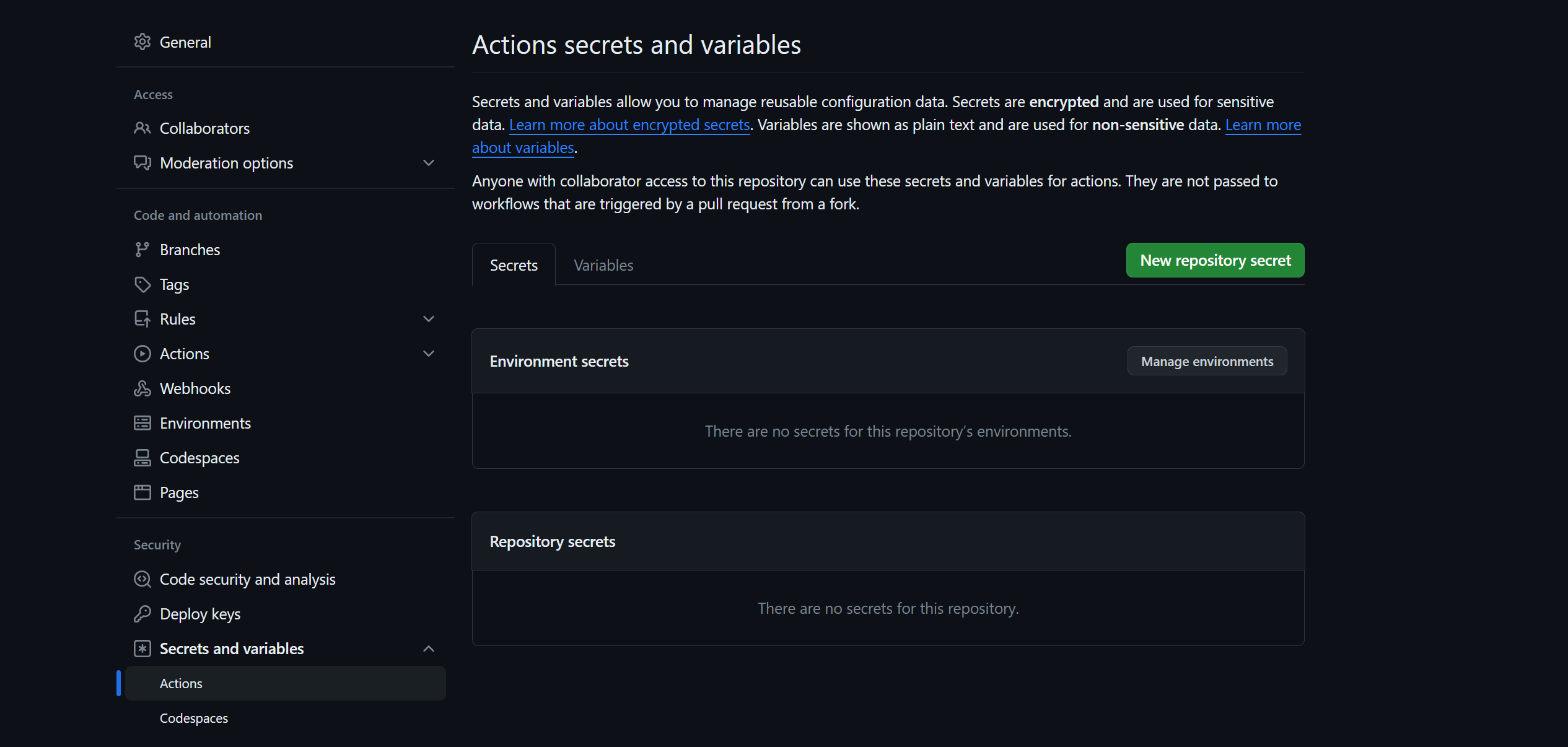


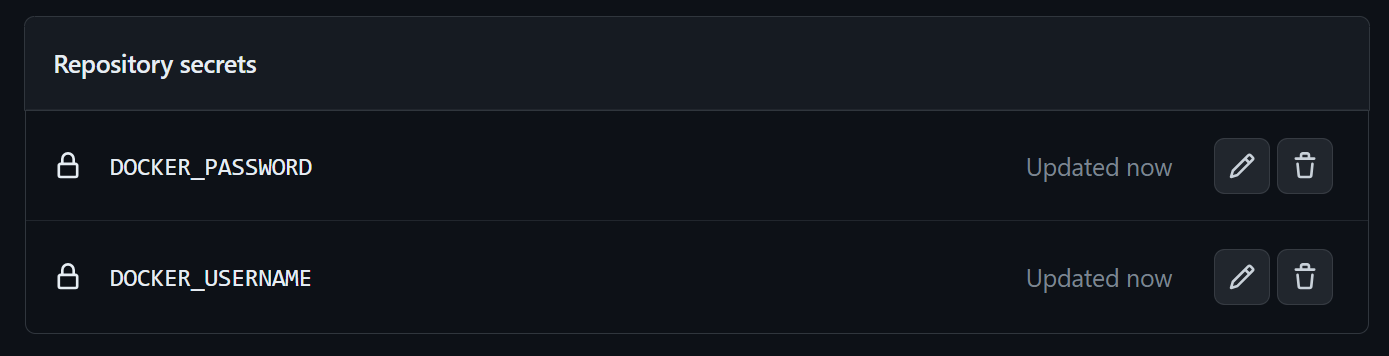
***Experiment-4***

1. Add Docker file to your repository.

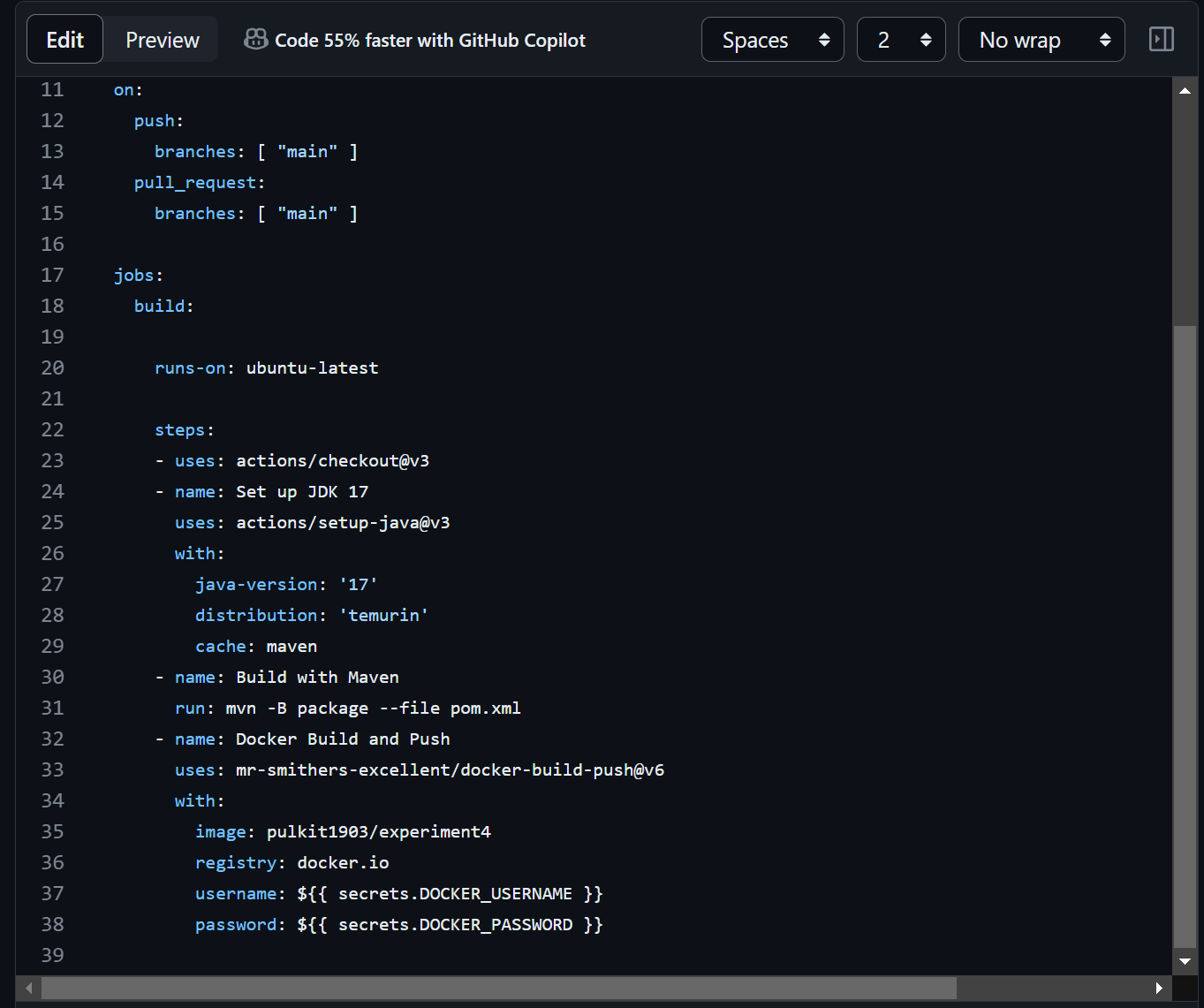


1. Add Secrets for Docker Hub.



