Jenkins

Table of Contents

Jenkins

Jenkins is a powerful application that allows continuous integration and continuous delivery of projects, regardless of the platform you are working on. It is a free source that can handle any kind of build or continuous integration. You can integrate Jenkins with a number of testing and deployment technologies. In this notes, we will explain how you can use Jenkins to build and test your software projects continuously.

Continuous Integration and Continuous Delivery (CI/CD) are essential practices in modern software development that aim to streamline and automate the process of building, testing, and deploying code changes. These practices help development teams deliver software more efficiently, with higher quality, and at a faster pace.

Continuous Integration (CI):
CI is the practice of automatically integrating code changes from multiple developers into a shared repository on a frequent basis, typically multiple times a day. The main goal of CI is to detect integration issues early in the development cycle, thereby reducing the chances of conflicts and bugs that arise when multiple developers work on the same codebase.
Key features of CI:

Automatic code integration: Developers frequently merge their code changes into a central repository, where automated build and testing processes are triggered.
Automated build and tests: CI systems automatically compile the code, run unit tests, and perform other validation checks to ensure the codebase remains stable.

Early feedback: By catching issues early, CI allows developers to fix problems quickly, preventing the accumulation of bugs.

Continuous Delivery (CD):
CD is an extension of CI that focuses on automating the software release process. It ensures that code changes are always in a deployable state and ready for production release. The ultimate goal of CD is to enable frequent and reliable software releases to end-users with minimal manual intervention.

Key features of CD

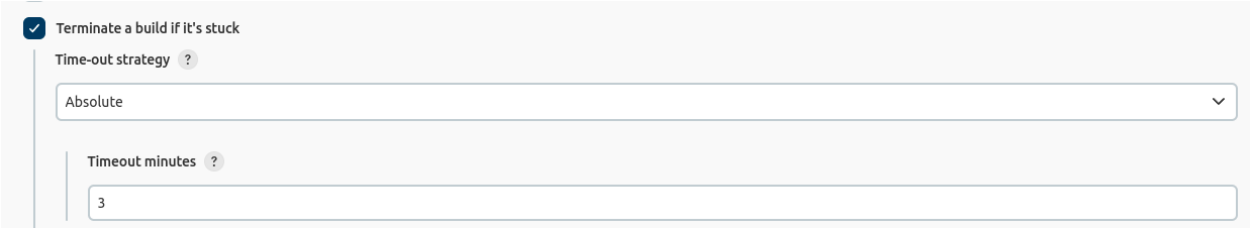
- **Automated deployment:** The CD pipeline automates the deployment of code changes to various environments, including staging and production, with consistent configurations.
- **Release orchestration:** CD pipelines manage the entire release process, coordinating tasks such as database updates and infrastructure provisioning.
- **Rollbacks and monitoring:** CD ensures that automated rollbacks are possible if any issues arise during deployment. Monitoring is also integrated to track the application's health post-deployment.

Benefits of CI/CD

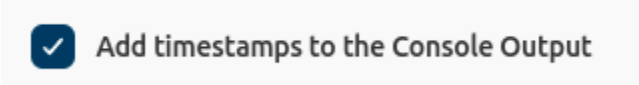
- **Faster time-to-market:** CI/CD streamlines the development and deployment process, reducing the time it takes to deliver new features and updates to end-users.
- **Higher software quality:** Automated testing and validation catch bugs early, leading to a more stable and reliable codebase.
- **Risk reduction:** Frequent integration and automated deployment enable faster bug identification and recovery, minimizing risks associated with manual processes.
- **Increased collaboration:** CI/CD encourages collaboration among developers, testers, and operations teams, fostering a culture of continuous improvement.

Jenkins features of every tool:

1. **Timeout-** It will be found under the Build Environment section-> Terminate a build if it's stuck.
   When the system sleeps for some time e.g. 240 seconds we define the Time-out strategy time e.g. 3 minutes. It means that a job will not build or fail.

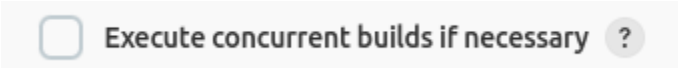


2. **Timestamp-** It will be found under the Build Environment section->Add timestamps to the Console Output which means it will notify the time whatever command is running at what time.

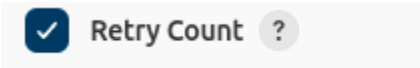


3. **Disable/Enable Job-** It will be found under the status of a specific project. You can use this feature to disable or enable the job build. If you want that nobody can build the job except you then, you can disable the job. And whenever you want to enable it you can do that too.

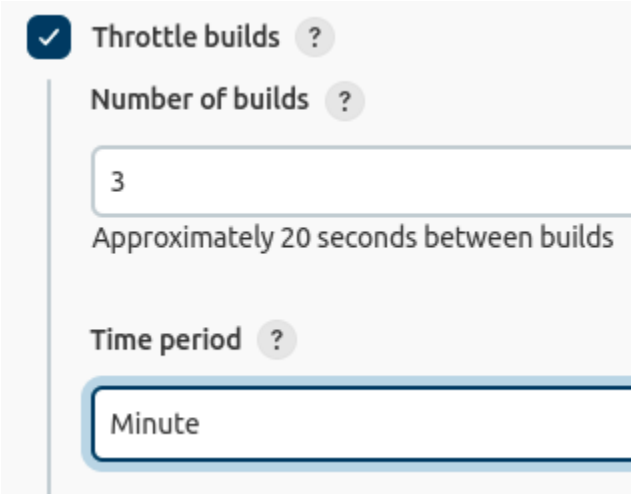
4. **Build a Job concurrently-** As you are aware you couldn't build a job concurrently. But if you want to do this then, you have to enable the feature that will be found under the section of Description named -> `Execute concurrent builds if necessary.`



5. **Retry count-** When your job is not built successfully. So you can retry as many as you want but for that, you have to enable it to define time and number of counts.

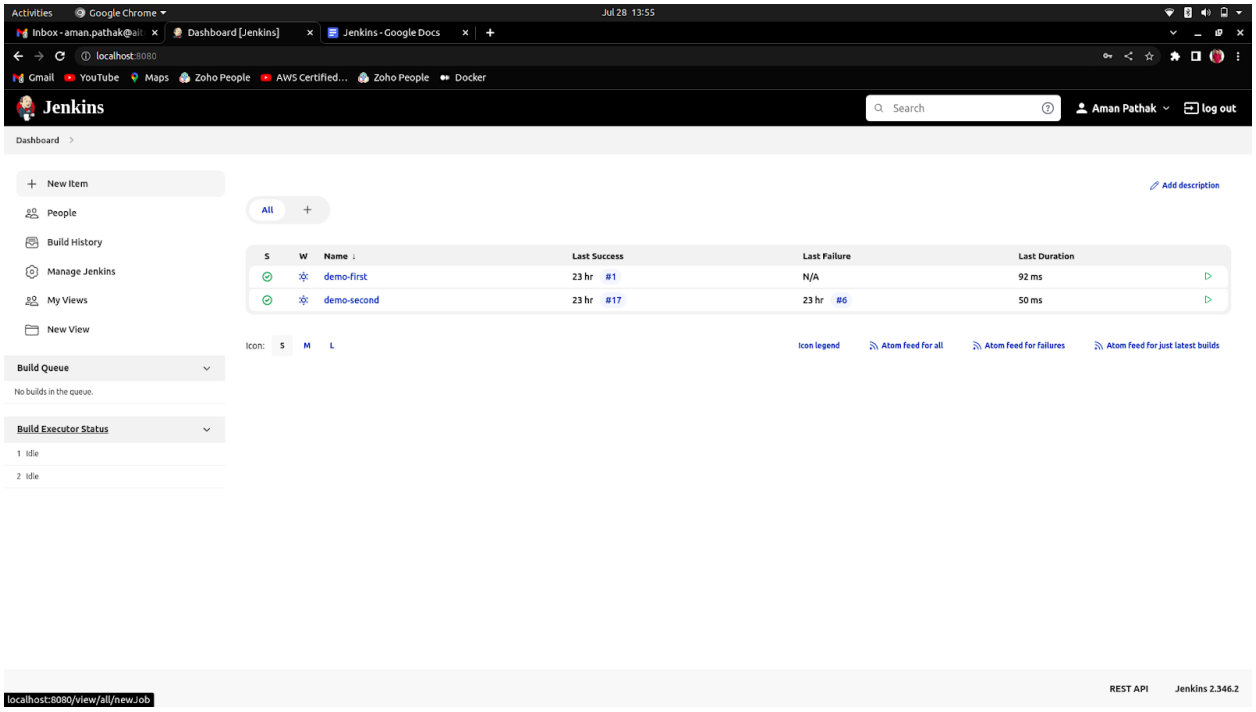


6. **Throttle Builds-** When you want that there should be a limited number of builds asked then, you will use throttle builds. For eg, the Number of builds is 3 and the Time period is minute. Then there are only 3 builds that will be successful and possible, not more than that.

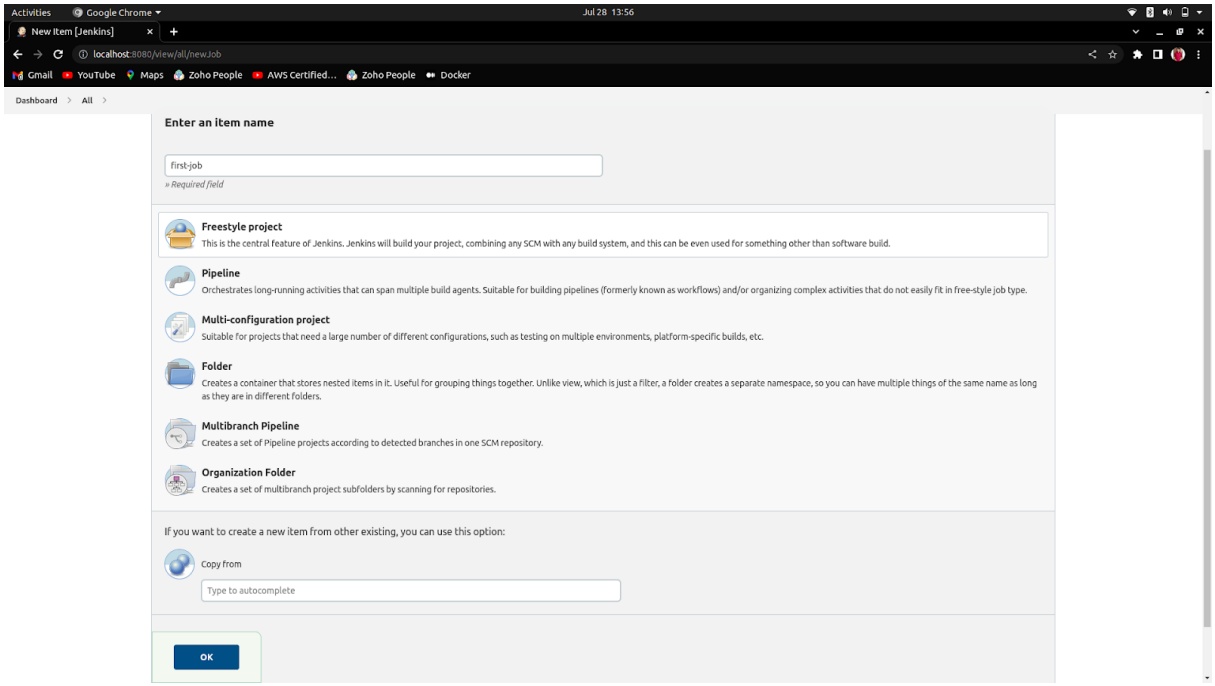


Creating First Job

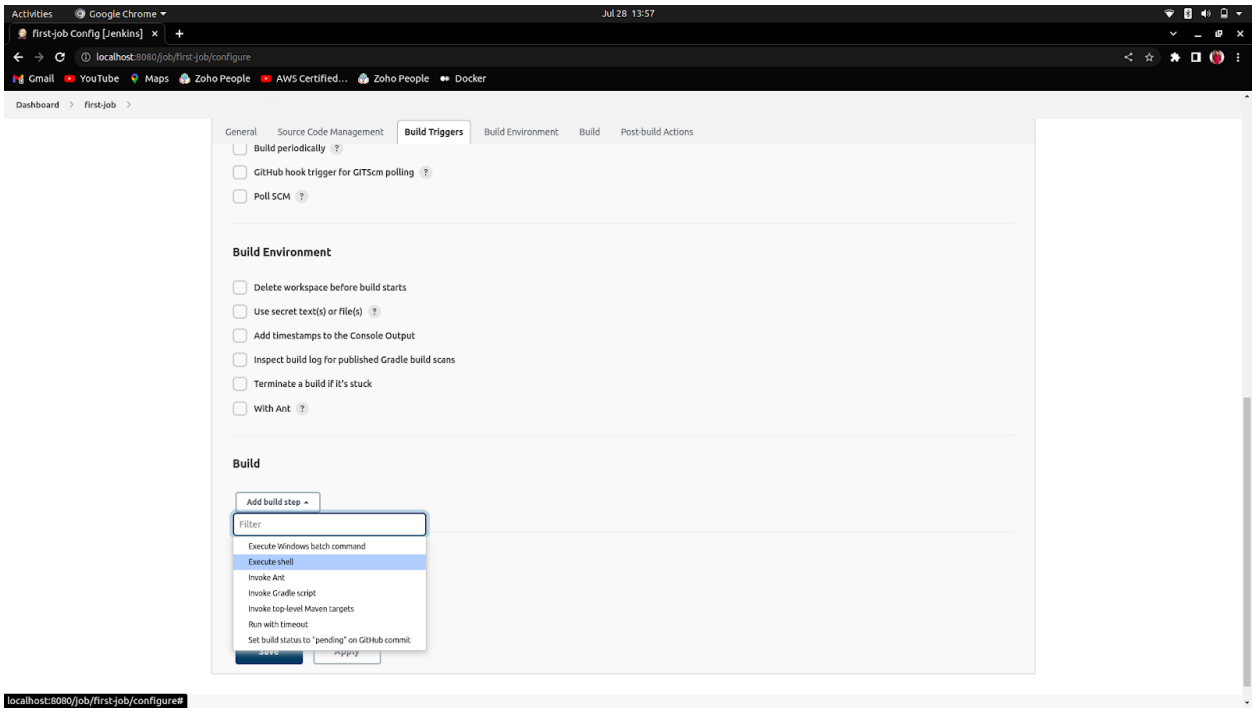
1. Click on New Item.

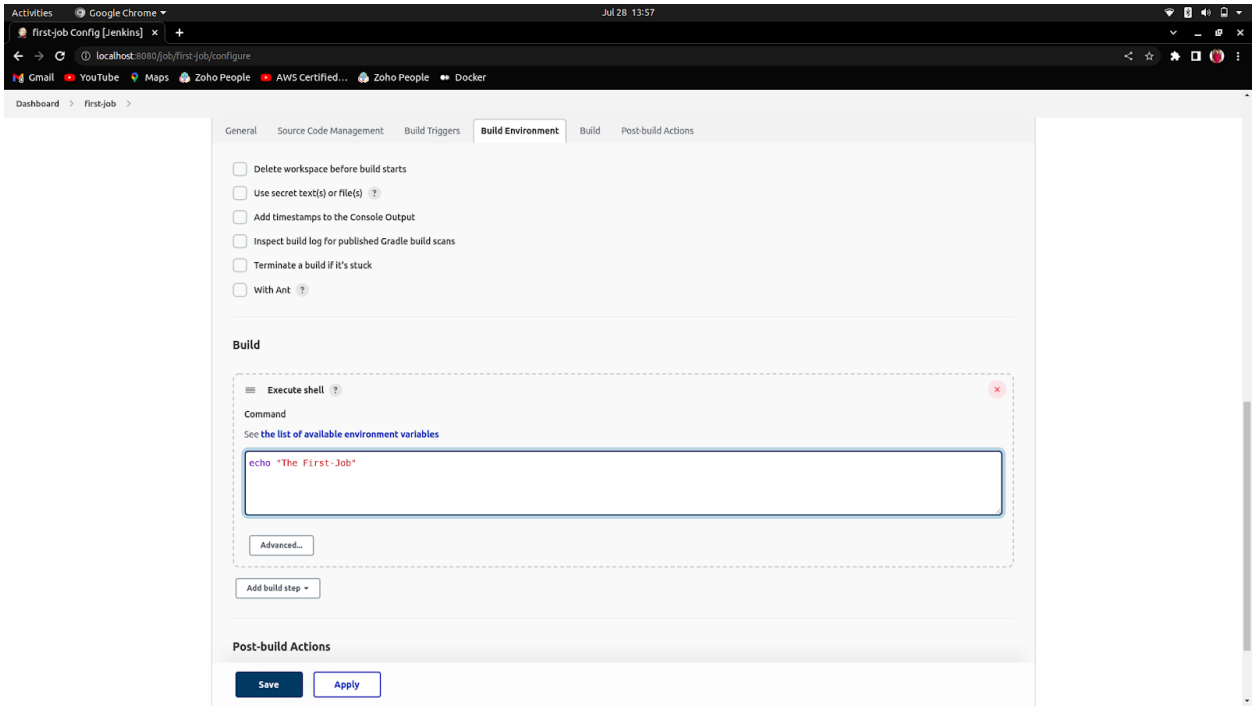


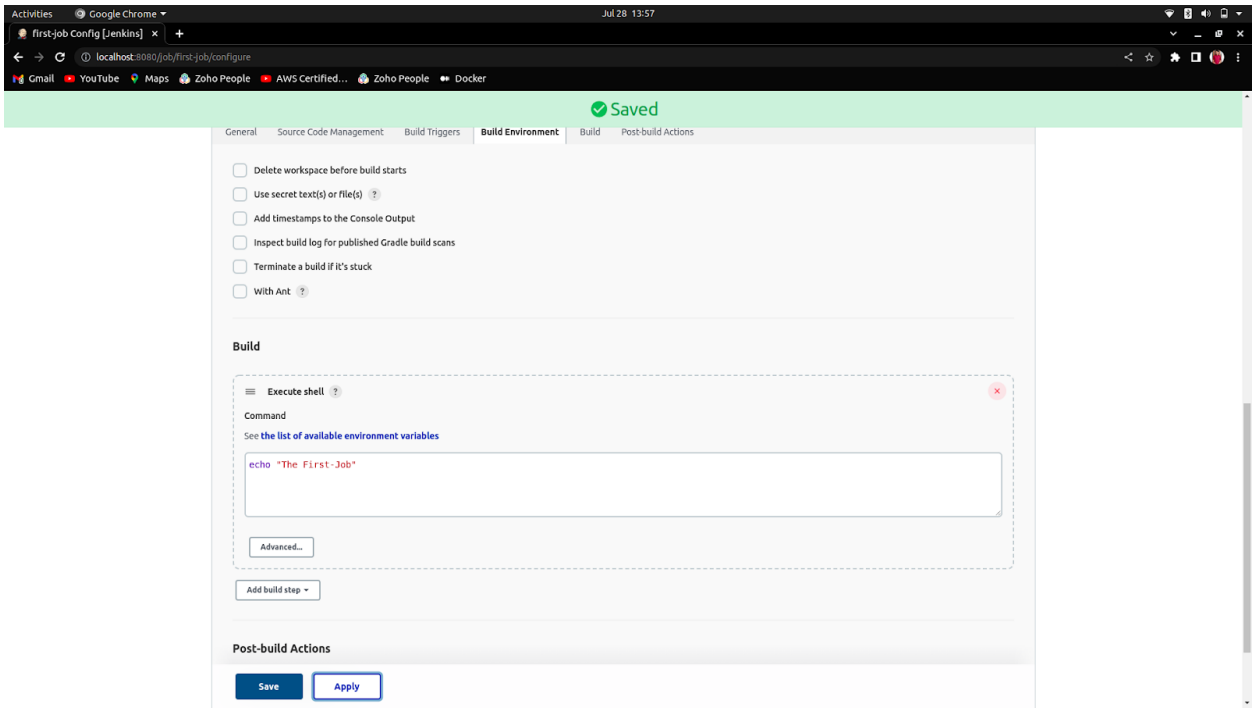
2. Enter the job name and choose <u>freestyle project.</u>



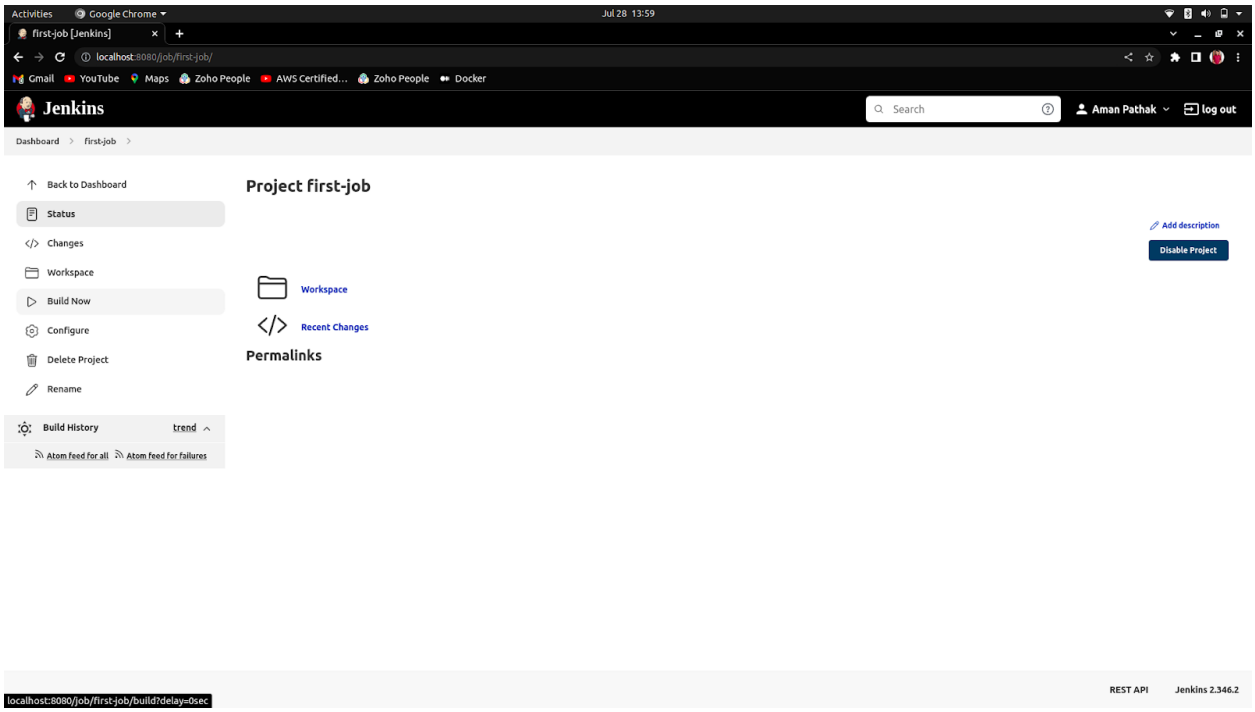
3. Follow the below steps.



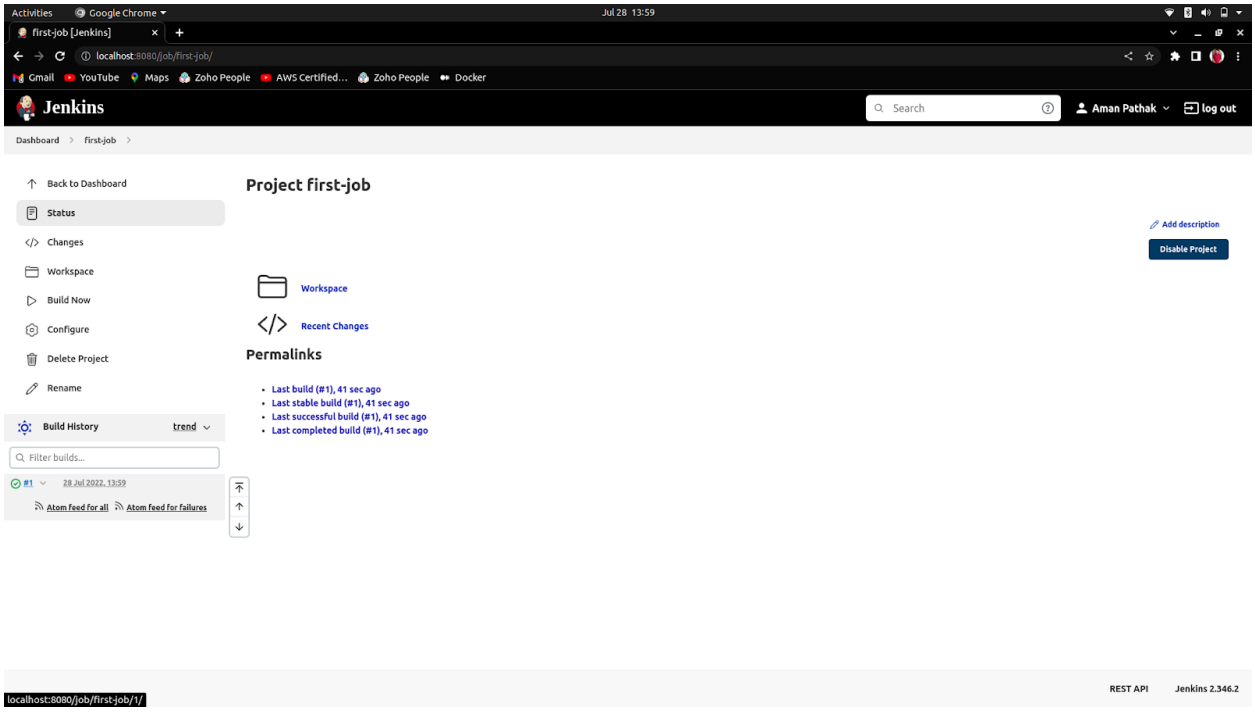

Activities
Google Chrome
Jul 28 13:57
first-job Config [Jenkins]
localhost:8080/job/first-job/configure
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
first-job
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Delete workspace before build starts
Use secret text(s) or file(s)
Add timestamps to the Console Output
Inspect build log for published Gradle build scans
Terminate a build if it's stuck
With Ant
Build
Execute shell
Command
See the list of available environment variables
echo "The First-Job"
Advanced...
Add build step
Post-build Actions
Save
Apply


Activities
Google Chrome
Jul 28 13:57
first-job Config [Jenkins]
localhost:8080/job/first-job/configure
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Saved
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Delete workspace before build starts
Use secret text(s) or file(s)
Add timestamps to the Console Output
Inspect build log for published Gradle build scans
Terminate a build if it's stuck
With Ant
Build
Execute shell
Command
See the list of available environment variables
echo "The First-Job"
Advanced...
Add build step
Post-build Actions
Save
Apply

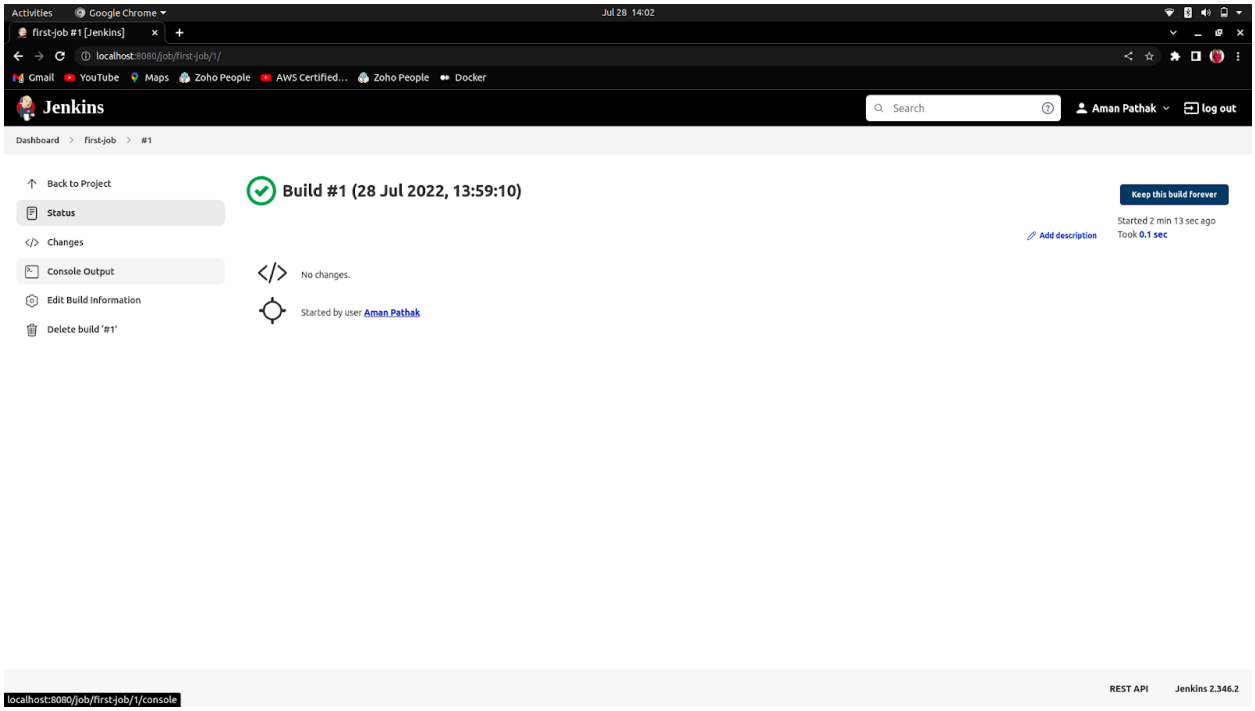
4. Click on Build Now



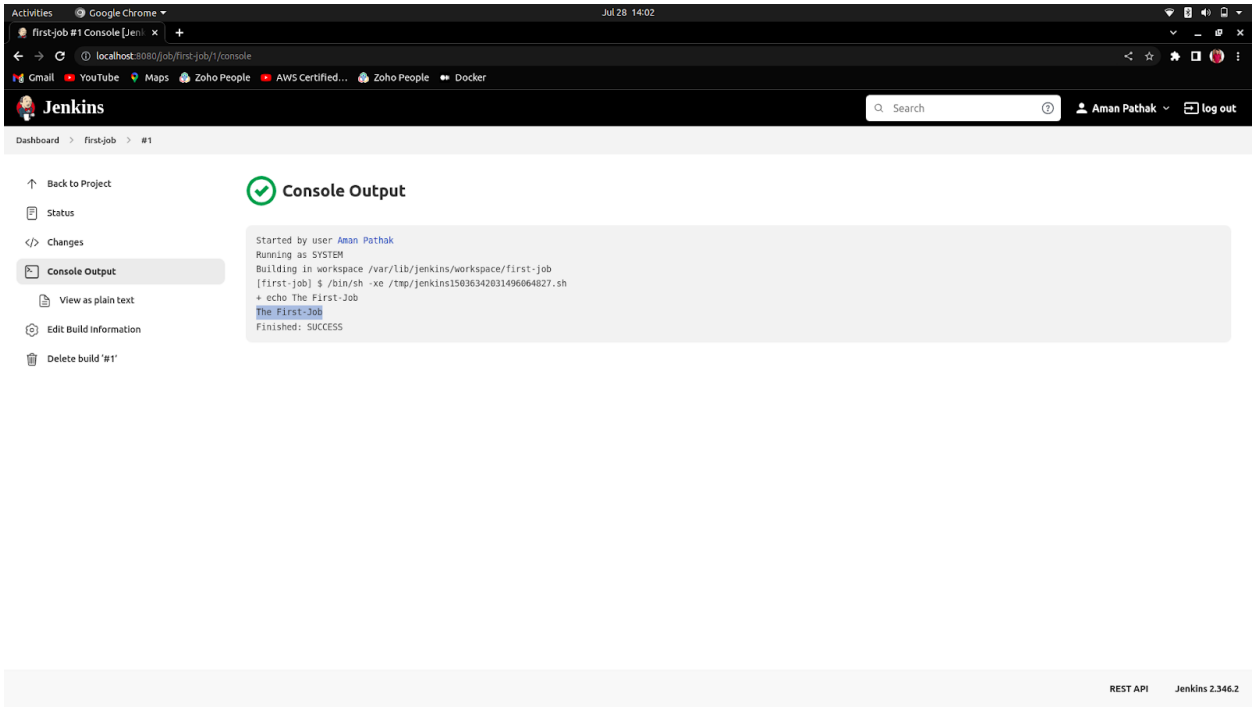
5. As the build is successful. Now click on the green color text (#1).



6. Now, To see the output of your created job. Click on the Console Output.

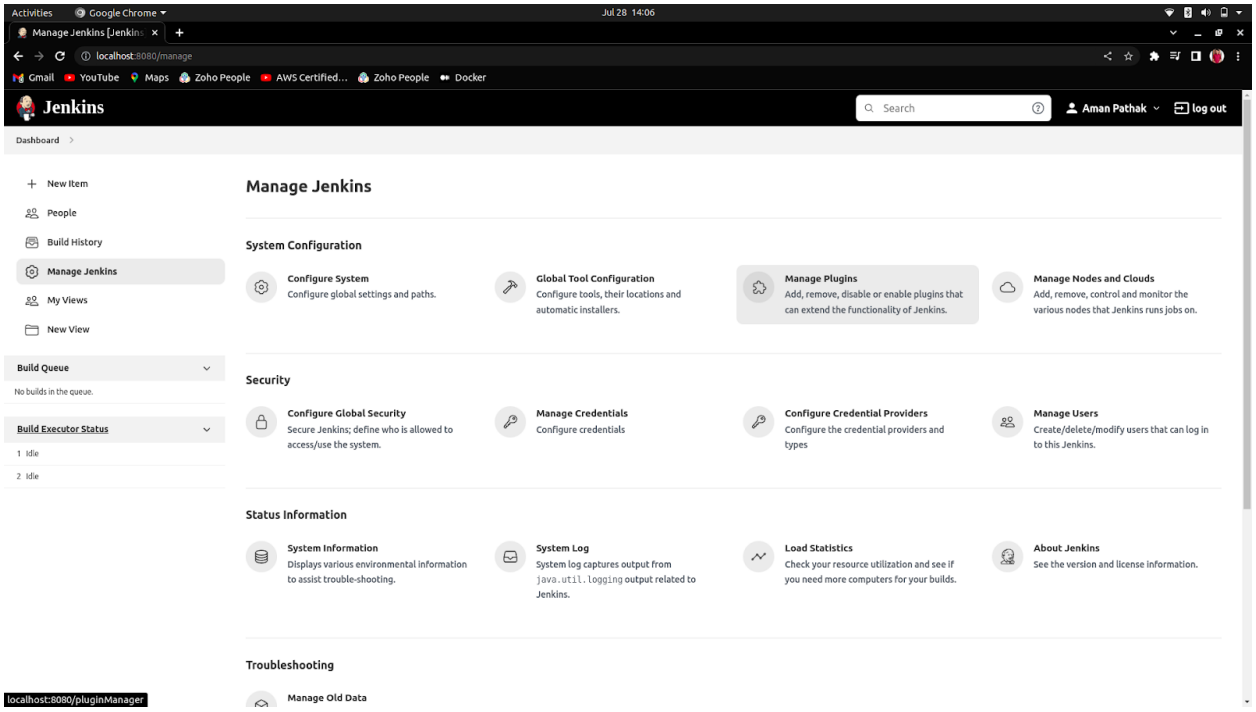


7. The Output:

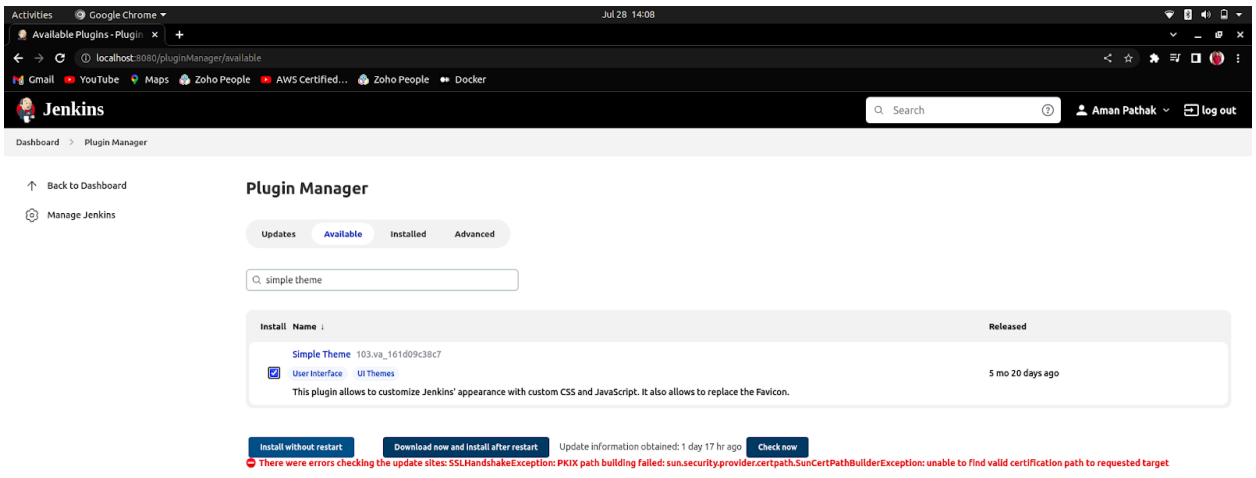


Change Jenkins Theme using Plugin

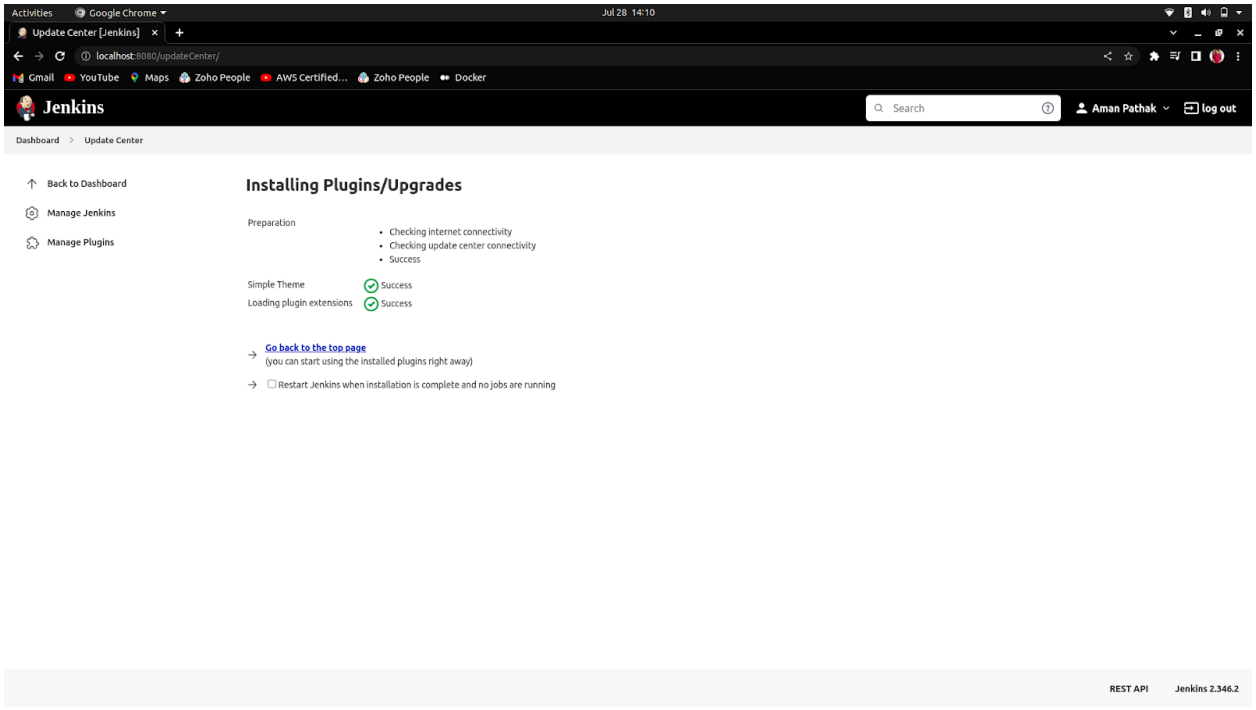
1. Click on Manage Jenkins -> Manage Plugins.

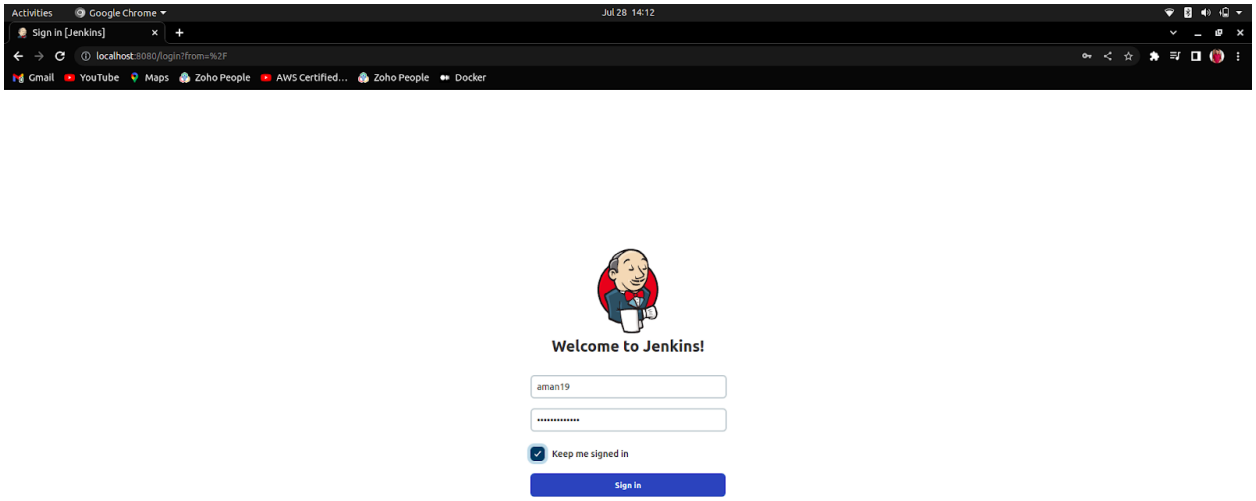


2. Now search the Simple theme plugin, checked it and install without restart.

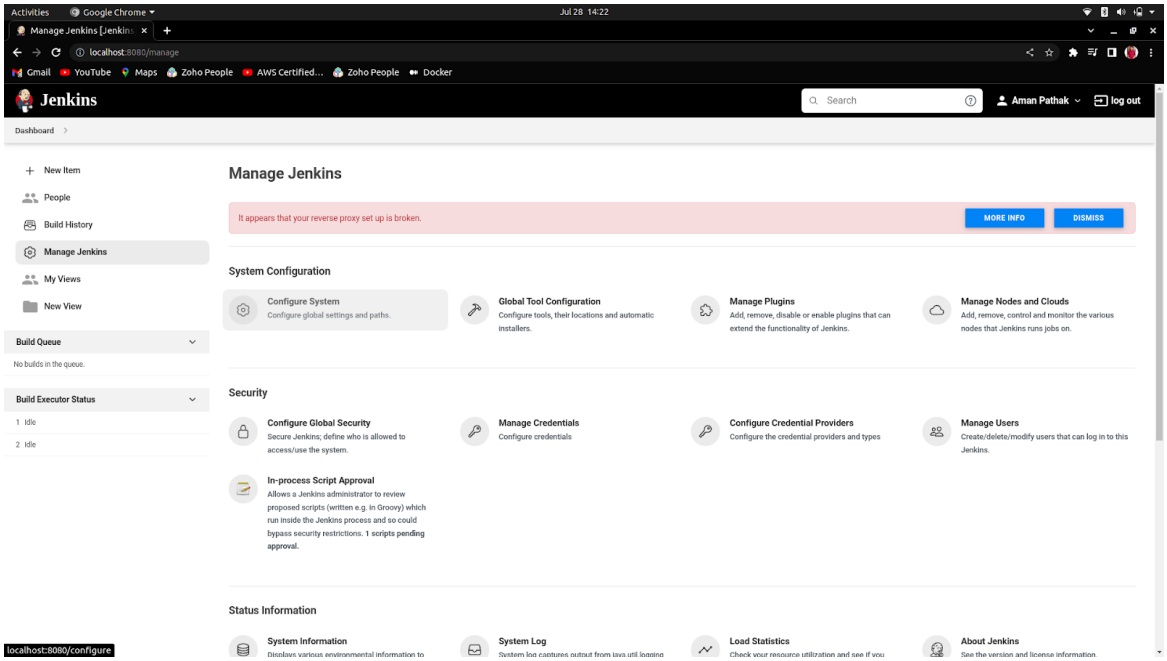


3. Now, that the plugin has been installed check the restart Jenkins server and log in again.

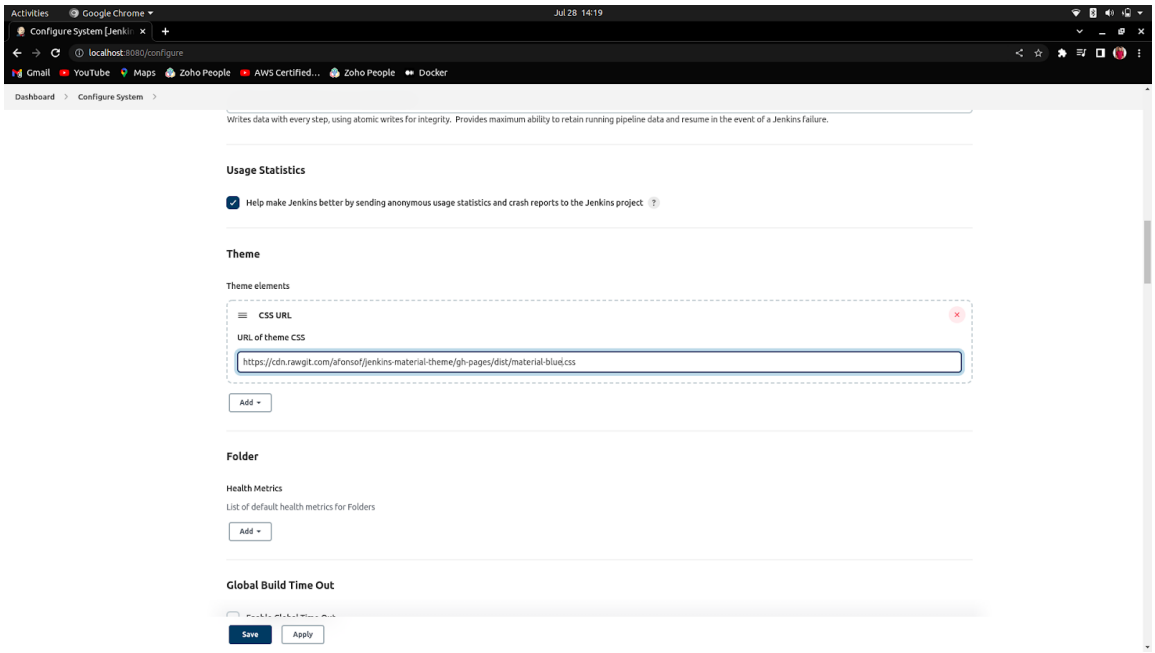




4. Now, go to the Manage Jenkins again and configure the Plugin by clicking on Configure System.



5. Now, enter the URL of CSS to change the theme and apply then, then save it.
URL Link-> Themes Link

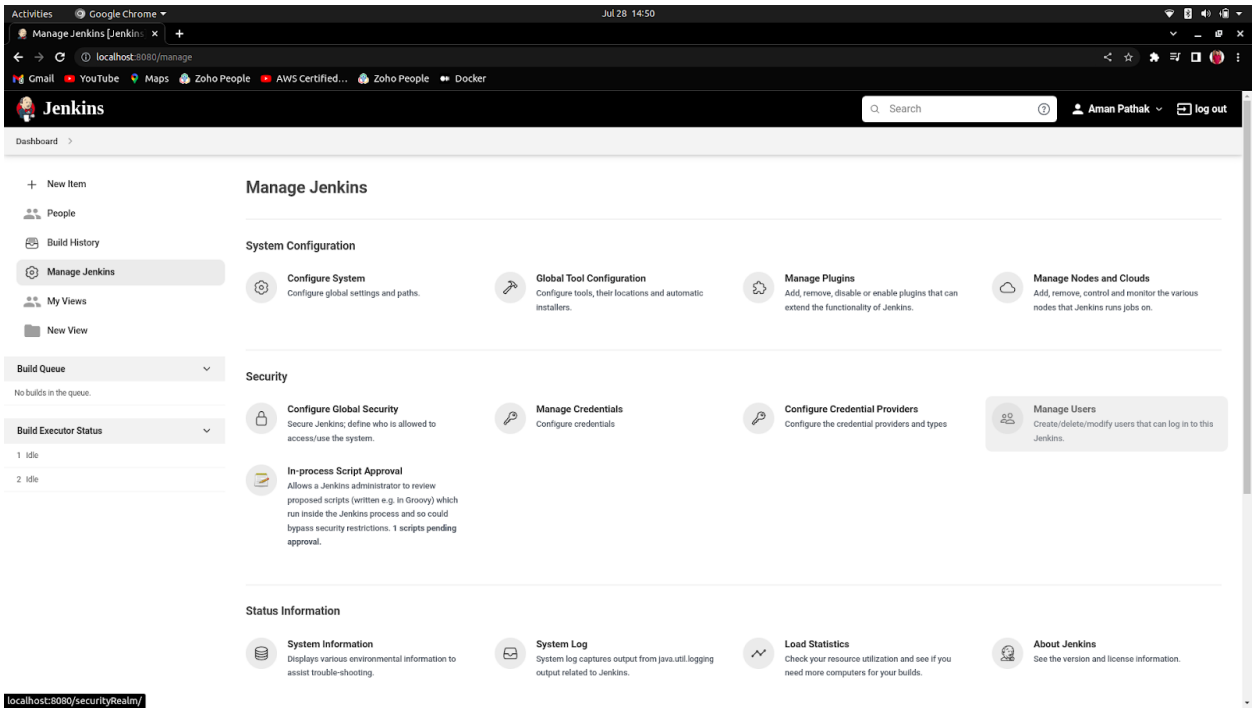


To change the Jenkins URL

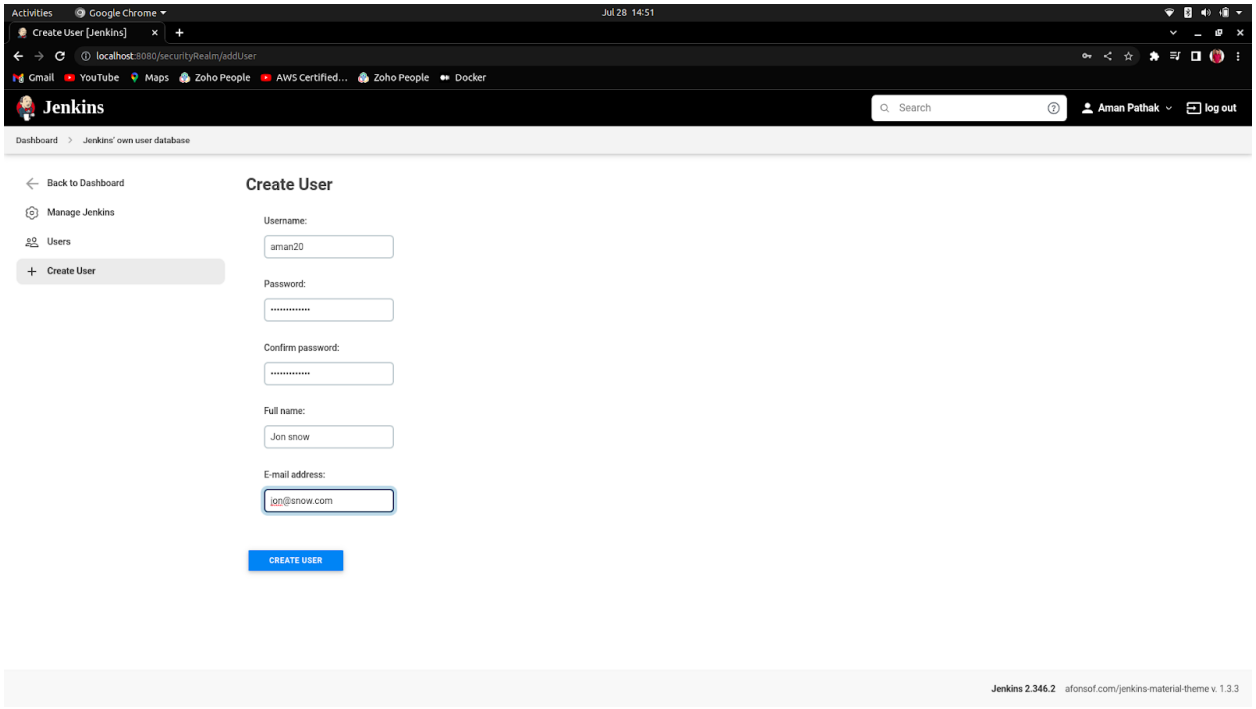
Manage Jenkins -> Configure System
Enter the desired location or URL.
By Default URL-> http://localhost:8080/

+Create a new user in Jenkins

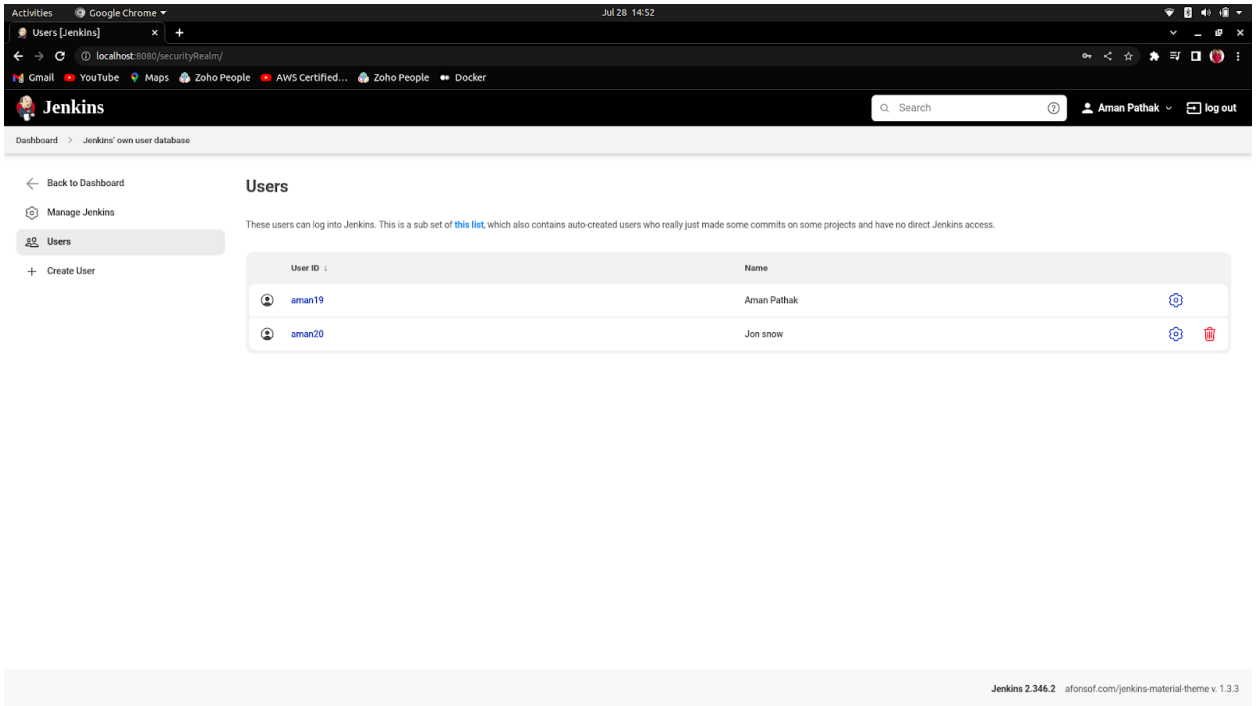
1. **Go to Manage Plugins -> Manage Users.**



2. Click on Create user. And enter the further details.



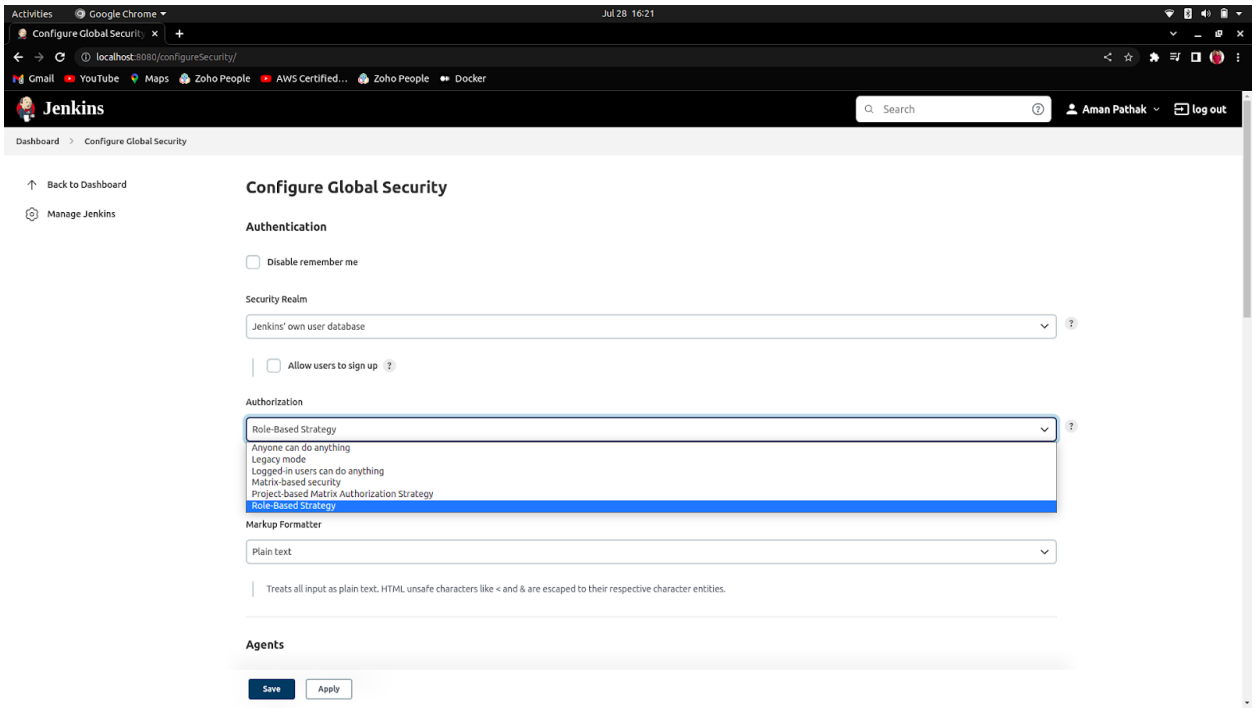
3. New User



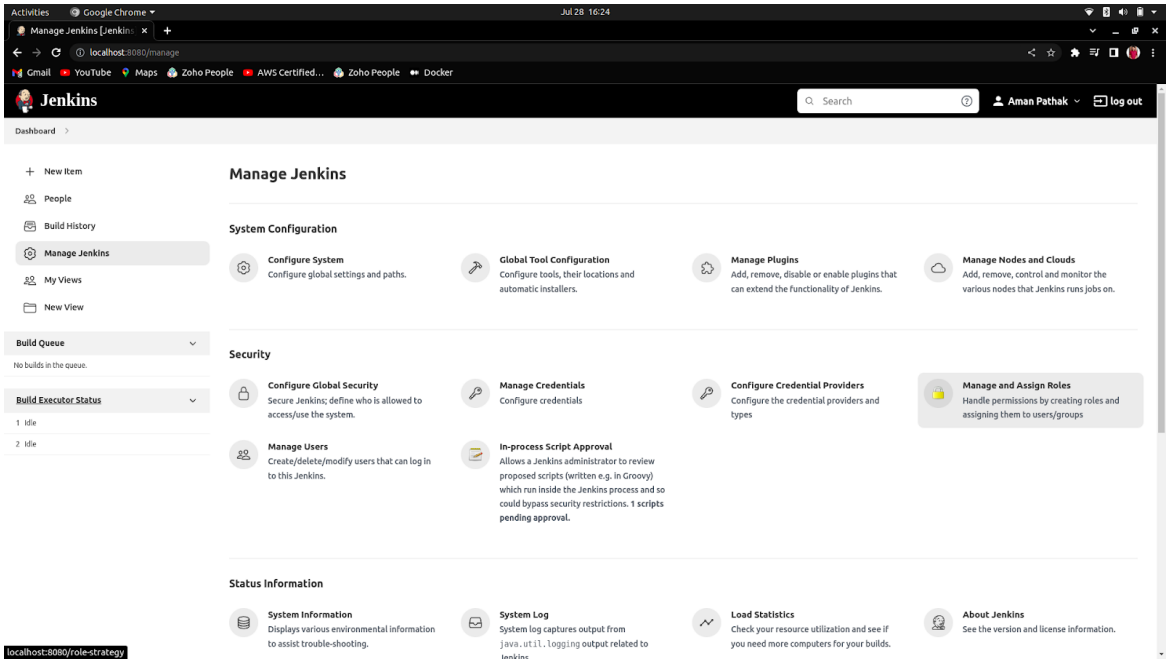
Jenkins Role-Based Access Control (RBAC)

1. **Install the plugin named <u>Role-based Authorization Strategy</u> by going into the Manage-Jenkins -> Manage Plugins - Available and then search the plugin name which is written above.**

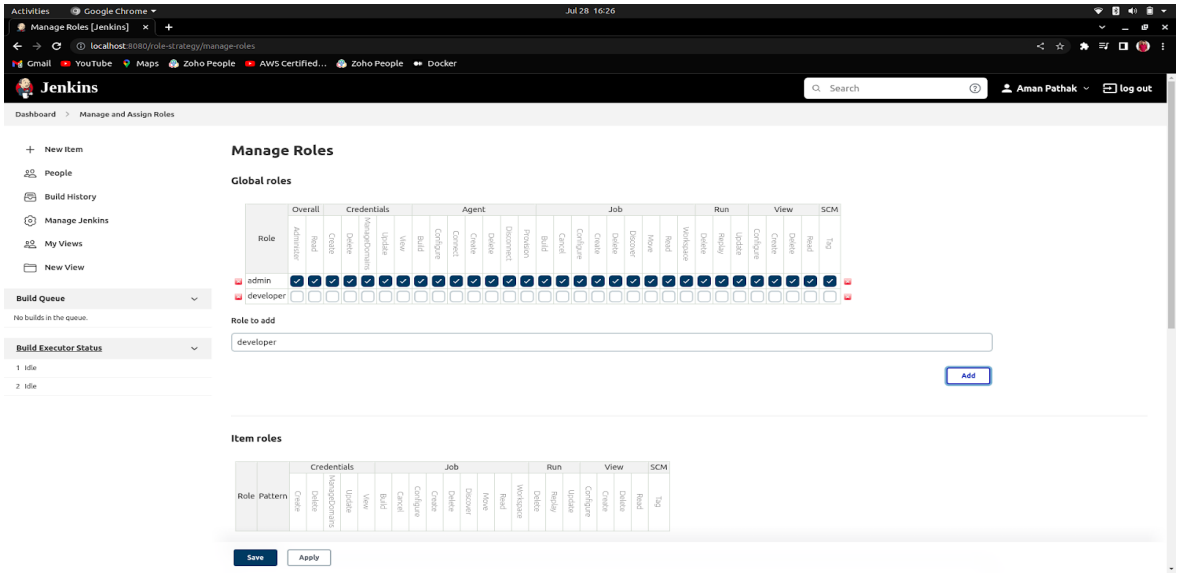
2. **Now, you have to configure this plugin by going into Manage Plugin -> Configure Global Security selecting the <u>Role-based Strategy,</u> and applying and saving the configuration.**



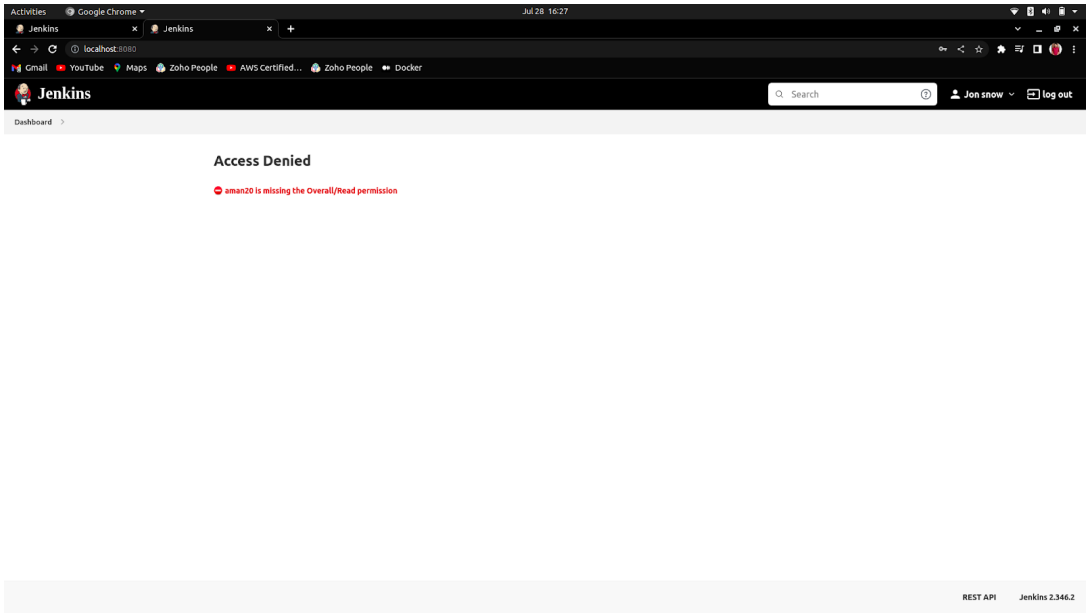
3. Now, the Manage and Assign roles option will appear under the security section in the Manage Jenkins.



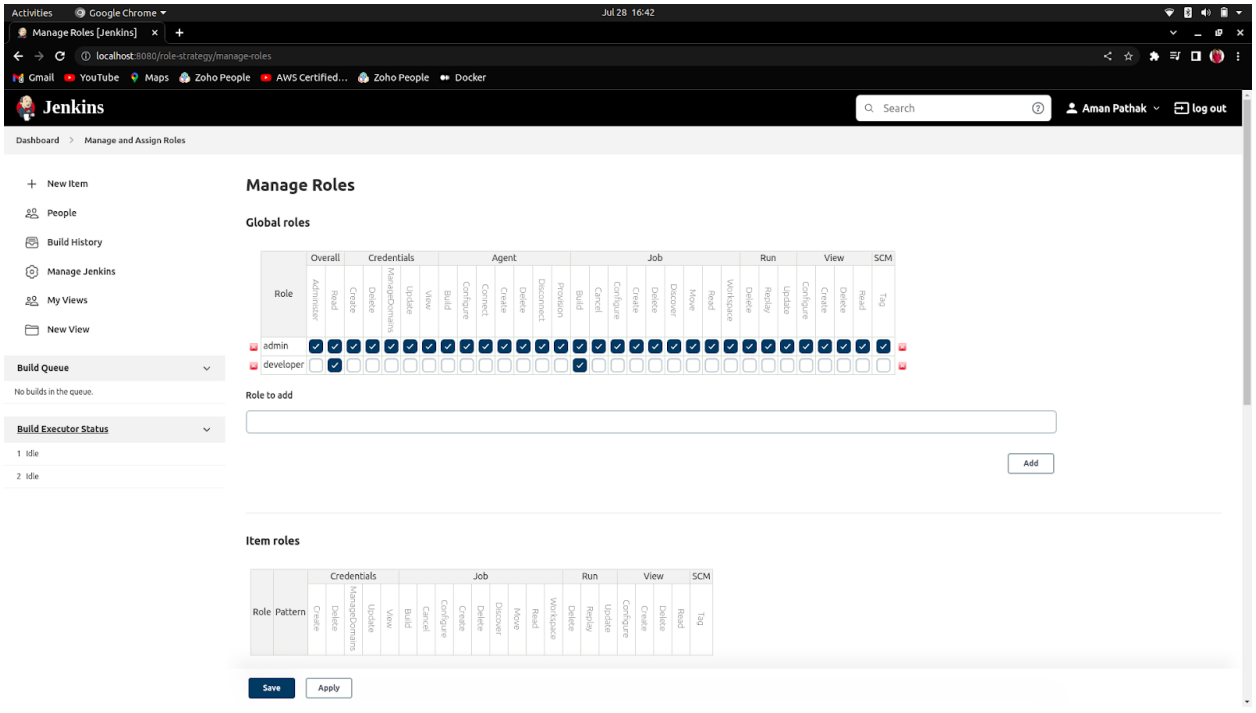
4. Now, go into the section and click on Manage Roles. Then, create the role named <developer> which has no access then apply and save.

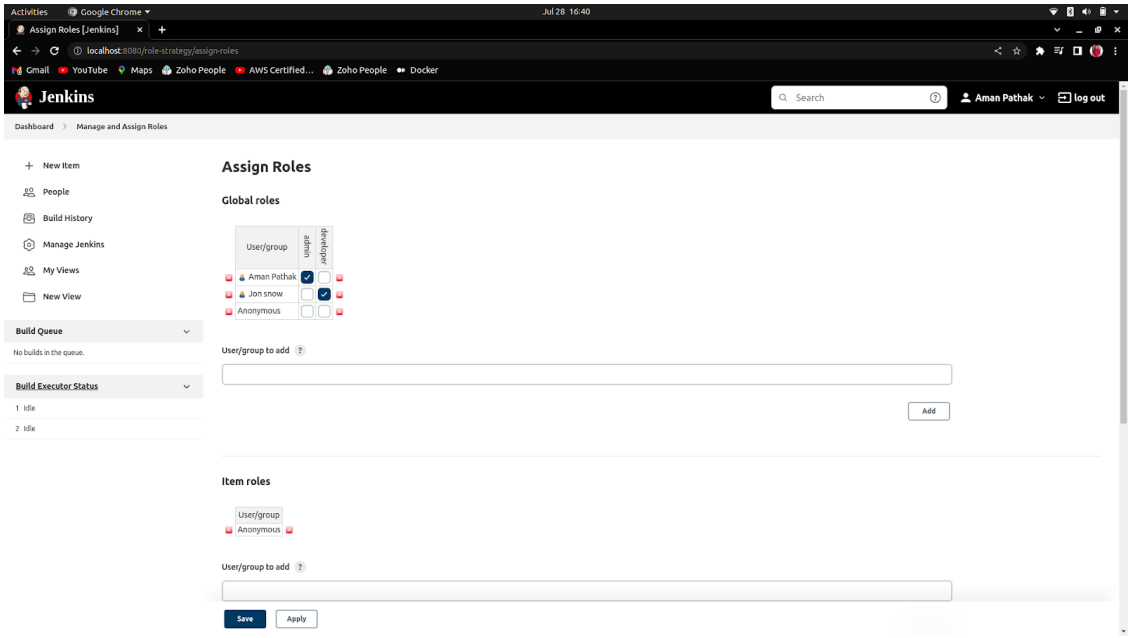


5. Now, when you log in with another user e.g. Jon Snow. You'll get this.

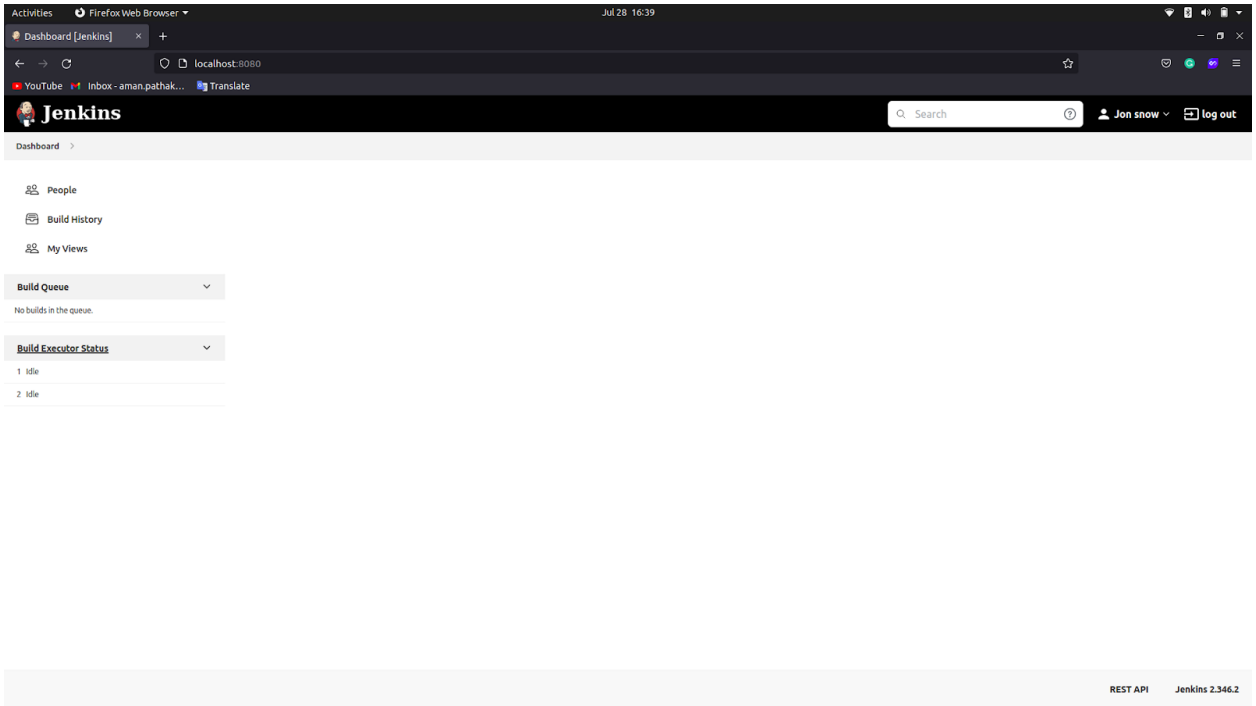


6. Now, go to the Assign Roles and add the user then, give the developer role to that user. Click on Apply and save.

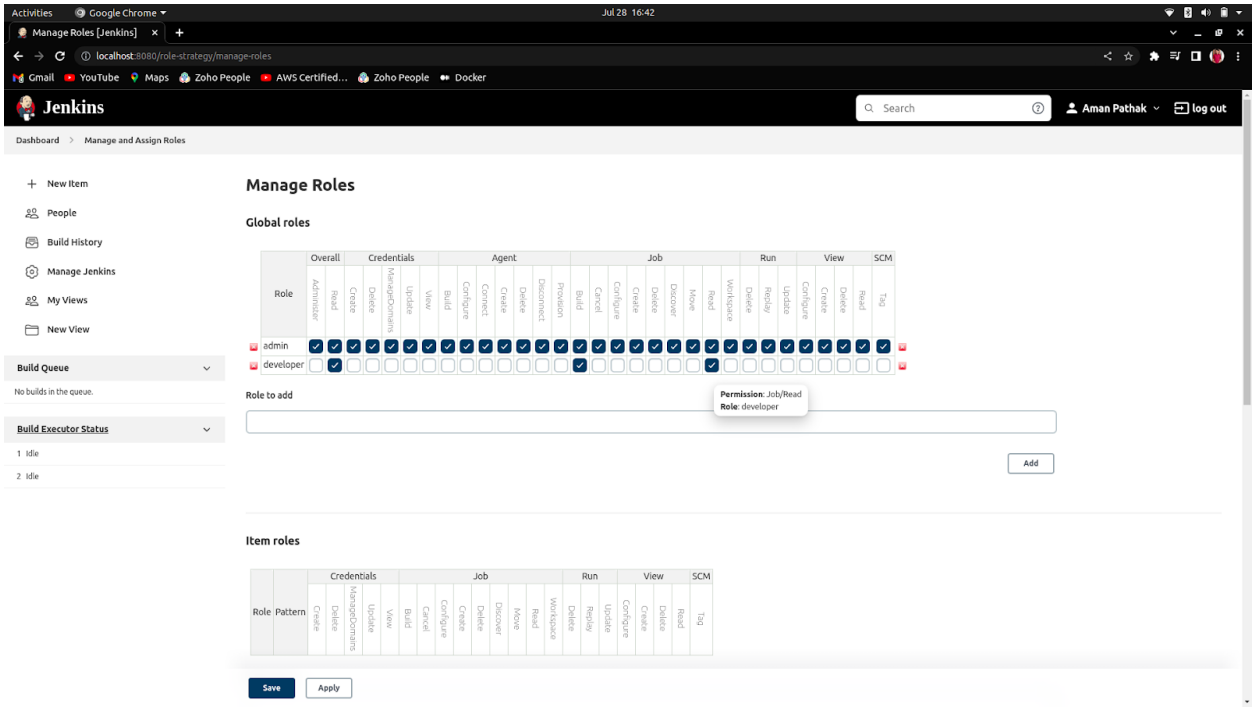


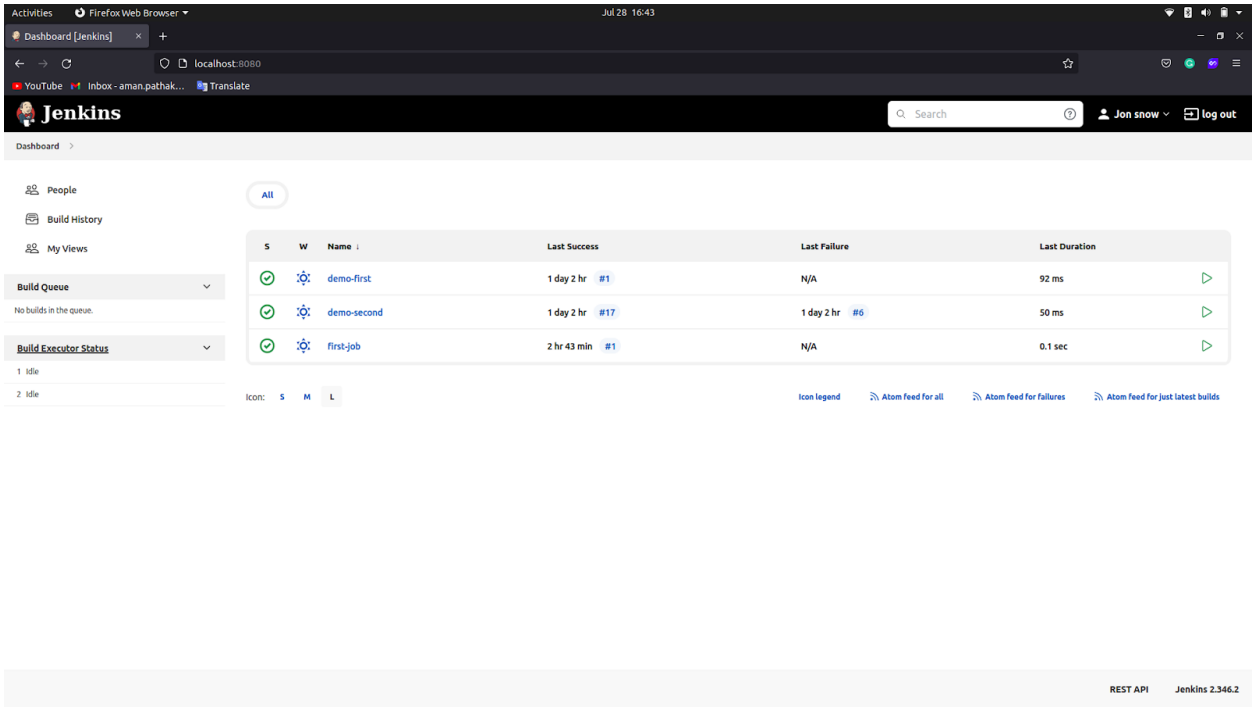


7. **If you give access by changing the roles and assigning the roles to read and build jobs to the other user, It will look like this.**



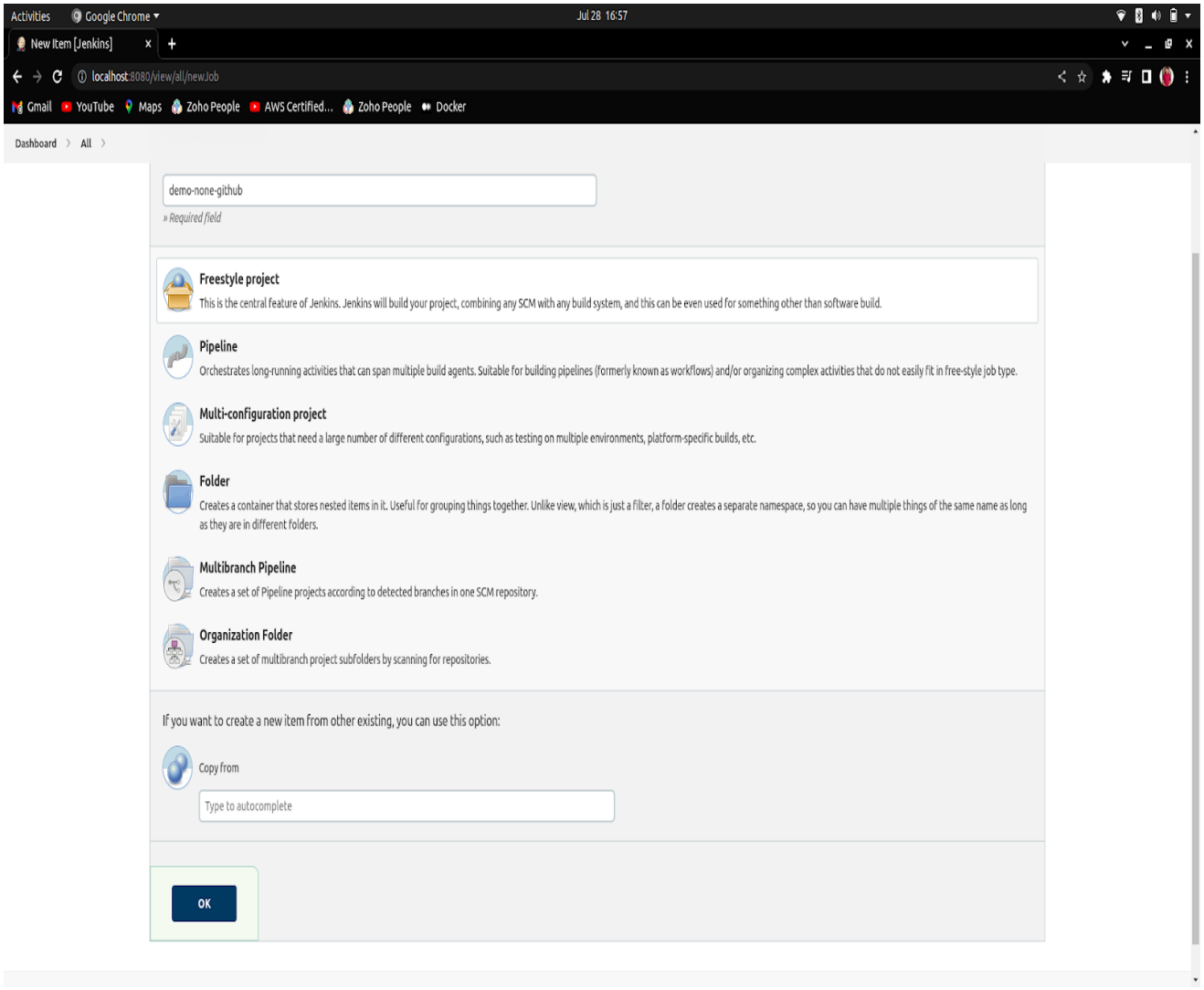
8. If you add one more permission which is job read then, the other can see the jobs.



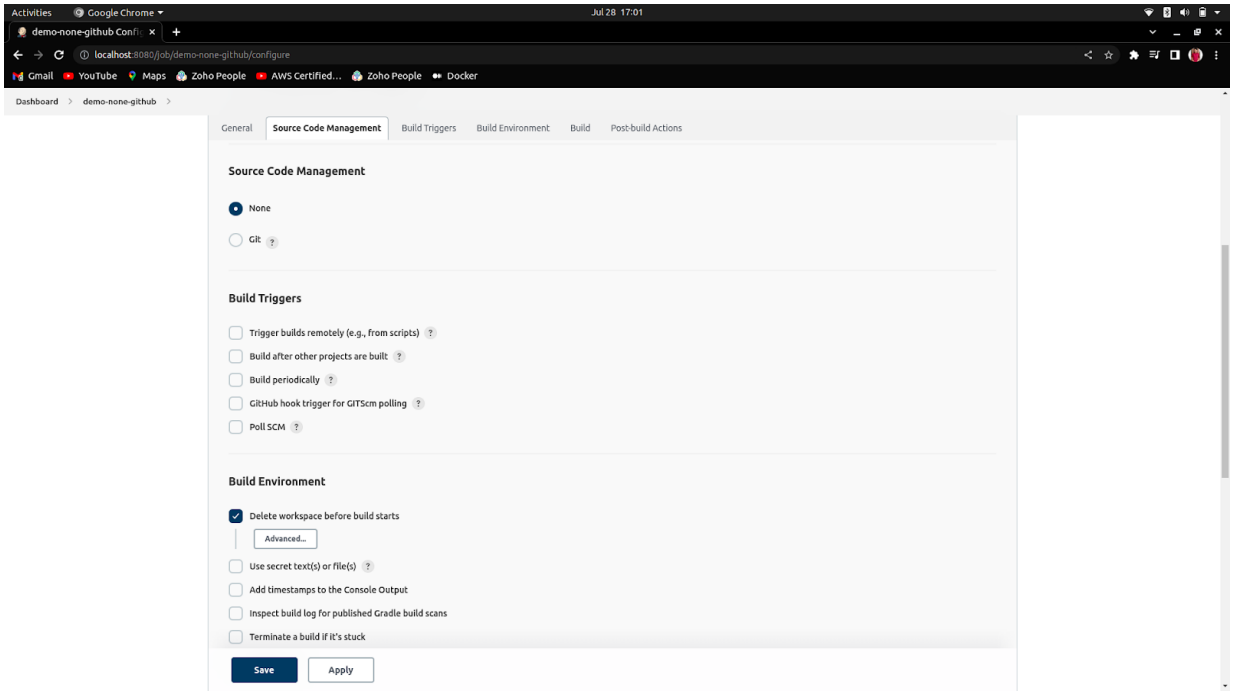


GitHub Job without GitHub Plugin

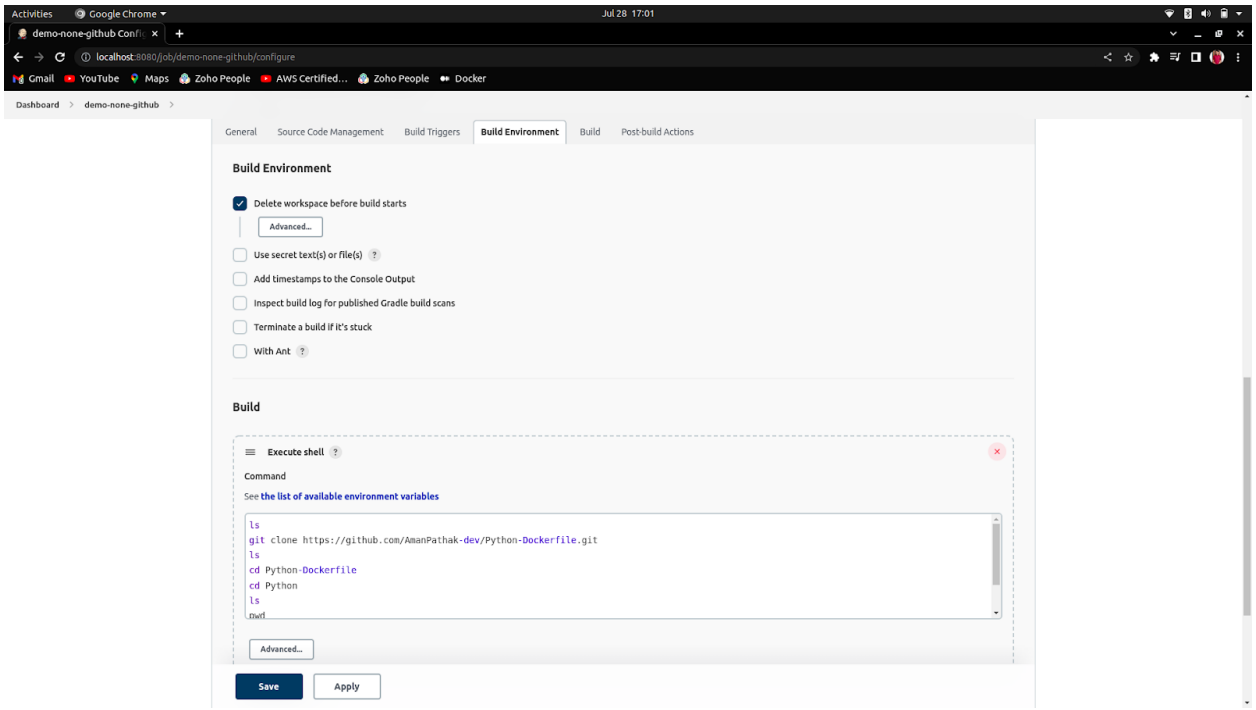
1. Create a Job with freestyle projects.



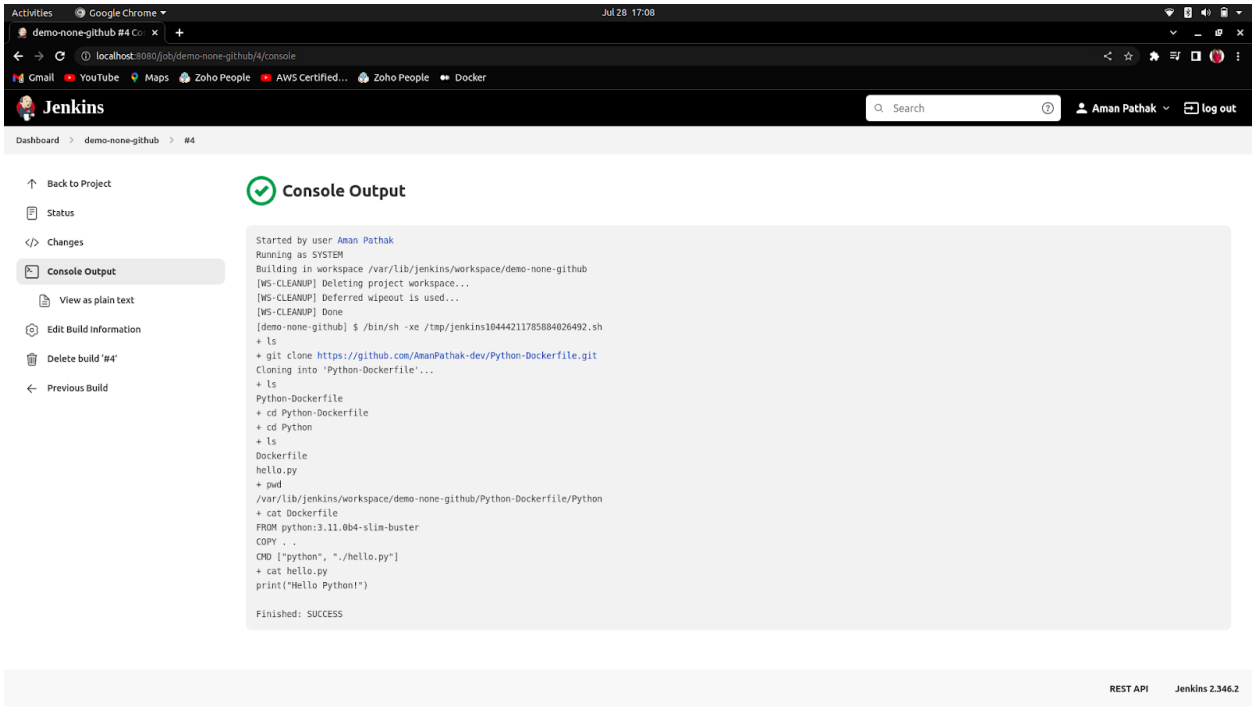
2. Take Source code Management as <u>None</u>(for not github).



3. **Build Environment -> Check the `Delete workspace before build starts`. It is checked because when you are going to build the job again it couldn't succeed with the <git clone> because the file will already exist. That's why we checked that.**
   Then, write your bash/shell script.