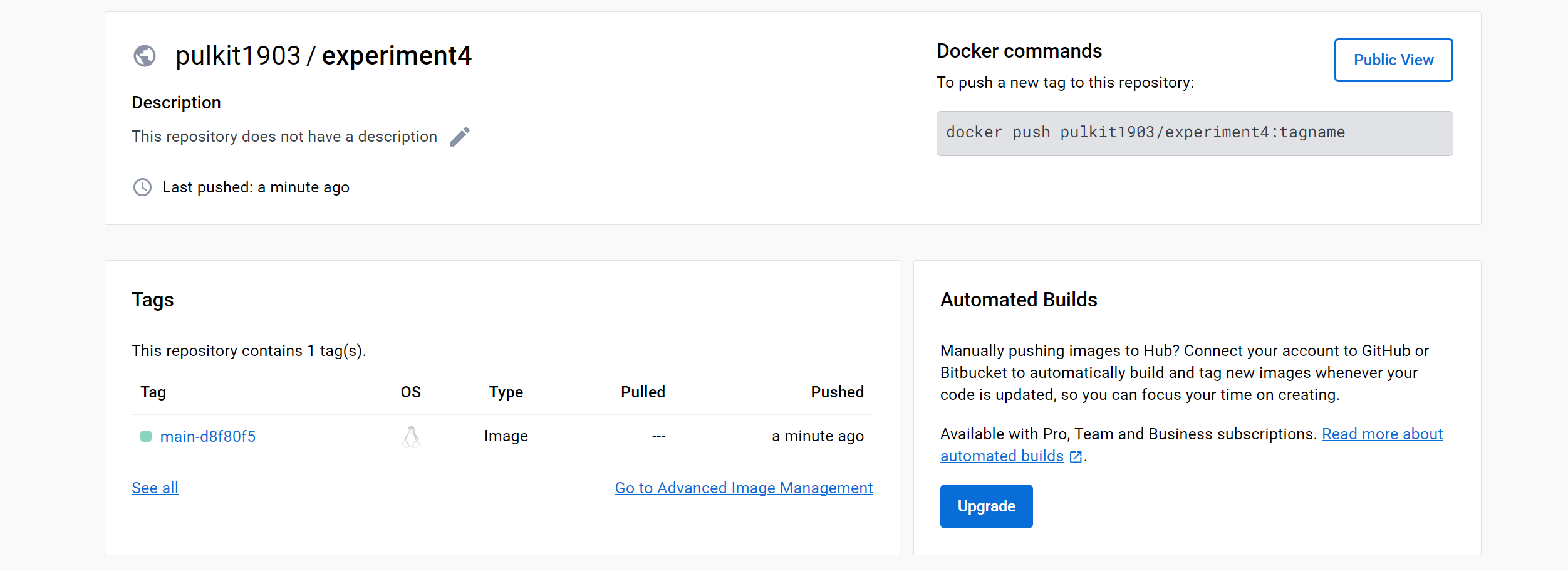


1. Edit the previous “maven.xml” file and commit changes.



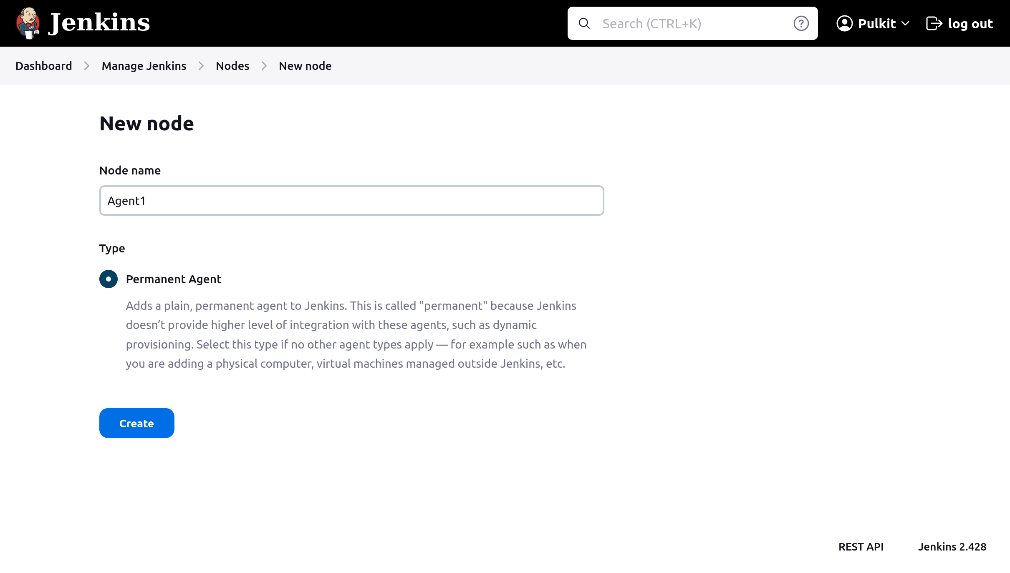
1. Jobs must have been completed and Docker Image will be uploaded to your Docker Hub.

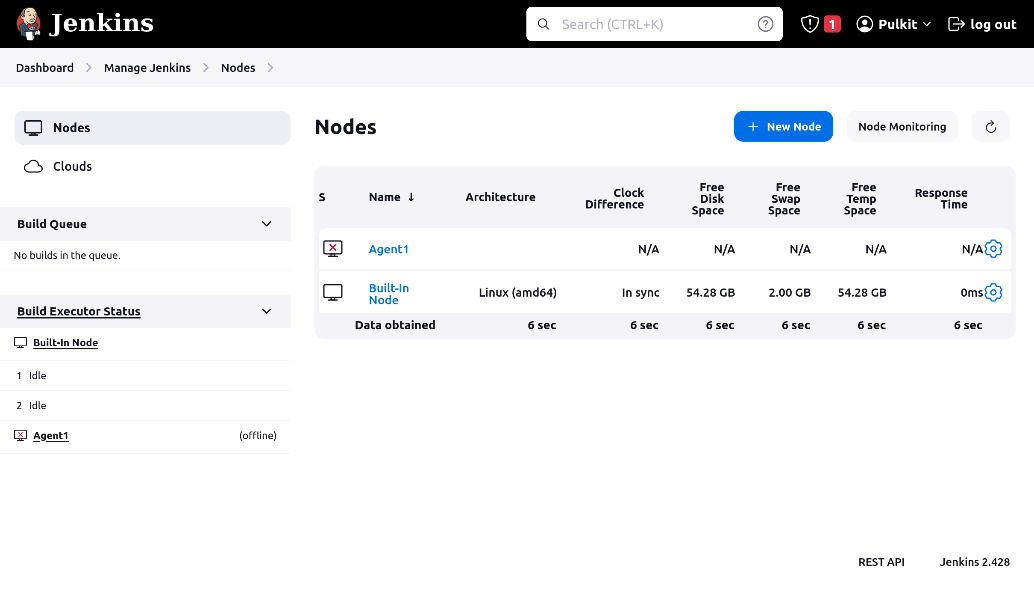


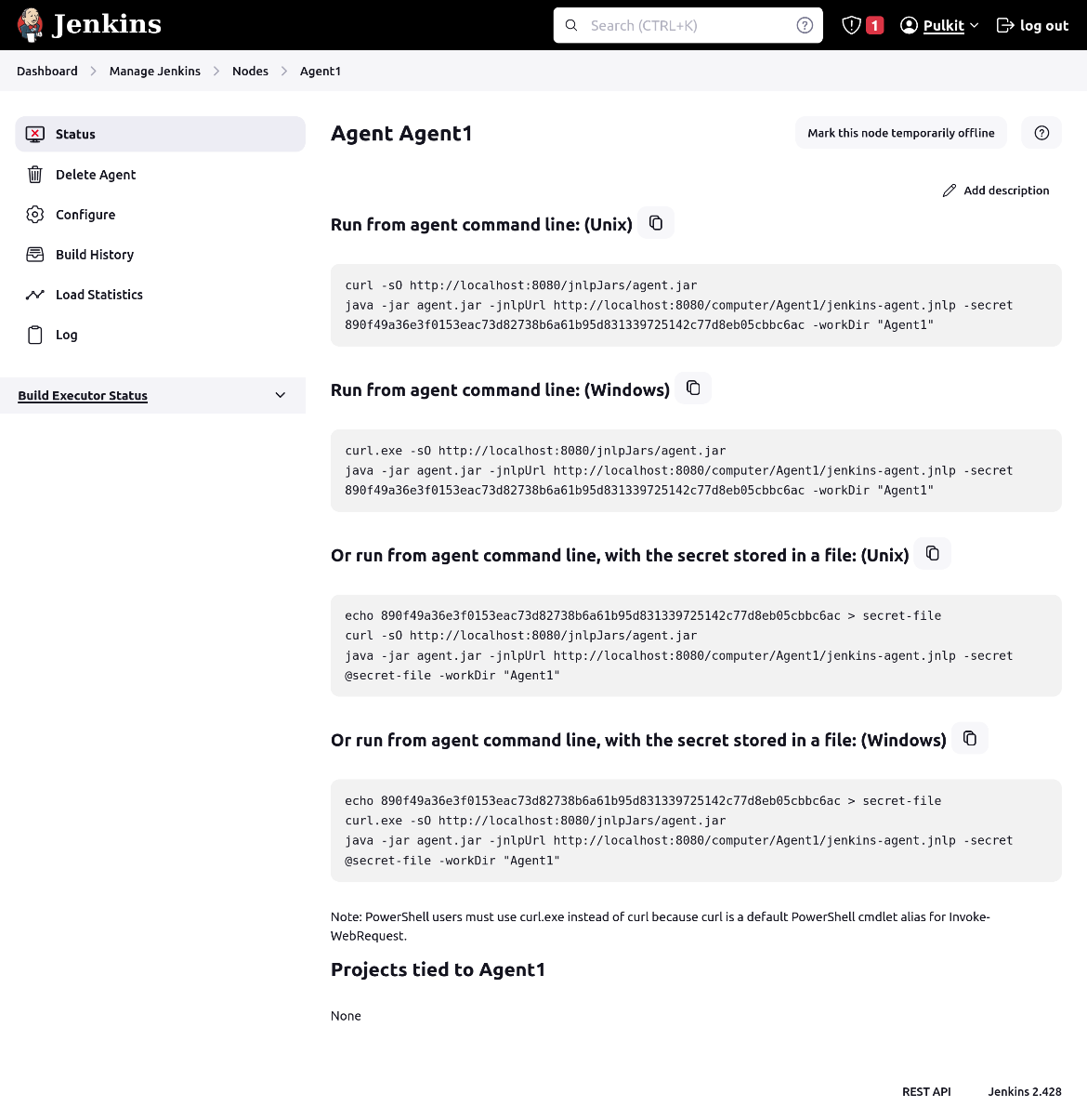


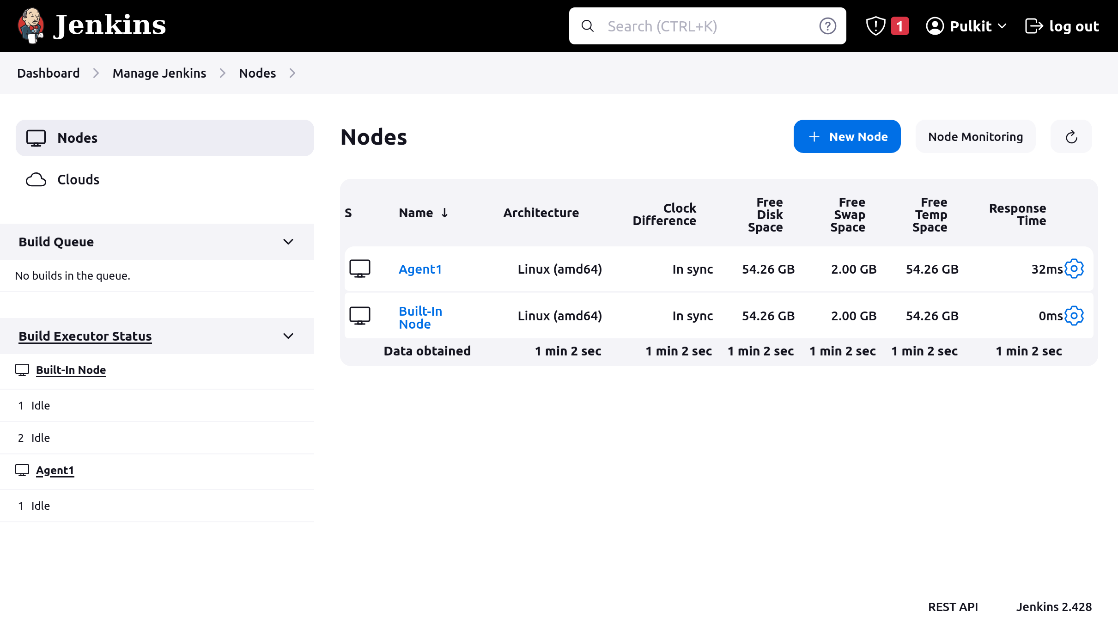