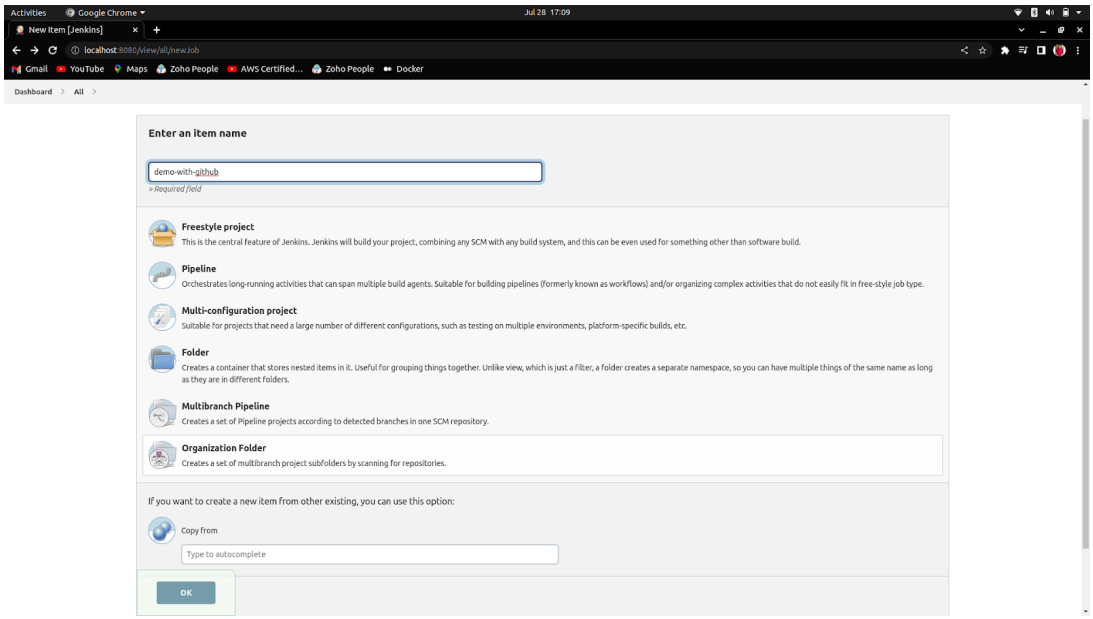


4. The Output

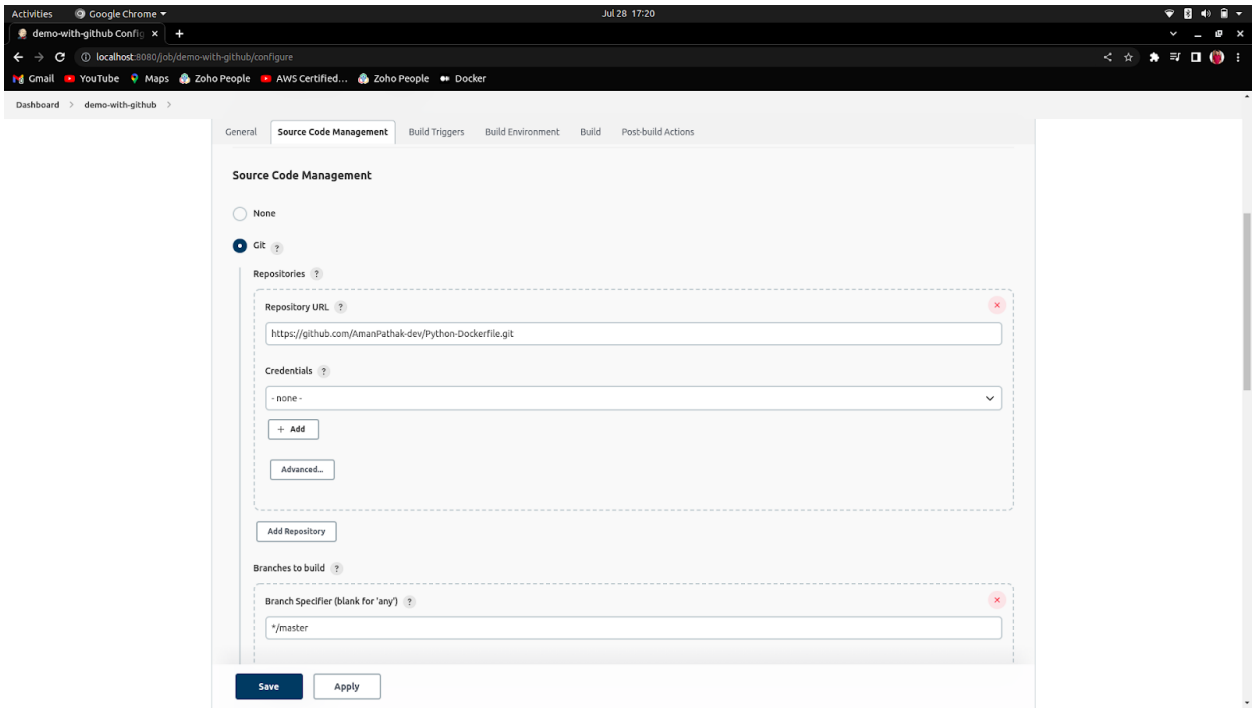

Activities
Google Chrome
Jul 28 17:08
demo-none-github #4 Co
localhost:8080/job/demo-none-github/4/console
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Jenkins
Search
Aman Pathak
log out
Dashboard
demo-none-github
#4
Back to Project
Status
Changes
Console Output
View as plain text
Edit Build Information
Delete build '#4'
Previous Build
Console Output
Started by user Aman Pathak
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/demo-none-github
[WS-CLEANUP] Deleting project workspace...
[WS-CLEANUP] Deferred wipeout is used...
[WS-CLEANUP] Done
[demo-none-github] $ /bin/sh -xe /tmp/jenkins10444211785884026492.sh
+ ls
+ git clone https://github.com/AmanPathak-dev/Python-Dockerfile.git
Cloning into 'Python-Dockerfile'...
+ ls
Python-Dockerfile
+ cd Python-Dockerfile
+ cd Python
+ ls
Dockerfile
hello.py
+ pwd
/var/lib/jenkins/workspace/demo-none-github/Python-Dockerfile/Python
+ cat Dockerfile
FROM python:3.11.0b4-slim-buster
COPY . .
CMD ["python", "./hello.py"]
+ cat hello.py
print("Hello Python!")
Finished: SUCCESS
REST API
Jenkins 2.346.2

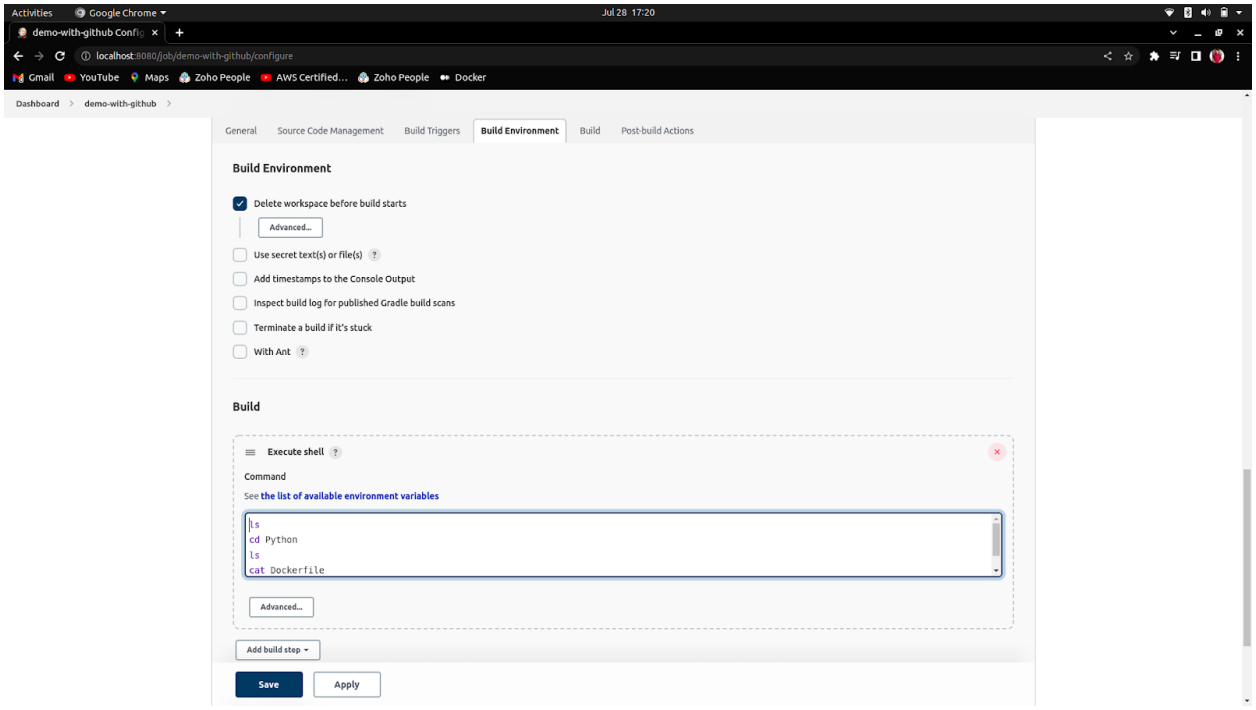
GitHub Job with GitHub Plugin

1. Create a freestyle project.

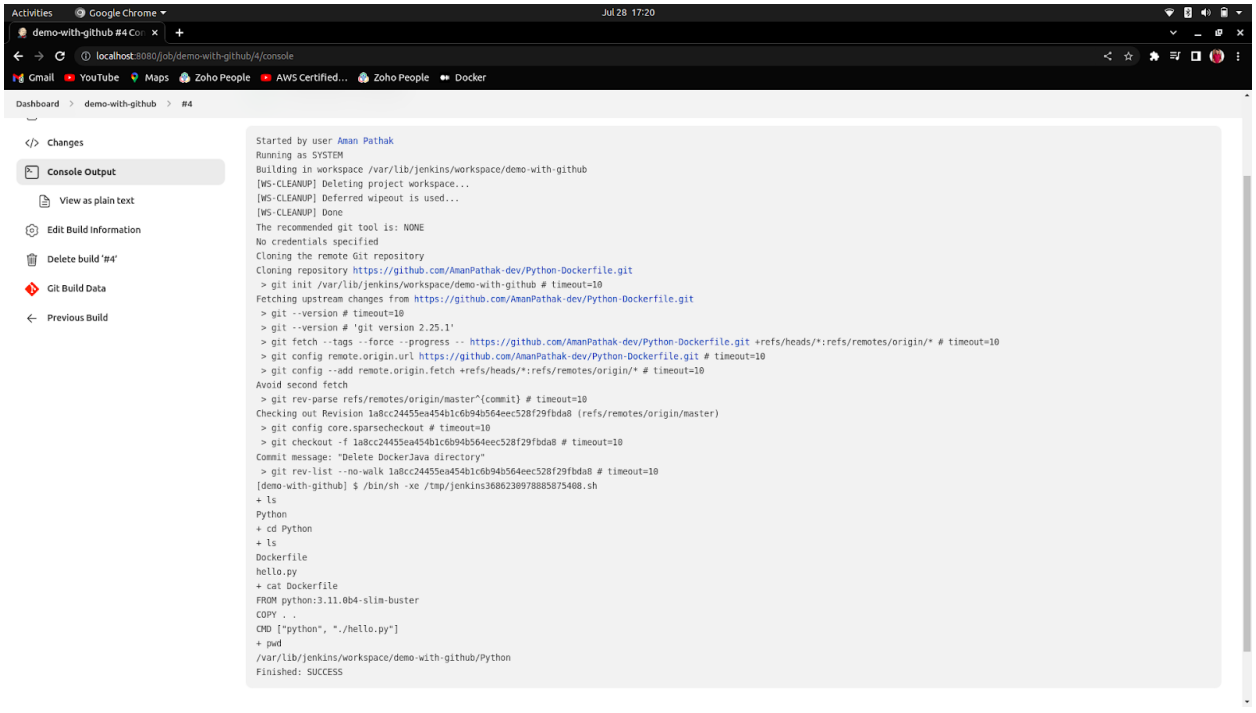


2. Source code Management must be GitHub.
   And add the further details as written below.



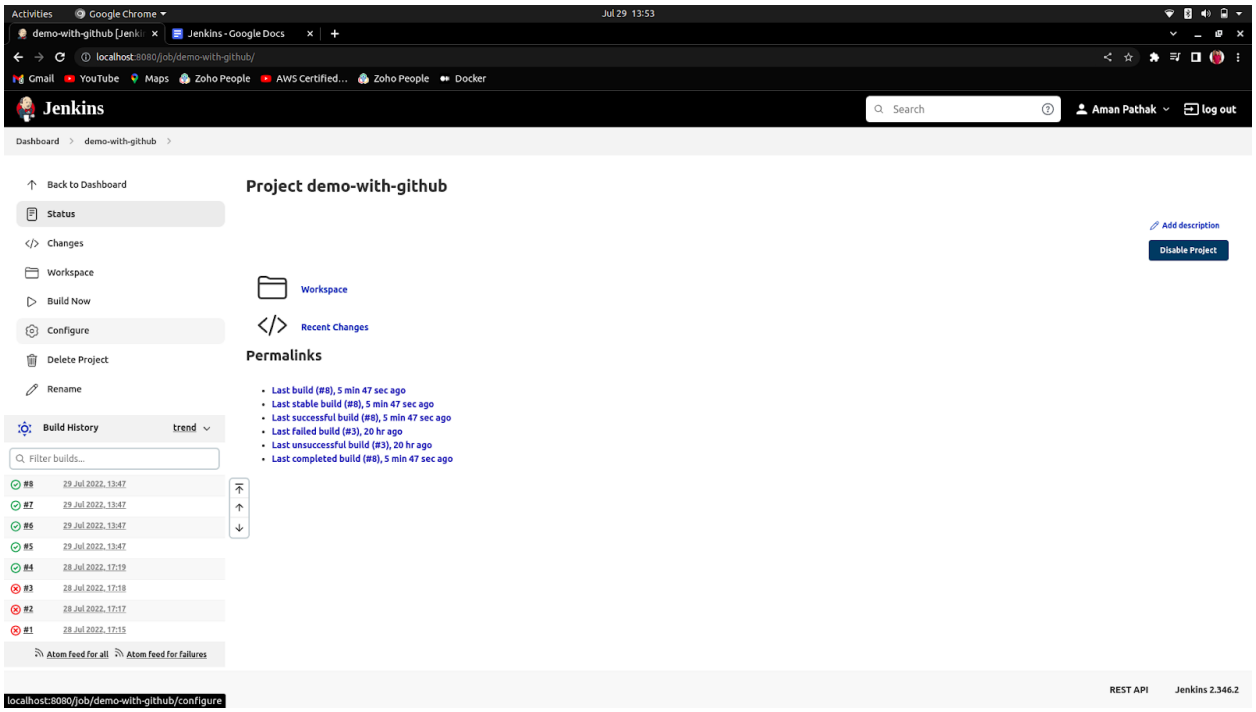

Activities
Google Chrome
Jul 28 17:20
demo-with-github Config
localhost:8080/job/demo-with-github/configure
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
demo-with-github
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Build Environment
Delete workspace before build starts
Advanced...
Use secret text(s) or file(s)
Add timestamps to the Console Output
Inspect build log for published Gradle build scans
Terminate a build if it's stuck
With Ant
Build
Execute shell
Command
See the list of available environment variables
ls
cd Python
ls
cat Dockerfile
Advanced...
Add build step
Save
Apply

The Output

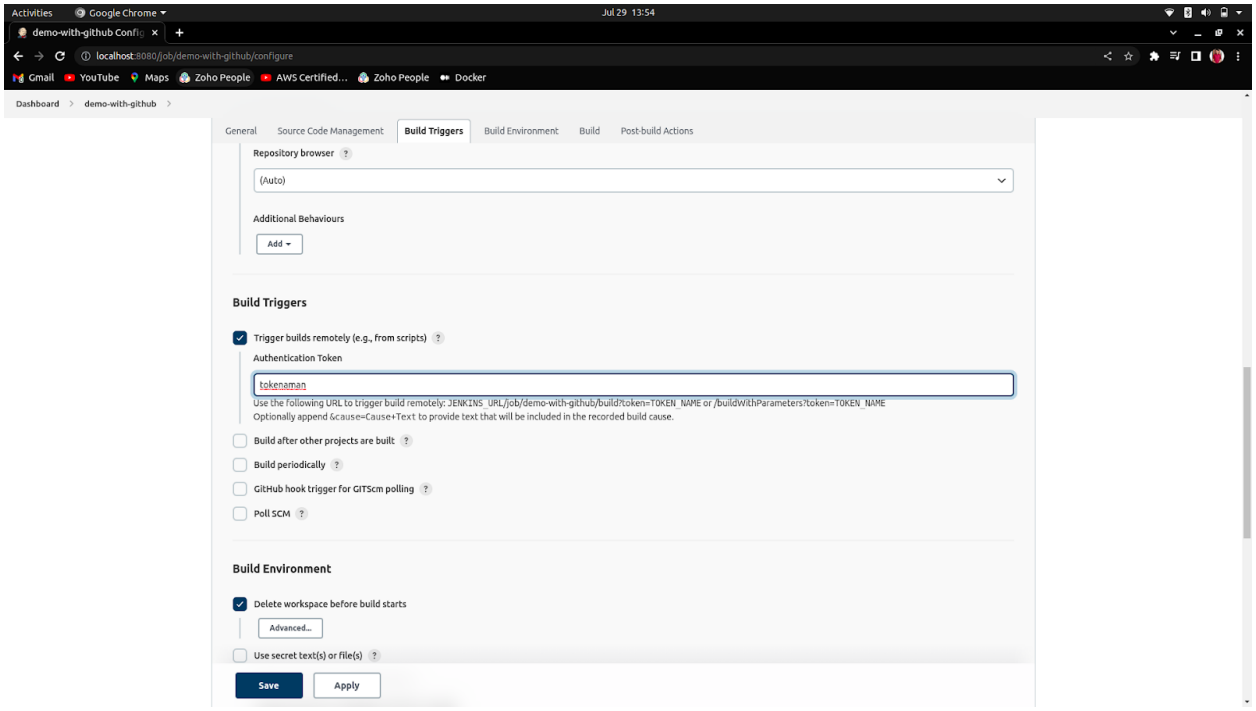

Activities
Google Chrome
Jul 28 17:20
demo-with-github #4 Co
localhost:8080/job/demo-with-github/4/console
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
demo-with-github
#4
Changes
Console Output
View as plain text
Edit Build Information
Delete build '#4'
Git Build Data
Previous Build
Started by user Aman Pathak
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/demo-with-github
[WS-CLEANUP] Deleting project workspace...
[WS-CLEANUP] Deferred wipeout is used...
[WS-CLEANUP] Done
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/AmanPathak-dev/Python-Dockerfile.git
 > git init /var/lib/jenkins/workspace/demo-with-github # timeout=10
Fetching upstream changes from https://github.com/AmanPathak-dev/Python-Dockerfile.git
 > git --version # timeout=10
 > git --version # 'git version 2.25.1'
 > git fetch --tags --force --progress -- https://github.com/AmanPathak-dev/Python-Dockerfile.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://github.com/AmanPathak-dev/Python-Dockerfile.git # timeout=10
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision 1a8cc24455ea454b1c6b94b564eec528f29fbda8 (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 1a8cc24455ea454b1c6b94b564eec528f29fbda8 # timeout=10
Commit message: "Delete DockerJava directory"
 > git rev-list --no-walk 1a8cc24455ea454b1c6b94b564eec528f29fbda8 # timeout=10
[demo-with-github] $ /bin/sh -xe /tmp/jenkins3686230978885875408.sh
+ ls
Python
+ cd Python
+ ls
Dockerfile
hello.py
+ cat Dockerfile
FROM python:3.11.0b4-slim-buster
COPY . .
CMD ["python", "./hello.py"]
+ pwd
/var/lib/jenkins/workspace/demo-with-github/Python
Finished: SUCCESS

How to build a job using Trigger Builds remotely (Authentication token)

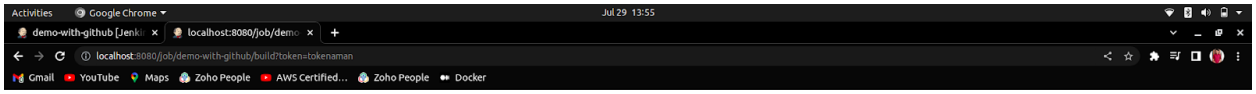
1. Go to any Job.



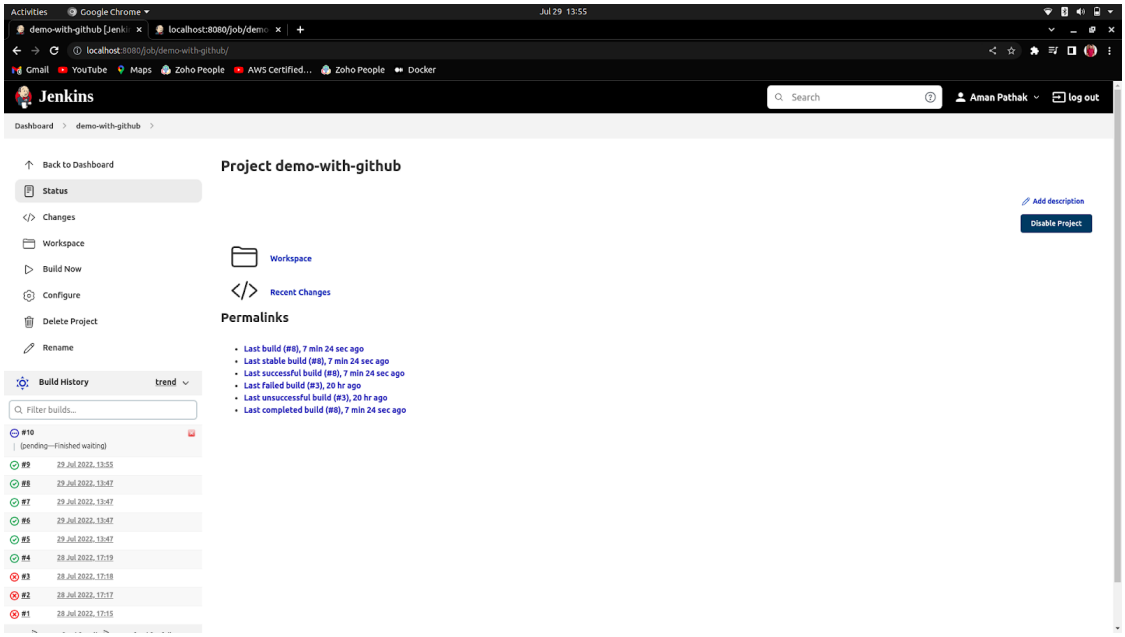
2. Enter the token name inside the Trigger build remotely section.



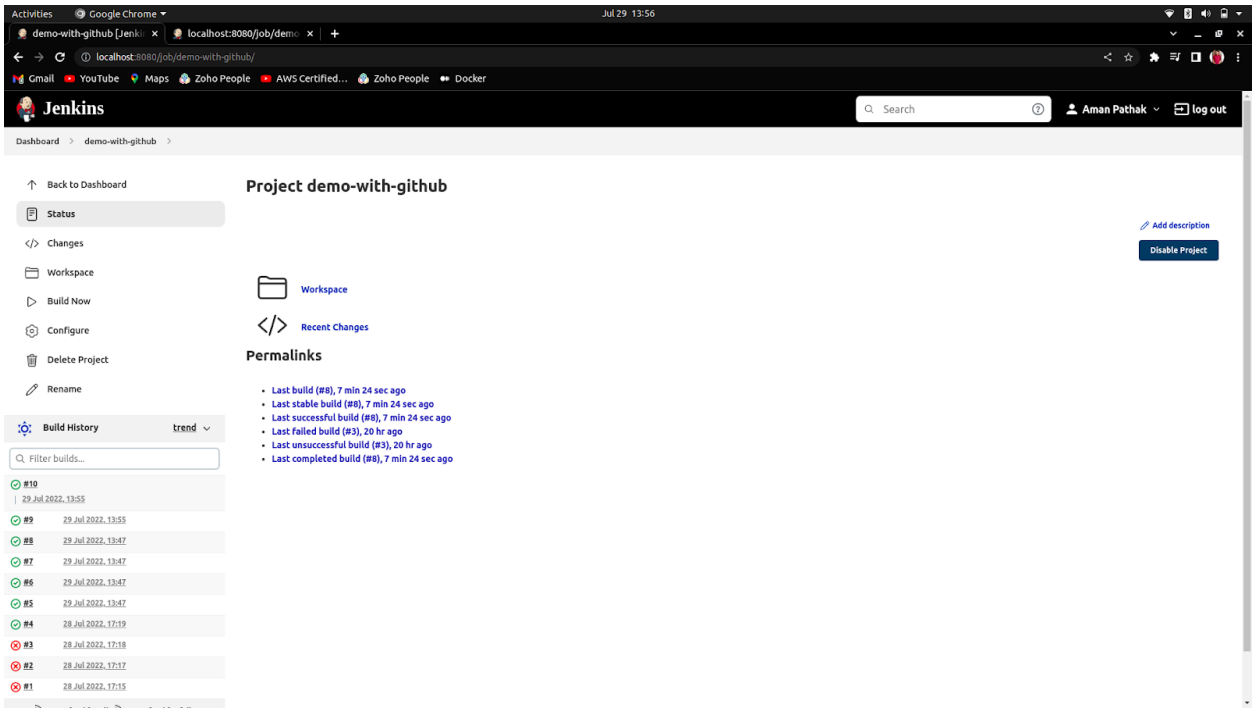
3. Now, cop the URL that was under the token name field and paste in the new tab



4. Now, come to the Jenkins dashboard.
   You will see the build has been automatically started.



Build Successful



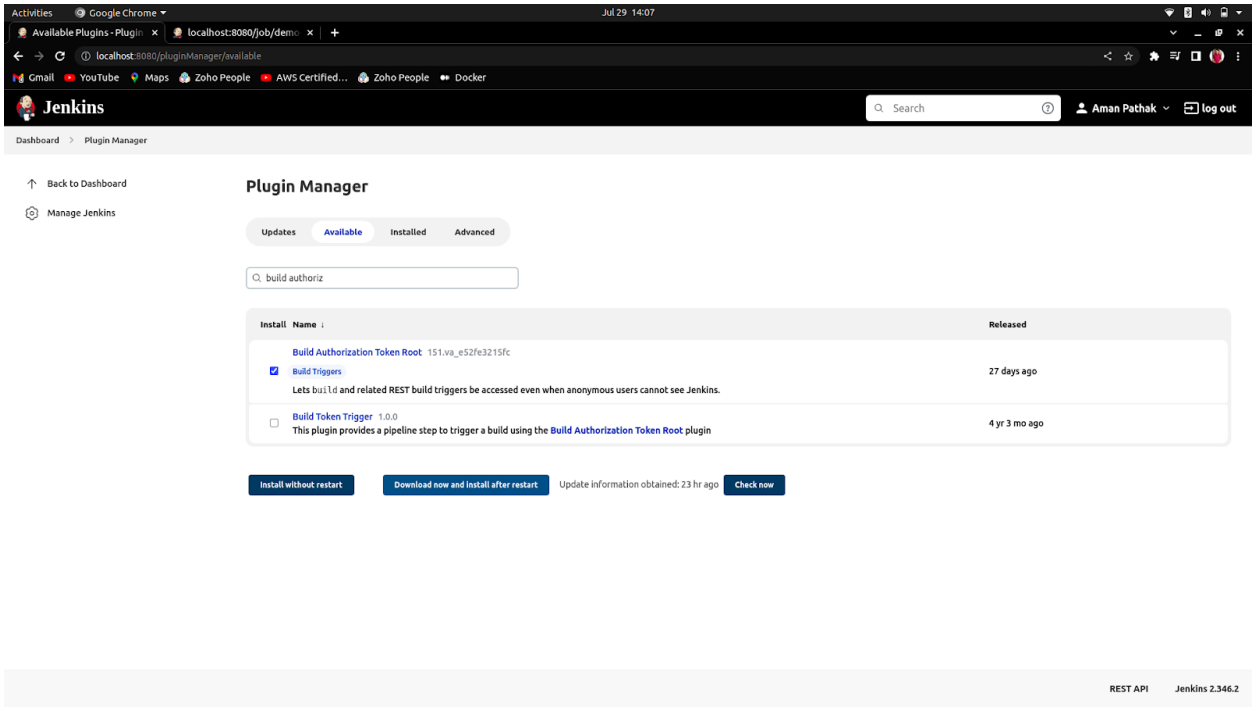
Build Triggers

1. Build Job through Incognito Mode and Terminal

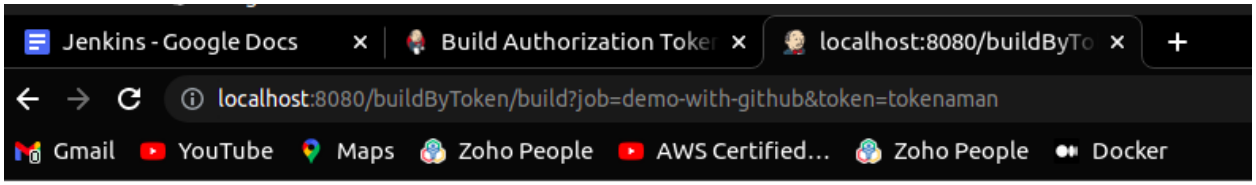
Why?

As I am building a job through just a URL. But when I put the same URL in incognito mode of any browser it asked me to log in again, to resolve that issue. Follow the steps below:

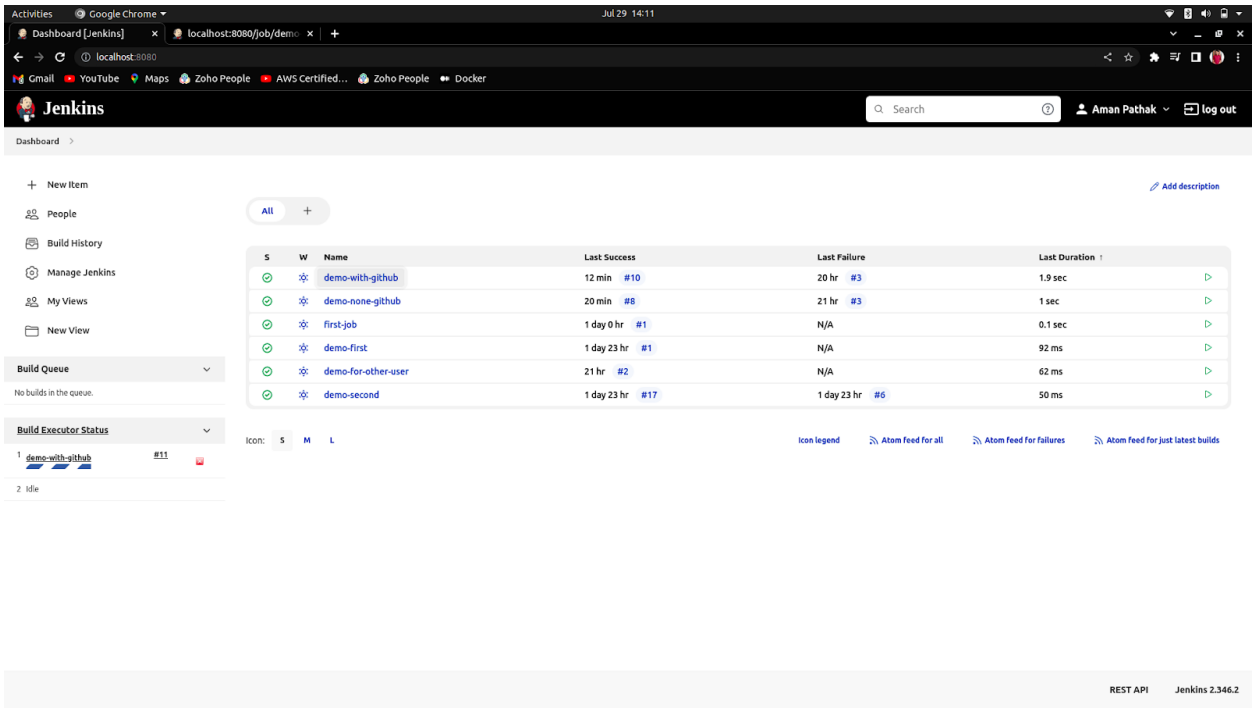
1. First of all, I need to install a plugin named *Build Authorization Token Root* as [Build Authorization Token Link Official Plugin]



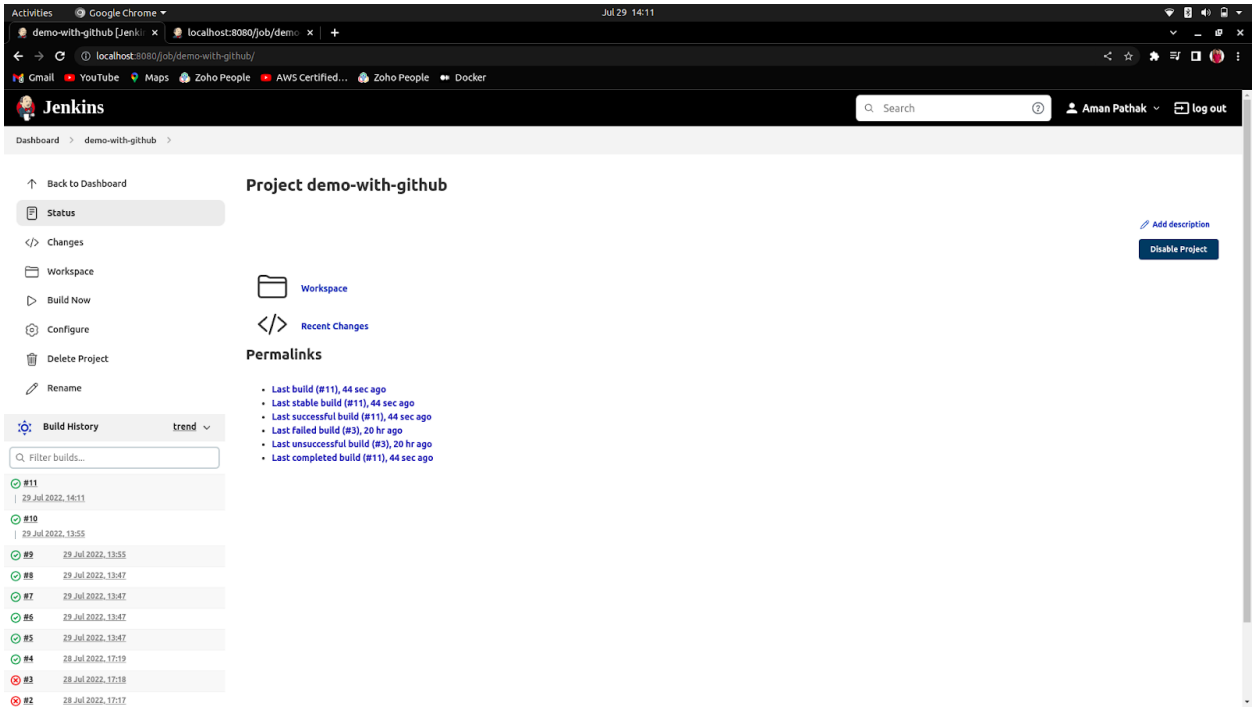
2. Enter the URL which is written in the tab.



3. The result you can see.



The build has been successful.

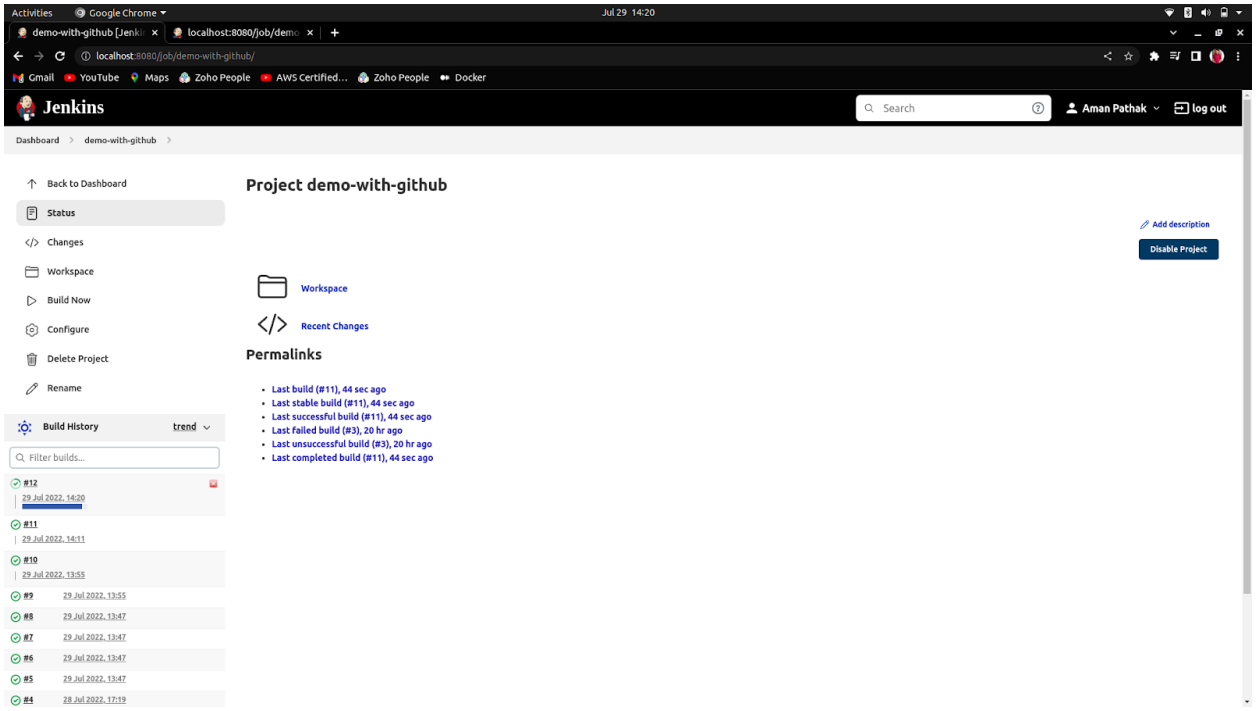


Through Terminal

1. Enter the same URL and make one change before the & symbol. Use backslash (\).

```
amanpathak@aman-pathak:~$ curl http://localhost:8080/buildByToken/build?job=demo-with-github\&token=tokenaman
amanpathak@aman-pathak:~$
```

2. The build has been successful.

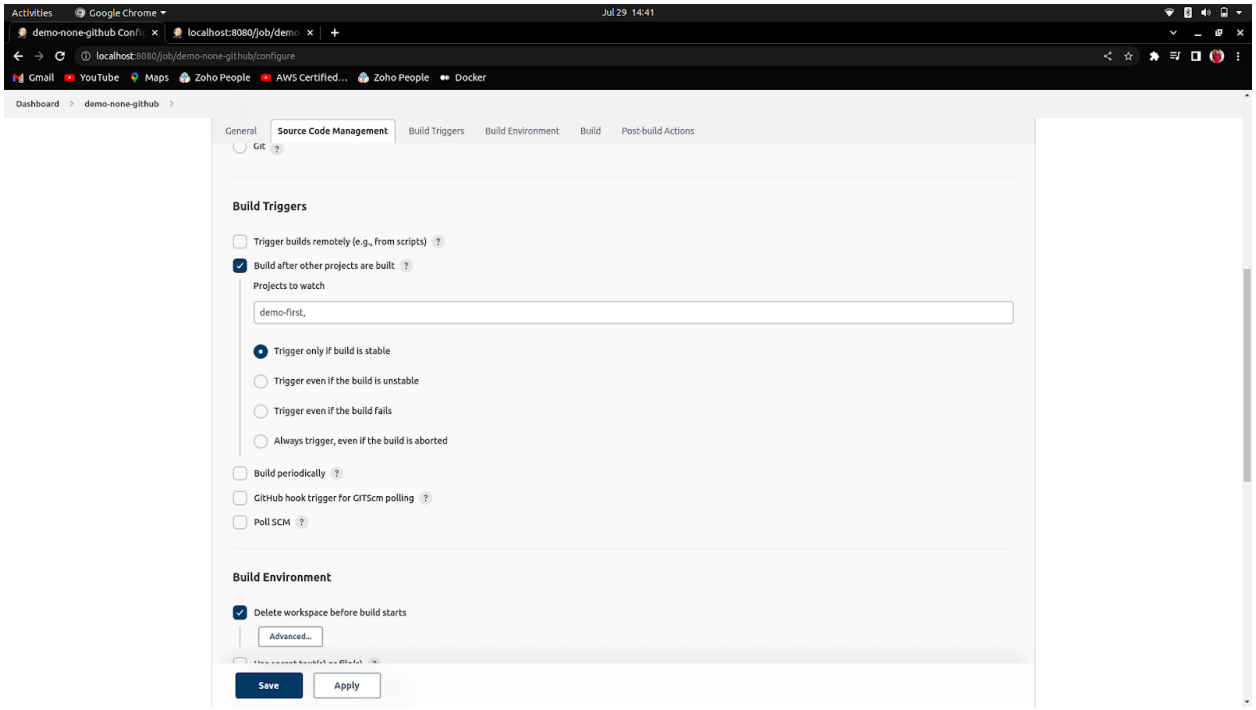


2. Build after other projects are built

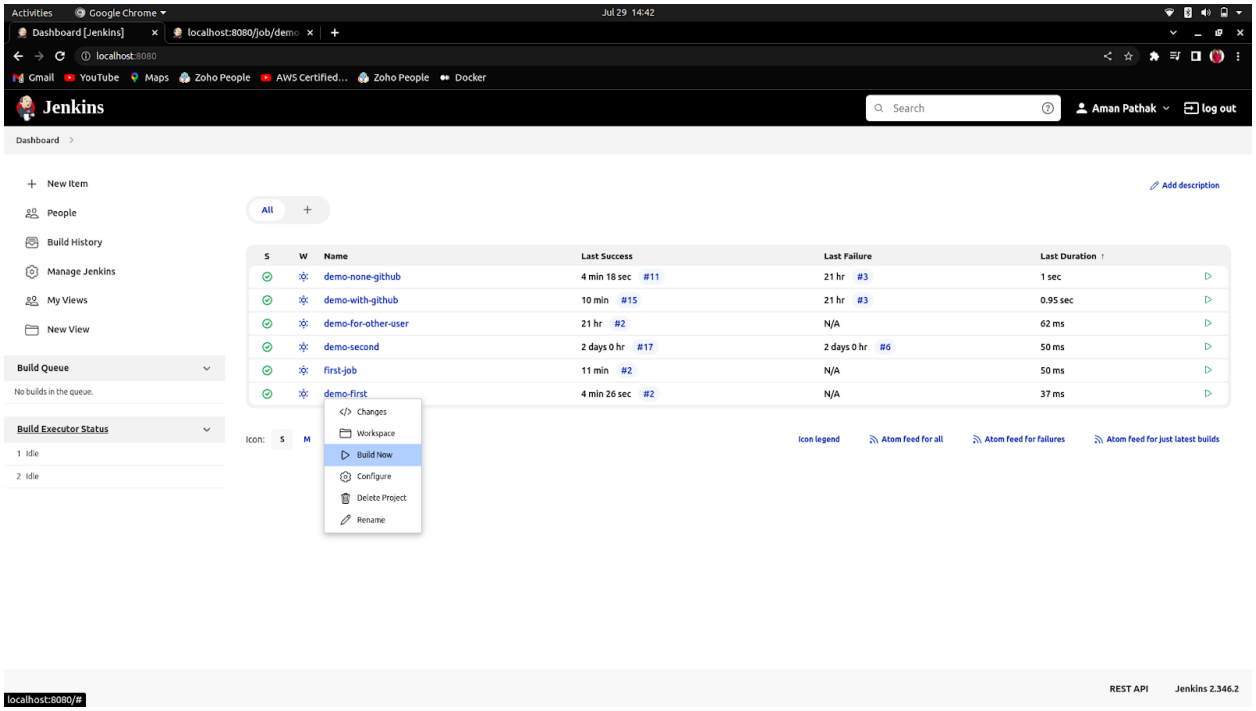
Why?

When you have to build your specific project after some desired projects. You can use this Build Trigger.

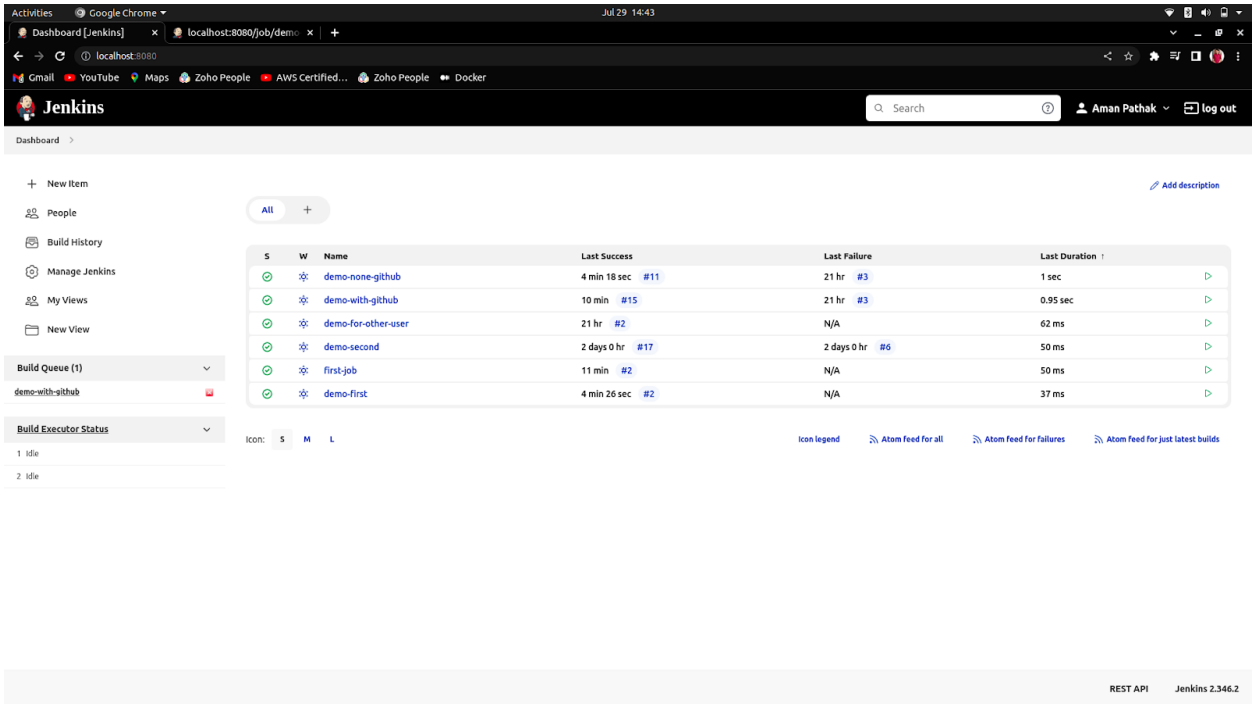
1. Enter the other project that you want to build first.



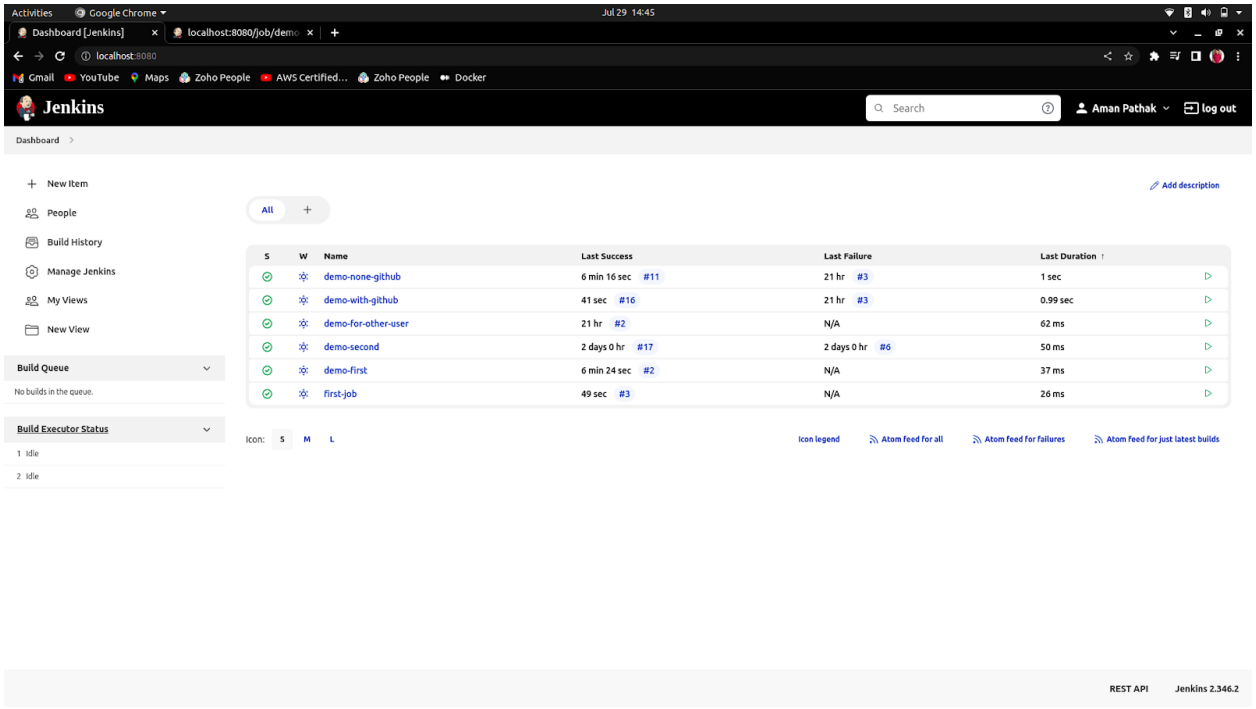
2. Now, run the project that you entered under the Projects to Watch section.