***Experiment-5***

1. Create 1st agent and make it online.

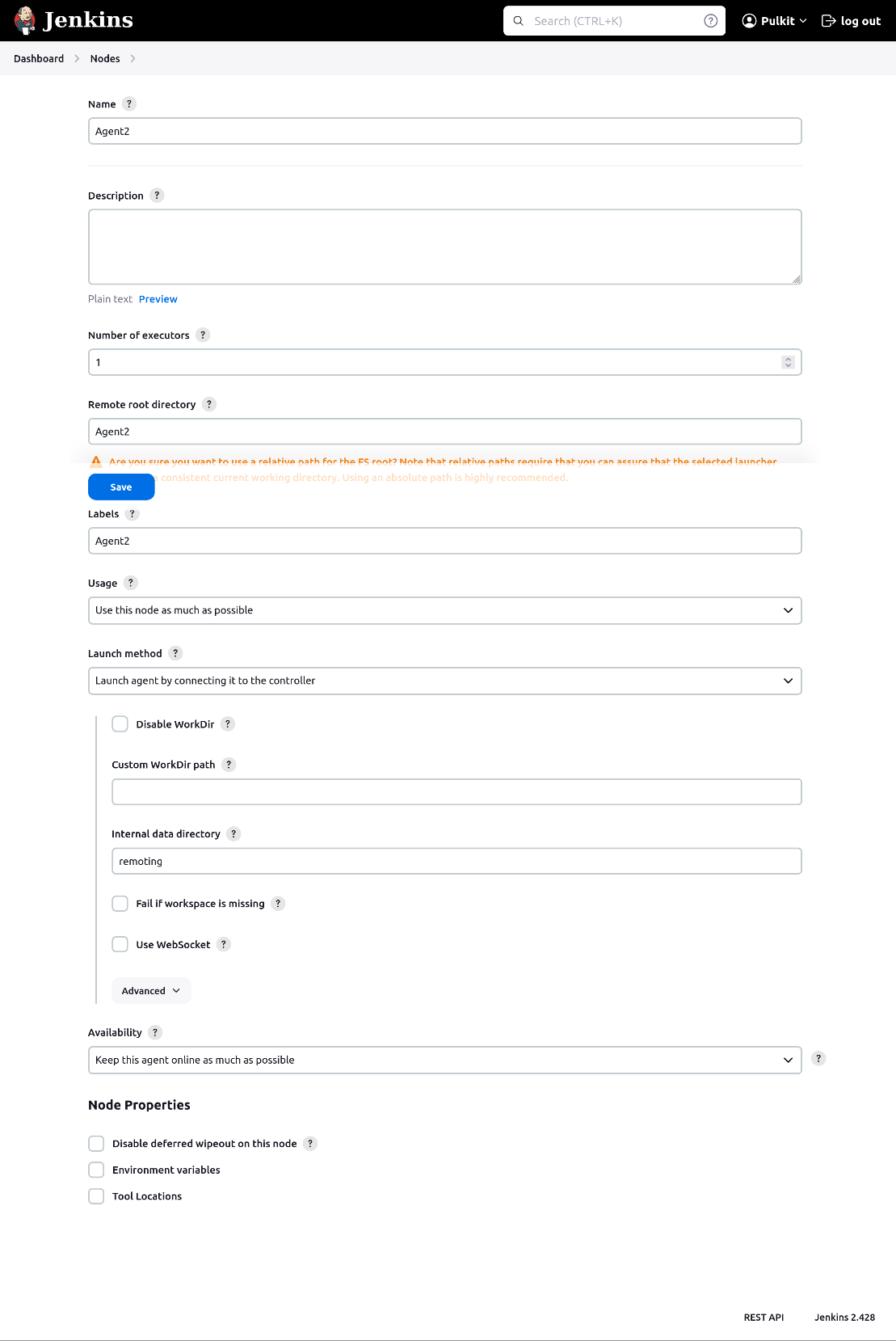
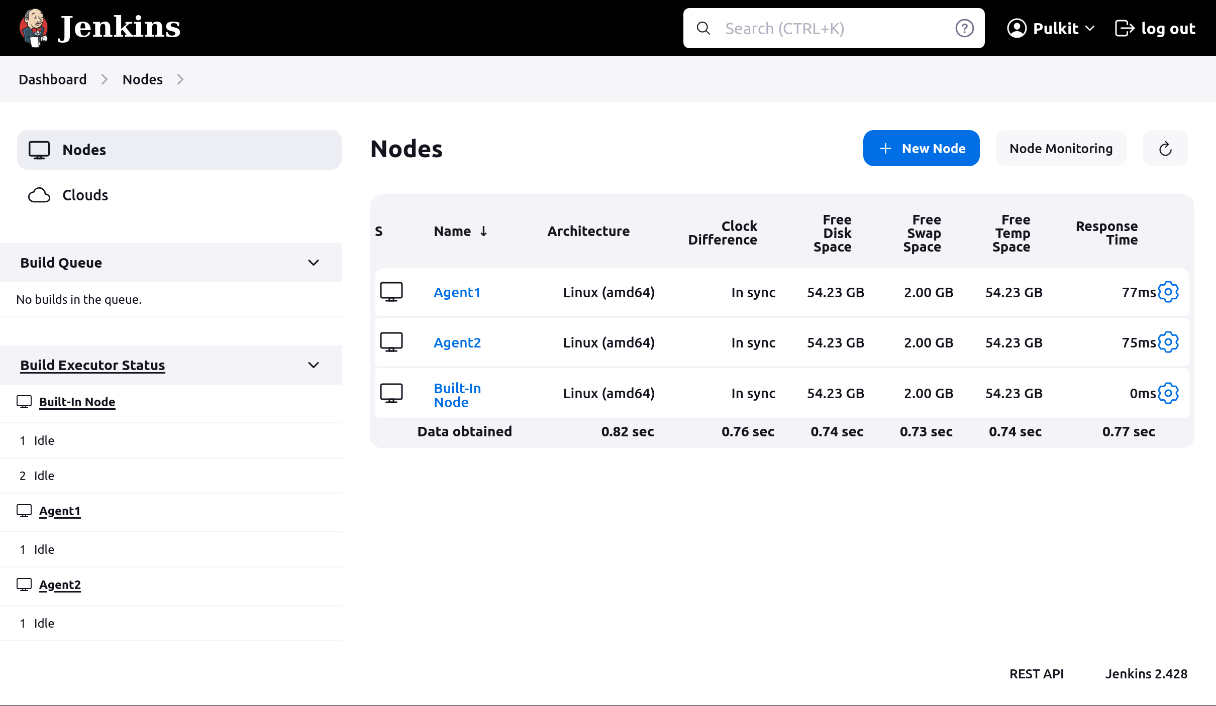