3. You will see on the left, the project is built.



4. If you observe, Both build numbers are incremented by 1.

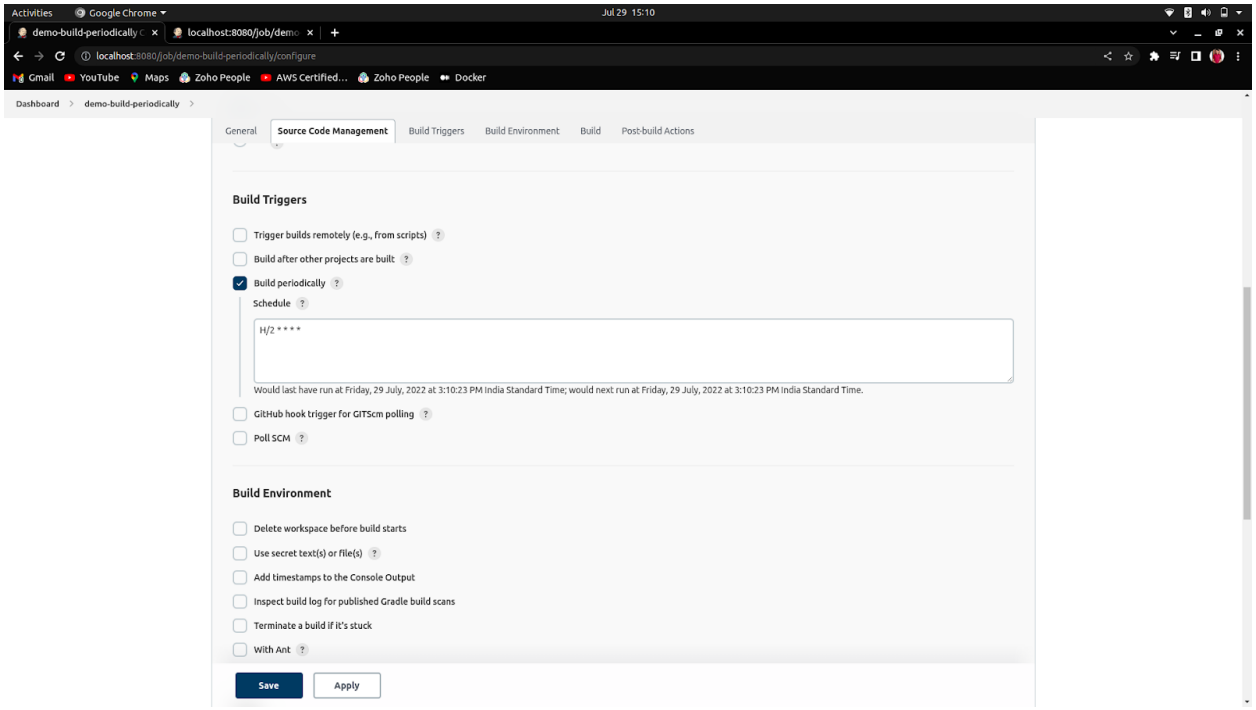


3. Build Job Periodically

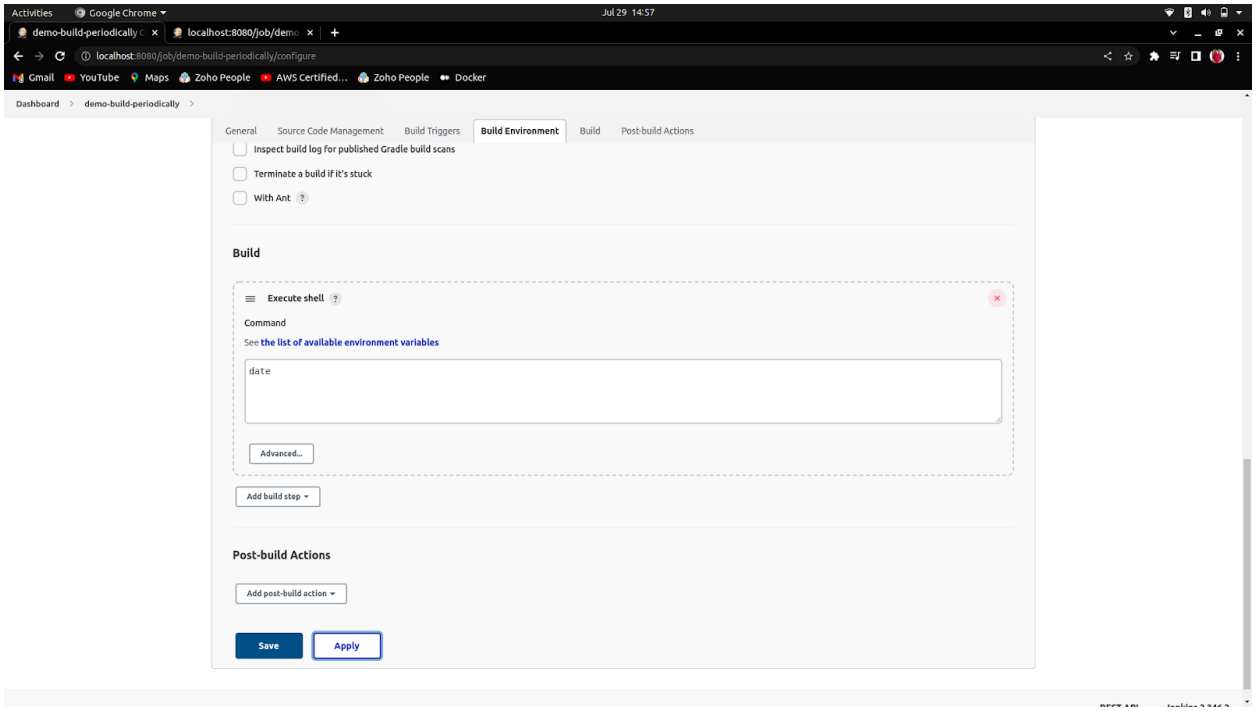
Why?

When you have to build your job at a specific time or same interval e.g. every two minutes. You can use Build periodically under the Build Triggers section.

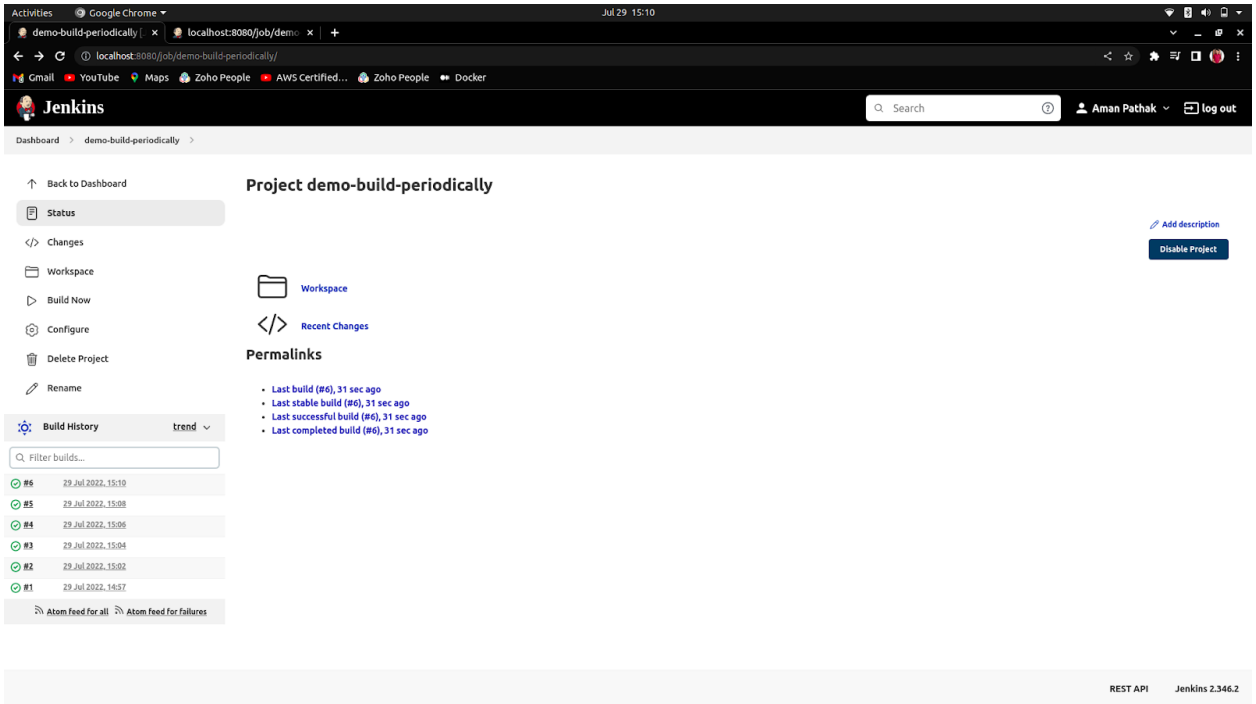
1. Write the time interval the same as written in the cron job.



2. Write the command and apply then save it.



3. Here, you can check it out. The build is automatically done every two minutes.

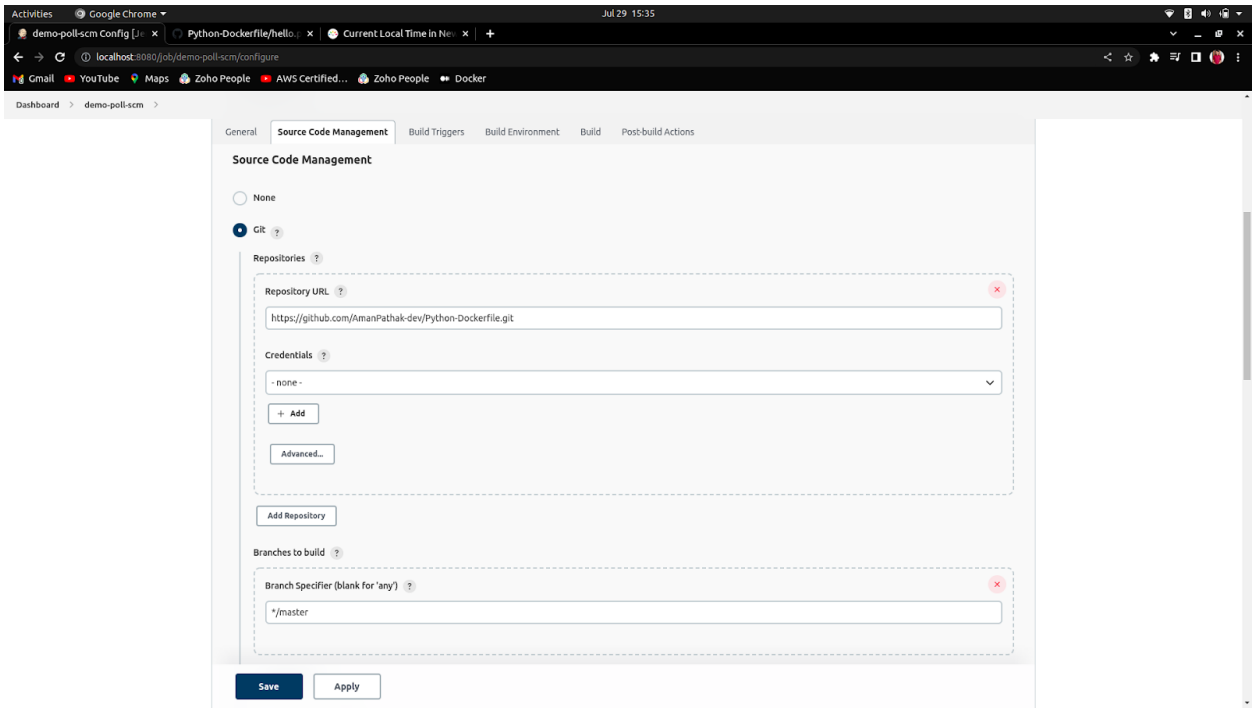


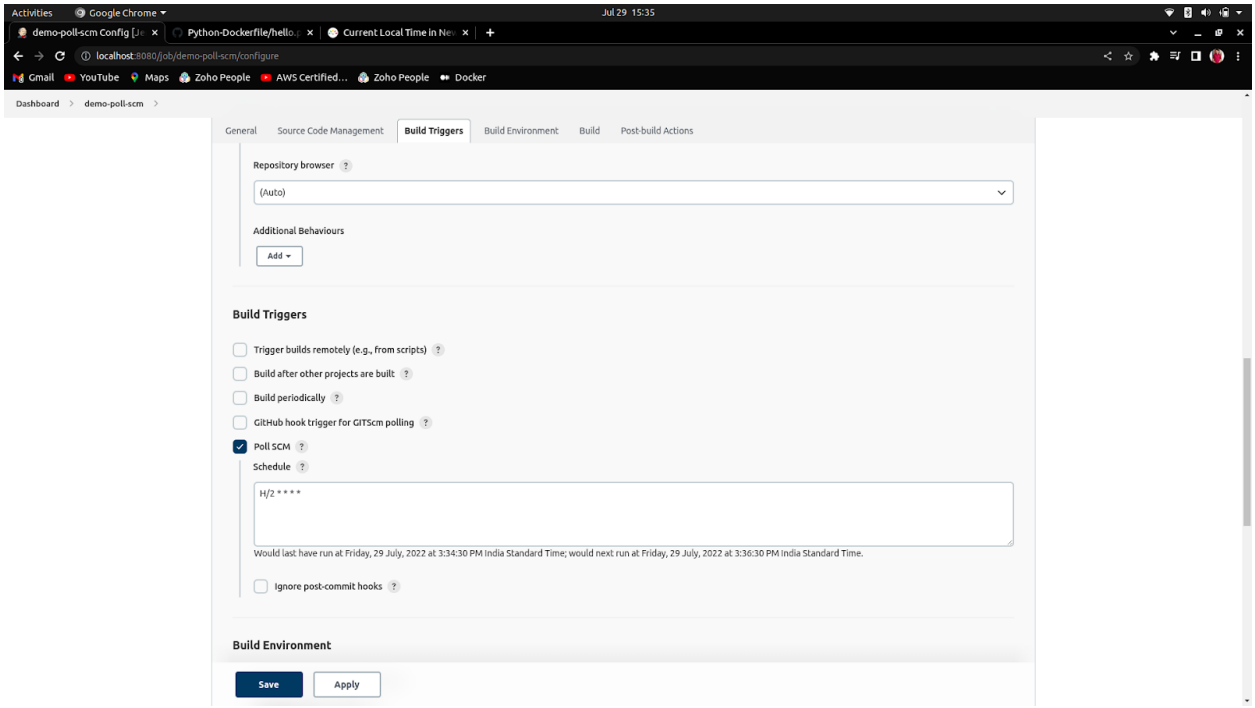
4. Poll SCM(Source Code Management)

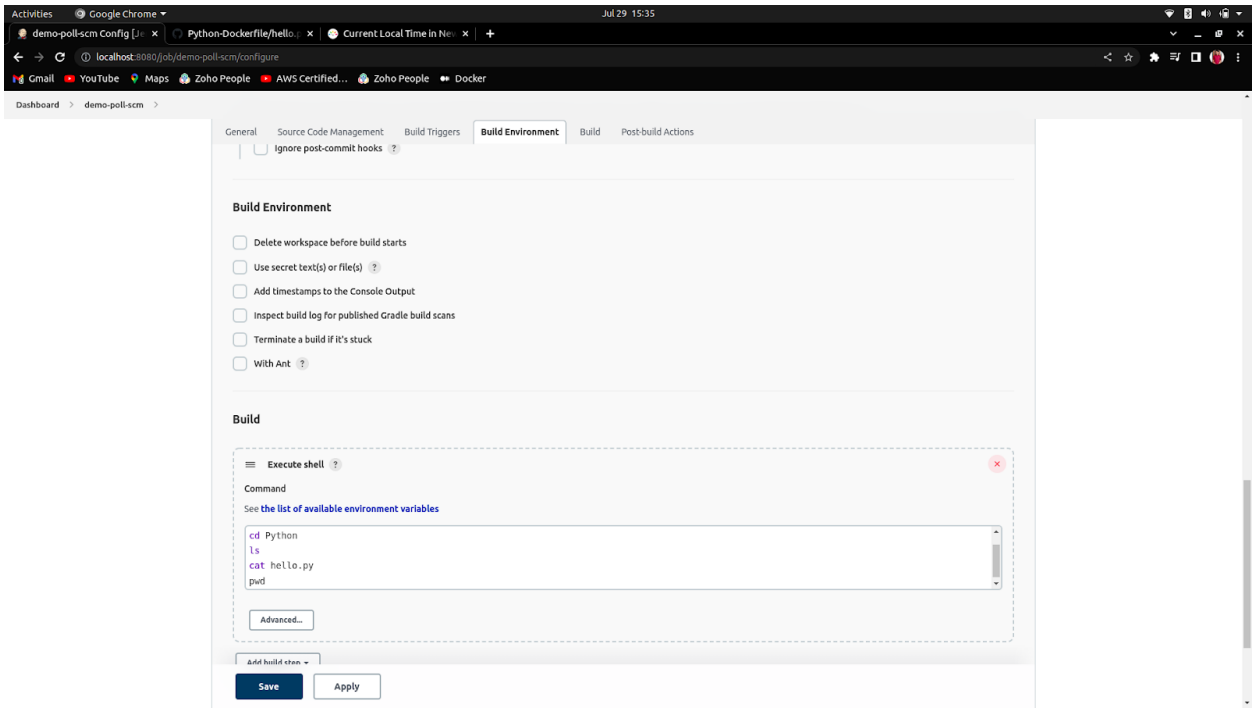
Why?

This Build trigger is the same as Build Periodically. But there is only one difference which is, that the project will fetch from the GitHub repository(mandatory) and if there is no commit in the repository then, the job will not build. But, if there is any commit change in the GitHub repository, then the job will build as a per-time scheduler which is also known as Cronjob/Crontab.

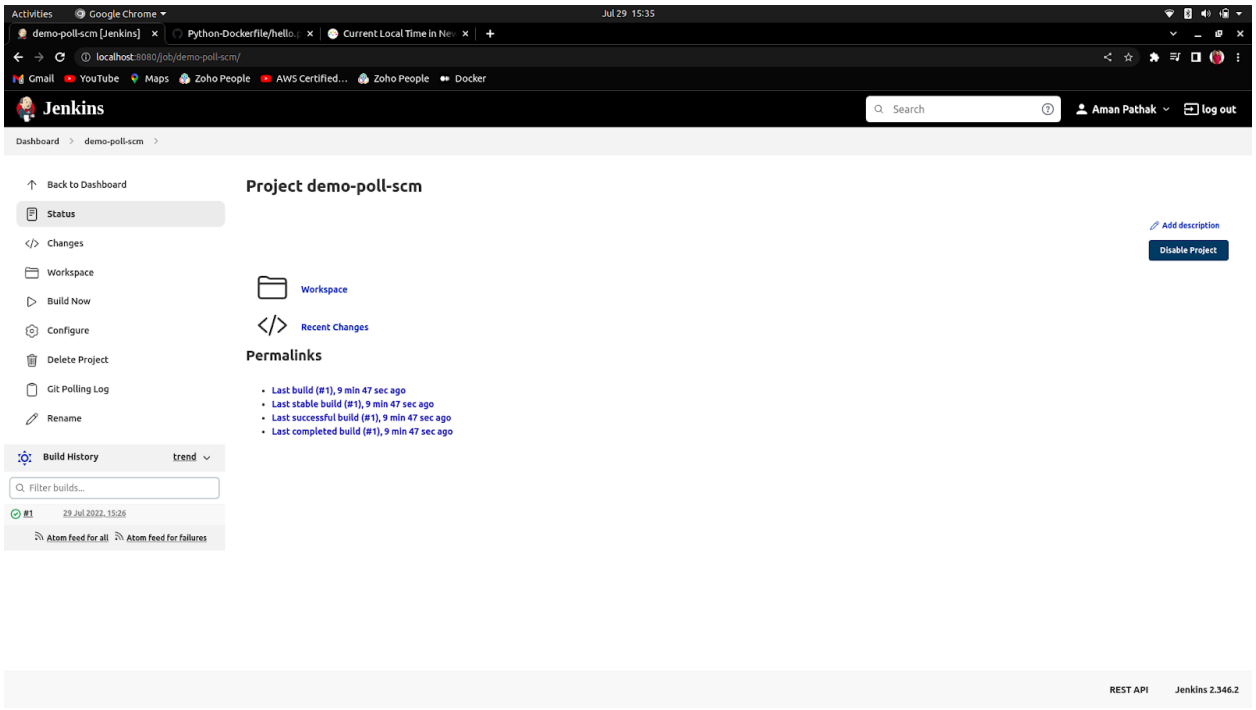
1. Configuration for the Project.



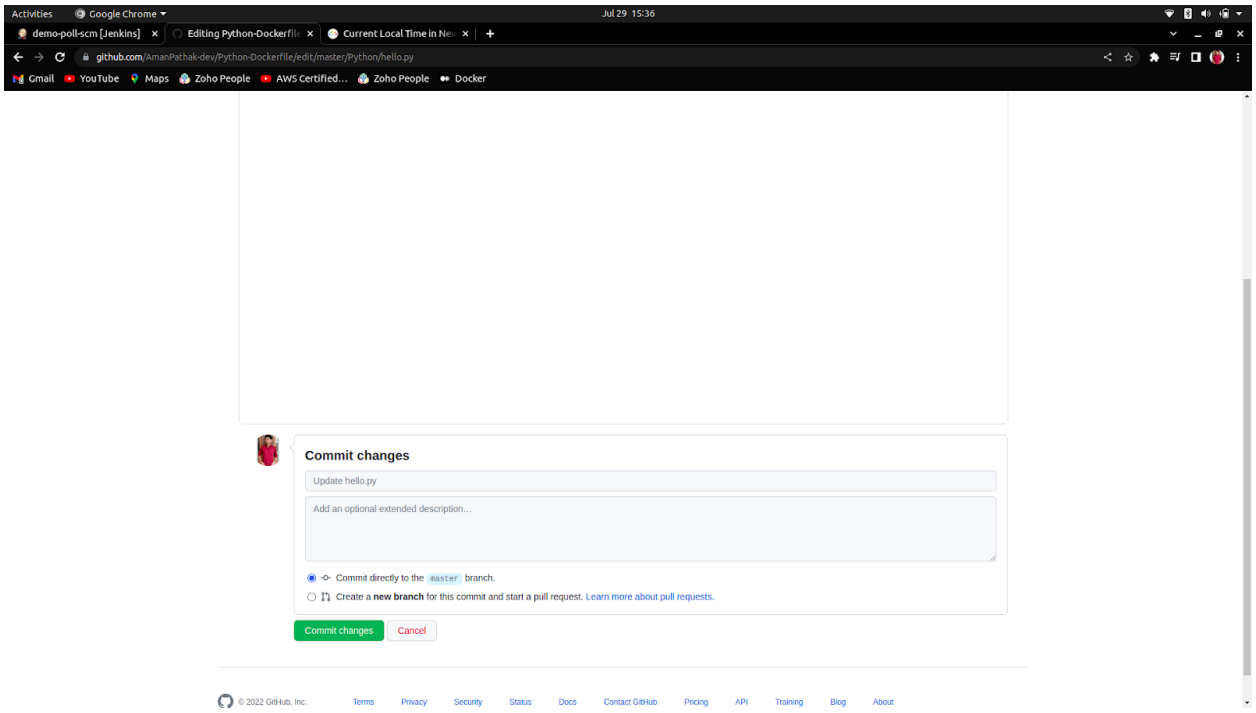

Activities
Google Chrome
Jul 29 15:35
demo-poll-scm Config [J
Python-Dockerfile/hello.
Current Local Time in Ne
localhost:8080/job/demo-poll-scm/configure
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
demo-poll-scm
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Repository browser
(Auto)
Additional Behaviours
Add
Build Triggers
Trigger builds remotely (e.g., from scripts)
Build after other projects are built
Build periodically
GitHub hook trigger for GITScm polling
Poll SCM
Schedule
H/2 * * * *
Would last have run at Friday, 29 July, 2022 at 3:34:30 PM India Standard Time; would next run at Friday, 29 July, 2022 at 3:36:30 PM India Standard Time.
Ignore post-commit hooks
Build Environment
Save
Apply


Activities
Google Chrome
Jul 29 15:35
demo-poll-scm Config [J
Python-Dockerfile/hello.
Current Local Time in Ne
localhost:8080/job/demo-poll-scm/configure
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
demo-poll-scm
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Ignore post-commit hooks
Build Environment
Delete workspace before build starts
Use secret text(s) or file(s)
Add timestamps to the Console Output
Inspect build log for published Gradle build scans
Terminate a build if it's stuck
With Ant
Build
Execute shell
Command
See the list of available environment variables
cd Python
ls
cat hello.py
pwd
Advanced...
Save
Apply

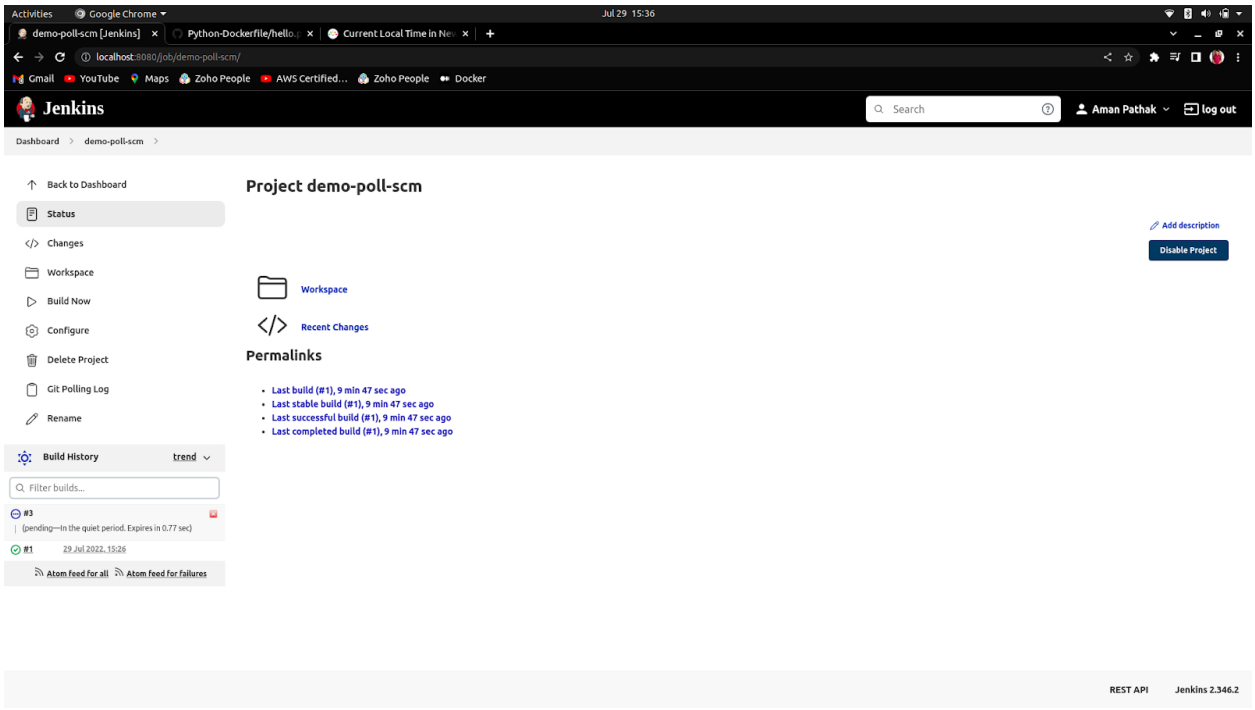
2. Job is built automatically. But then no job build



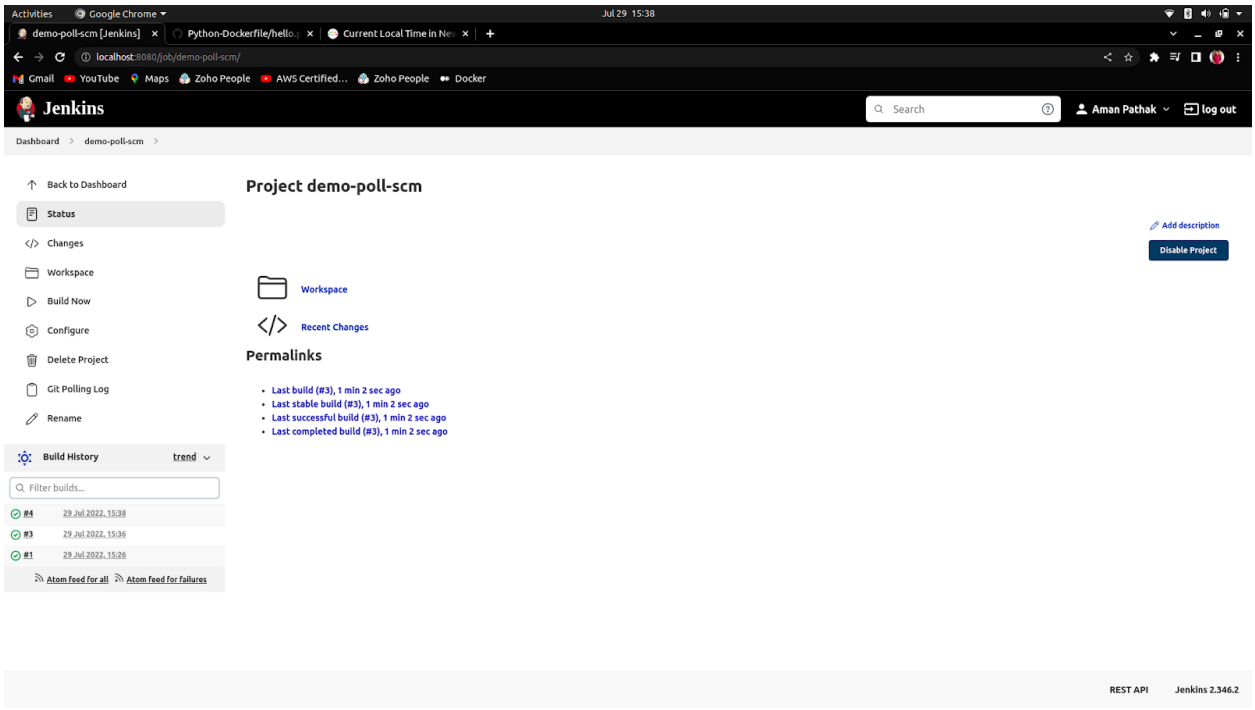
3. Changing some files data of the GitHub repository to build a job.



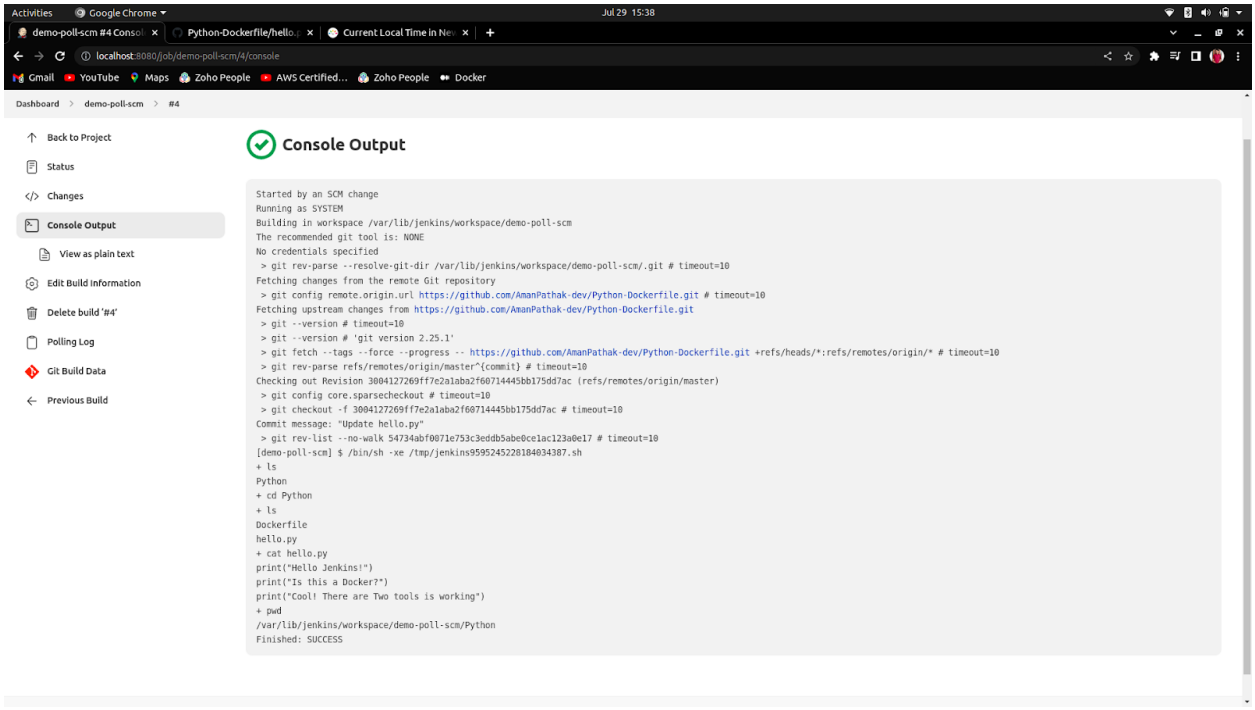
4. As GitHub repository has a new commit. Now, it is building a job.



5. The Build is successful.

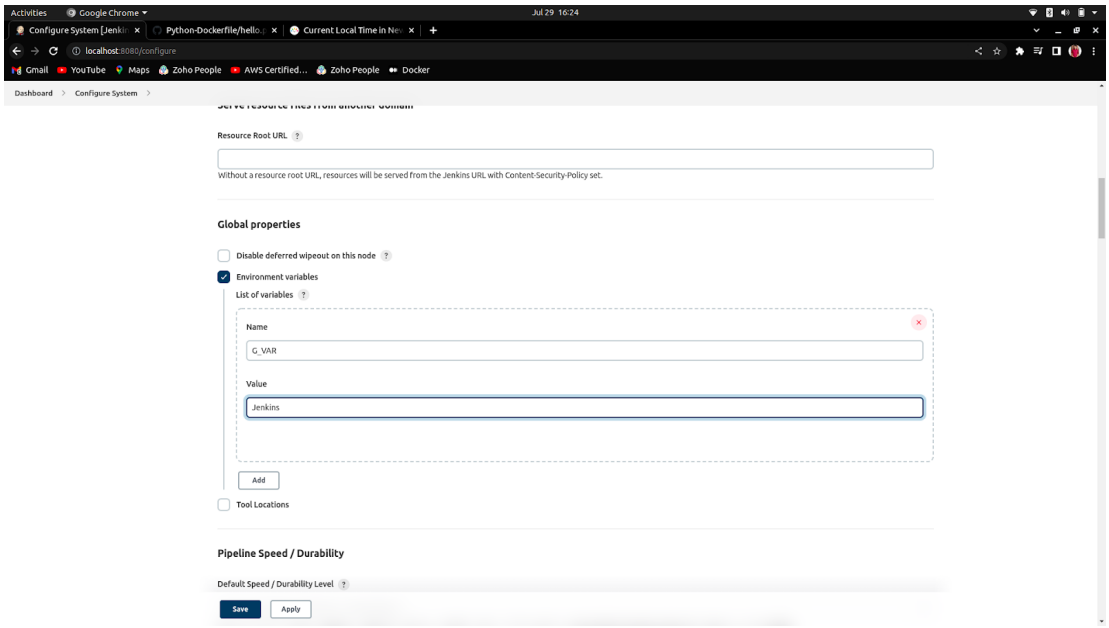


The Output

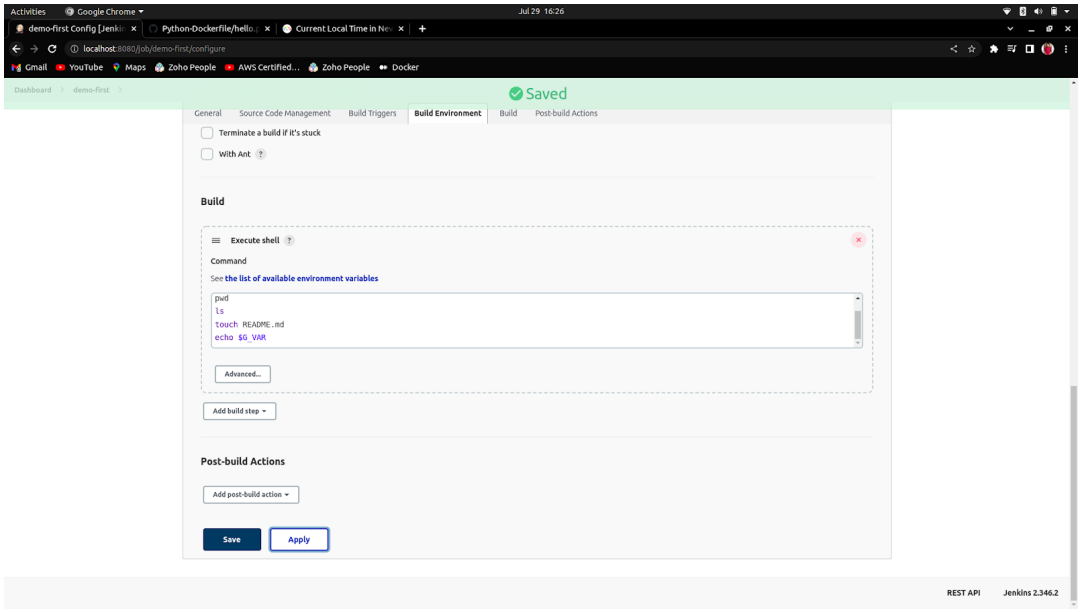

Activities
Google Chrome
Jul 29 15:38
demo-poll-scm #4 Consol
Python-Dockerfile/hello.
Current Local Time in Ne
localhost:8080/job/demo-poll-scm/4/console
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
demo-poll-scm
#4
Back to Project
Status
Changes
Console Output
View as plain text
Edit Build Information
Delete build '#4'
Polling Log
Git Build Data
Previous Build
Console Output
Started by an SCM change
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/demo-poll-scm
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/demo-poll-scm/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/AmanPathak-dev/Python-Dockerfile.git # timeout=10
Fetching upstream changes from https://github.com/AmanPathak-dev/Python-Dockerfile.git
> git --version # timeout=10
> git --version # 'git version 2.25.1'
> git fetch --tags --force --progress -- https://github.com/AmanPathak-dev/Python-Dockerfile.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision 3004127269ff7e2a1aba2f60714445bb175dd7ac (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f 3004127269ff7e2a1aba2f60714445bb175dd7ac # timeout=10
Commit message: "Update hello.py"
> git rev-list --no-walk 54734abf0071e753c3eddb5abe0ce1ac123a0e17 # timeout=10
[demo-poll-scm] $ /bin/sh -xe /tmp/jenkins9595245228184034387.sh
+ ls
Python
+ cd Python
+ ls
Dockerfile
hello.py
+ cat hello.py
print("Hello Jenkins!")
print("Is this a Docker?")
print("Cool! There are Two tools is working")
+ pwd
/var/lib/jenkins/workspace/demo-poll-scm/Python
Finished: SUCCESS

How to define variables globally

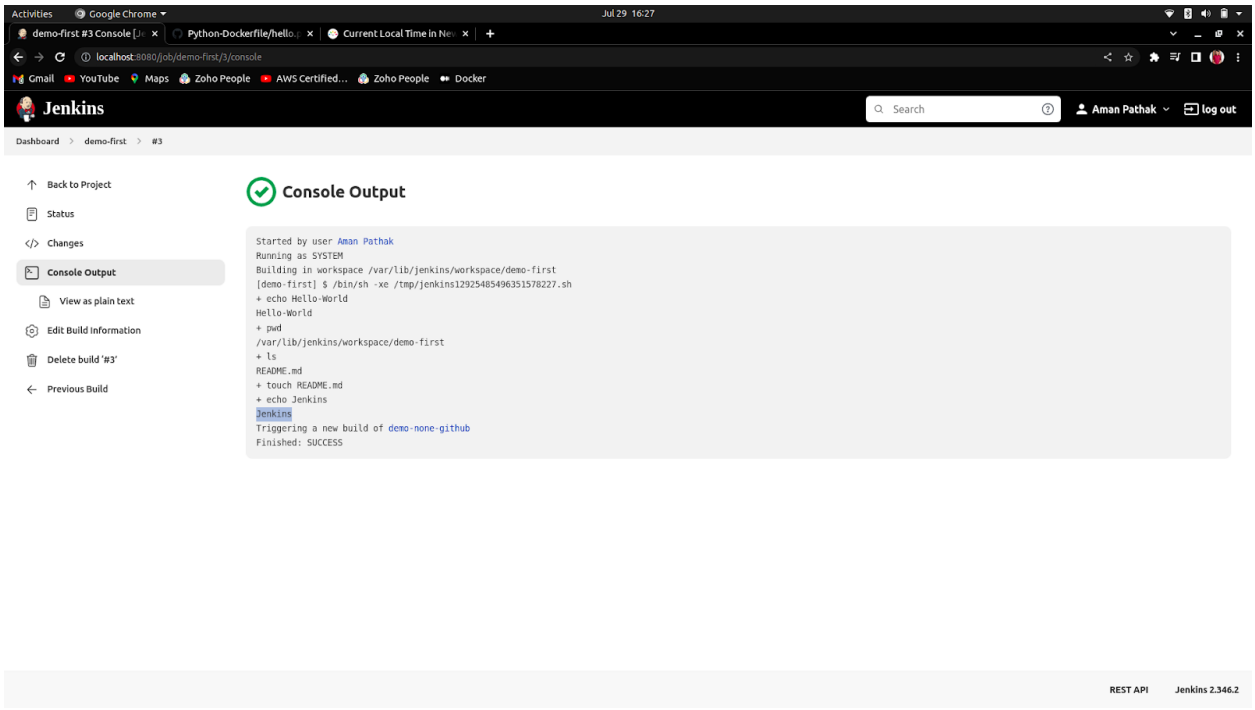
1. Go to the Manage Jenkins -> Configure System and scroll down then, look for Global properties.



2. Now, Create a job and print that value of variable by passing the environment variable.

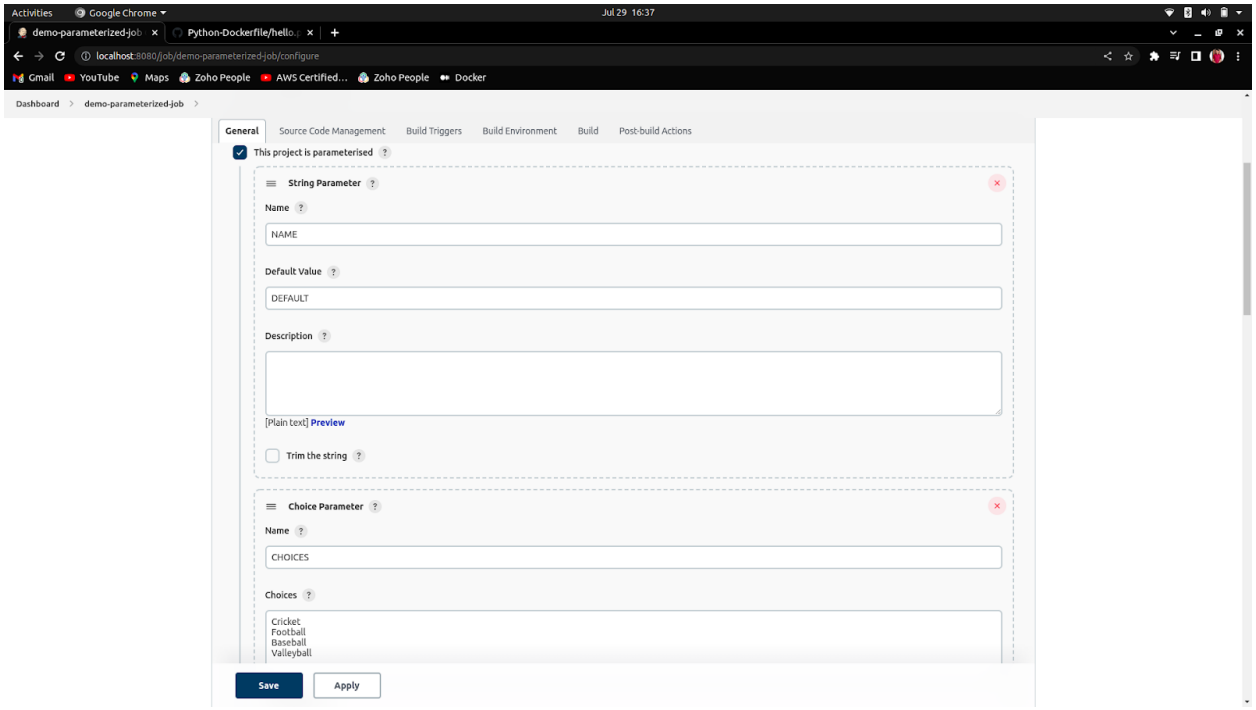


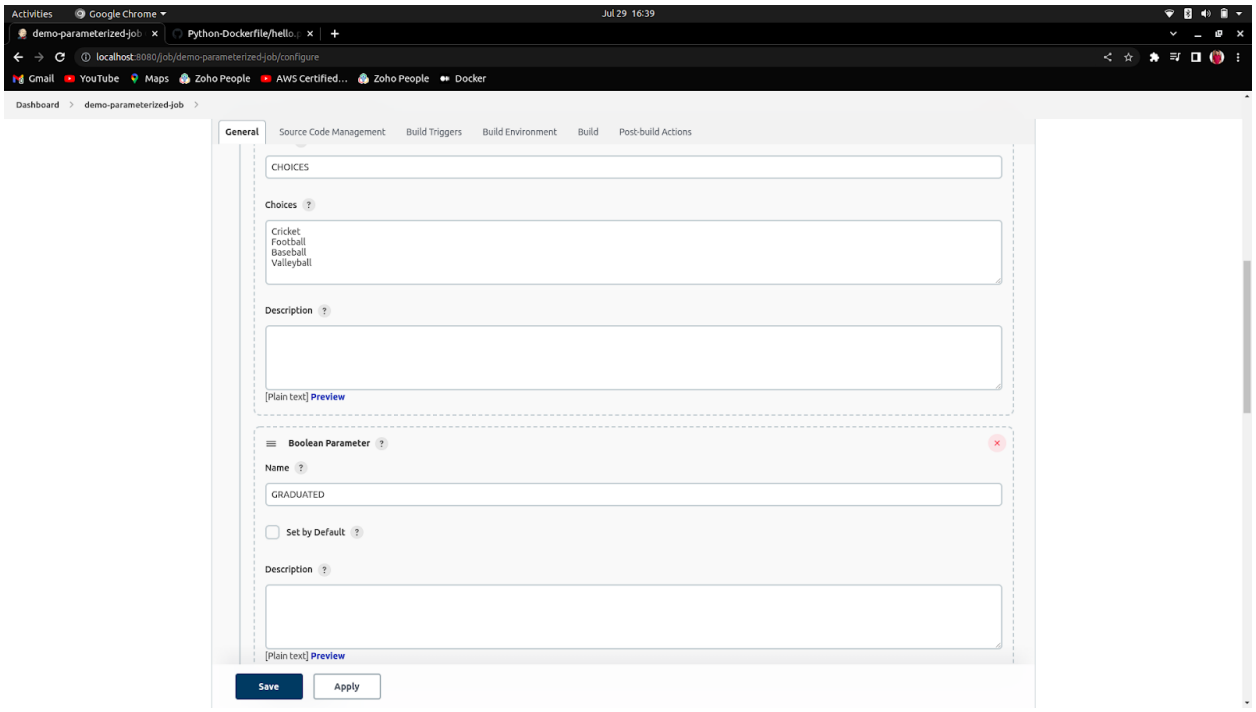
3. Here, you can see the value of variable is appeared.