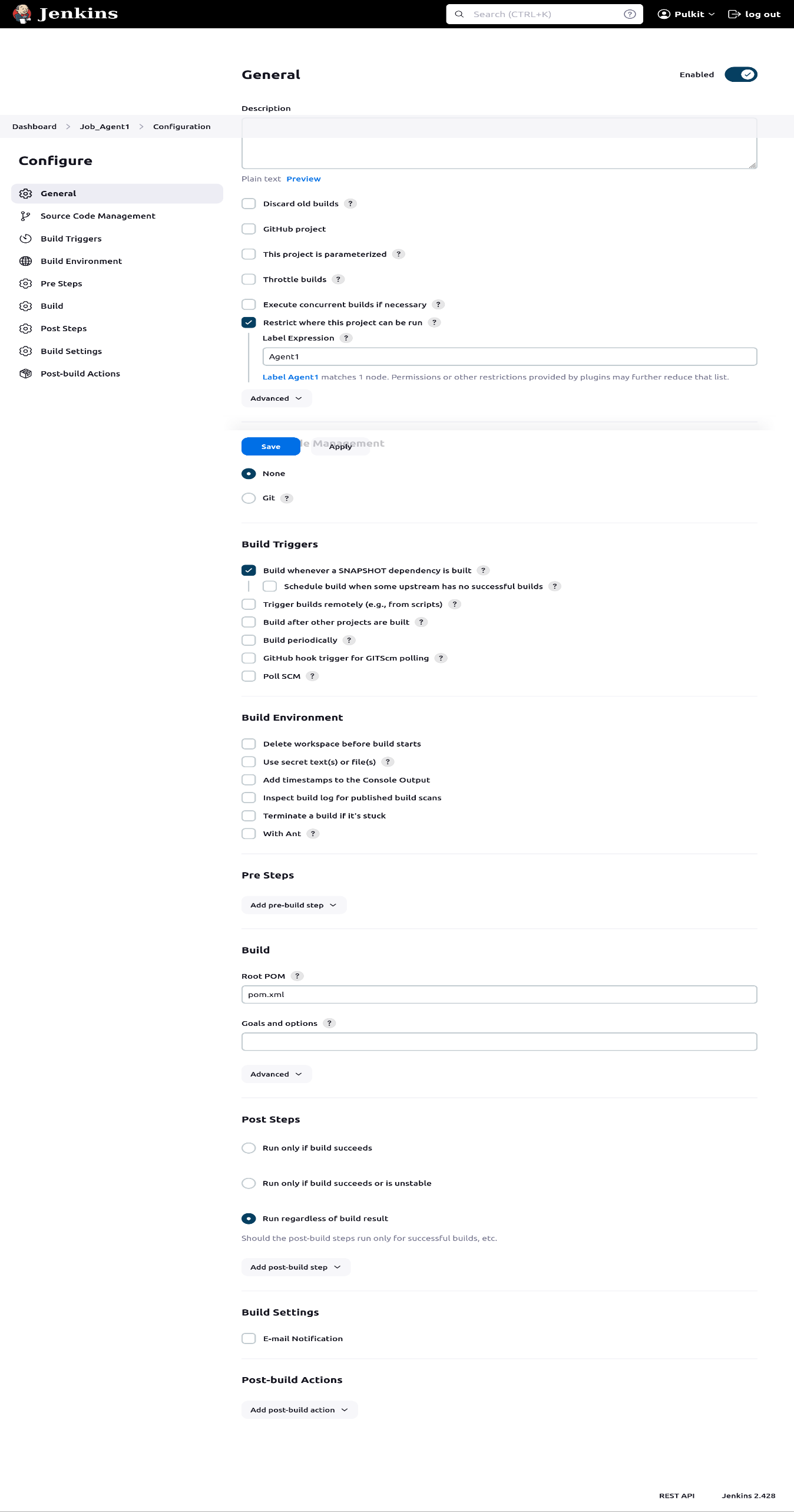


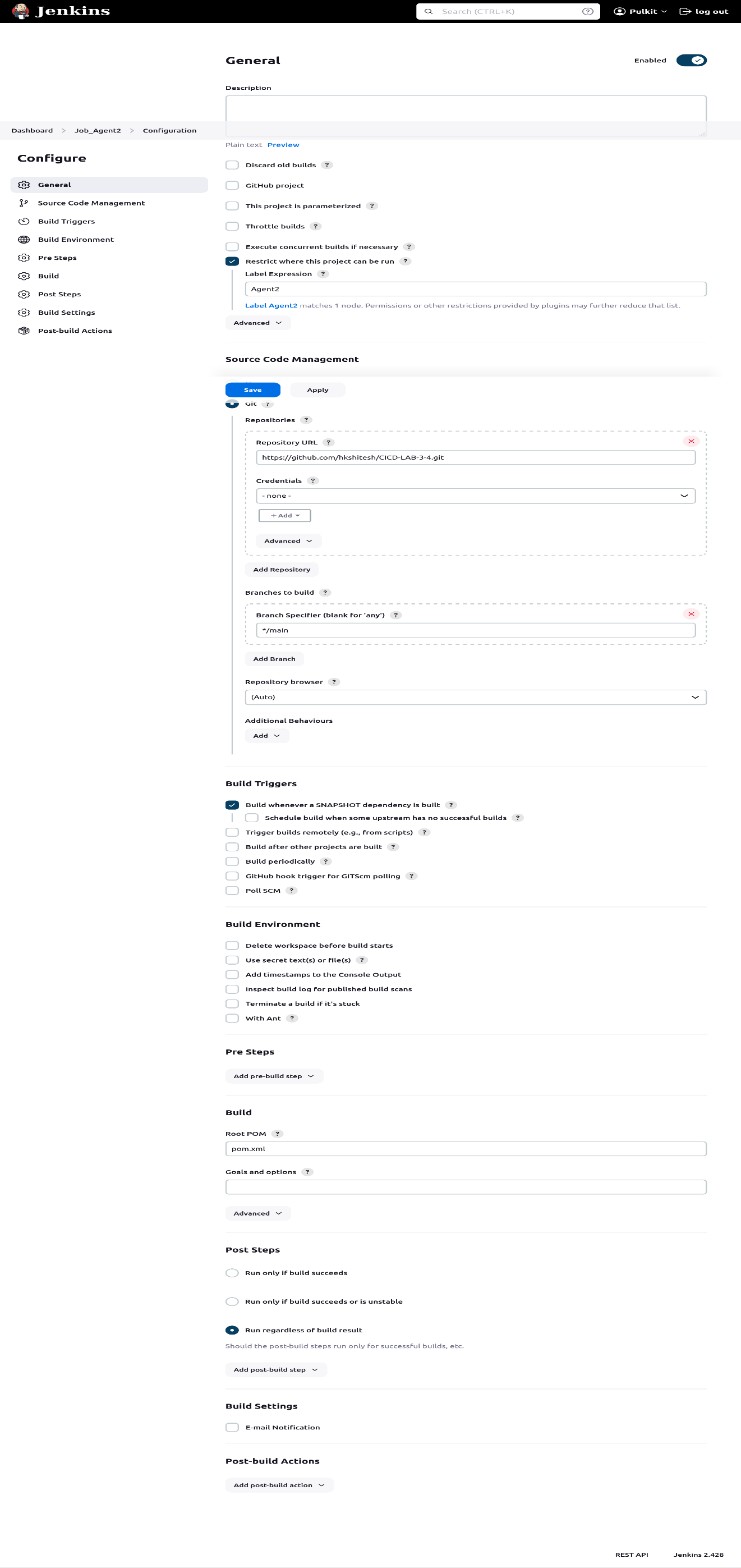


1. Create 2nd Agent and make it online with the help of above process.



1. Now assign one job to each Agent created.





***Experiment-6***

1. Create a Maven Project DO-JOB-1

Create another Pipeline DO-JOB-2



-> In the Build triggers section Check “Build after other Projects are Build” and Enter DO-JOB-1.