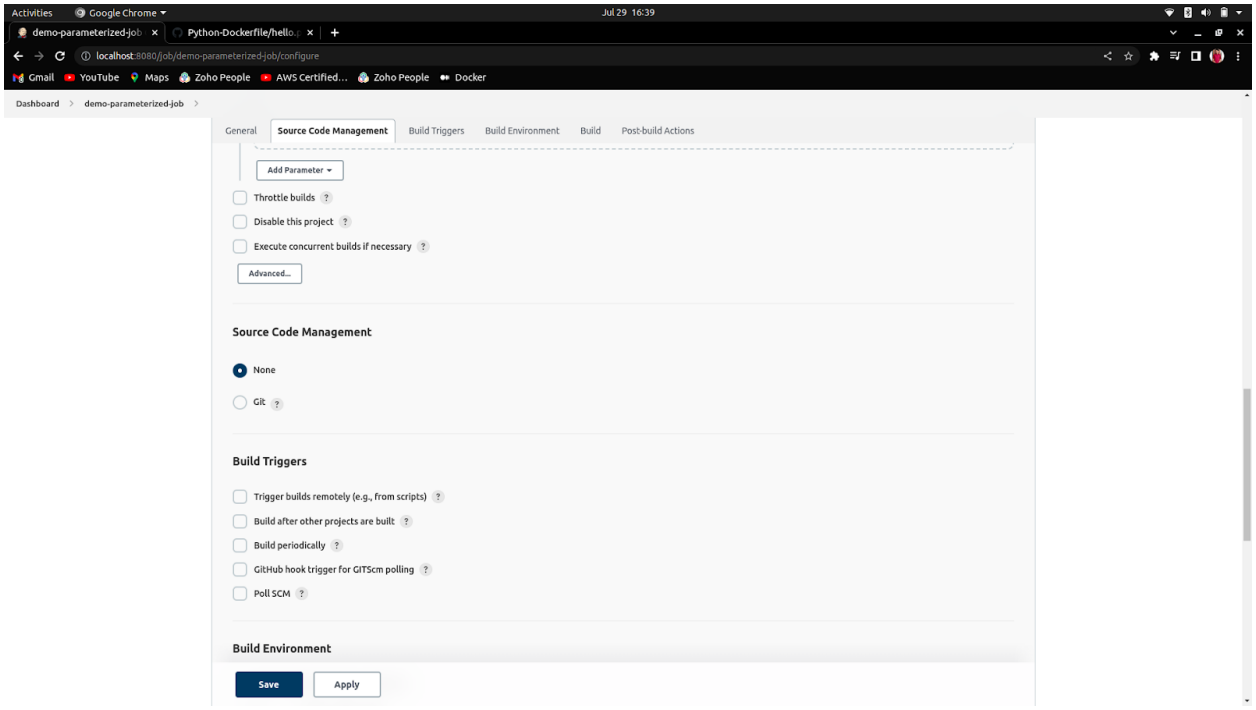


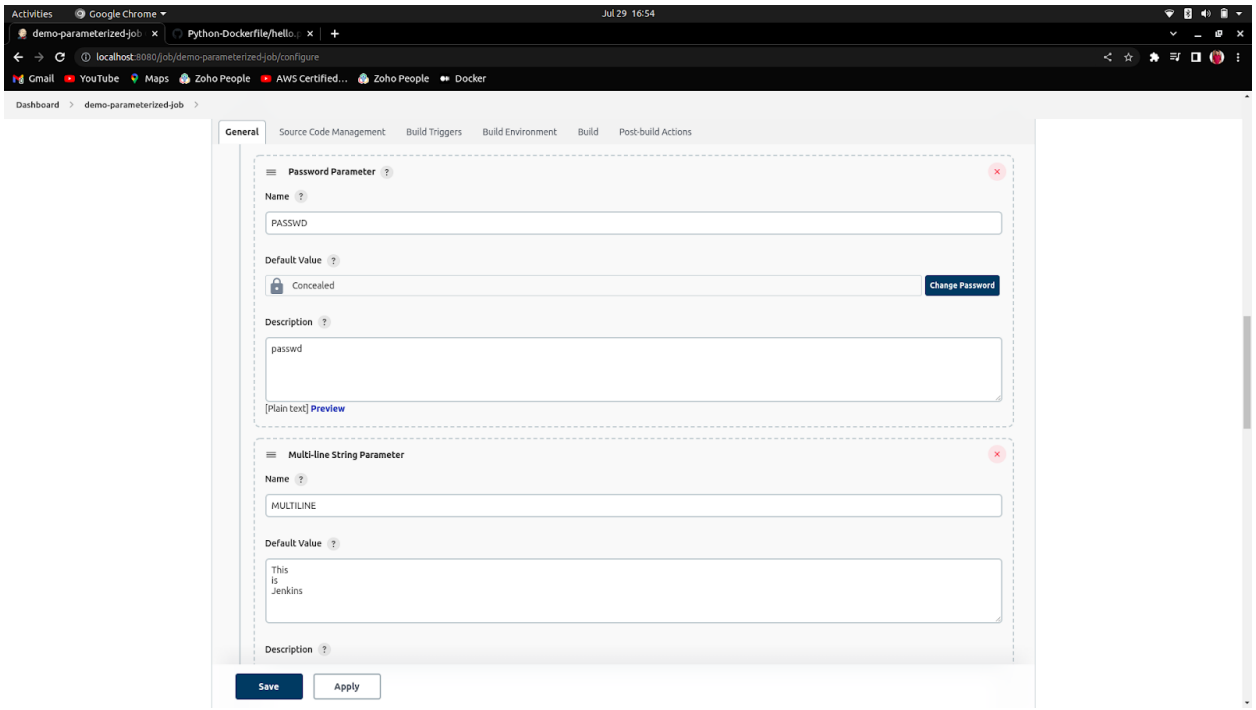
Parameterised in Job

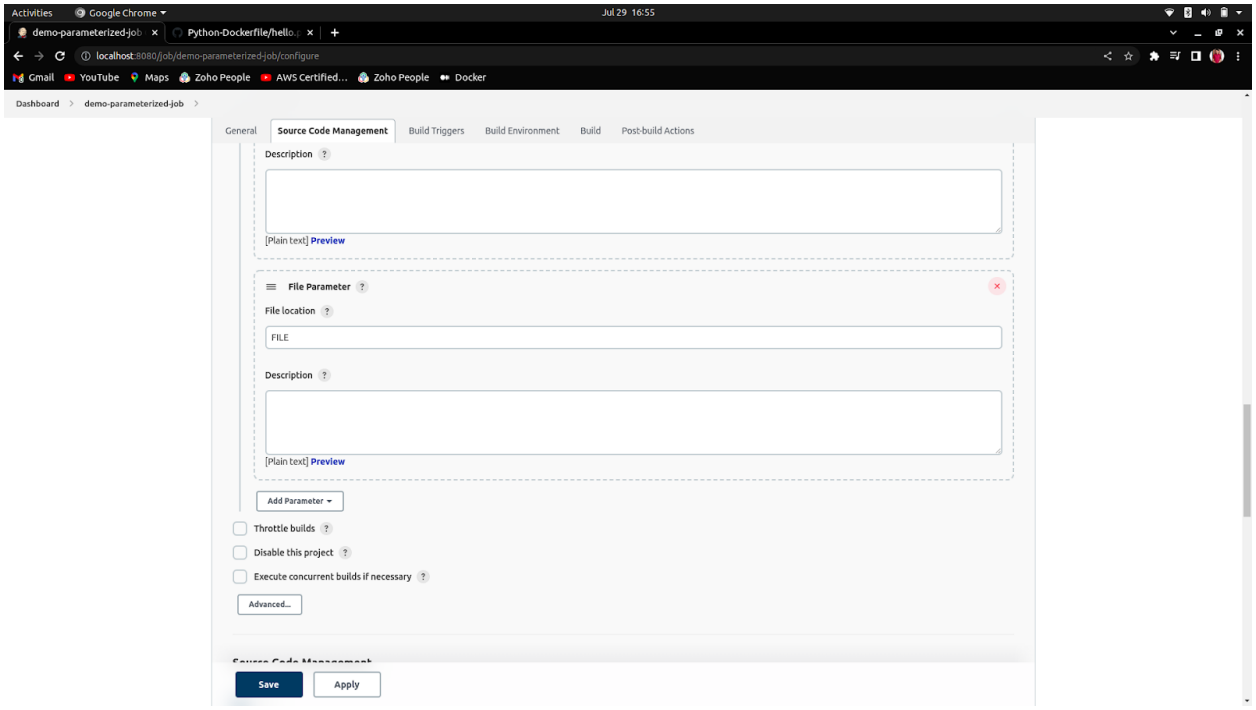
1. Define the variable and pass the default value.

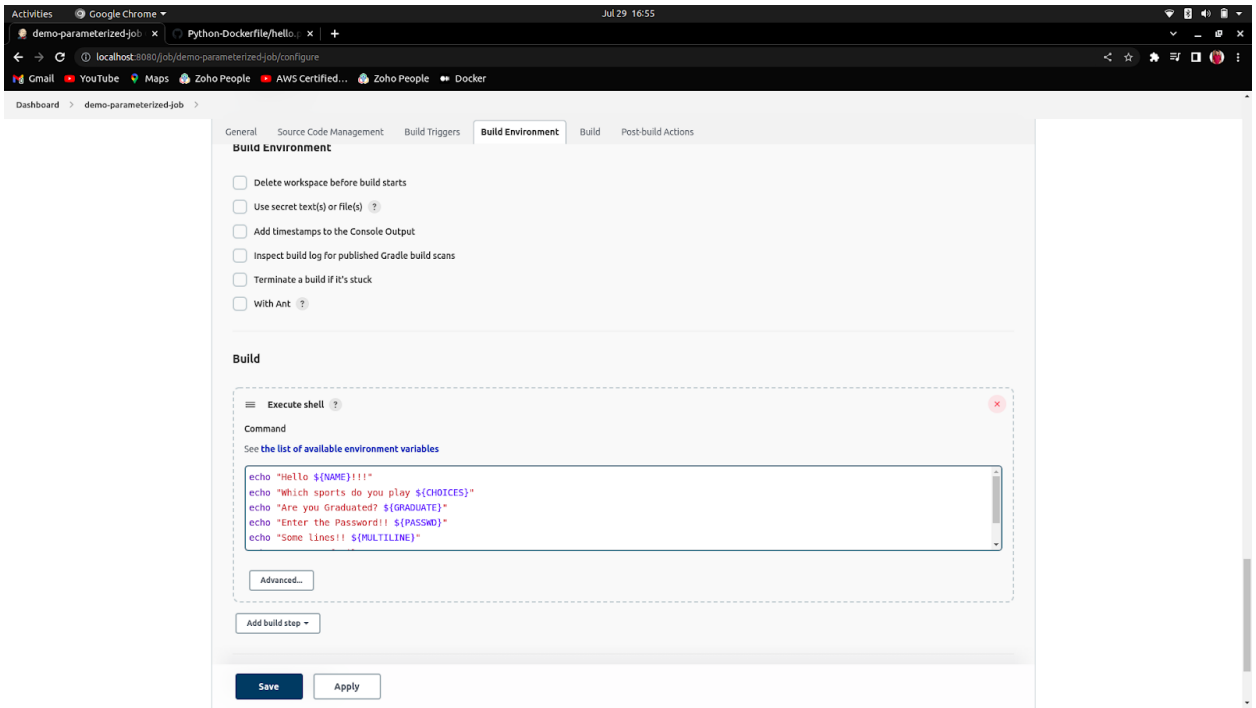





Activities
Google Chrome
Jul 29 16:39
demo-parameterized-job
Python-Dockerfile/hello...
localhost:8080/job/demo-parameterized-job/configure
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
demo-parameterized-job
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Add Parameter
Throttle builds
Disable this project
Execute concurrent builds if necessary
Advanced...
Source Code Management
None
Git
Build Triggers
Trigger builds remotely (e.g., from scripts)
Build after other projects are built
Build periodically
GitHub hook trigger for GITScm polling
Poll SCM
Build Environment
Save
Apply

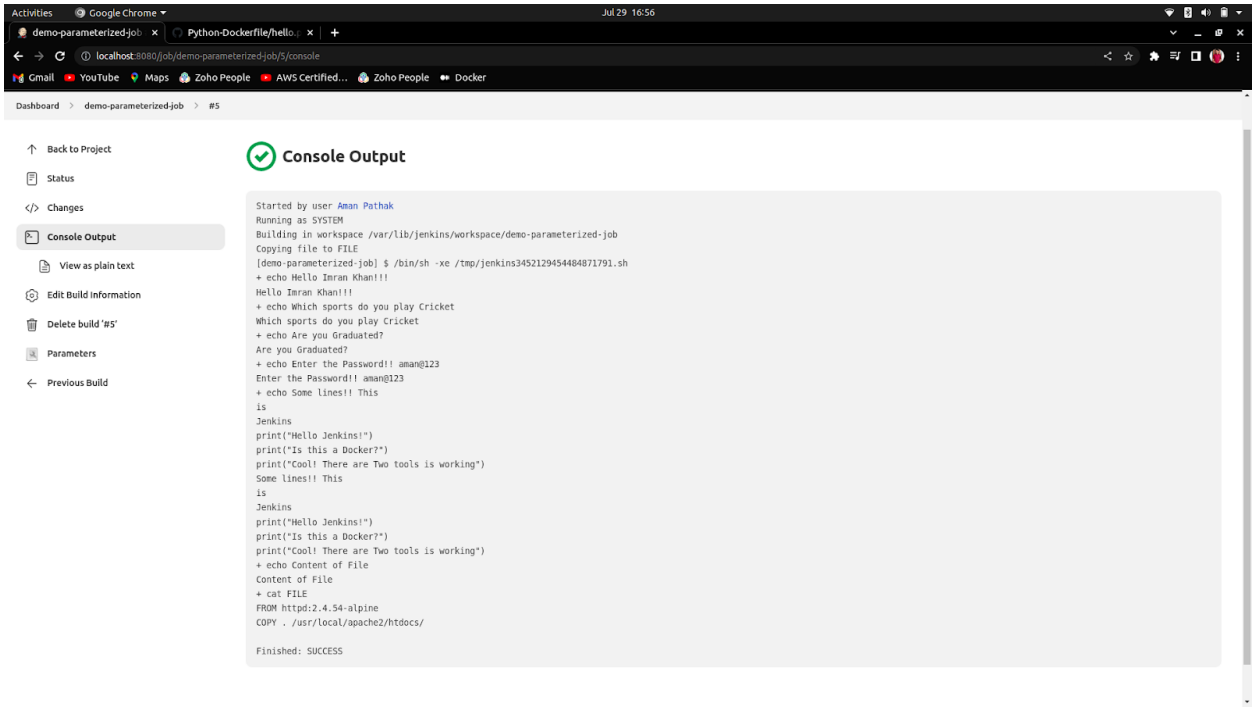

Activities
Google Chrome
Jul 29 16:54
demo-parameterized-job
Python-Dockerfile/hello...
localhost:8080/job/demo-parameterized-job/configure
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
demo-parameterized-job
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Password Parameter
Name
PASSWD
Default Value
Concealed
Change Password
Description
passwd
[Plain text] Preview
Multi-line String Parameter
Name
MULTILINE
Default Value
This
is
Jenkins
Description
Save
Apply





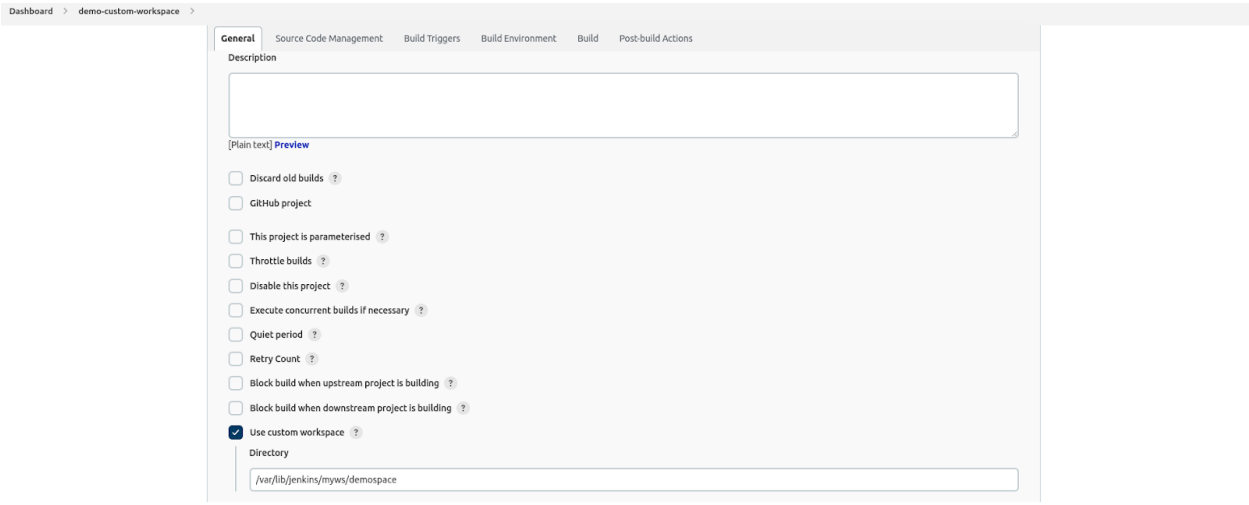

```
echo "Are you Graduated? ${GRADUATE}"
echo "Enter the Password!! ${PASSWD}"
echo "Some lines!! ${MULTILINE}"
echo "Content of File"
cat FILE
```

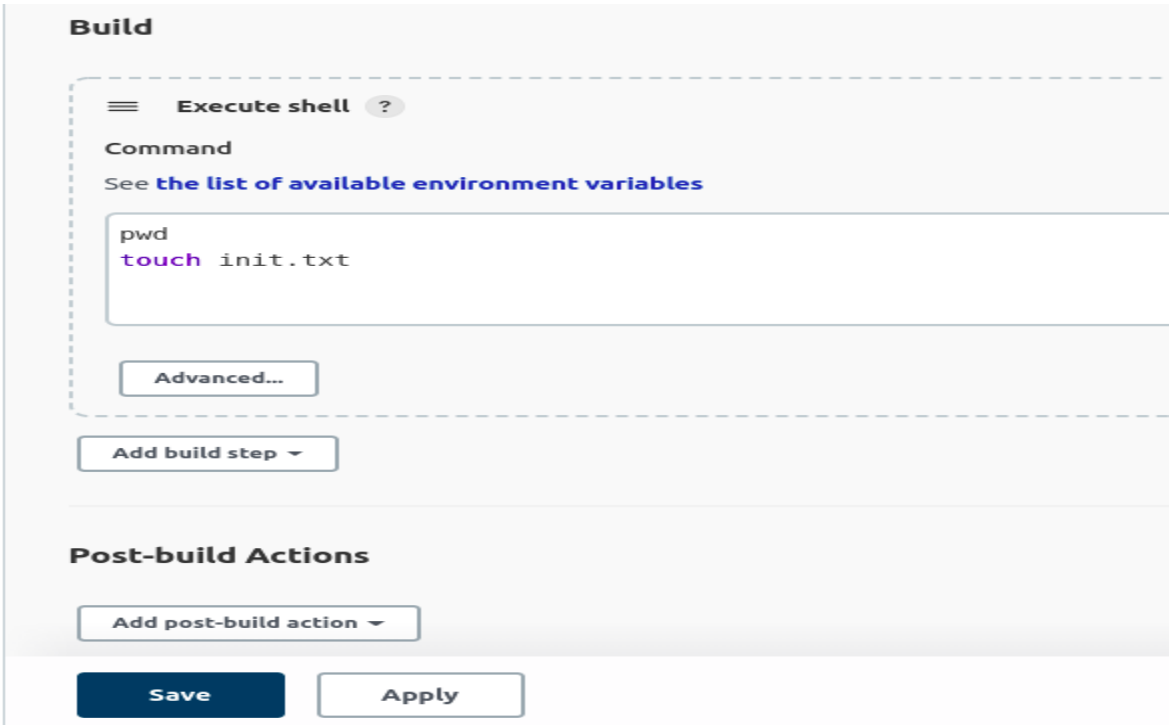
Console Output



Custom Workspace

1. Create a new job and do the configuration as given below:





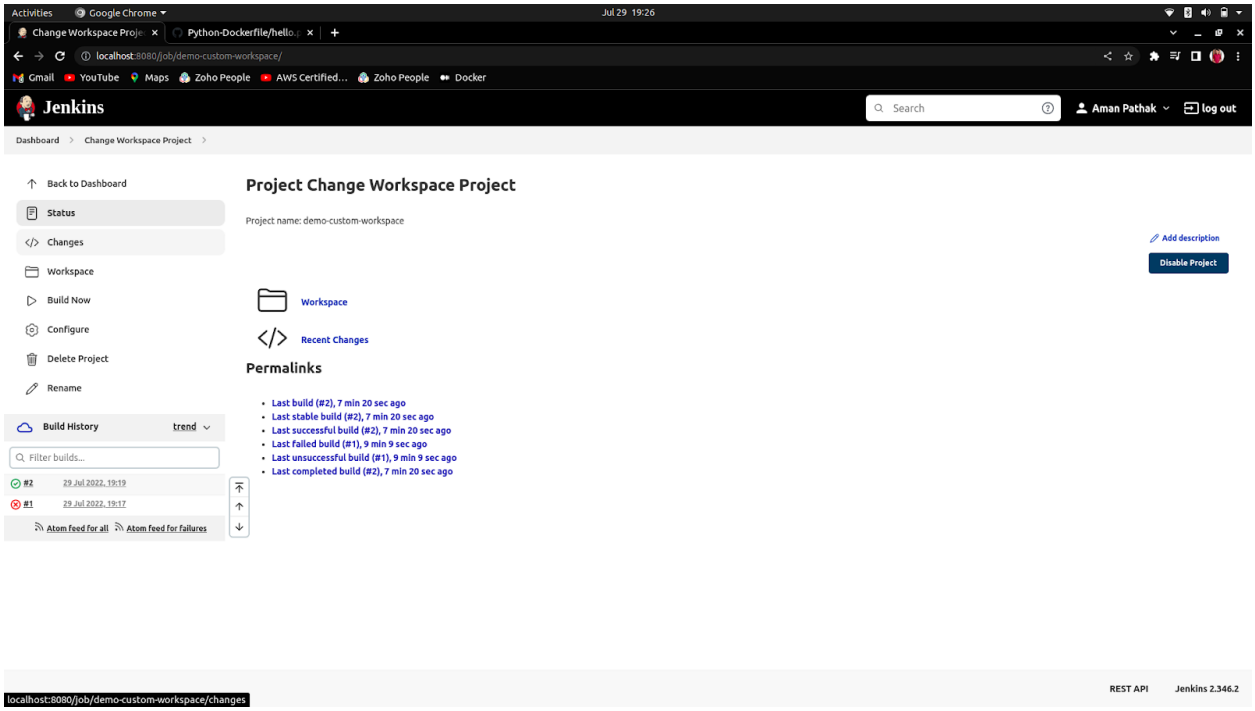
2. Now, the directory has been created and file too.
   To check it, go to your terminal.

```
amanpathak@aman-pathak:/var/lib/jenkins$ cd myws/
amanpathak@aman-pathak:/var/lib/jenkins/myws$ ls
demospace
amanpathak@aman-pathak:/var/lib/jenkins/myws$ cd demospace/
amanpathak@aman-pathak:/var/lib/jenkins/myws/demospace$ ls
init.txt
```

Change display name and project name

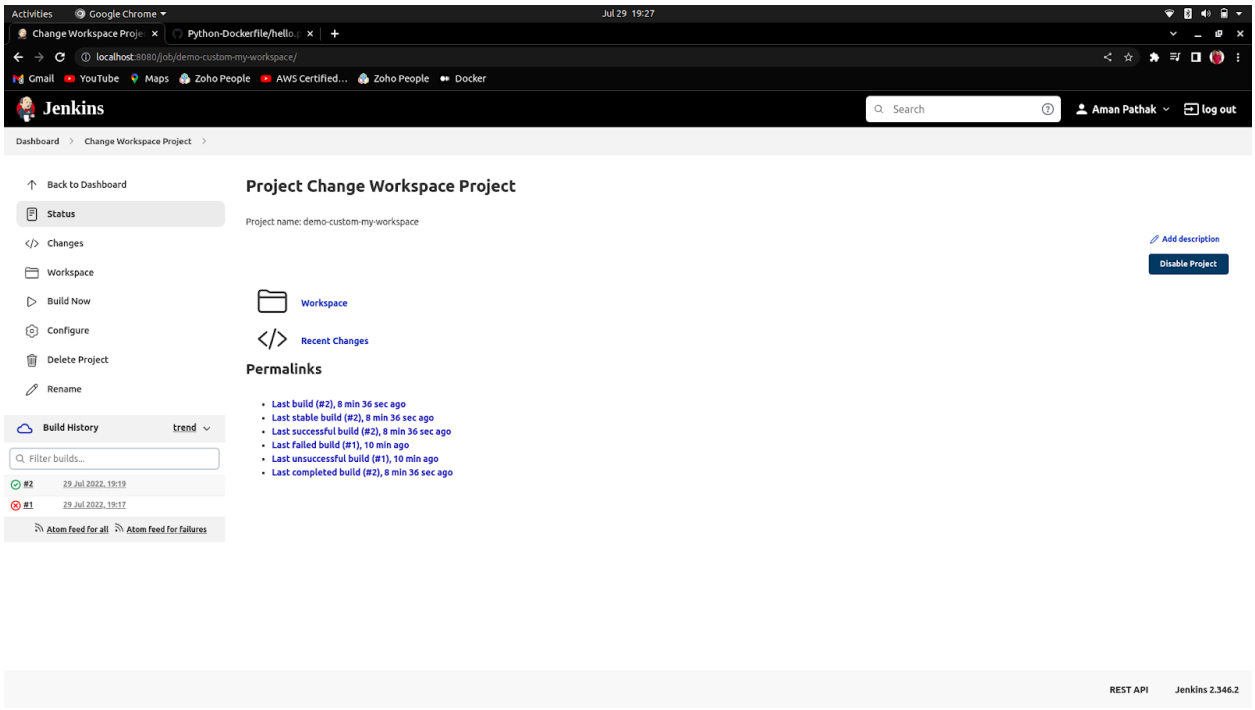
1. Go to in the Configure of the Specific Project-> Enter the desired name that you want in the text field.

The Output



But the Project hasn't changed. For that you have to click on the Rename button which is left on the screen and enter the desired name.

Now, The project name has changed too.



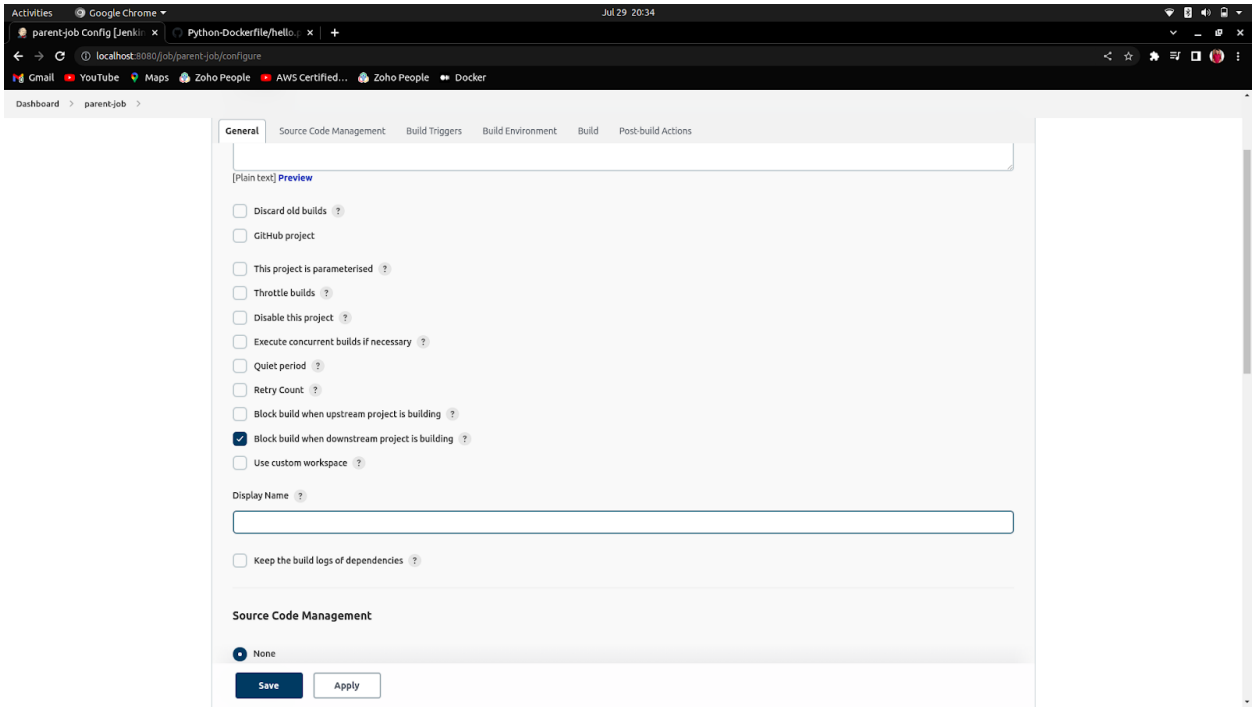
Building Upstream and Downstream Project

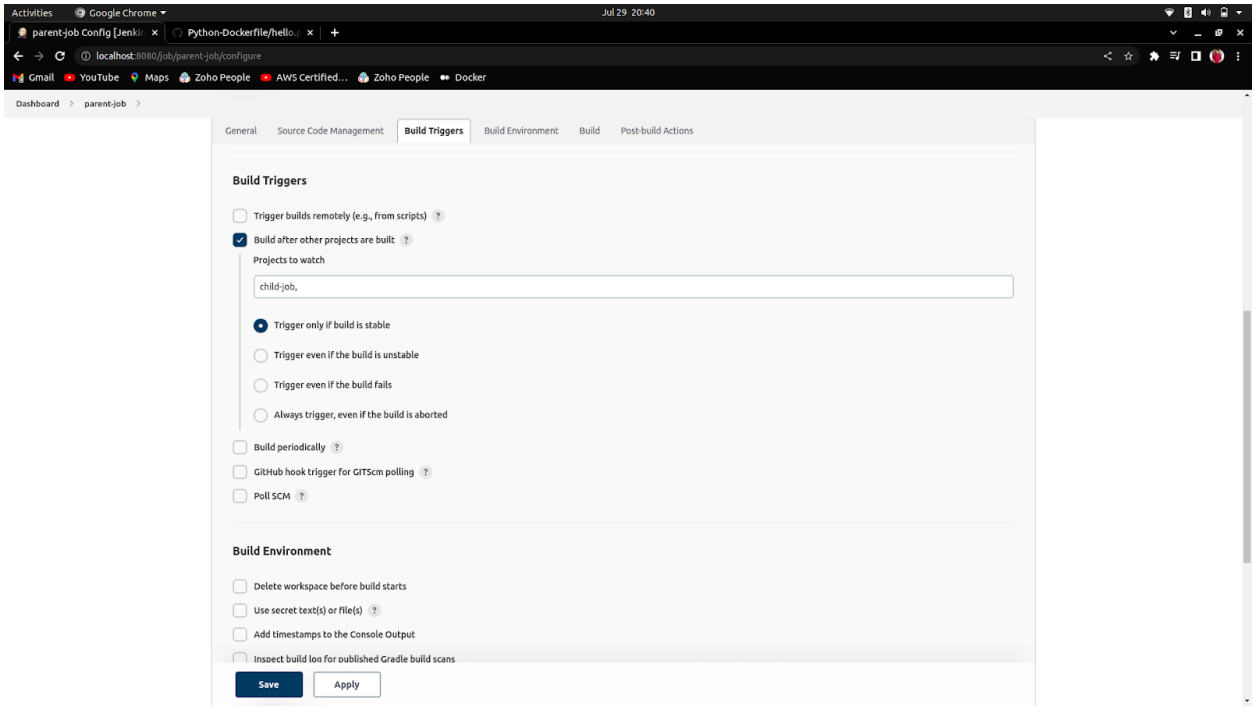
Upstream- Parent job
Downstream- Child job

That job that is triggering another job is known as the Parent job and that job that is triggered by another job/parent job is known as the Child job.

Block build when the upstream project is building

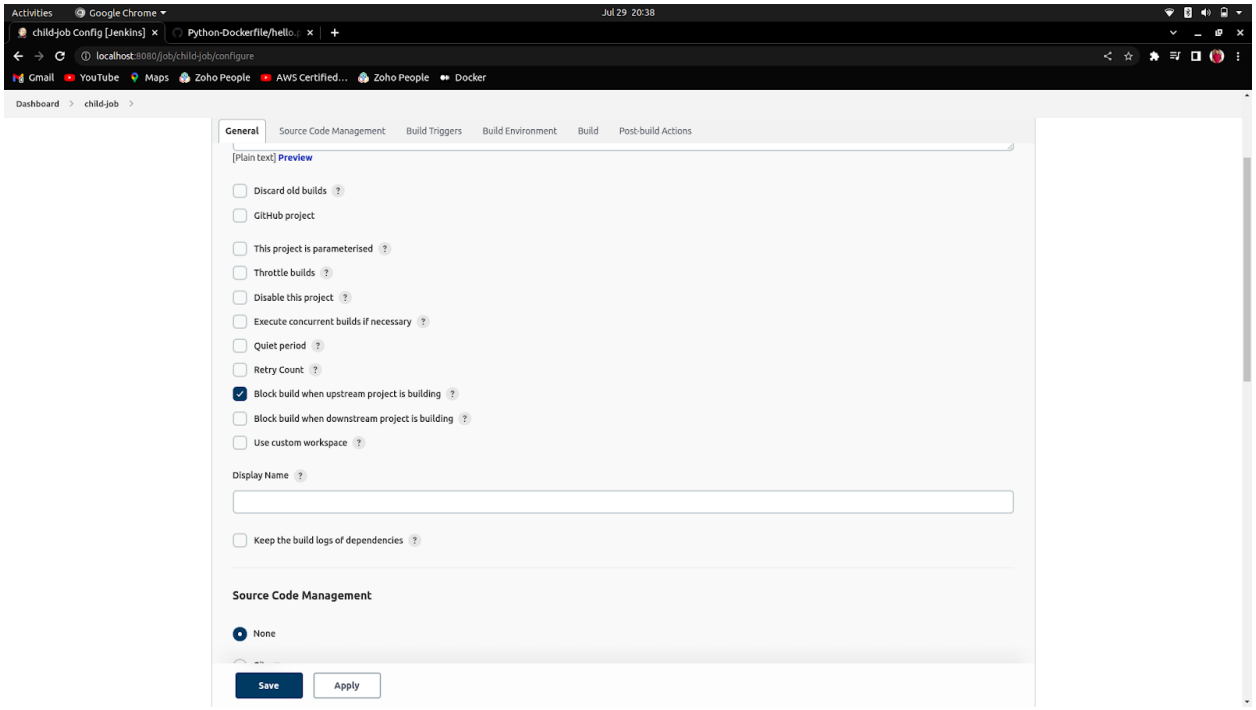
Here, the scenario is if you are building a parent job then, you couldn't build a child job at the same time. To build the child job, you have to complete the build of the parent job first.

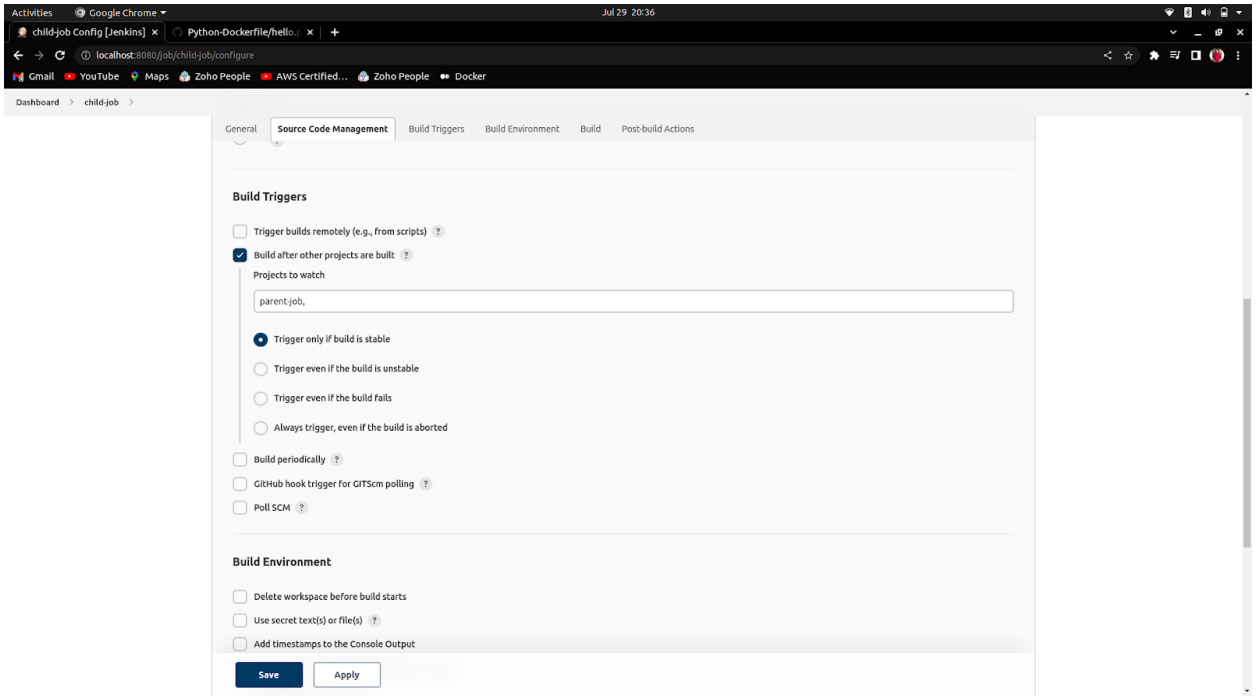


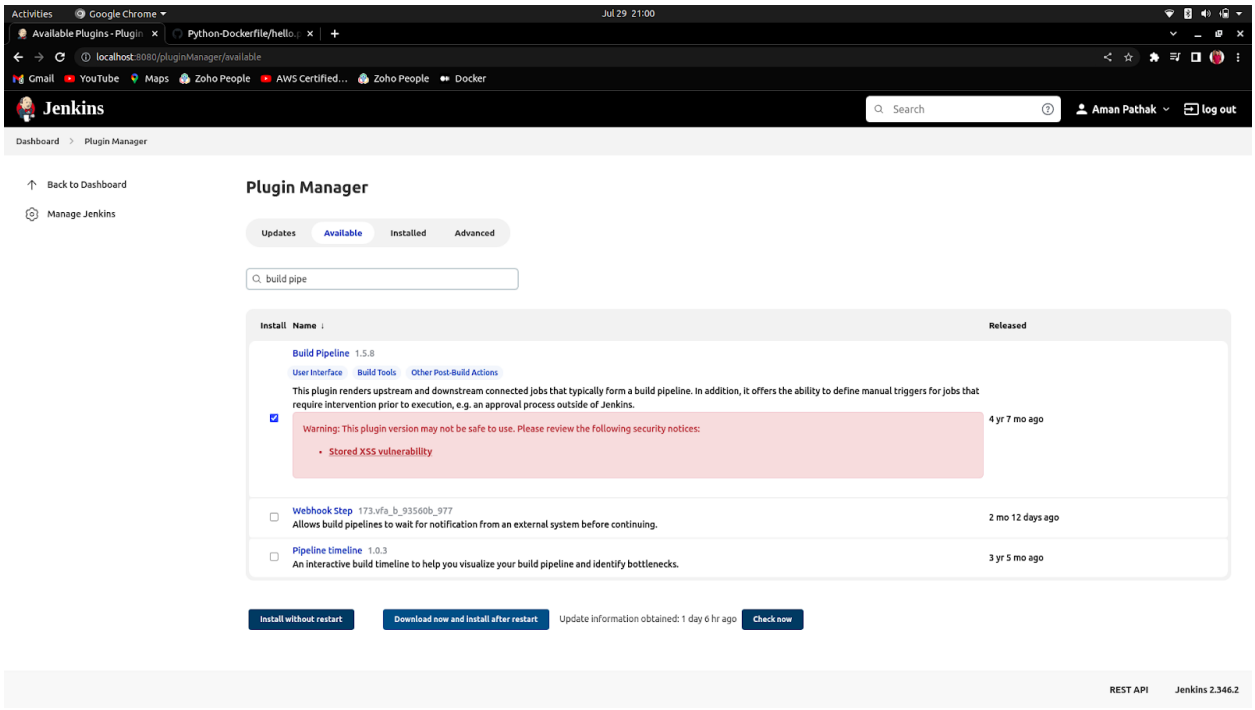

Activities
Google Chrome
Jul 29 20:40
parent-job Config [Jenkin
Python-Dockerfile/hello.
localhost:8080/job/parent-job/configure
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
parent-job
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Build Triggers
Trigger builds remotely (e.g., from scripts)
Build after other projects are built
Projects to watch
child-job,
Trigger only if build is stable
Trigger even if the build is unstable
Trigger even if the build fails
Always trigger, even if the build is aborted
Build periodically
GitHub hook trigger for GITScm polling
Poll SCM
Build Environment
Delete workspace before build starts
Use secret text(s) or file(s)
Add timestamps to the Console Output
Inspect build log for published Gradle build scans
Save
Apply

Block build when downstream project is building

Here, the scenario is if you are building a child job then, you couldn’t build a parent job at the same time. To build the parent job, you have to complete the build of the child job first.
Complete Reverse of the previous one.