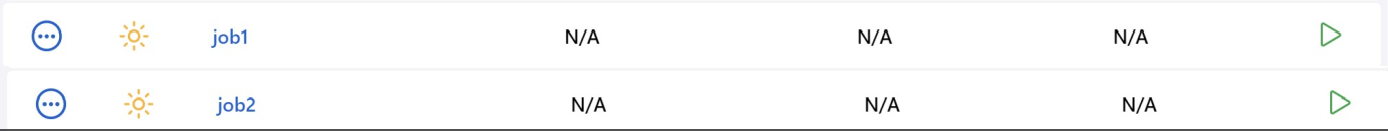


1. Create two more pipelines DO-JOB-3 DO-JOB-4, creating a longer and branched chain.

Add DO-JOB-2 as Upstream in DO-JOB-3



Add DO-JOB-2 and DO-JOB-3 in DO-JOB-4



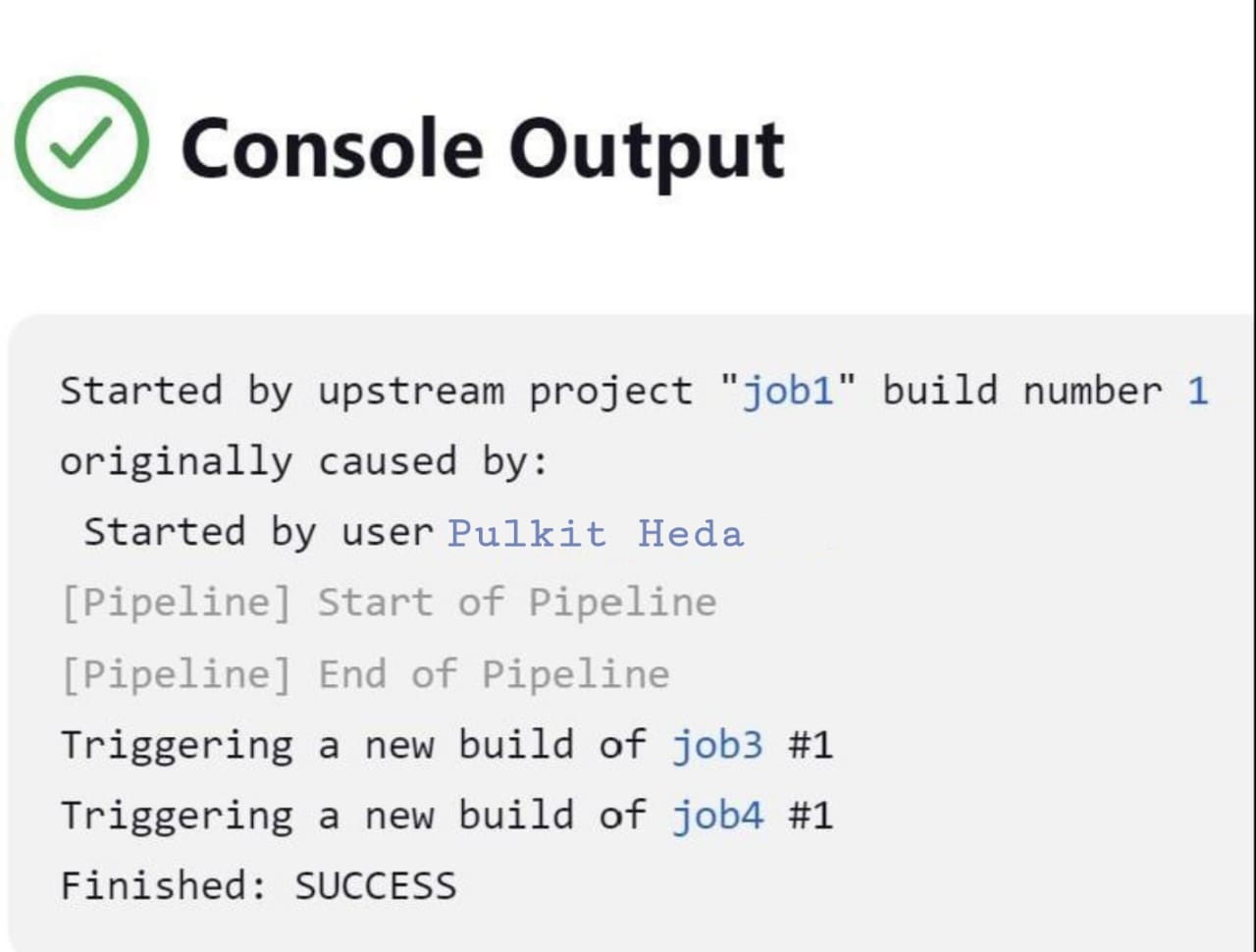
Test the complete pipeline chain by running DO-JOB-1

Checking Logs

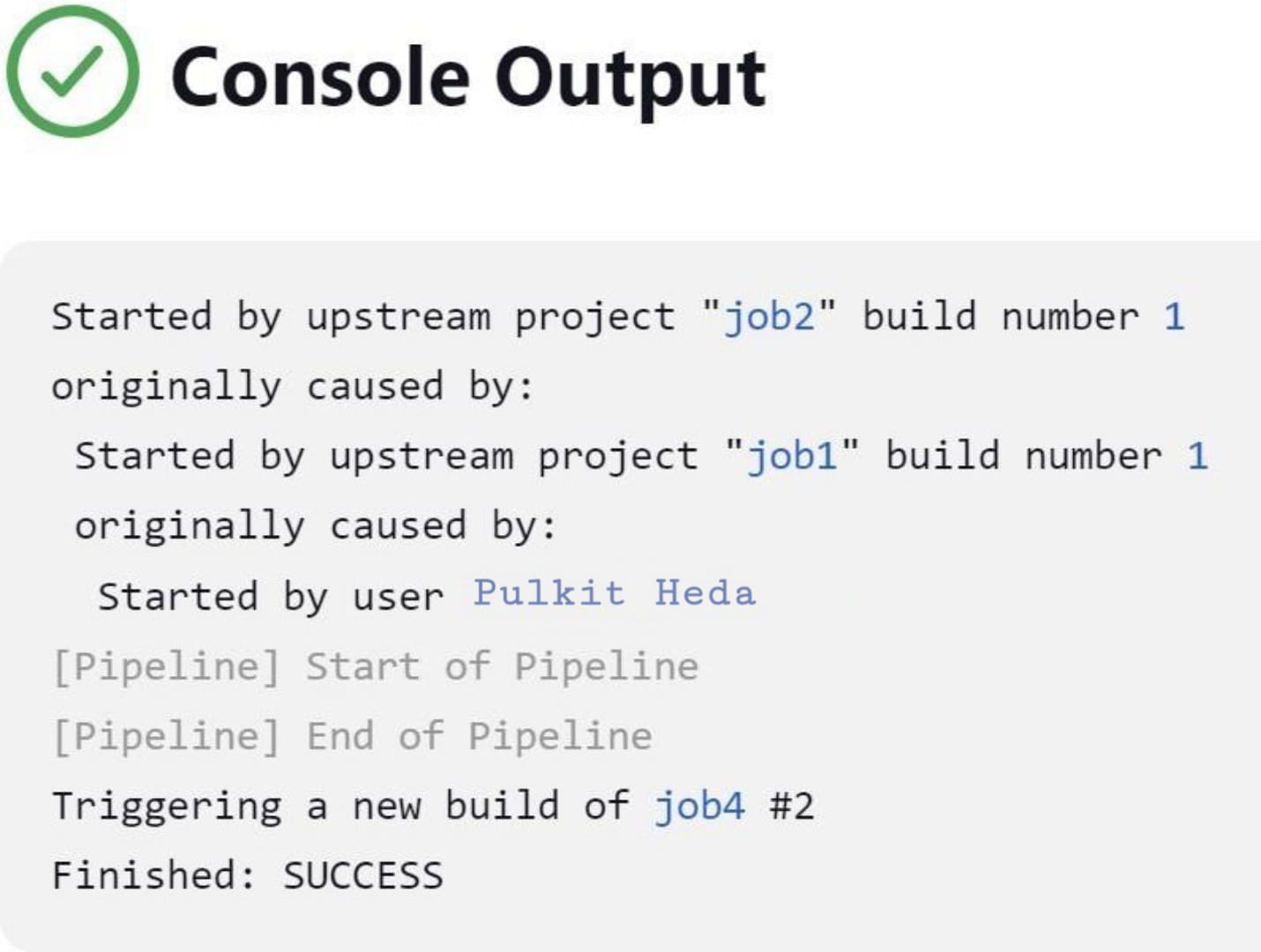
DO-JOB-1 triggered DO-JOB-2 once it succeeded.



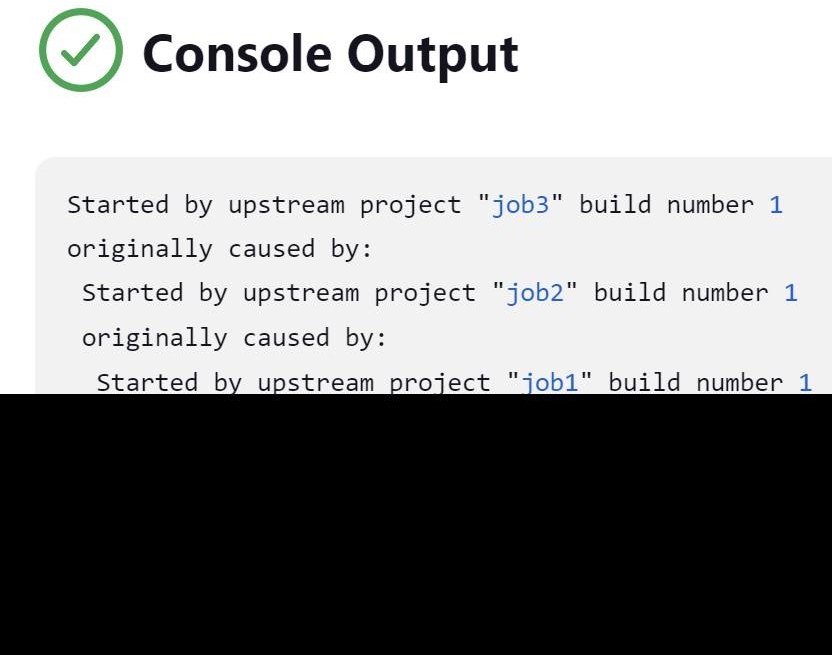
Similarly, DO-JOB-2 triggers DO-JOB3 and then DO-JOB-4 respectively.



DO-JOB-3 logs

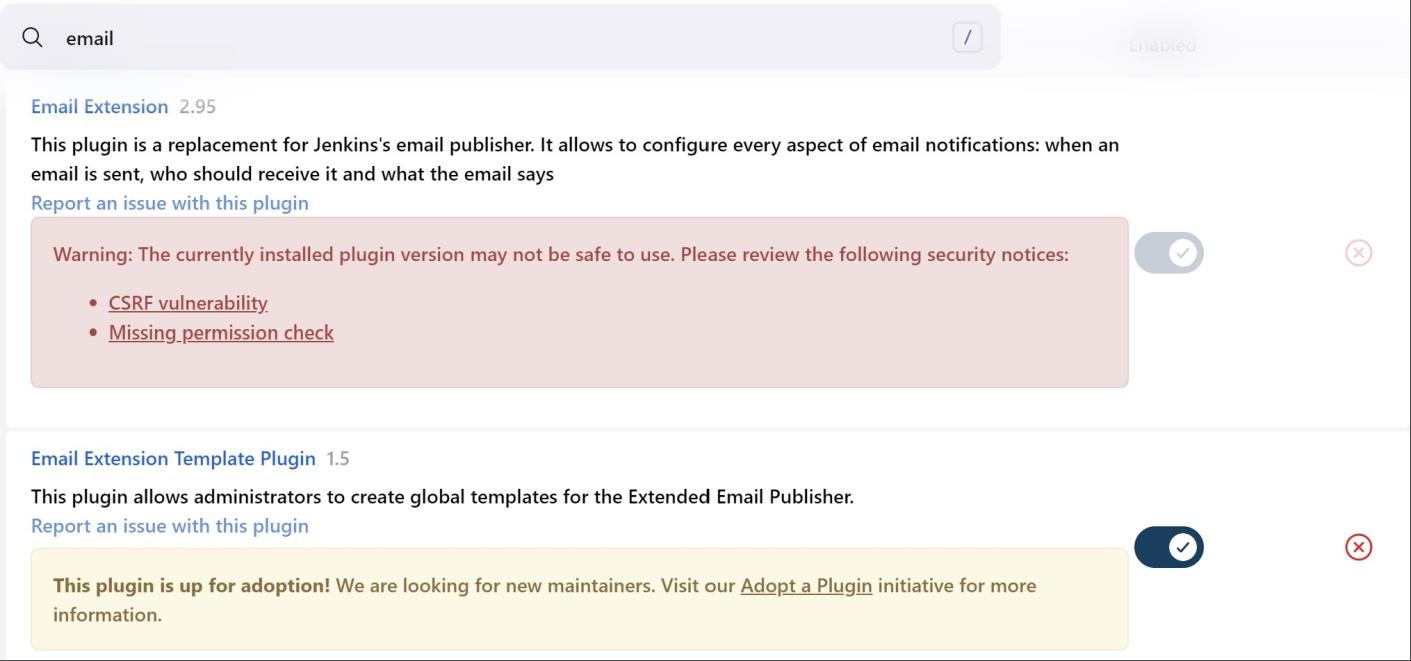


DO-JOB-4 logs



***Experiment 7***

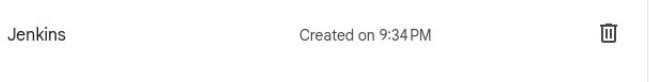
1. Install Email Extension and Email Extension Template Plugin.