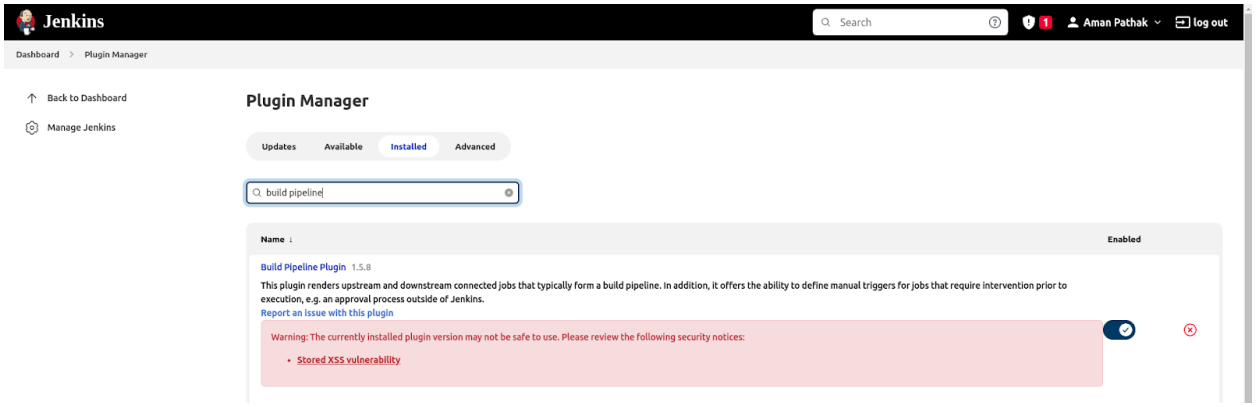



Activities
Google Chrome
Jul 29 20:36
child-job Config [Jenkins]
Python-Dockerfile/hello...
localhost:8080/job/child-job/configure
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
child-job
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Build Triggers
Trigger builds remotely (e.g., from scripts)
Build after other projects are built
Projects to watch
parent-job,
Trigger only if build is stable
Trigger even if the build is unstable
Trigger even if the build fails
Always trigger, even if the build is aborted
Build periodically
GitHub hook trigger for GITScm polling
Poll SCM
Build Environment
Delete workspace before build starts
Use secret text(s) or file(s)
Add timestamps to the Console Output
Save
Apply

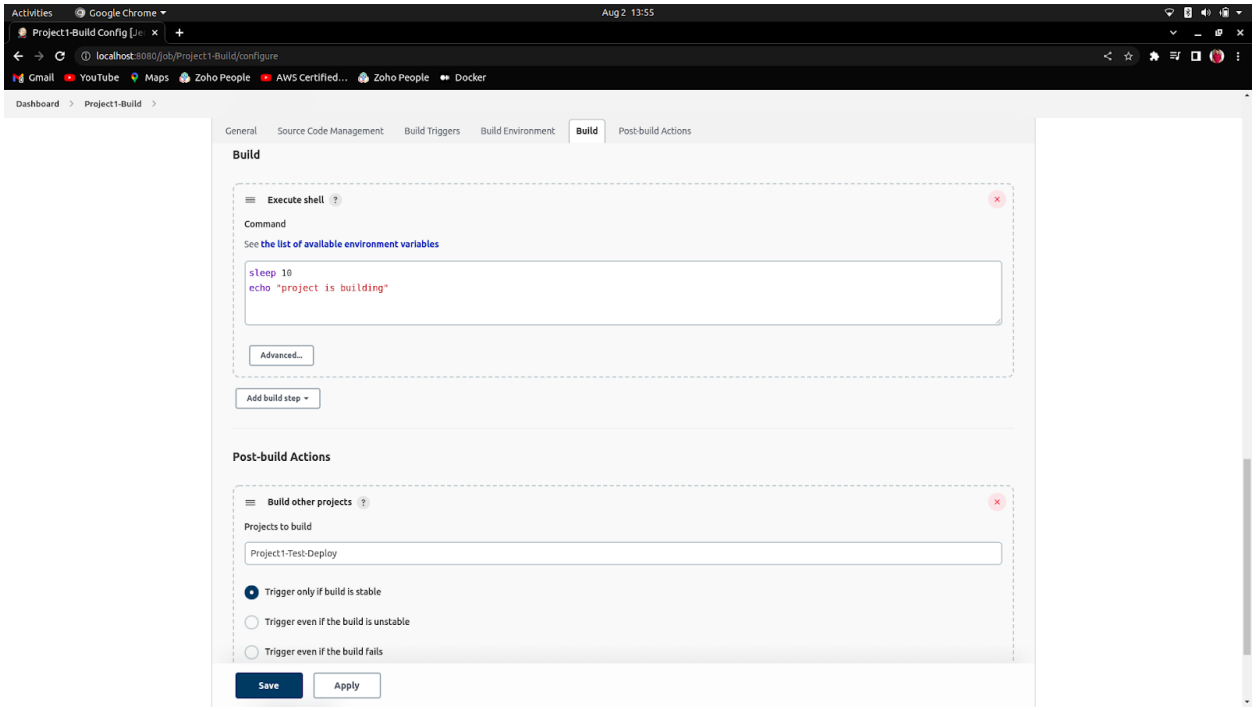

Activities
Google Chrome
Jul 29 21:00
Available Plugins - Plugin
Python-Dockerfile/hello.
localhost:8080/pluginManager/available
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Jenkins
Search
Aman Pathak
log out
Dashboard
Plugin Manager
Back to Dashboard
Manage Jenkins
Plugin Manager
Updates
Available
Installed
Advanced
build pipe
Install
Name ↓
Released
Build Pipeline 1.5.8
User Interface
Build Tools
Other Post-Build Actions
This plugin renders upstream and downstream connected jobs that typically form a build pipeline. In addition, it offers the ability to define manual triggers for jobs that require intervention prior to execution, e.g. an approval process outside of Jenkins.
Warning: This plugin version may not be safe to use. Please review the following security notices:
Stored XSS vulnerability
4 yr 7 mo ago
Webhook Step 173.vfa_b_93560b_977
Allows build pipelines to wait for notification from an external system before continuing.
2 mo 12 days ago
Pipeline timeline 1.0.3
An interactive build timeline to help you visualize your build pipeline and identify bottlenecks.
3 yr 5 mo ago
Install without restart
Download now and install after restart
Update information obtained: 1 day 6 hr ago
Check now
REST API
Jenkins 2.346.2

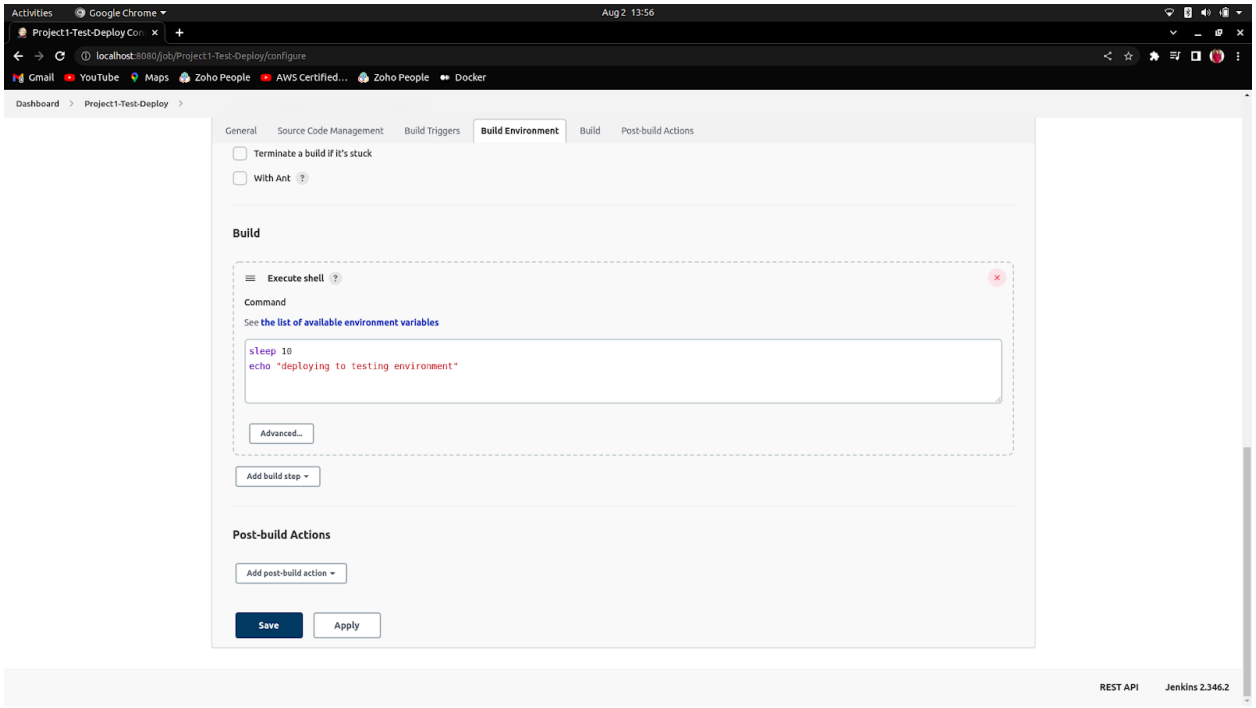
Jenkins Pipeline using Build Pipeline

1. First of all, Install the plugin named Build Pipeline.

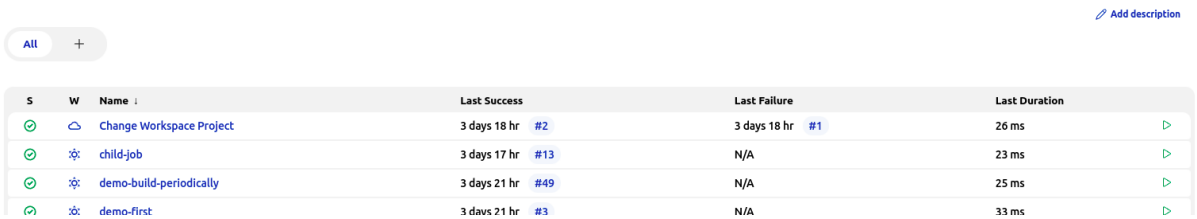


2. Create two Jobs, first will be Parent and the second will be Child job and assign the Post-build actions in Parent Job for Child job as per below:

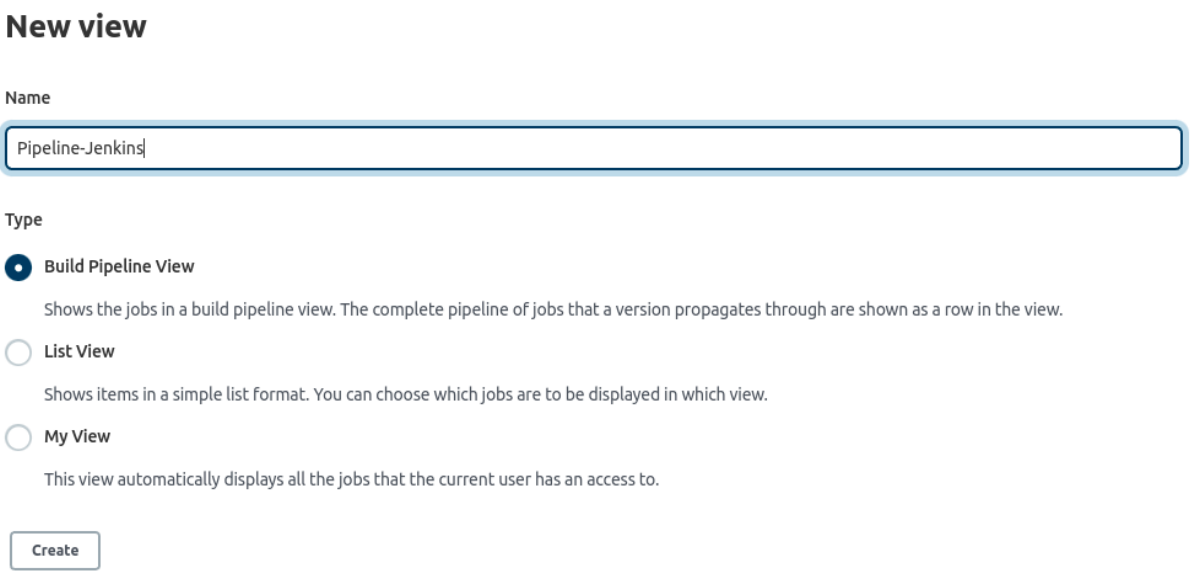




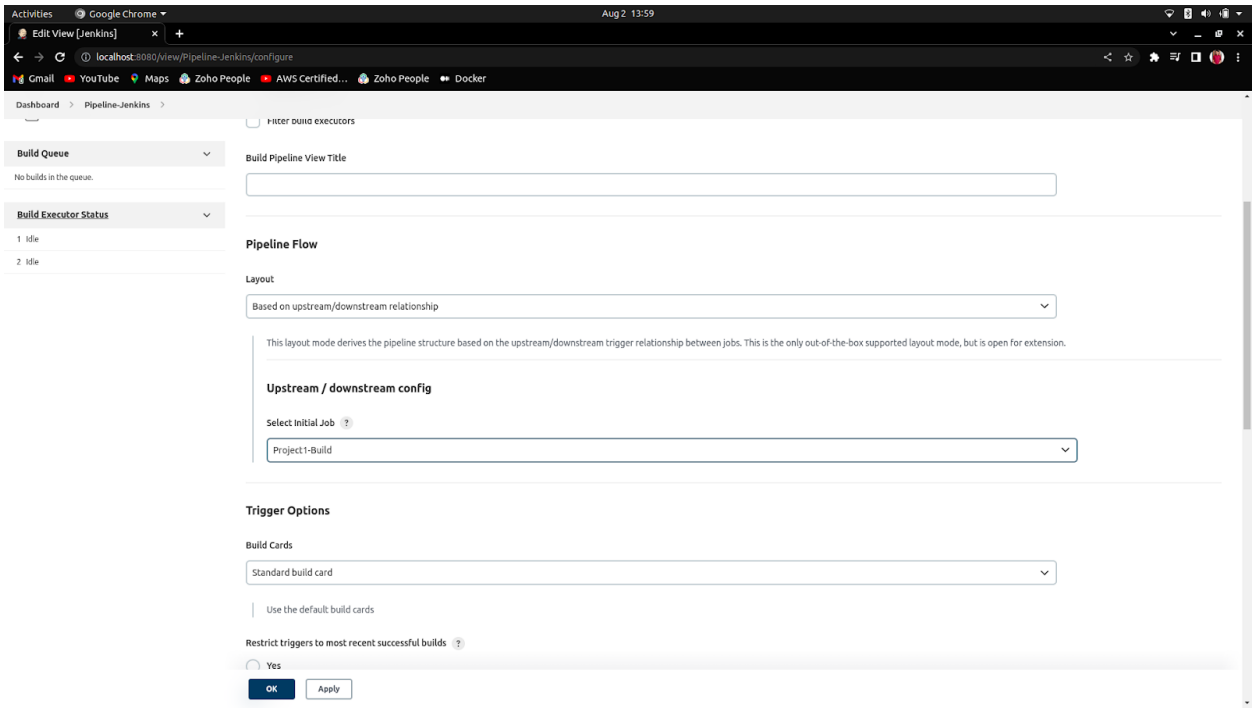
3. Click on “+” to create the Build Pipeline.



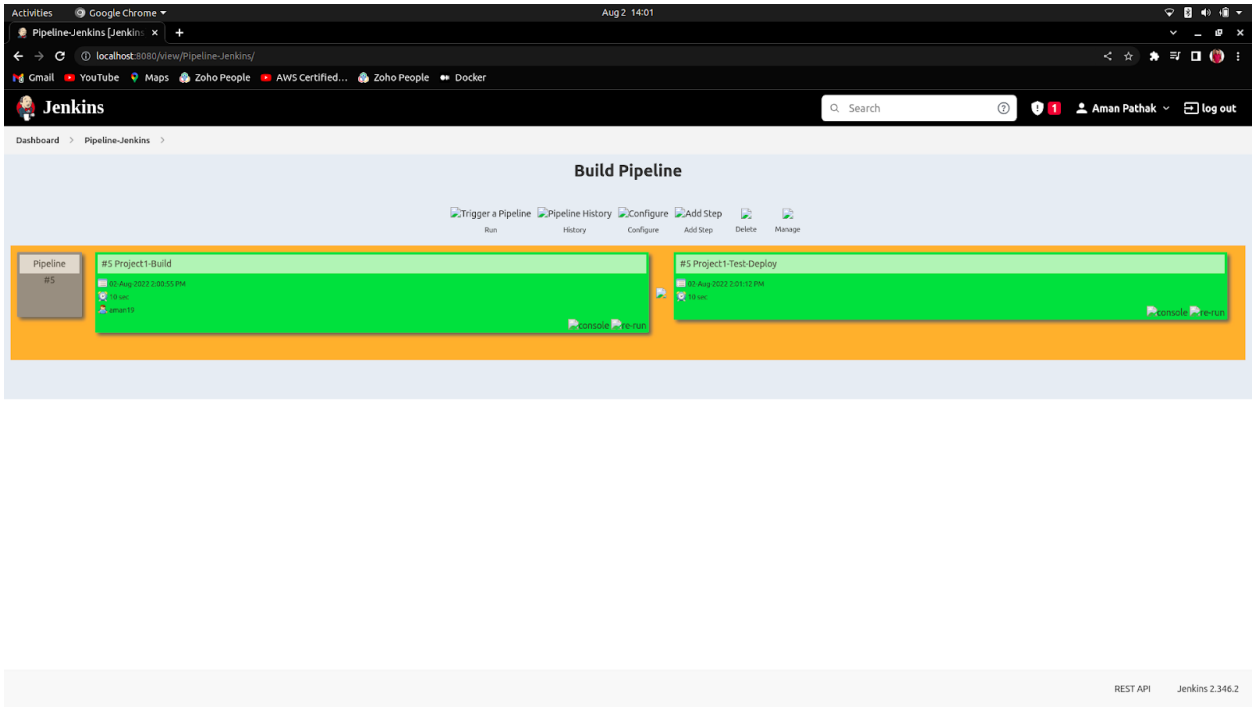
4. Choose Type as “Build Pipeline view” and assign the name of your Pipeline.



5. Now, all you have to do is “Select the Initial Job” which will be your Parent's job.



6. In the last, Click on the run button and the result will be in front of you like this:

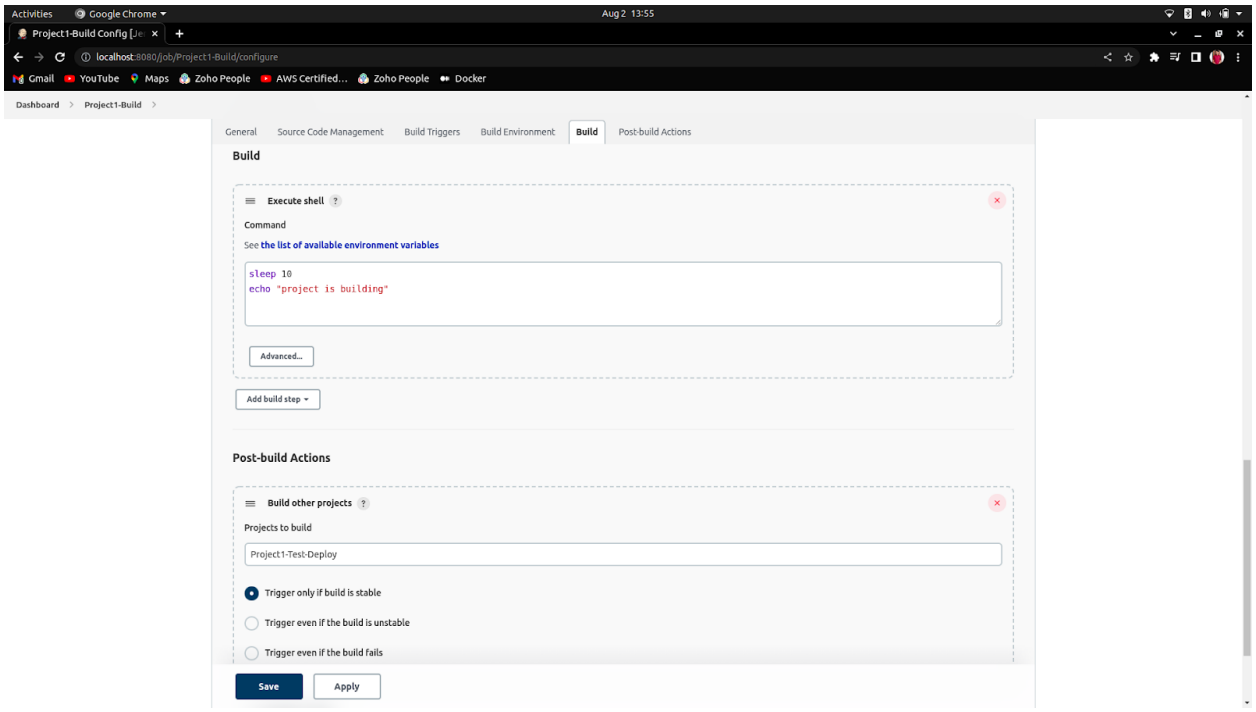


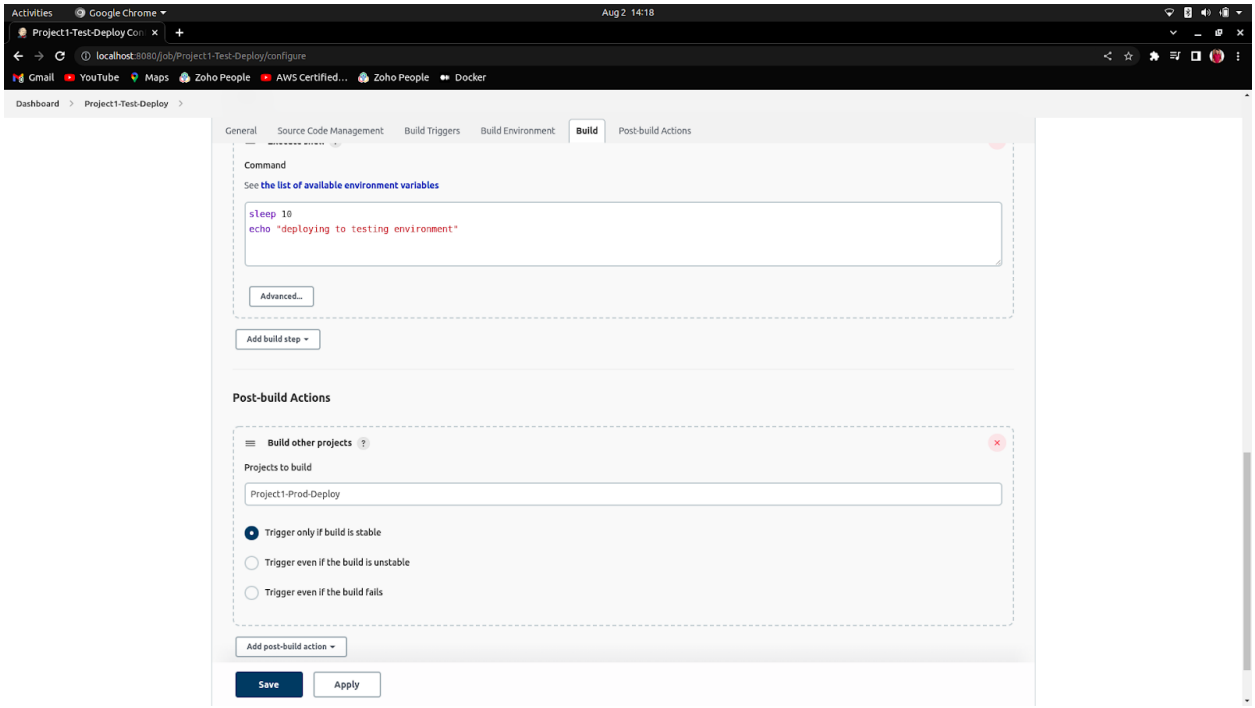
Continuous Deployment vs Continuous Delivery

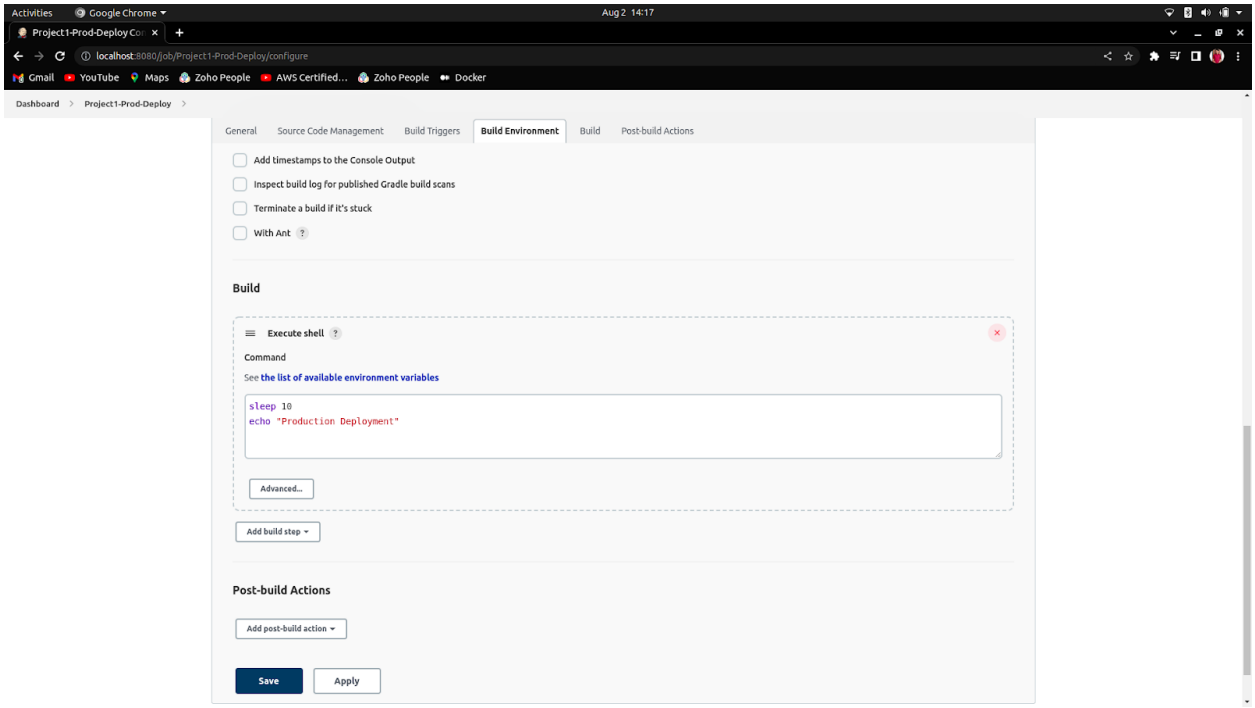
In Continuous Deployment, whenever the code changes to an application is released automatically into the production environment. There is no inclusion of human intervention or manual click.

E.g

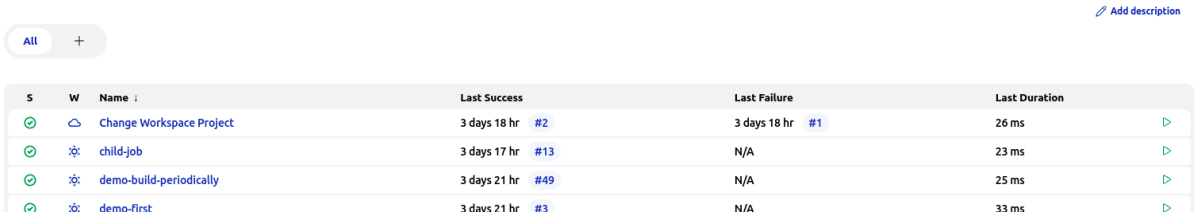
1. Create three Jobs, first will be GrandParent, the second will be Parent job and the third will be Child job and assign the Post-build actions in Grand Parent Job for Parent job and in Parent Job for Child Job as per below:



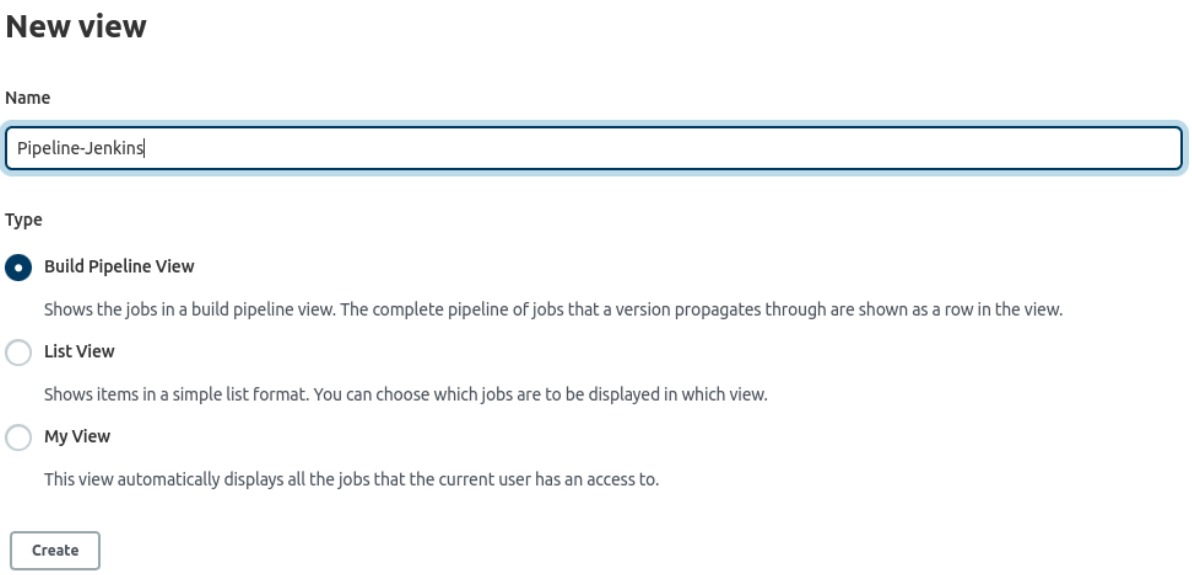

Activities
Google Chrome
Aug 2 14:18
Project1-Test-Deploy Con
localhost:8080/job/Project1-Test-Deploy/configure
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
Project1-Test-Deploy
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Command
See the list of available environment variables
sleep 10
echo "deploying to testing environment"
Advanced...
Add build step
Post-build Actions
Build other projects
Projects to build
Project1-Prod-Deploy
Trigger only if build is stable
Trigger even if the build is unstable
Trigger even if the build fails
Add post-build action
Save
Apply


Activities
Google Chrome
Aug 2 14:17
Project1-Prod-Deploy Con
localhost:8080/job/Project1-Prod-Deploy/configure
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
Project1-Prod-Deploy
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Add timestamps to the Console Output
Inspect build log for published Gradle build scans
Terminate a build if it's stuck
With Ant
Build
Execute shell
Command
See the list of available environment variables
sleep 10
echo "Production Deployment"
Advanced...
Add build step
Post-build Actions
Add post-build action
Save
Apply

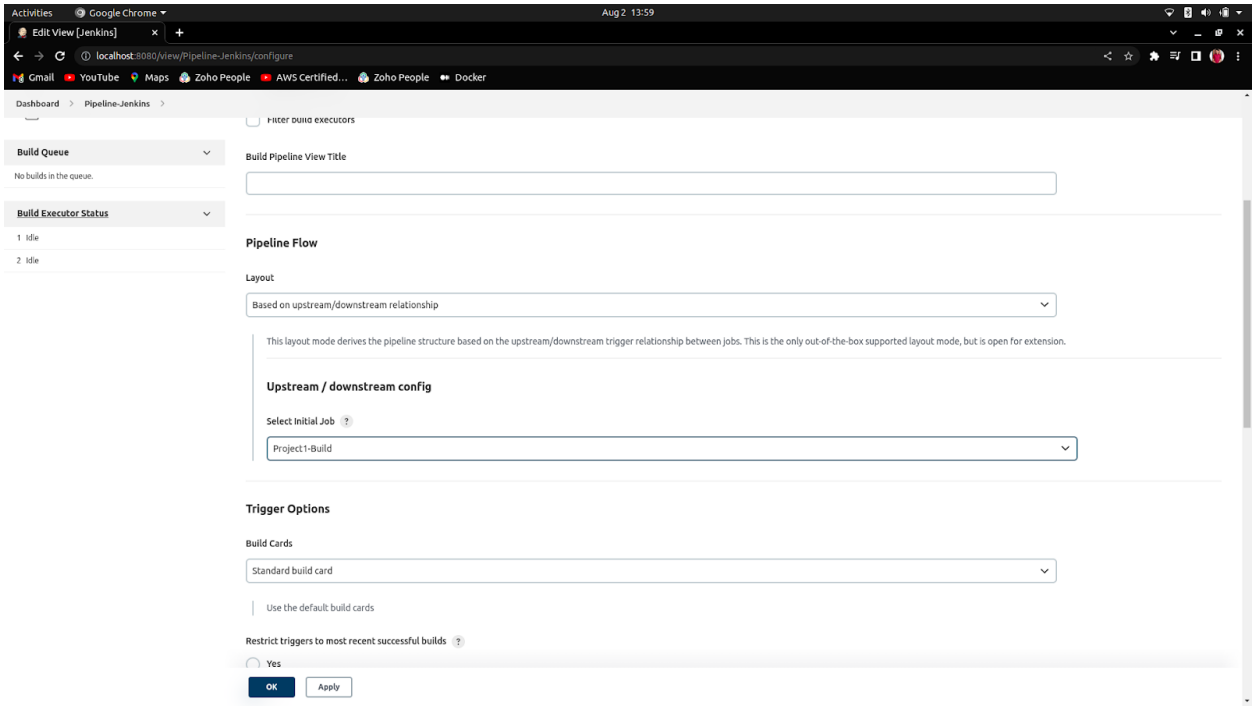
2. Click on “+” to create the Build Pipeline.



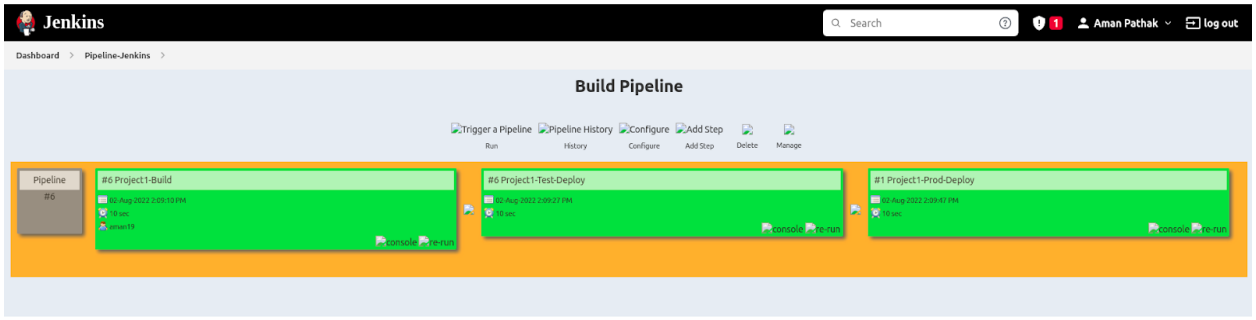
3. Choose “Build Pipeline view” and assign the name of your Pipeline.



4. Now, all you have to do is “Select the Initial Job” which will be your Parent's job.



5. The Final Output:

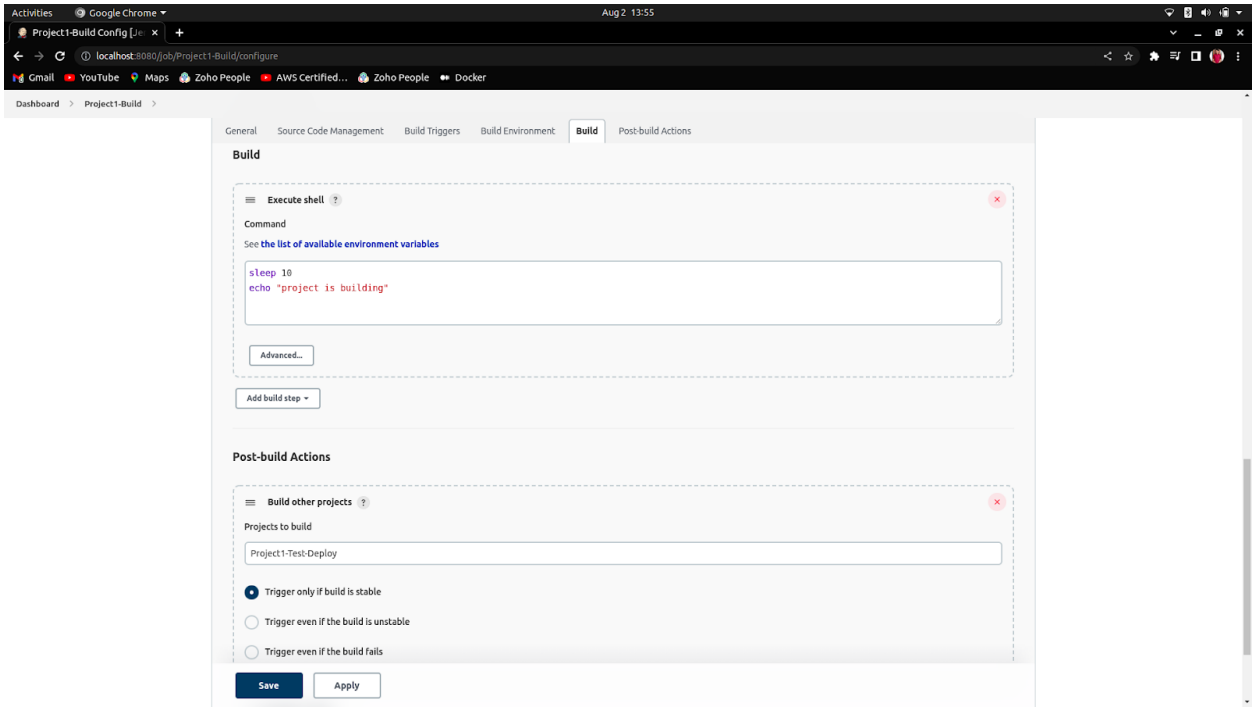


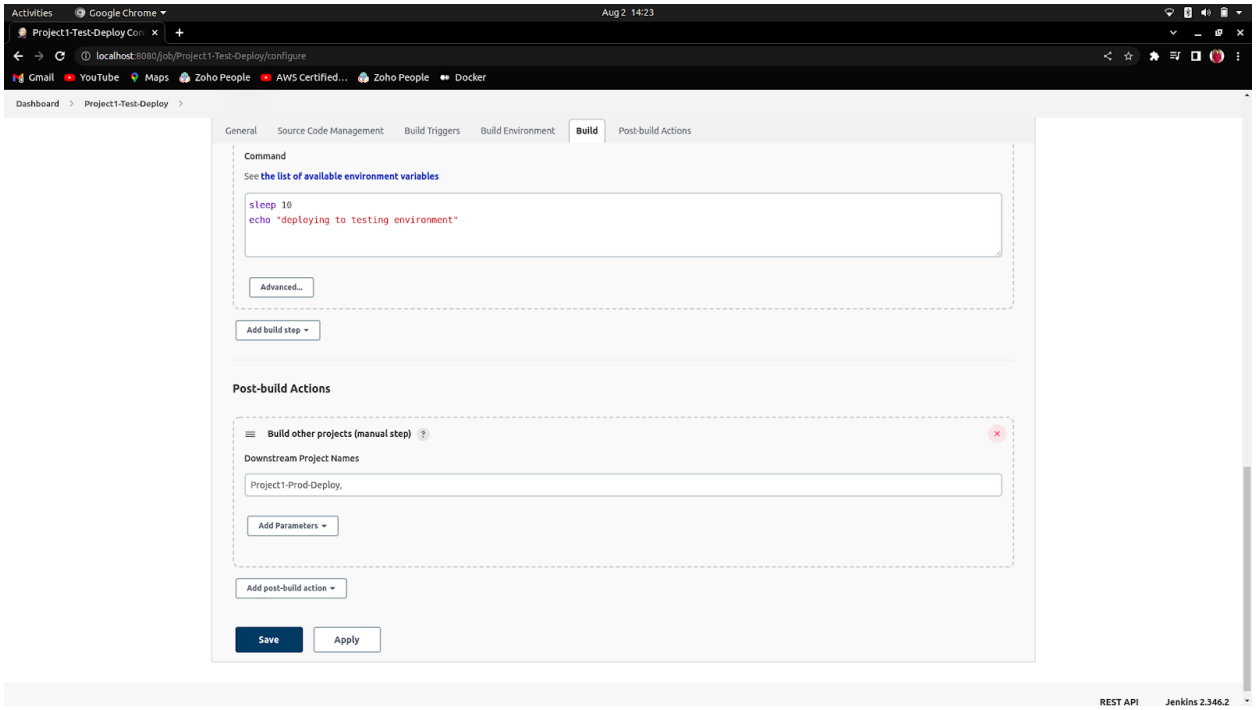
Continuous Delivery

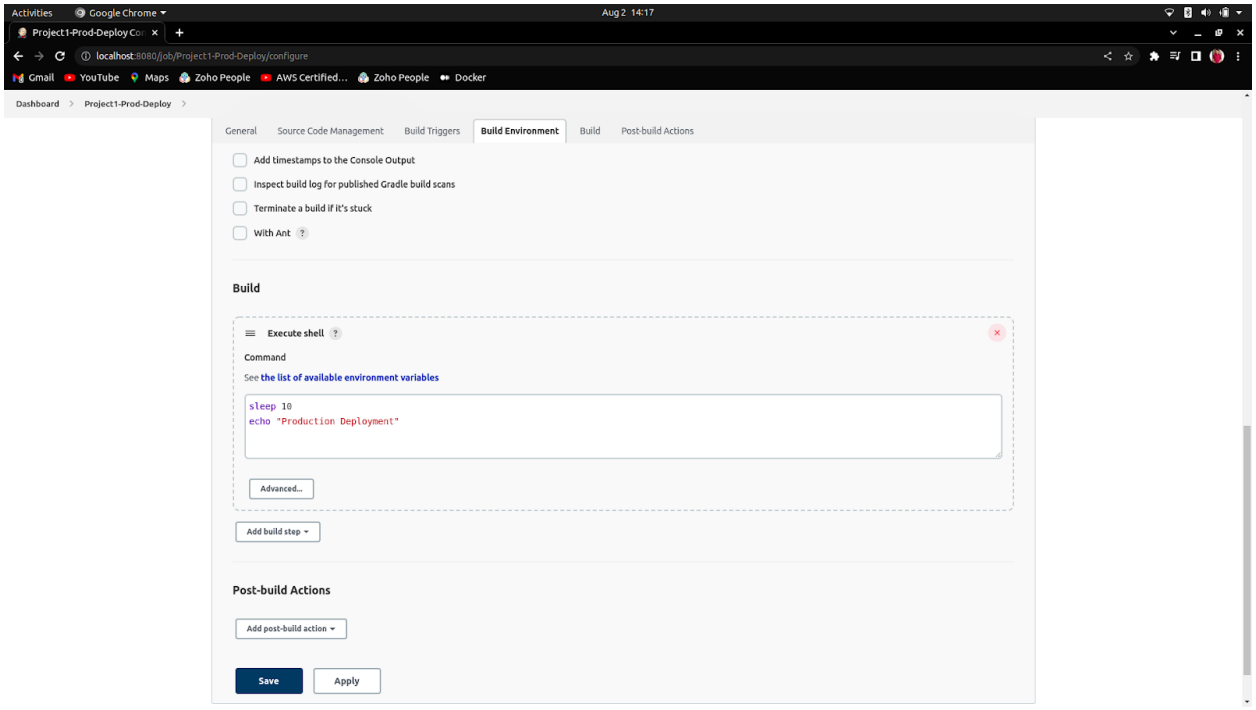
In Continuous Delivery, whenever the code changes to an application are released only by human intervention or through manual clicks into the production environment.There is no any Automation in the Continuous Delivery.

E.g,

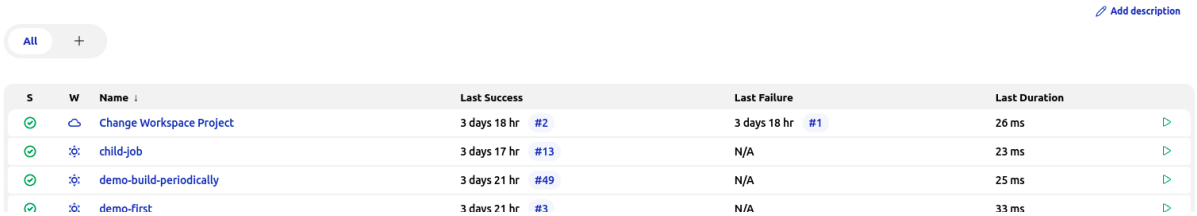
1. Create three Job, first will be GrandParent, the second will be Parent job and third will be Child job and assign the Post-build actions in Grand Parent Job for Parent job and rest the last will be **Manually** as per below:



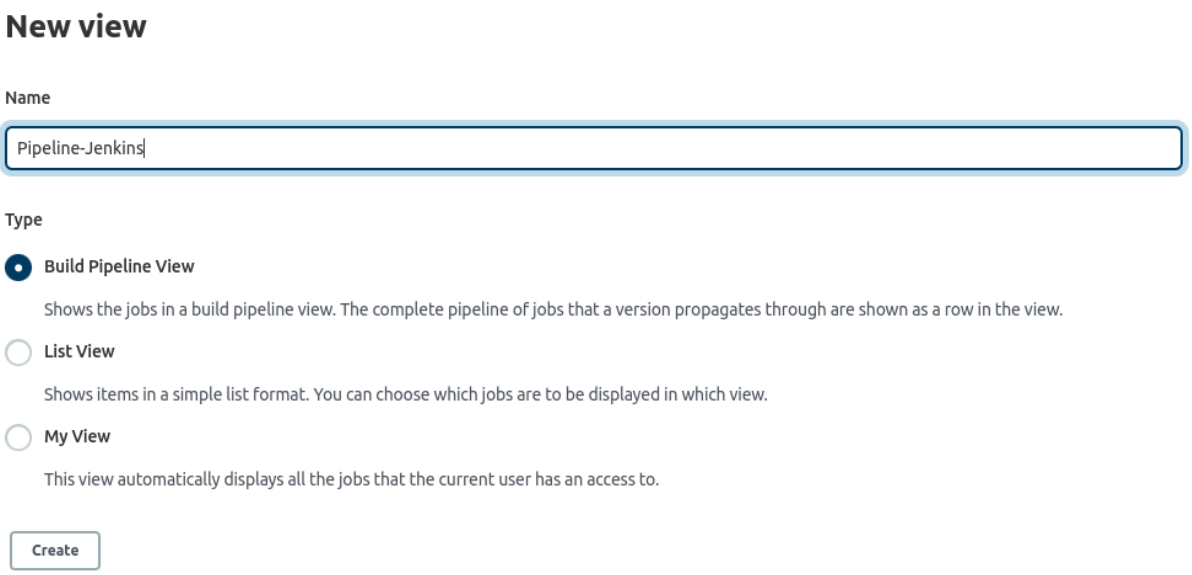

Activities
Google Chrome
Aug 2 14:23
Project1-Test-Deploy Con
localhost:8080/job/Project1-Test-Deploy/configure
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
Project1-Test-Deploy
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Command
See the list of available environment variables
sleep 10
echo "deploying to testing environment"
Advanced...
Add build step
Post-build Actions
Build other projects (manual step)
Downstream Project Names
Project1-Prod-Deploy,
Add Parameters
Add post-build action
Save
Apply
REST API
Jenkins 2.346.2


Activities
Google Chrome
Aug 2 14:17
Project1-Prod-Deploy Con
localhost:8080/job/Project1-Prod-Deploy/configure
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
Project1-Prod-Deploy
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Add timestamps to the Console Output
Inspect build log for published Gradle build scans
Terminate a build if it's stuck
With Ant
Build
Execute shell
Command
See the list of available environment variables
sleep 10
echo "Production Deployment"
Advanced...
Add build step
Post-build Actions
Add post-build action
Save
Apply

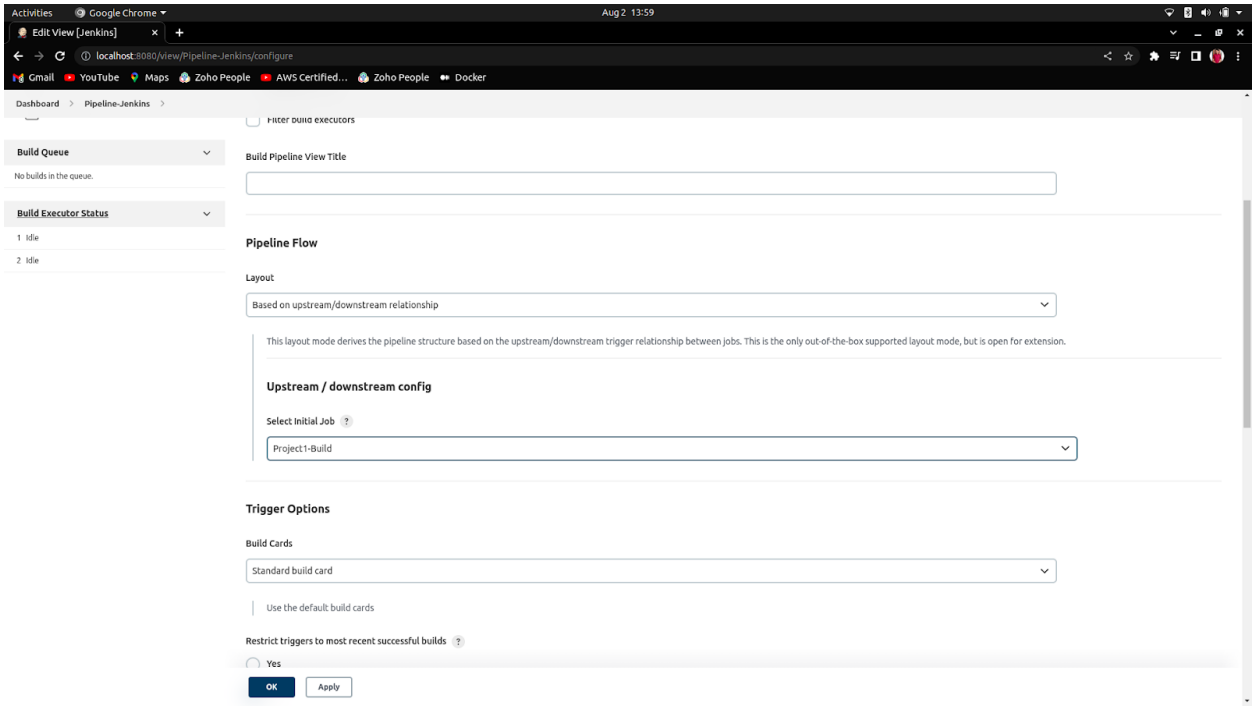
2. Click on “+” to create the Build Pipeline.



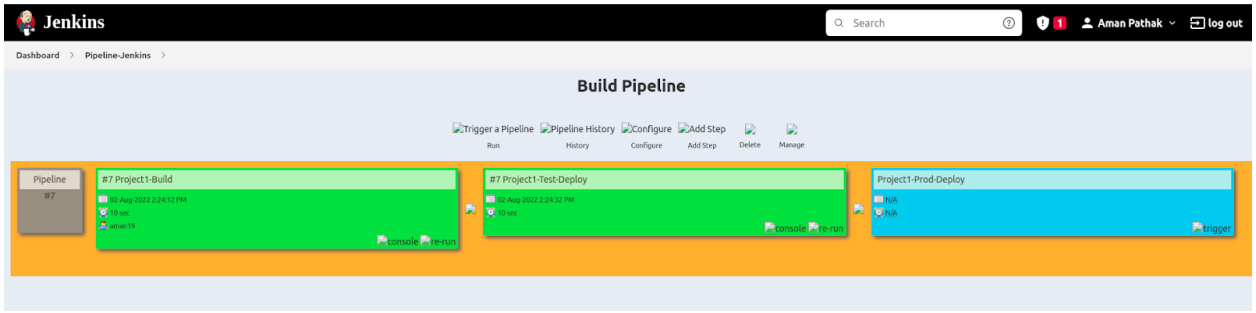
3. Choose Type “Build Pipeline view” and assign the name of your Pipeline.



4. Now, all you have to do is “Select the Initial Job” and which will be your Parent job.

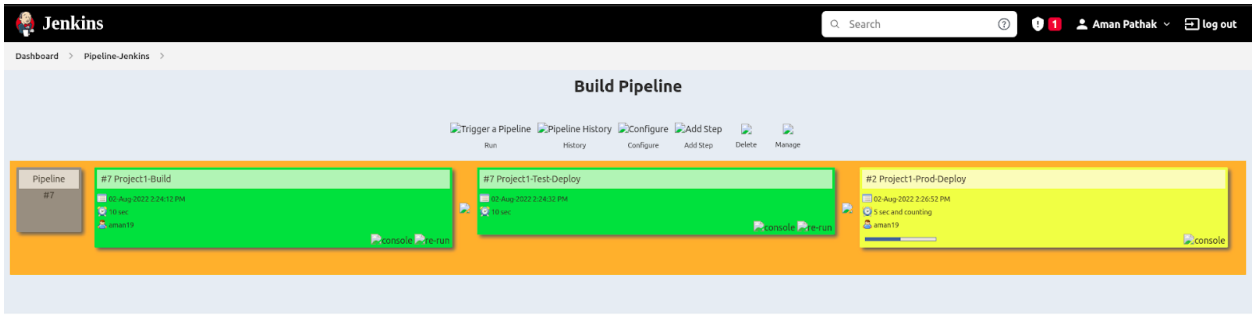


5. Now, when you run the Pipeline then you will face like this.

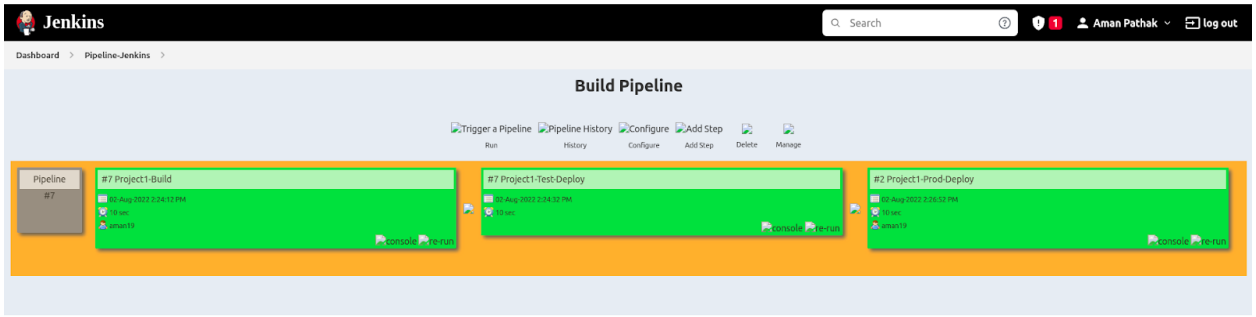


The last job will not run automatically as it is built with manual intervention.