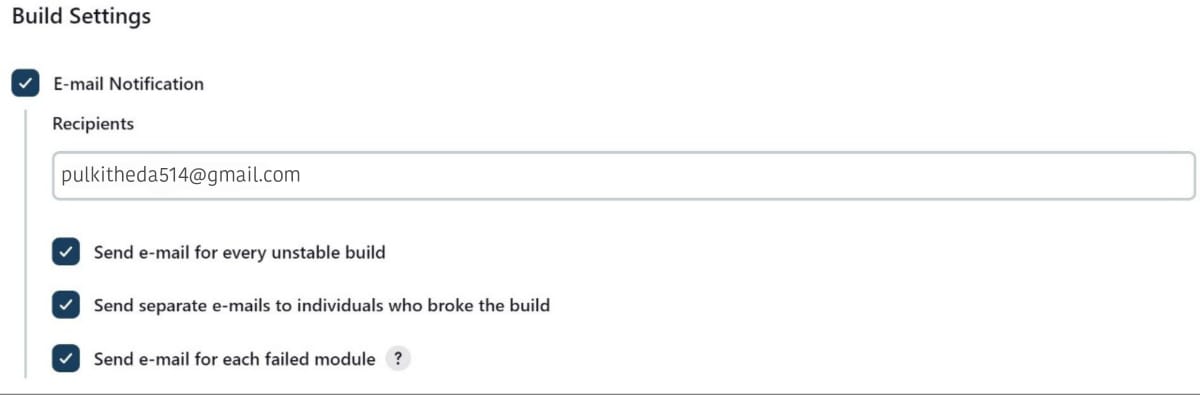


1. Configure SMTP server by going to Dashboard > Manage Jenkins > System > Email Notification.

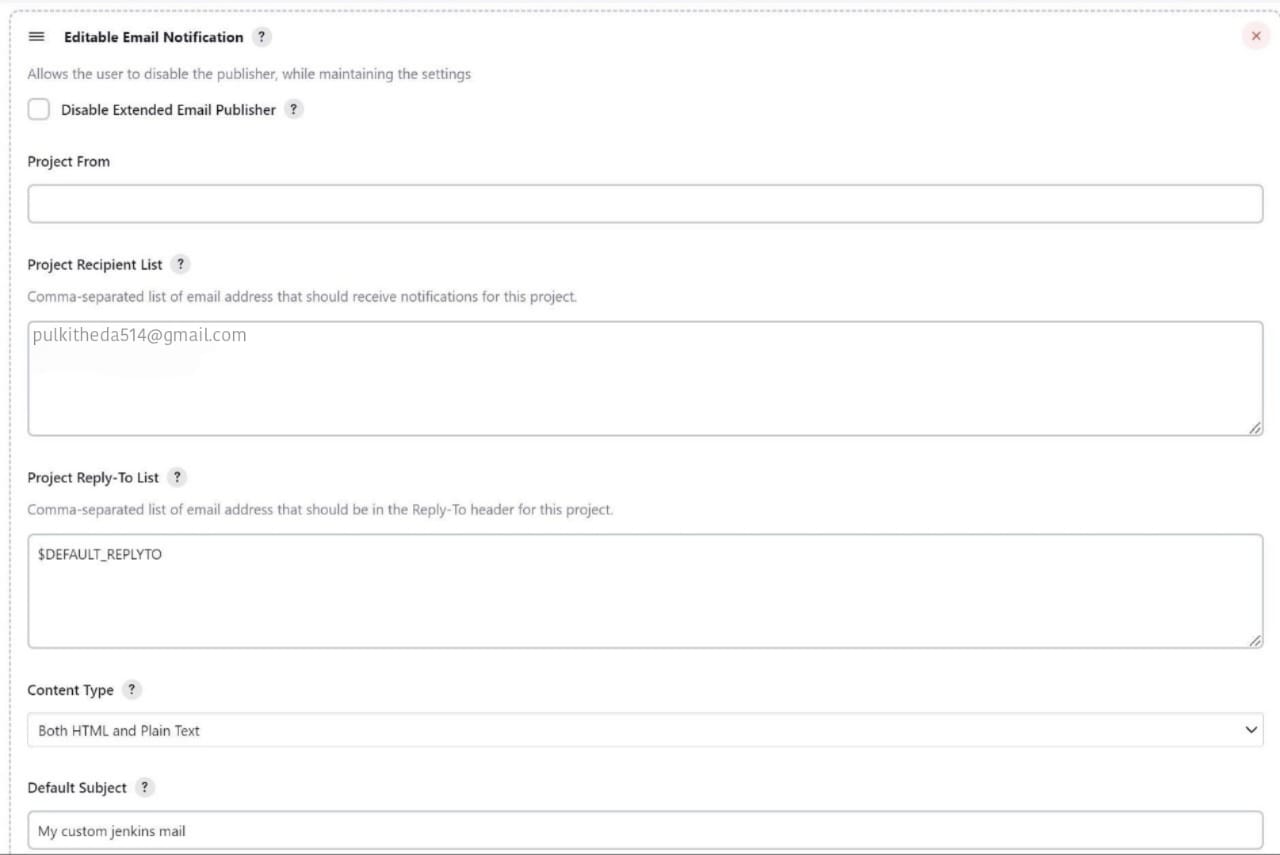
Note: get the password for google app passwords (and a password for your Jenkins app)



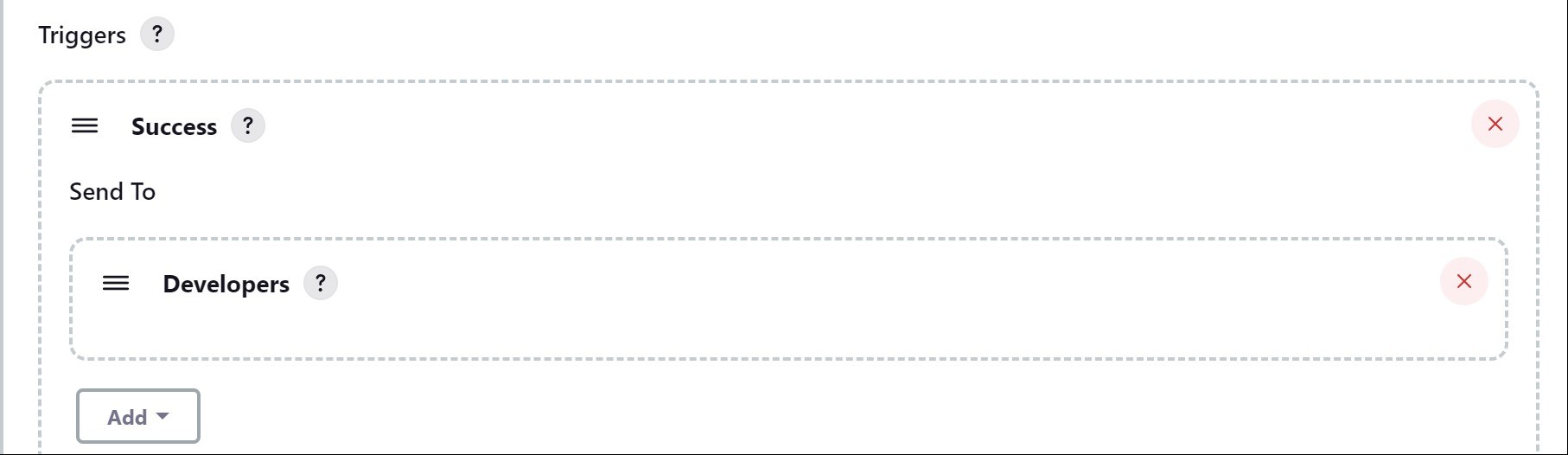
1. Create a Job and add email in Build Settings.



1. Add a Post Build Step for sending Editable Email Notification and add email.



1. We can also choose to send Mails on Build Success by choosing the advanced options. (Add Trigger Option)



1. Testing mail functionality.