6. So, to run that build too. You have to click on the trigger or run that build(on right bottom).

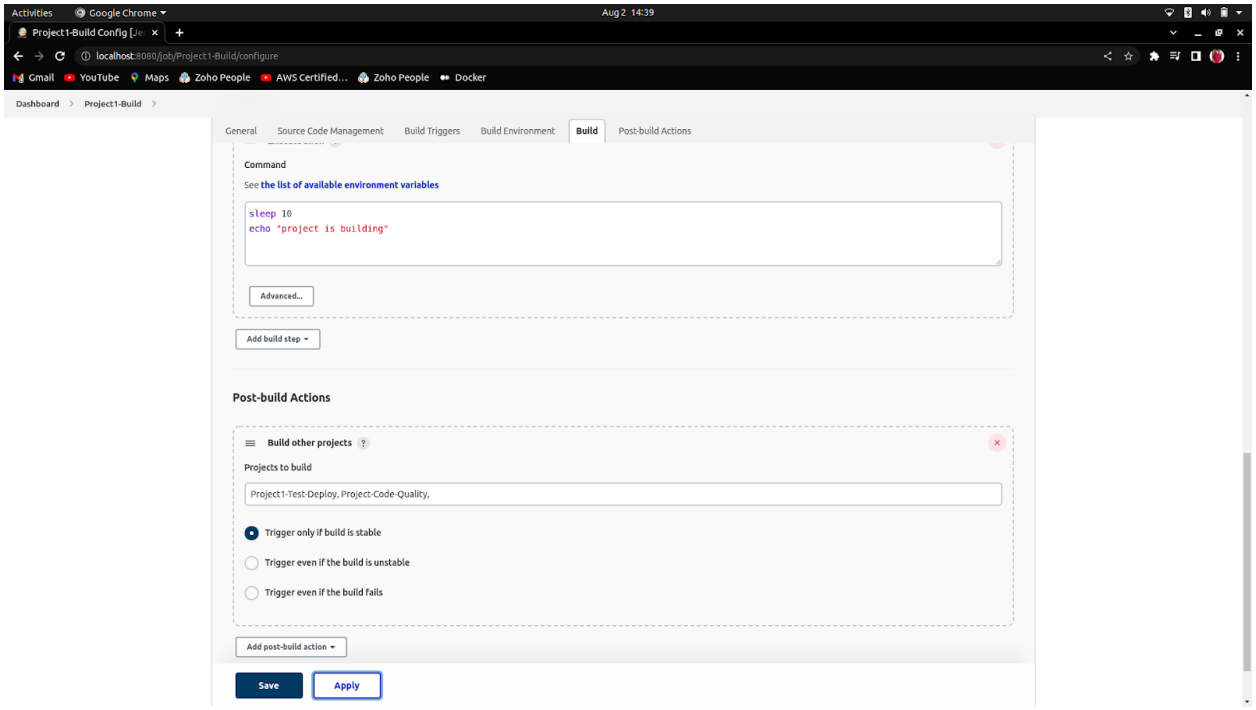


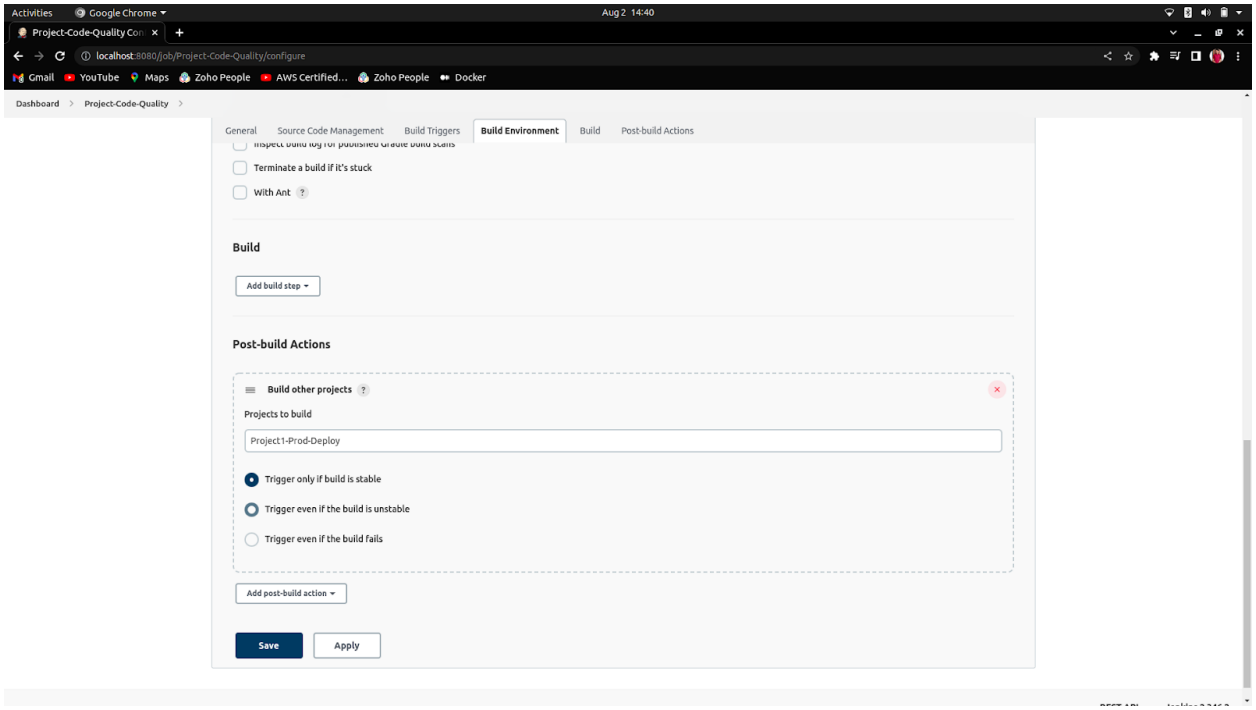
7. The Final Output:

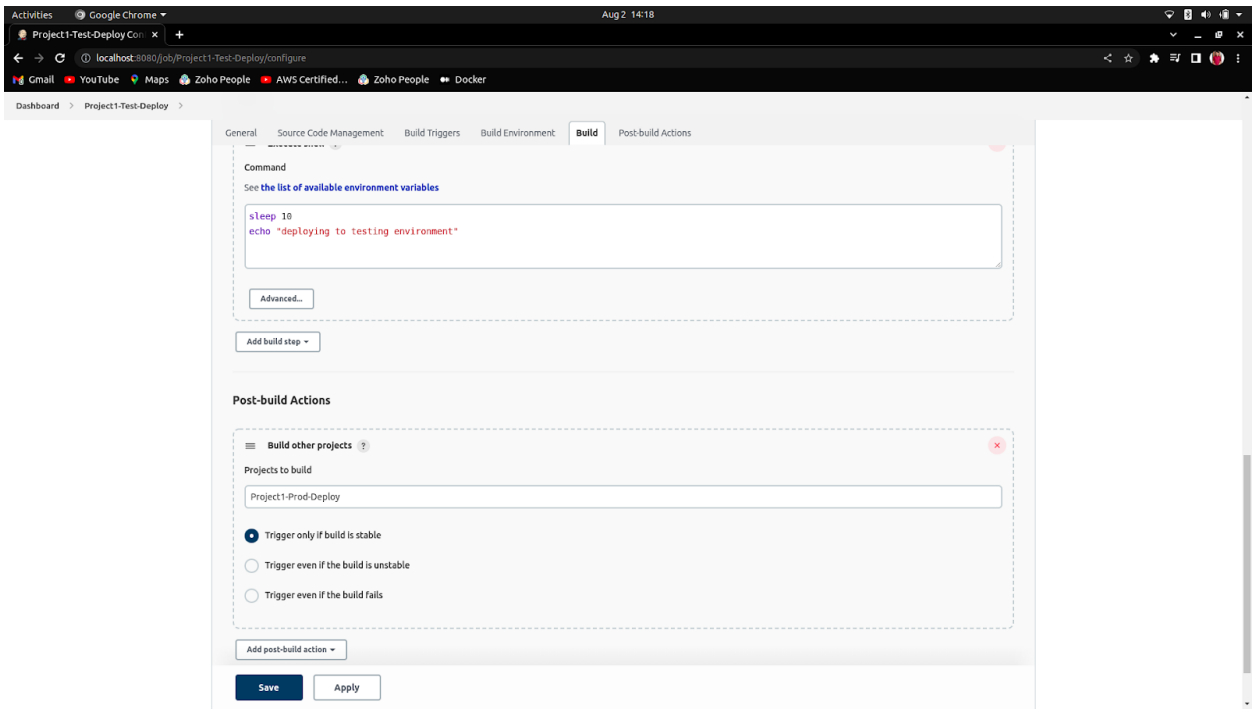


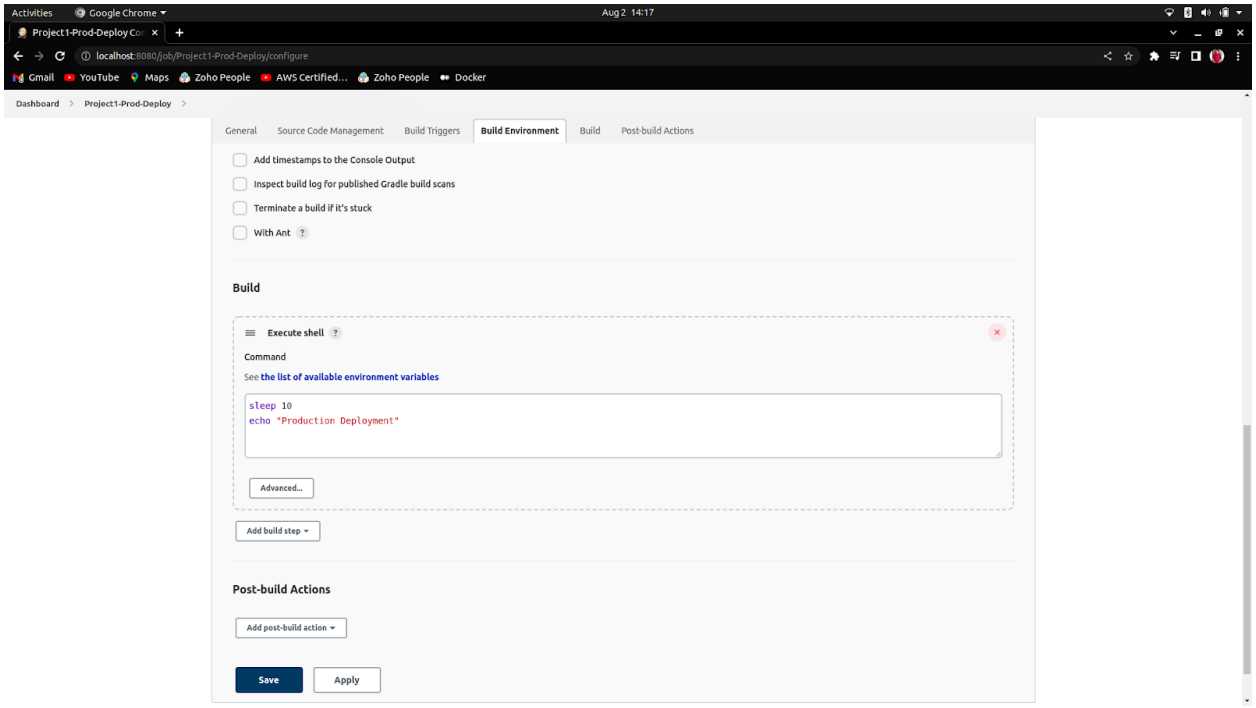
Run Two Job Parallel in Jenkins Pipeline

1. Create three Job, first will be GrandParent, the second will be Parent job and third will be Child job and assign the Post-build actions in Grand Parent Job for Parent job and in Parent Job for Child Job as per below:

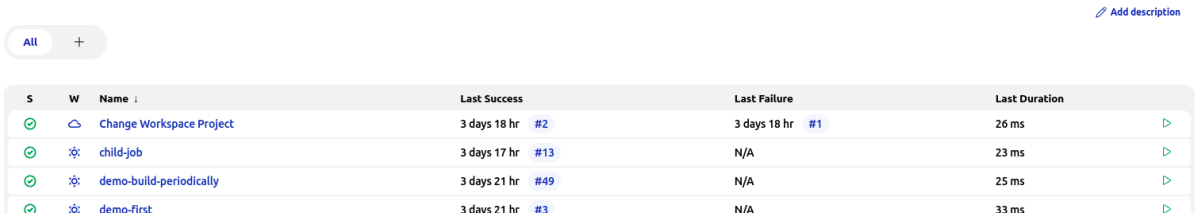



Activities
Google Chrome
Aug 2 14:40
Project-Code-Quality Con
localhost:8080/job/Project-Code-Quality/configure
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
Project-Code-Quality
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Terminate a build if it's stuck
With Ant
Build
Add build step
Post-build Actions
Build other projects
Projects to build
Project1-Prod-Deploy
Trigger only if build is stable
Trigger even if the build is unstable
Trigger even if the build fails
Add post-build action
Save
Apply

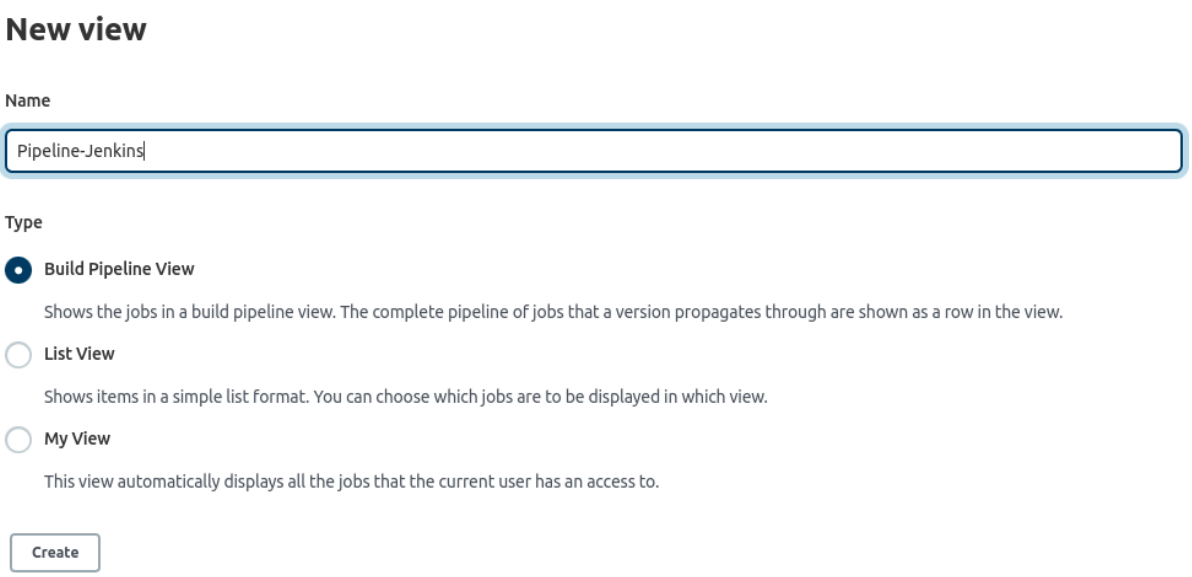

Activities
Google Chrome
Aug 2 14:18
Project1-Test-Deploy Con
localhost:8080/job/Project1-Test-Deploy/configure
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
Project1-Test-Deploy
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Command
See the list of available environment variables
sleep 10
echo "deploying to testing environment"
Advanced...
Add build step
Post-build Actions
Build other projects
Projects to build
Project1-Prod-Deploy
Trigger only if build is stable
Trigger even if the build is unstable
Trigger even if the build fails
Add post-build action
Save
Apply



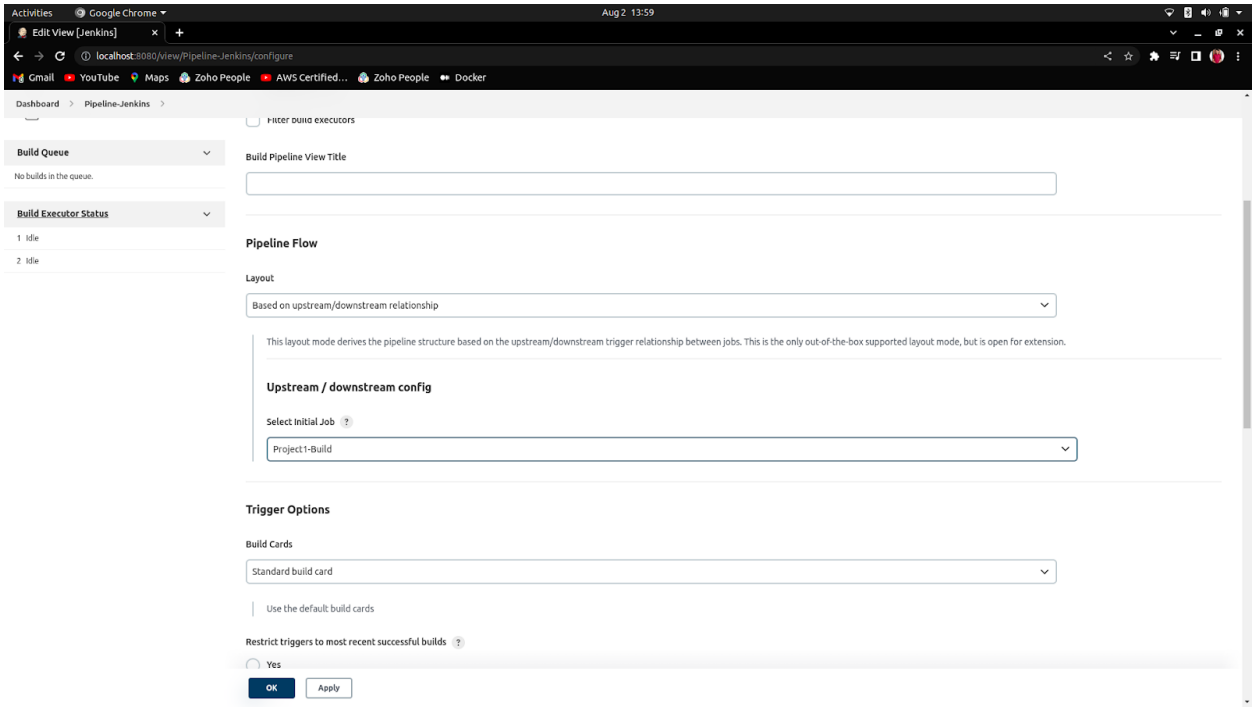
6. Click on “+” to create the Build Pipeline.



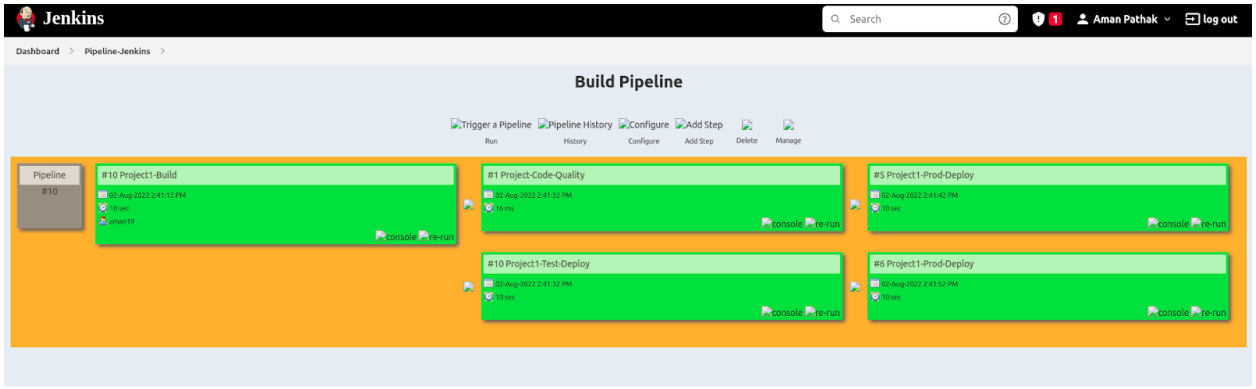
7. Choose Type “Build Pipeline view” and assign the name of your Pipeline.



8. Now, all you have to do is “Select the Initial Job” and which will be your Parent job.



9. The Final Output:



Deploy WAR to Tomcat Server through Jenkins (Automation)

Reference:

- Web Link
- Video Link

1. First of all, Configure the Tomcat Manager by following the below link:
   Tomcat Configuration with Manager

   Apache Tomcat 9.0.65 Download Link

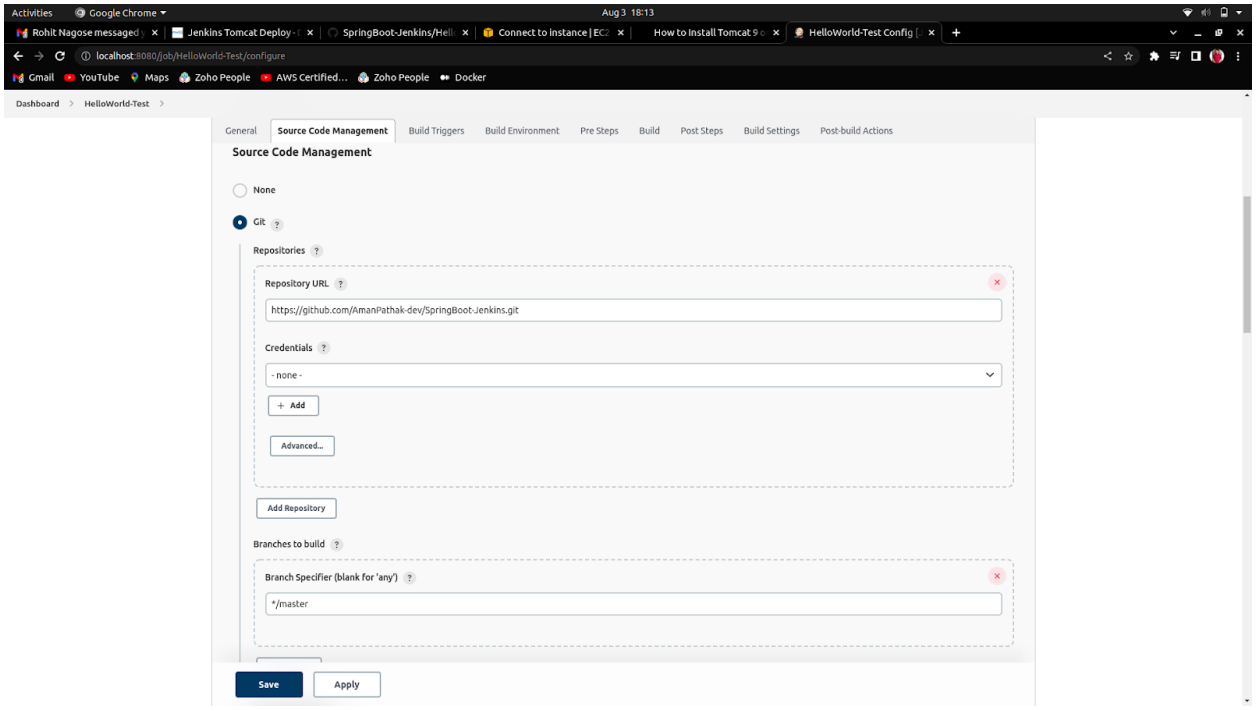
```
<role rolename="admin-gui"/>
<role rolename="manager-script"/>
<role rolename="manager-gui"/>
<user username="aman" password="mypassword" roles="admin-gui,manager-gui,manager-script"/>
```

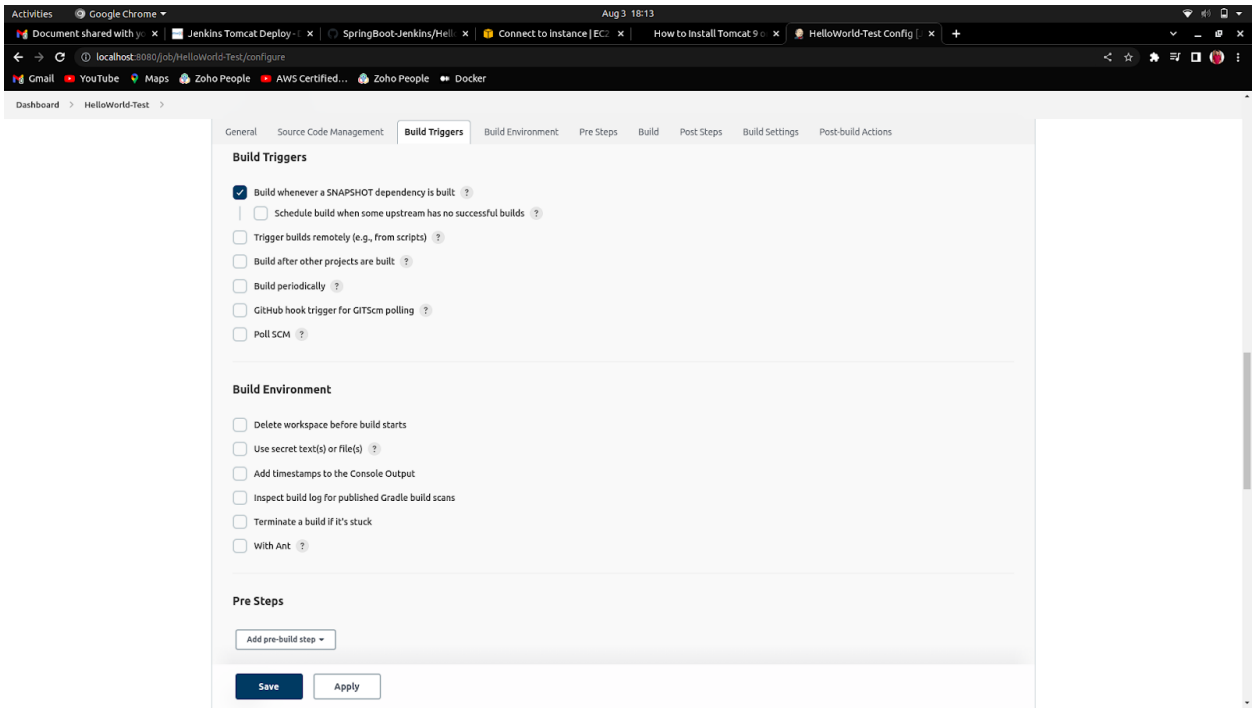
   The Output should be like this:

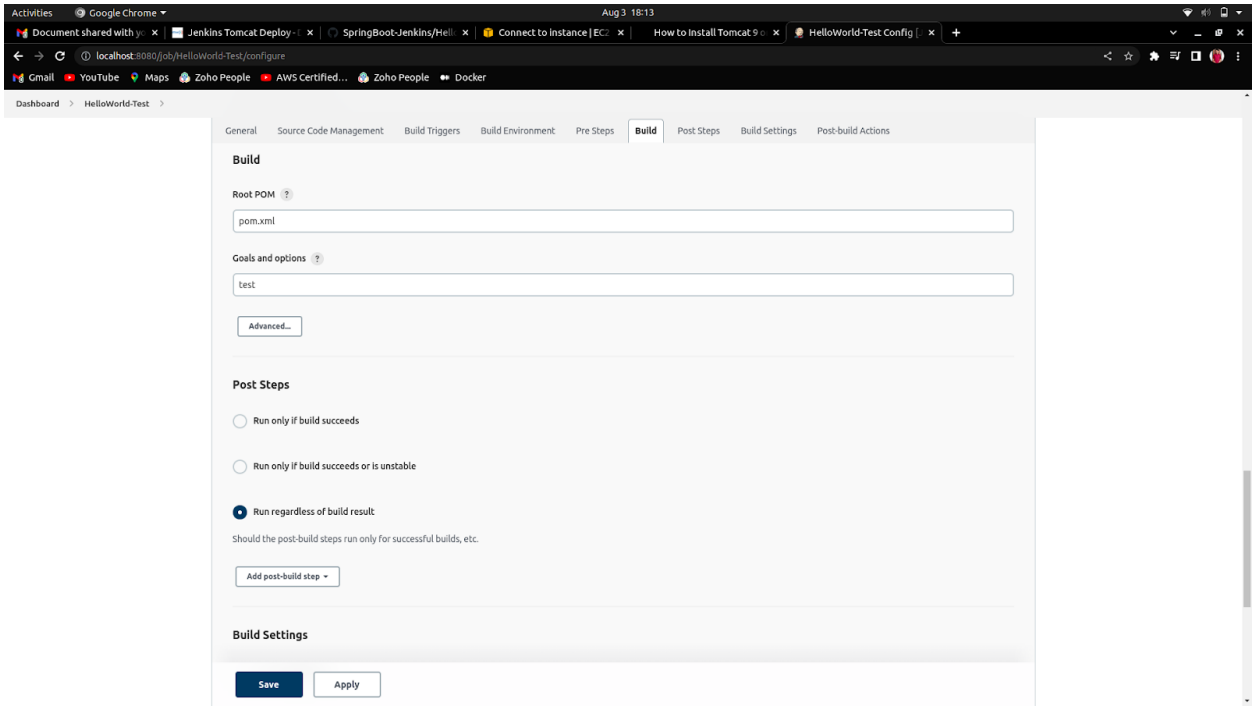
2. Now, You have to Create Four Jobs:
    a. **HelloWorld-Test:** This Job will perform the testing of the Project which is a Spring Boot-based Project.
    b. **Hello-World-Build:** This Job will perform the building of the Spring Boot Project which will be .war file as in the result.
    c. **HelloWorld-Deploy-Test:** In this Job, the deployment will be performed as a testing deployment to check whether the Job is successful or not.
    d. **HelloWorld-Deploy-Prod:** This job will perform the same task that was performed by the last job (HelloWorld-Deploy-Test) but it will be on the Production level where the inclusion is present of human intervention or manual clicks.

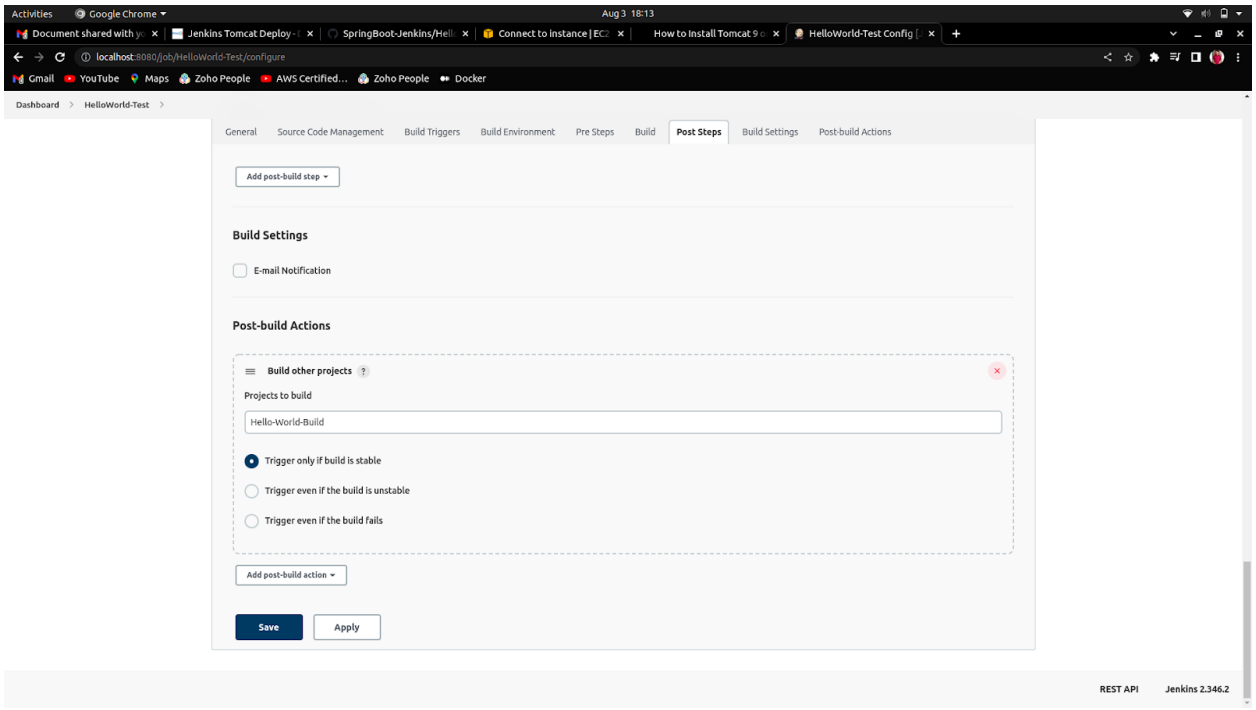
Now, All the jobs have their configuration, which is given below sequentially:

a. **HelloWorld-Test**

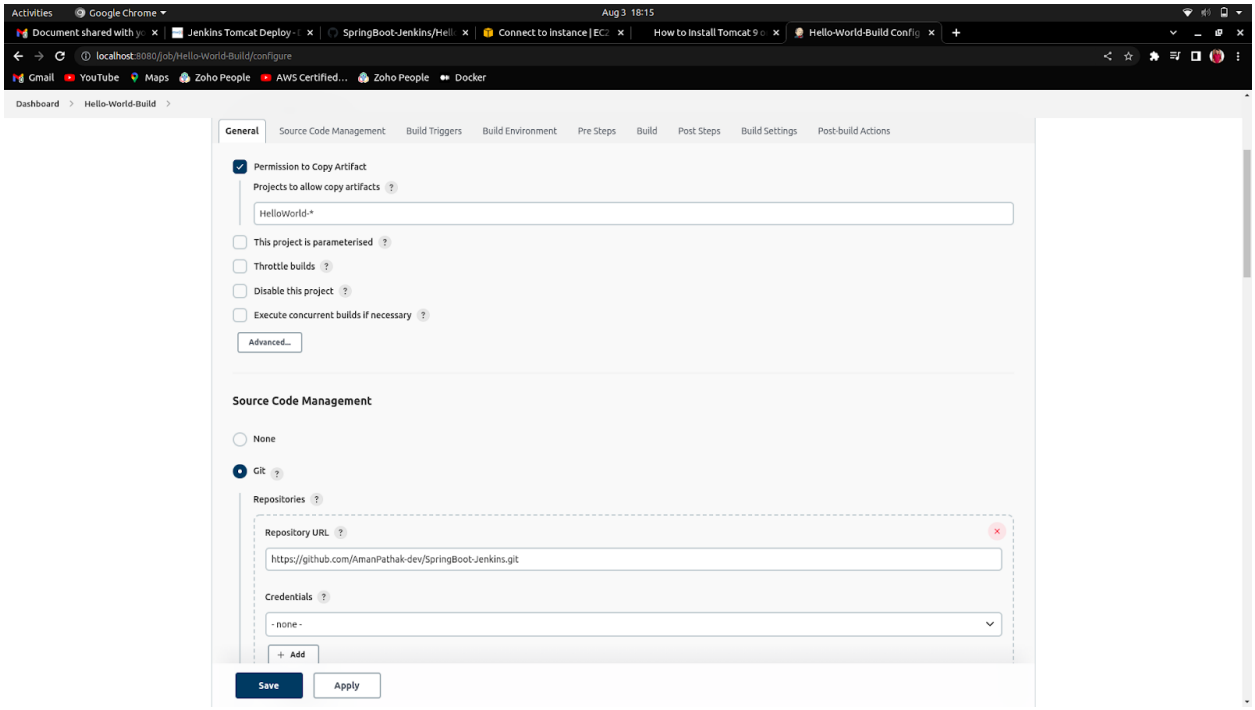

Activities
Google Chrome
Aug 3 18:13
Rohit Nagose messaged
Jenkins Tomcat Deploy
SpringBoot-Jenkins/Hell
Connect to instance | EC2
How to Install Tomcat 9
HelloWorld-Test Config
localhost:8080/job/HelloWorld-Test/configure
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
HelloWorld-Test
General
Source Code Management
Build Triggers
Build Environment
Pre Steps
Build
Post Steps
Build Settings
Post-build Actions
Source Code Management
None
Git
Repositories
Repository URL
https://github.com/AmanPathak-dev/SpringBoot-Jenkins.git
Credentials
- none -
Add
Advanced...
Add Repository
Branches to build
Branch Specifier (blank for 'any')
*/master
Save
Apply

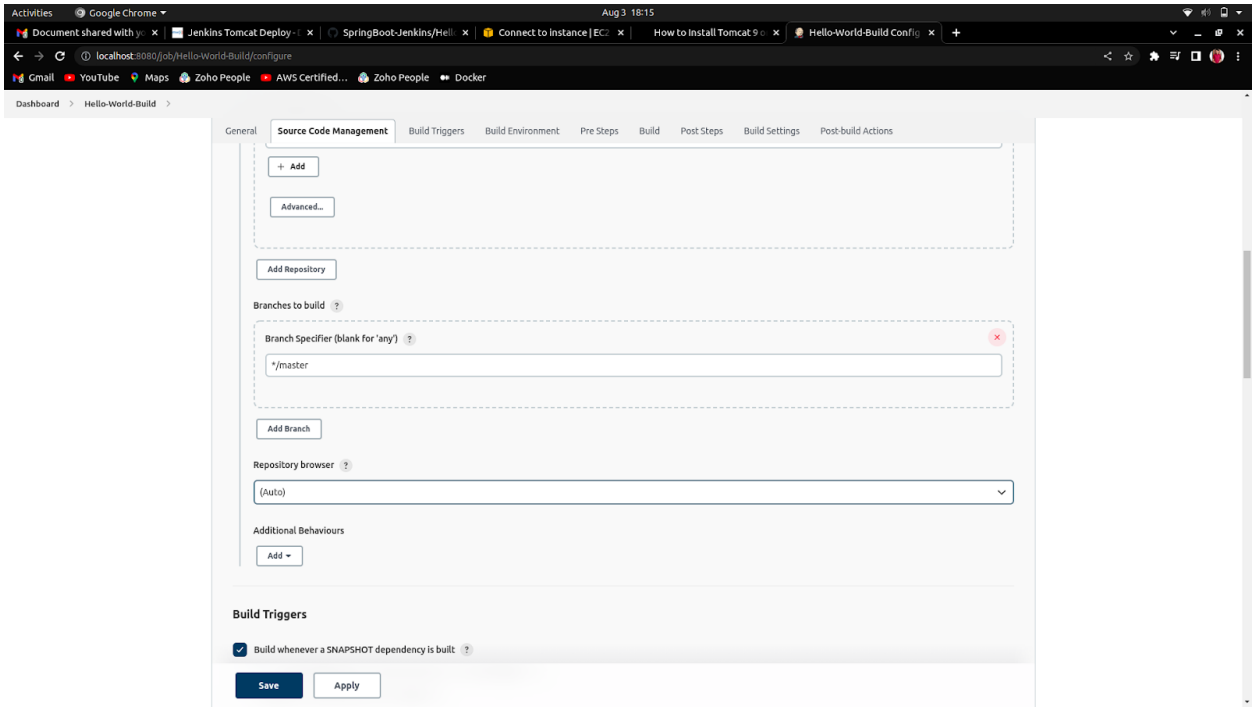

Activities
Google Chrome
Aug 3 18:13
Document shared with
Jenkins Tomcat Deploy
SpringBoot-Jenkins/Hell
Connect to instance | EC2
How to Install Tomcat 9
HelloWorld-Test Config
localhost:8080/job/HelloWorld-Test/configure
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
HelloWorld-Test
General
Source Code Management
Build Triggers
Build Environment
Pre Steps
Build
Post Steps
Build Settings
Post-build Actions
Build Triggers
Build whenever a SNAPSHOT dependency is built
Schedule build when some upstream has no successful builds
Trigger builds remotely (e.g., from scripts)
Build after other projects are built
Build periodically
GitHub hook trigger for GITScm polling
Poll SCM
Build Environment
Delete workspace before build starts
Use secret text(s) or file(s)
Add timestamps to the Console Output
Inspect build log for published Gradle build scans
Terminate a build if it's stuck
With Ant
Pre Steps
Add pre-build step
Save
Apply

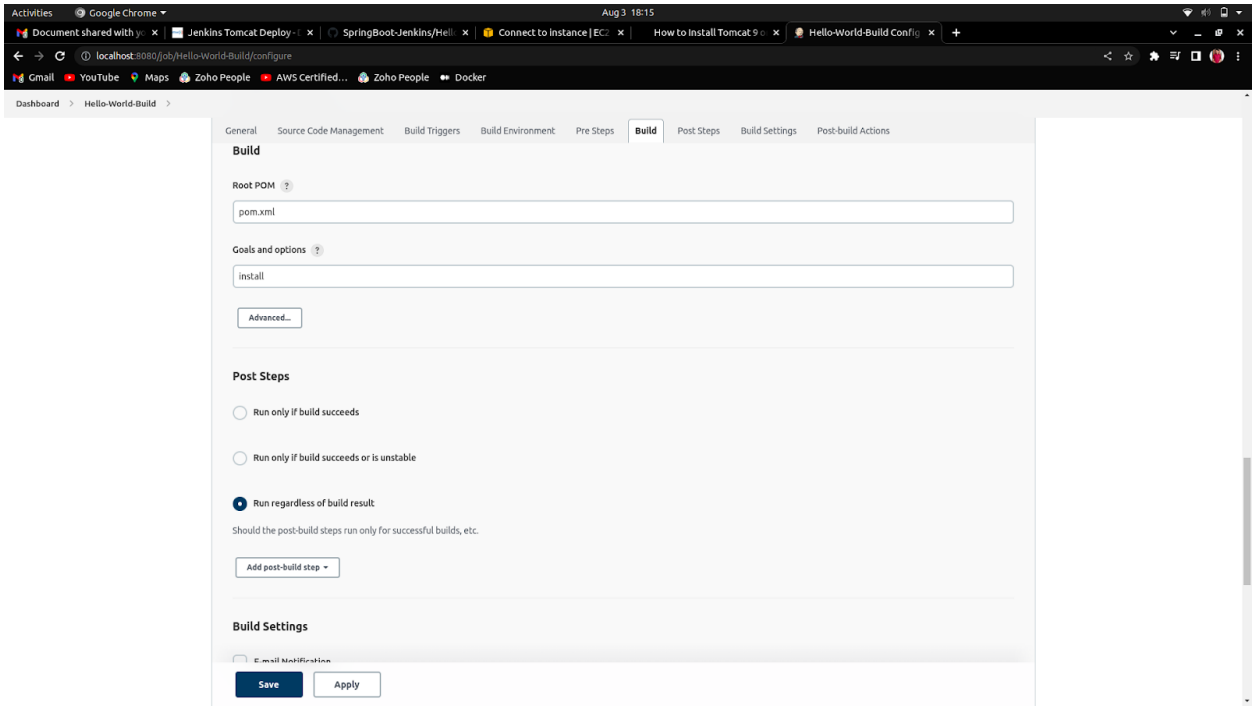

Activities
Google Chrome
Aug 3 18:13
Document shared with
Jenkins Tomcat Deploy
SpringBoot-Jenkins/Hell
Connect to instance | EC2
How to Install Tomcat 9
HelloWorld-Test Config
localhost:8080/job/HelloWorld-Test/configure
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
HelloWorld-Test
General
Source Code Management
Build Triggers
Build Environment
Pre Steps
Build
Post Steps
Build Settings
Post-build Actions
Build
Root POM
pom.xml
Goals and options
test
Advanced...
Post Steps
Run only if build succeeds
Run only if build succeeds or is unstable
Run regardless of build result
Should the post-build steps run only for successful builds, etc.
Add post-build step
Build Settings
Save
Apply

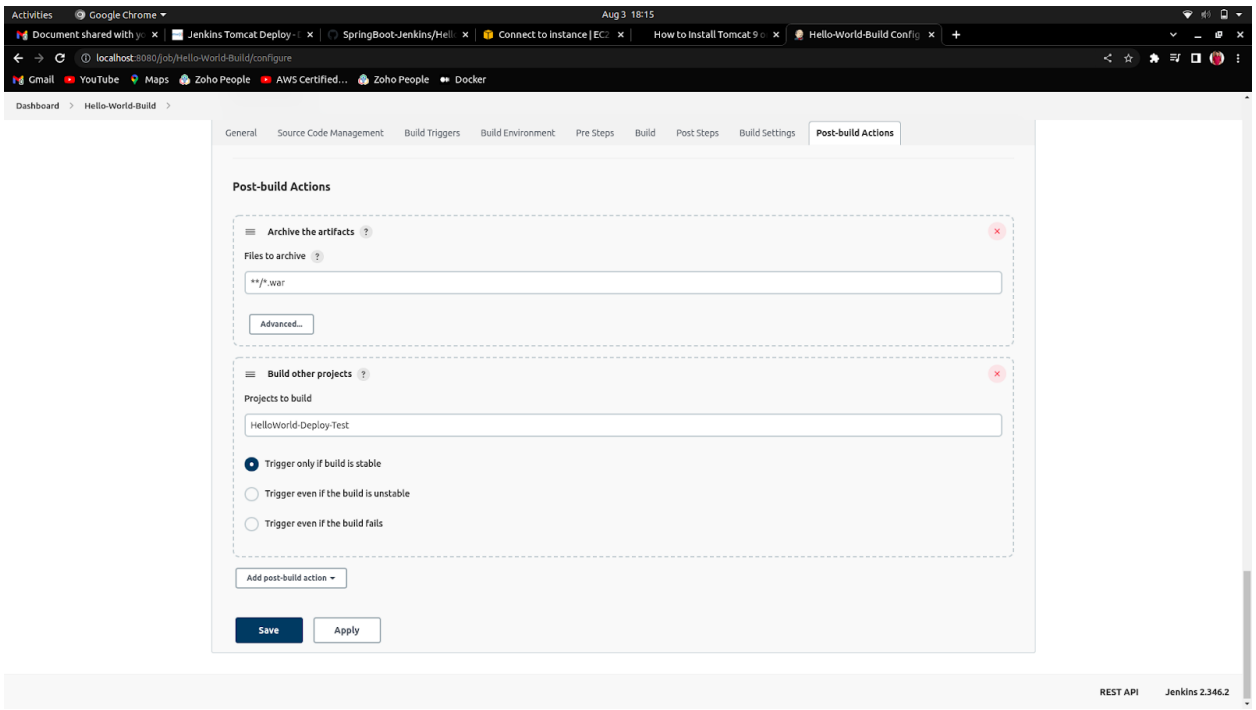

Activities
Google Chrome
Aug 3 18:13
Document shared with
Jenkins Tomcat Deploy
SpringBoot-Jenkins/Hell
Connect to instance | EC2
How to Install Tomcat 9
HelloWorld-Test Config
localhost:8080/job/HelloWorld-Test/configure
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
HelloWorld-Test
General
Source Code Management
Build Triggers
Build Environment
Pre Steps
Build
Post Steps
Build Settings
Post-build Actions
Add post-build step
Build Settings
E-mail Notification
Post-build Actions
Build other projects
Projects to build
Hello-World-Build
Trigger only if build is stable
Trigger even if the build is unstable
Trigger even if the build fails
Add post-build action
Save
Apply
REST API
Jenkins 2.346.2

b. Hello-World-Build:

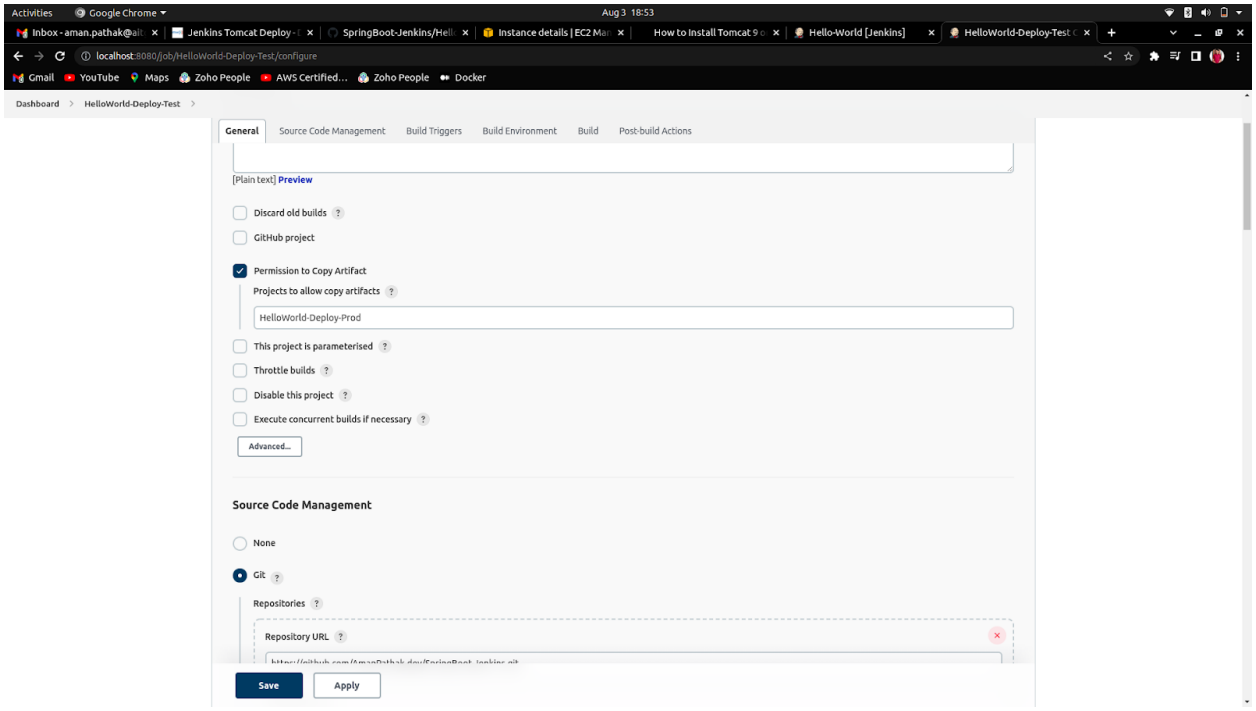


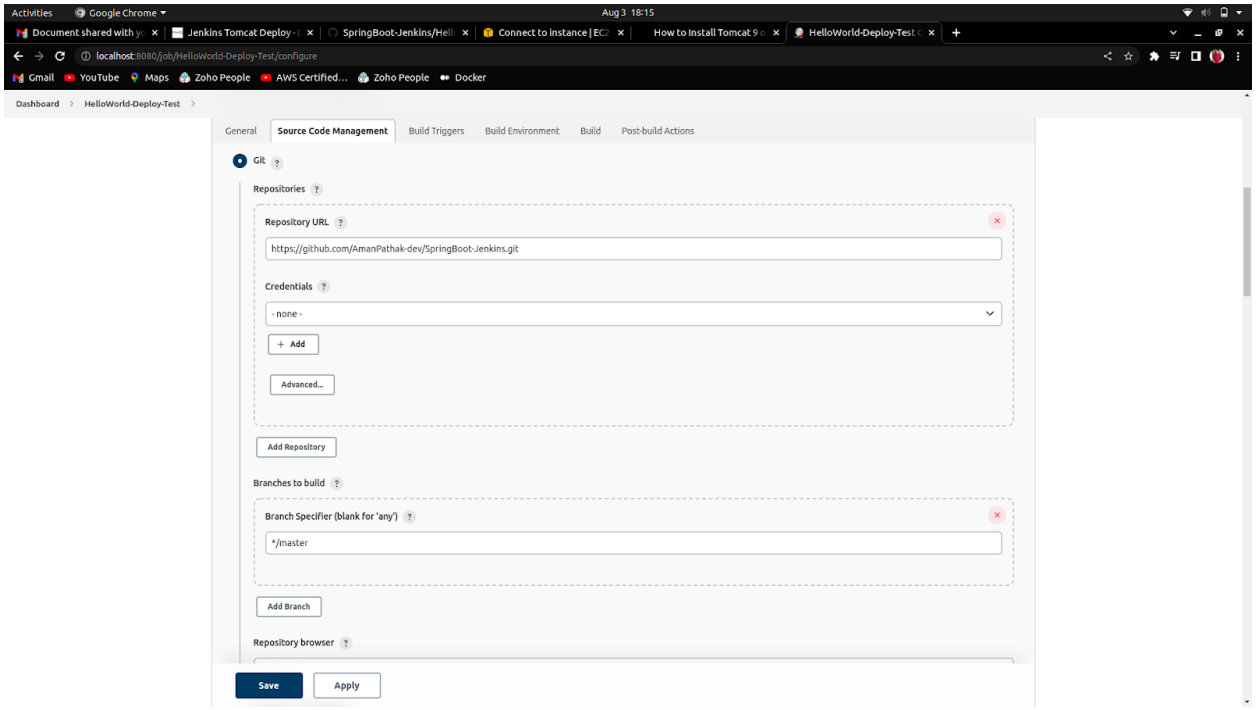


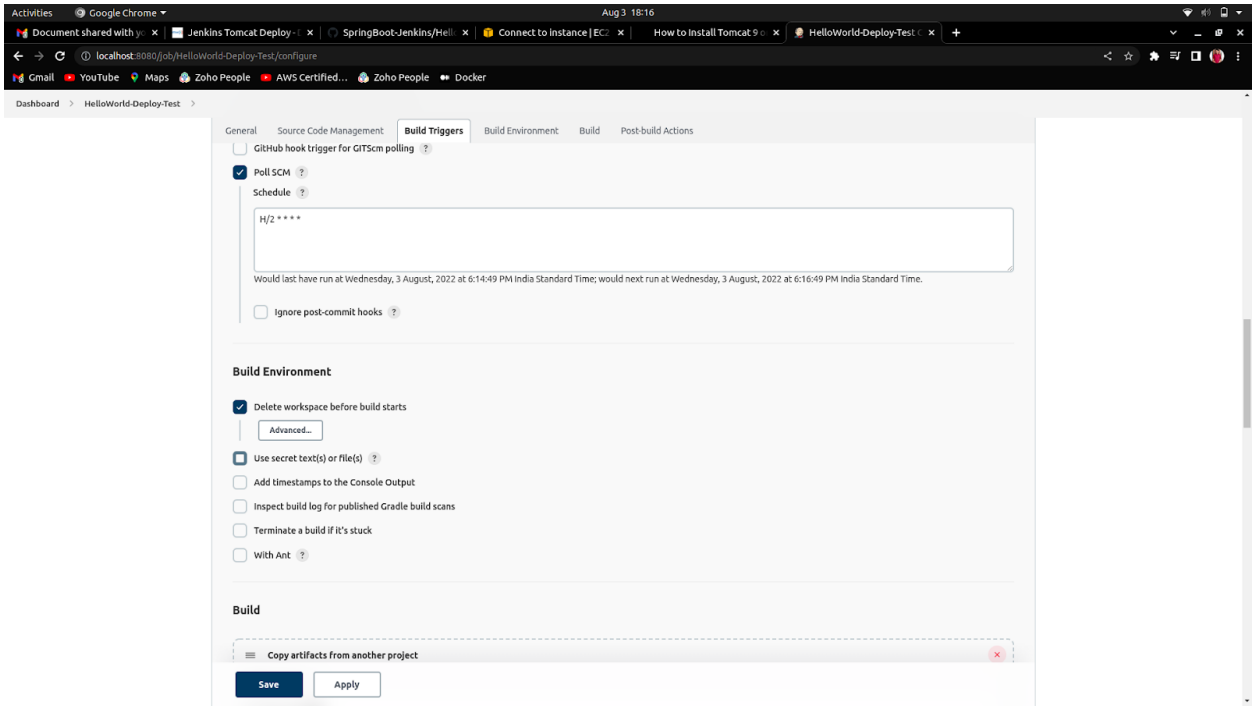

Activities
Google Chrome
Aug 3 18:15
Document shared with
Jenkins Tomcat Deploy
SpringBoot-Jenkins/Hell
Connect to instance | EC2
How to Install Tomcat 9
Hello-World-Build Config
localhost:8080/job/Hello-World-Build/configure
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
Hello-World-Build
General
Source Code Management
Build Triggers
Build Environment
Pre Steps
Build
Post Steps
Build Settings
Post-build Actions
Build
Root POM
pom.xml
Goals and options
install
Advanced...
Post Steps
Run only if build succeeds
Run only if build succeeds or is unstable
Run regardless of build result
Should the post-build steps run only for successful builds, etc.
Add post-build step
Build Settings
Save
Apply

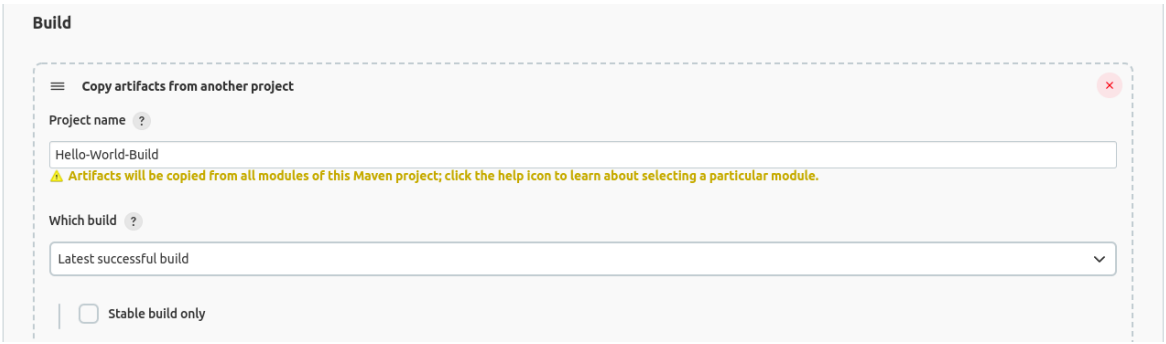

Activities
Google Chrome
Aug 3 18:15
Document shared with
Jenkins Tomcat Deploy
SpringBoot-Jenkins/Hell
Connect to instance | EC2
How to Install Tomcat 9
Hello-World-Build Config
localhost:8080/job/Hello-World-Build/configure
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
Hello-World-Build
General
Source Code Management
Build Triggers
Build Environment
Pre Steps
Build
Post Steps
Build Settings
Post-build Actions
Post-build Actions
Archive the artifacts
Files to archive
**/*.war
Advanced...
Build other projects
Projects to build
HelloWorld-Deploy-Test
Trigger only if build is stable
Trigger even if the build is unstable
Trigger even if the build fails
Add post-build action
Save
Apply
REST API
Jenkins 2.346.2

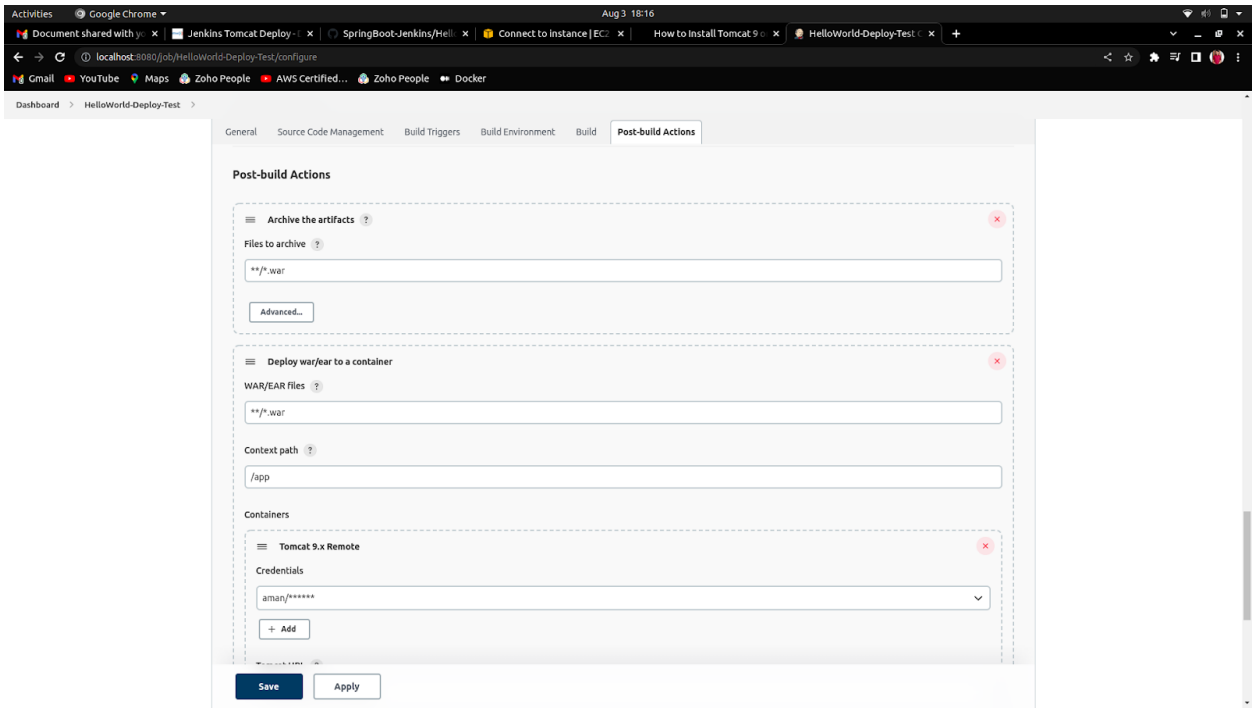
c. HelloWorld-Deploy-Test

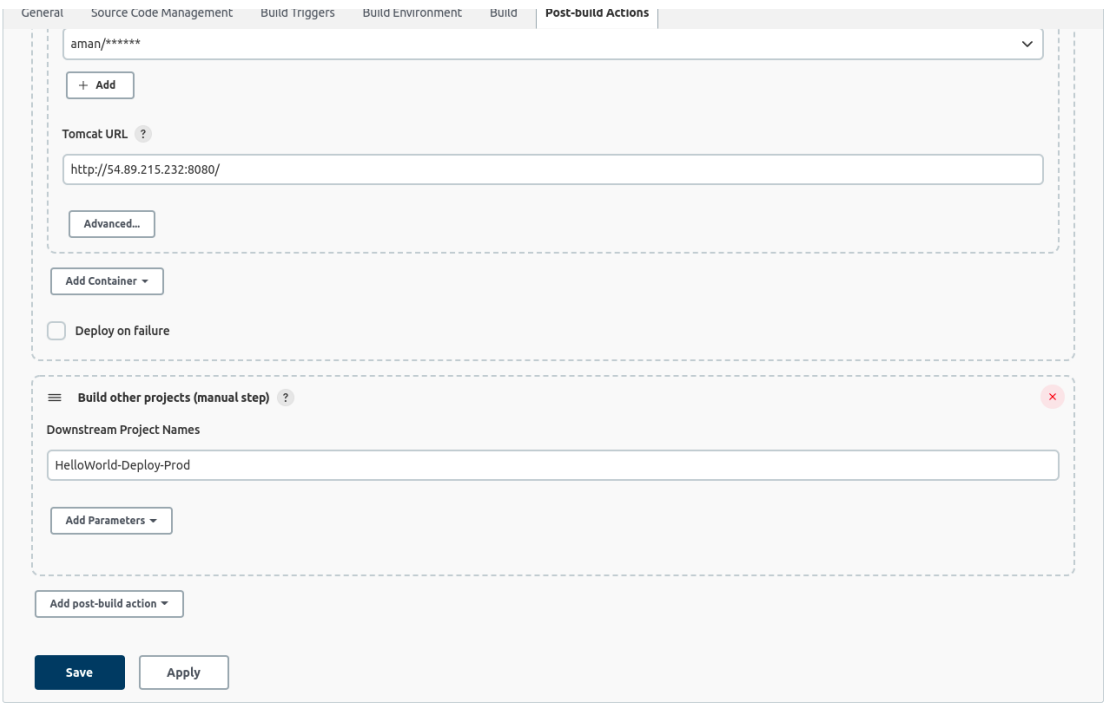




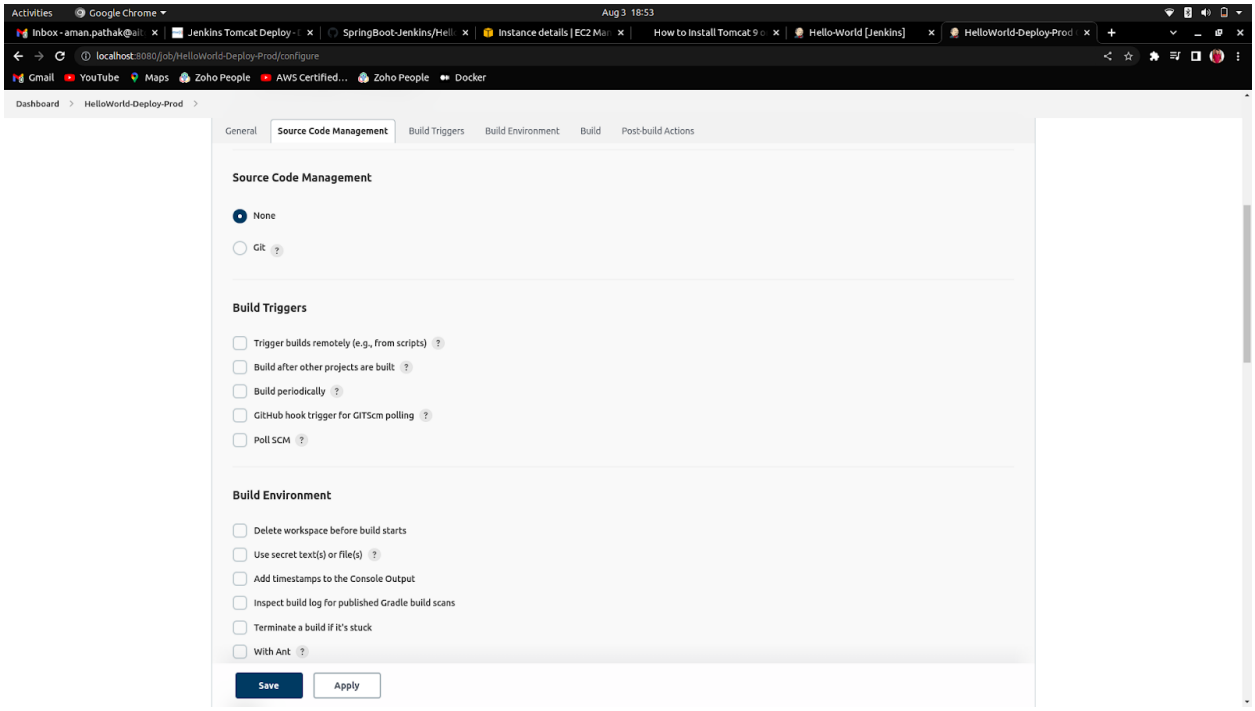

Activities
Google Chrome
Aug 3 18:16
Document shared with yo
Jenkins Tomcat Deploy-
SpringBoot-Jenkins/Hell
Connect to instance | EC2
How to Install Tomcat 9 o
HelloWorld-Deploy-Test
localhost:8080/job/HelloWorld-Deploy-Test/configure
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
HelloWorld-Deploy-Test
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
GitHub hook trigger for GITScm polling
Poll SCM
Schedule
H/2 * * * *
Would last have run at Wednesday, 3 August, 2022 at 6:14:49 PM India Standard Time; would next run at Wednesday, 3 August, 2022 at 6:16:49 PM India Standard Time.
Ignore post-commit hooks
Build Environment
Delete workspace before build starts
Advanced...
Use secret text(s) or file(s)
Add timestamps to the Console Output
Inspect build log for published Gradle build scans
Terminate a build if it's stuck
With Ant
Build
Copy artifacts from another project
Save
Apply

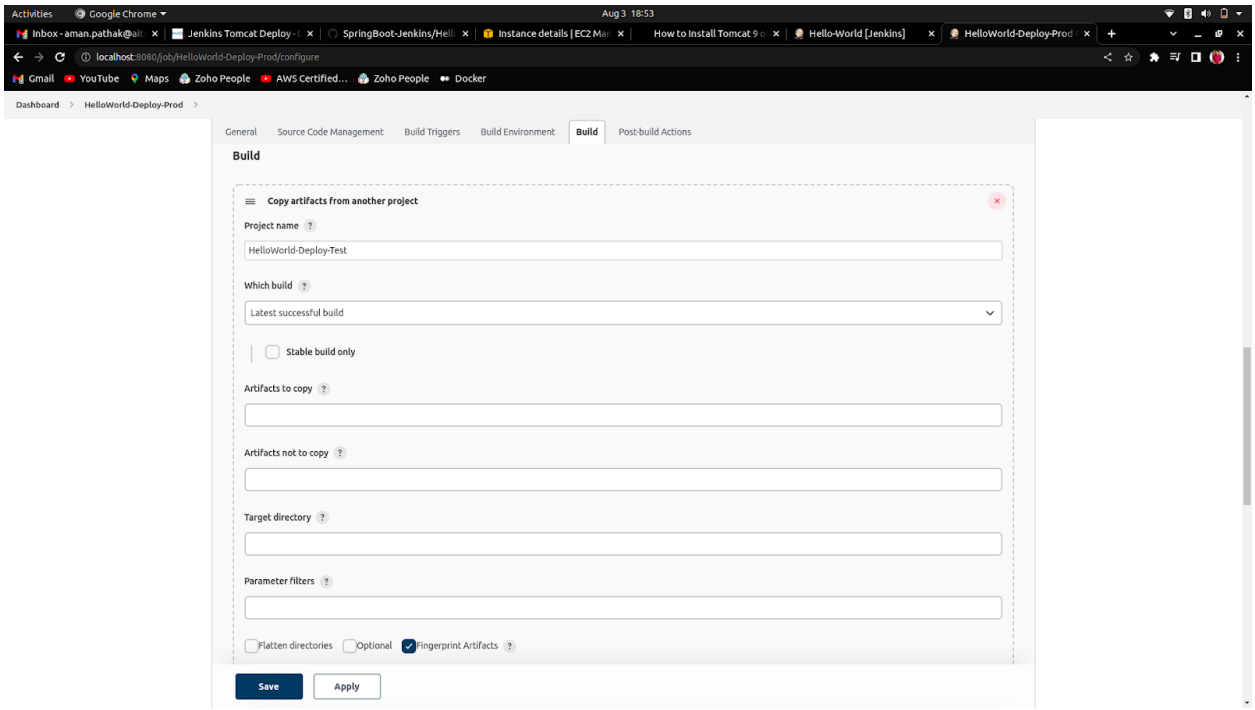

Build
Copy artifacts from another project
Project name
Hello-World-Build
Artifacts will be copied from all modules of this Maven project; click the help icon to learn about selecting a particular module.
Which build
Latest successful build
Stable build only

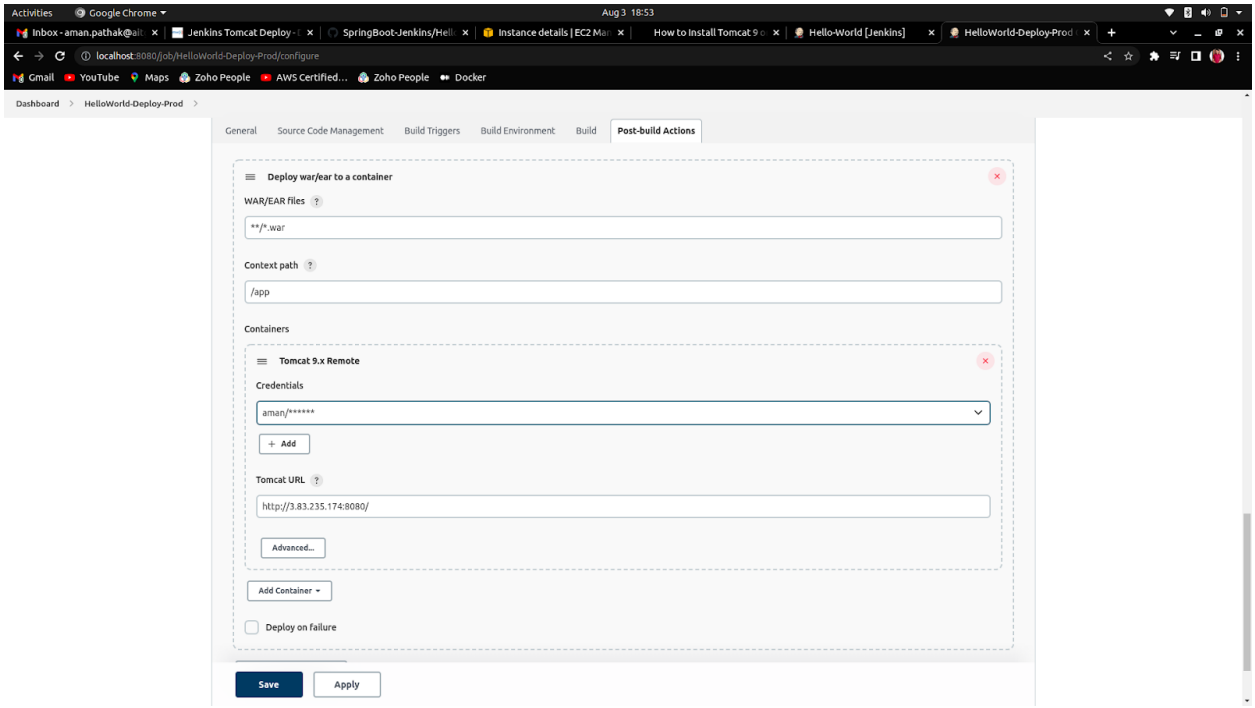

Activities
Google Chrome
Aug 3 18:16
Document shared with
Jenkins Tomcat Deploy
SpringBoot-Jenkins/Hell
Connect to instance | EC2
How to Install Tomcat 9
HelloWorld-Deploy-Test
localhost:8080/job/HelloWorld-Deploy-Test/configure
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
HelloWorld-Deploy-Test
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Post-build Actions
Archive the artifacts
Files to archive
**/*.war
Advanced...
Deploy war/ear to a container
WAR/EAR files
**/*.war
Context path
/app
Containers
Tomcat 9.x Remote
Credentials
aman/******
Add
Save
Apply


General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
aman/******
Add
Tomcat URL
http://54.89.215.232:8080/
Advanced...
Add Container
Deploy on failure
Build other projects (manual step)
Downstream Project Names
HelloWorld-Deploy-Prod
Add Parameters
Add post-build action
Save
Apply

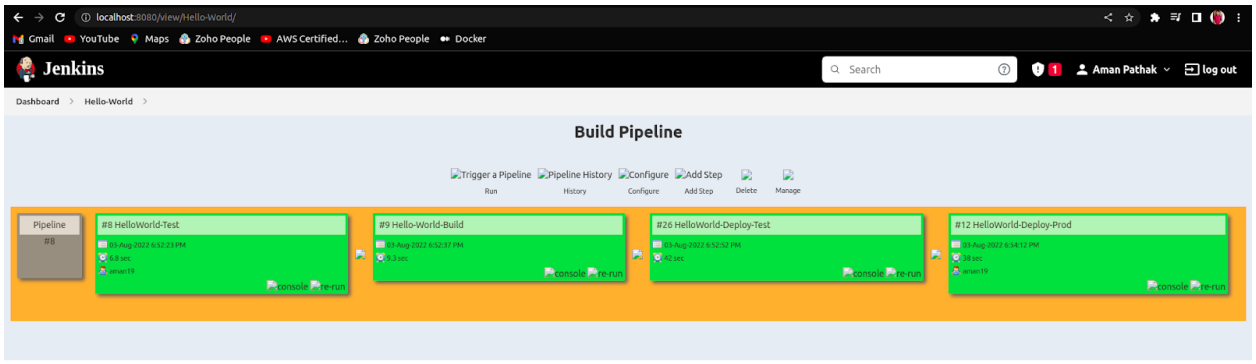
d. HelloWorld-Deploy-Prod

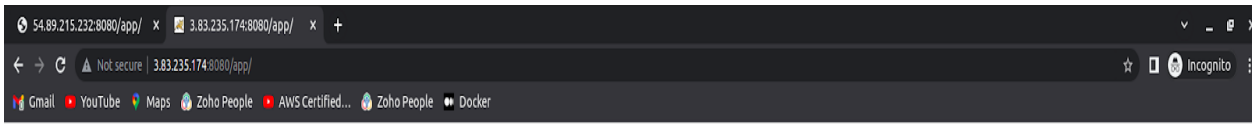





Activities
Google Chrome
Aug 3 18:53
Inbox - aman.pathak@alt
Jenkins Tomcat Deploy-
SpringBoot-Jenkins/Hell
Instance details | EC2 Man
How to Install Tomcat 9 o
Hello-World [Jenkins]
HelloWorld-Deploy-Prod
localhost:8080/job/HelloWorld-Deploy-Prod/configure
Gmail
YouTube
Maps
Zoho People
AWS Certified...
Zoho People
Docker
Dashboard
HelloWorld-Deploy-Prod
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Deploy war/ear to a container
WAR/EAR files
**/*.war
Context path
/app
Containers
Tomcat 9.x Remote
Credentials
aman/******
Add
Tomcat URL
http://3.83.235.174:8080/
Advanced...
Add Container
Deploy on failure
Save
Apply

The Output



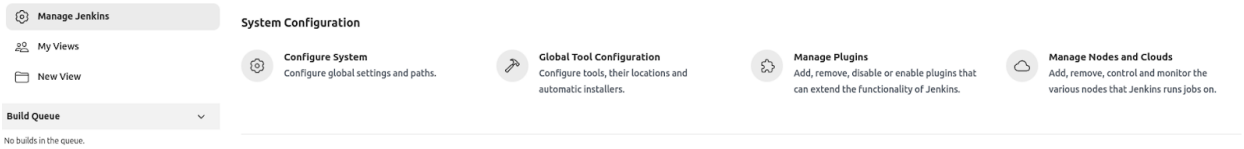


This is All About Spring Boot! This is The Second Version of The Spring Boot

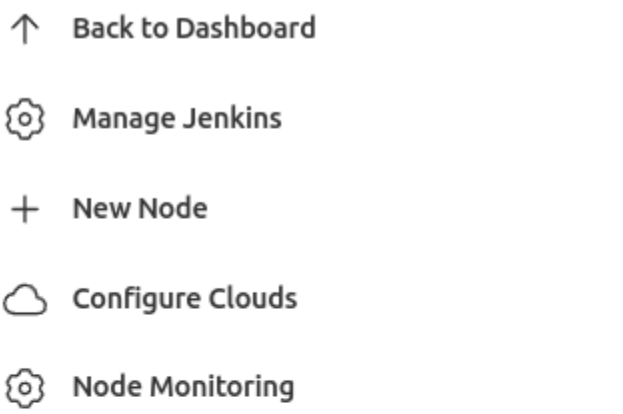
This is All About Spring Boot! This is The Second Version of The Spring Boot

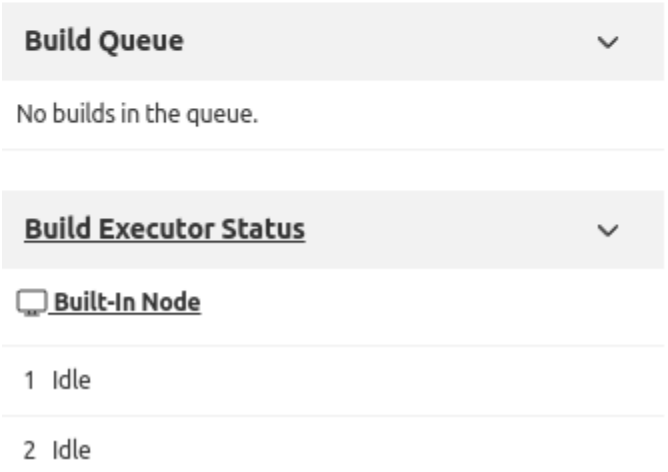
Creating Jenkins Slave

1. First of all, go into Manage Jenkins -> Manage Nodes and Clouds.

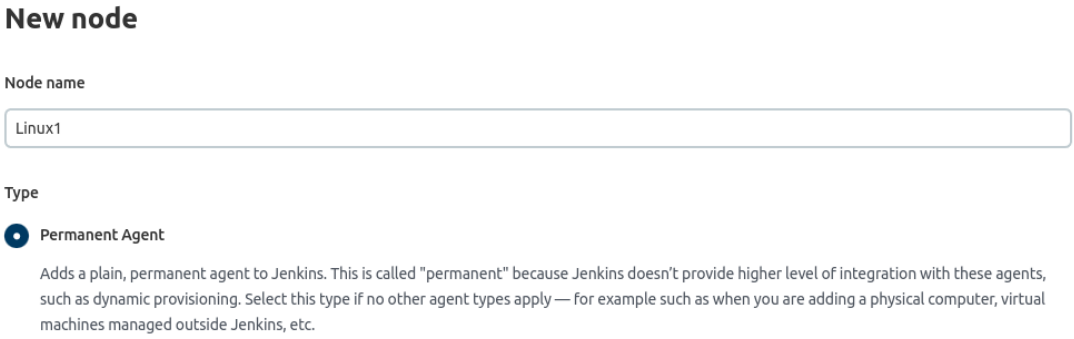


2. To create the new Slave, Click on the new node.





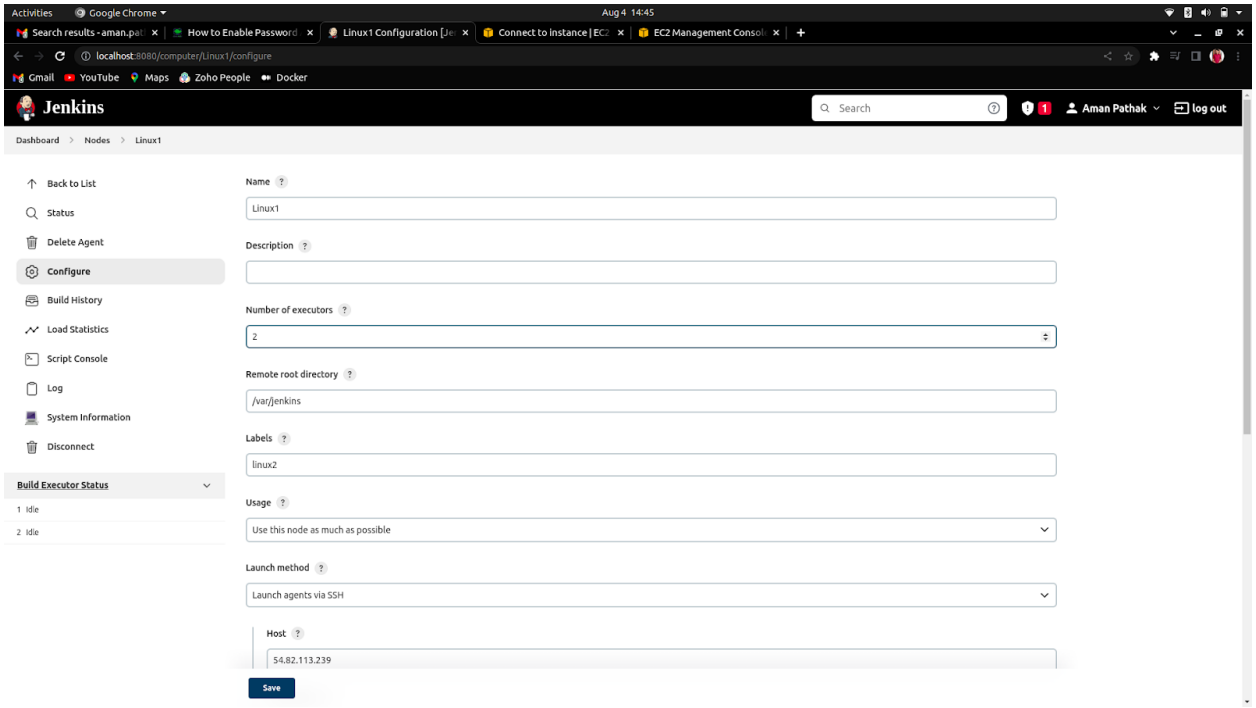
3. Give the name to your slave and check the radio button of Permanent Agent then, click on the Create button.

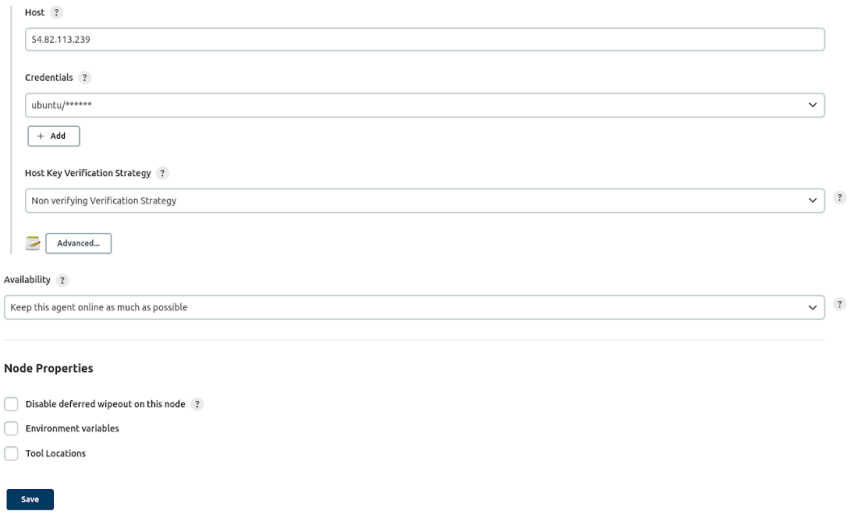


4. Before going to the subsequent Configurations of your Jenkins slave,
   Follow some steps:
   a. Create an EC2 Instance with Inbound rule 22(SSH) at least.
   b. SSH into your instance.
   c. Install OpenJDK by the command:
      sudo apt install openjdk-8-jre-headless -y
   d. Create a directory named jenkins inside the /var directory.
   e. Change the owner of that directory by the given command:
      sudo chown ubuntu:ubuntu jenkins/
   f. Now, change the password of your EC2 Instance by the following commands:
      sudo passwd ubuntu
      sudo vim sshd_config
      sudo systemctl restart jenkins

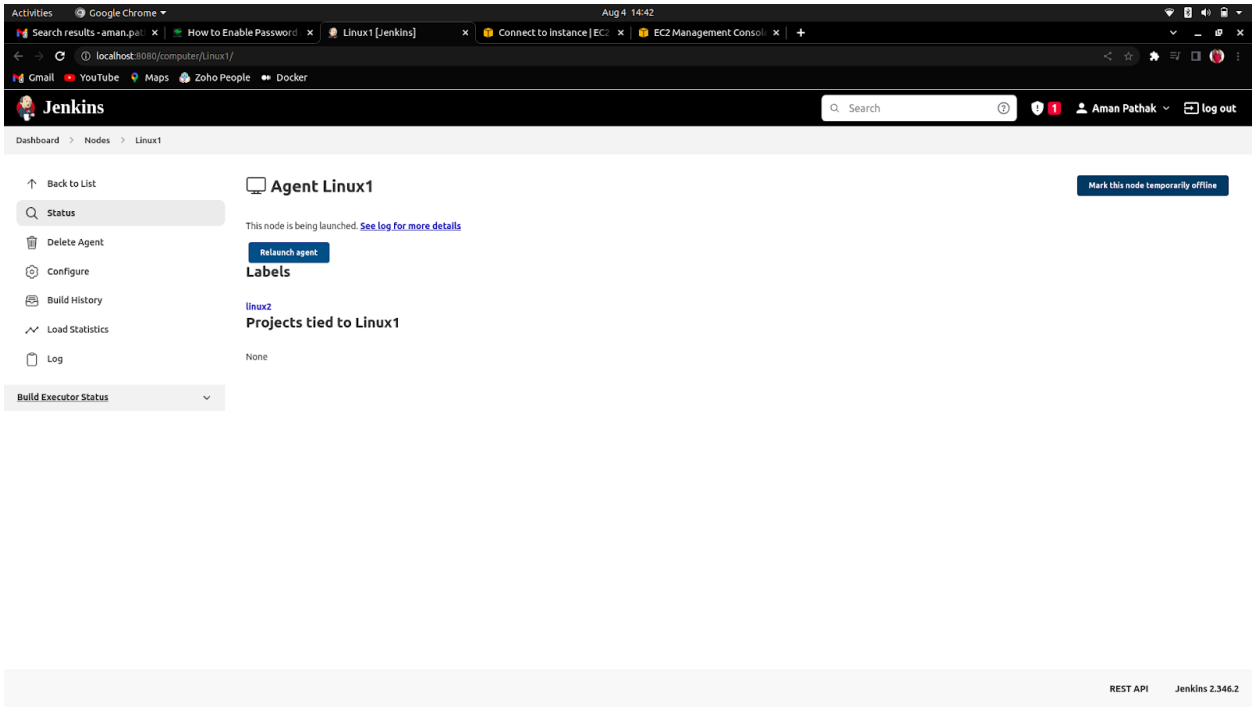
      For more reference, follow the link below:
      Change EC2 Instance Password

5. Enter the number of executors as per your choice, then enter the Remote root directory which will be /var/jenkins then, the launch type must be “Launch agents via SSH” Enter the Host Name which will be the Public IP of your instance, then add the credentials of your EC2 Instance (username= ubuntu and password= aman@123) and
   In the last Host, the Key verification Strategy must be “Non-verifying Verification Strategy”.

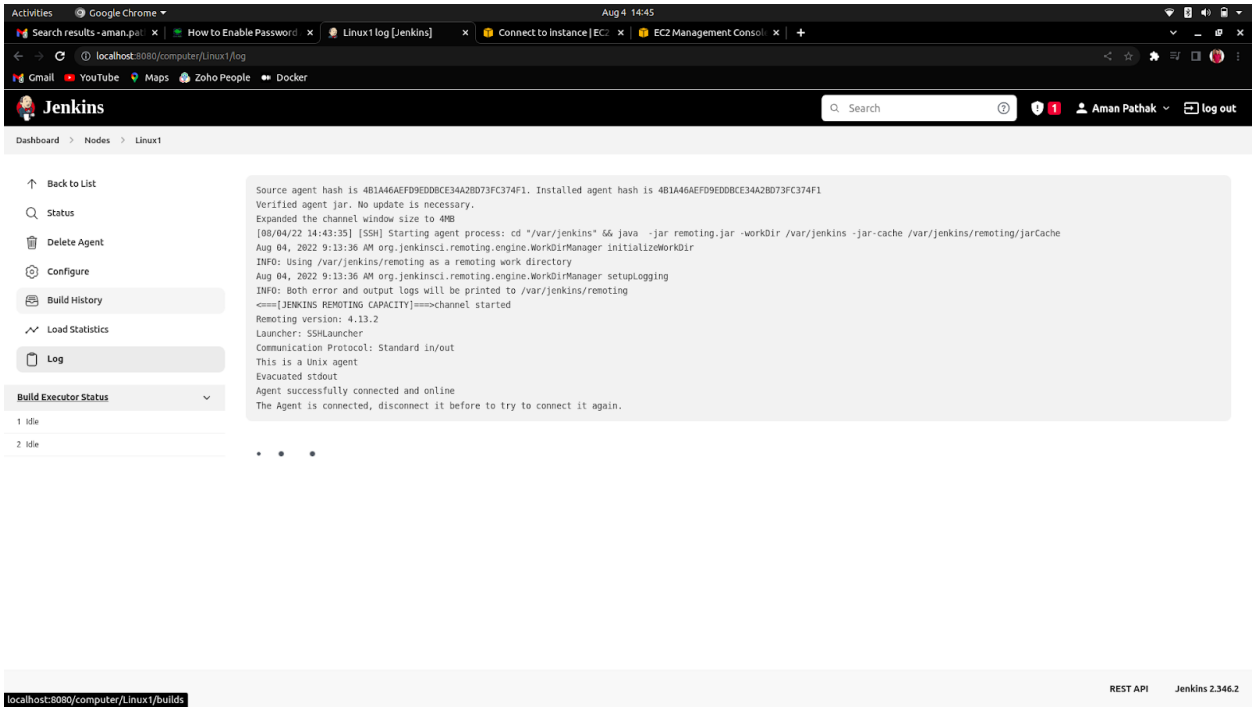



Host
54.82.113.239
Credentials
ubuntu/******
Add
Host Key Verification Strategy
Non verifying Verification Strategy
Advanced...
Availability
Keep this agent online as much as possible
Node Properties
Disable deferred wipeout on this node
Environment variables
Tool Locations
Save

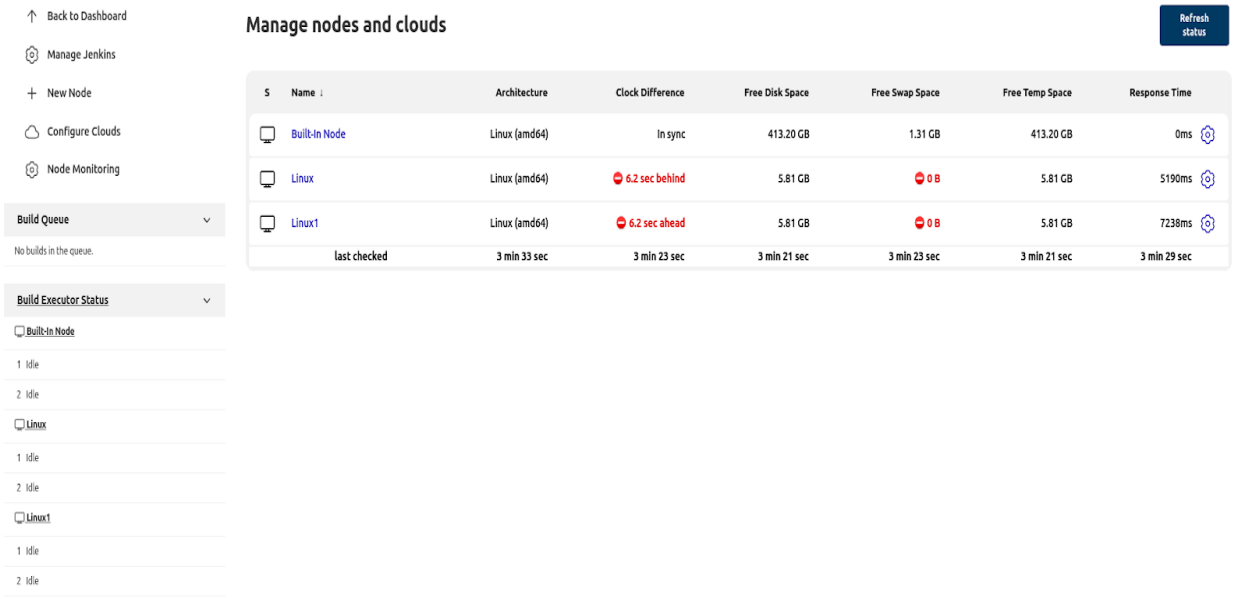
6. Now go to that Slave and click on Relaunch agent.



7. As the Slave is connected.

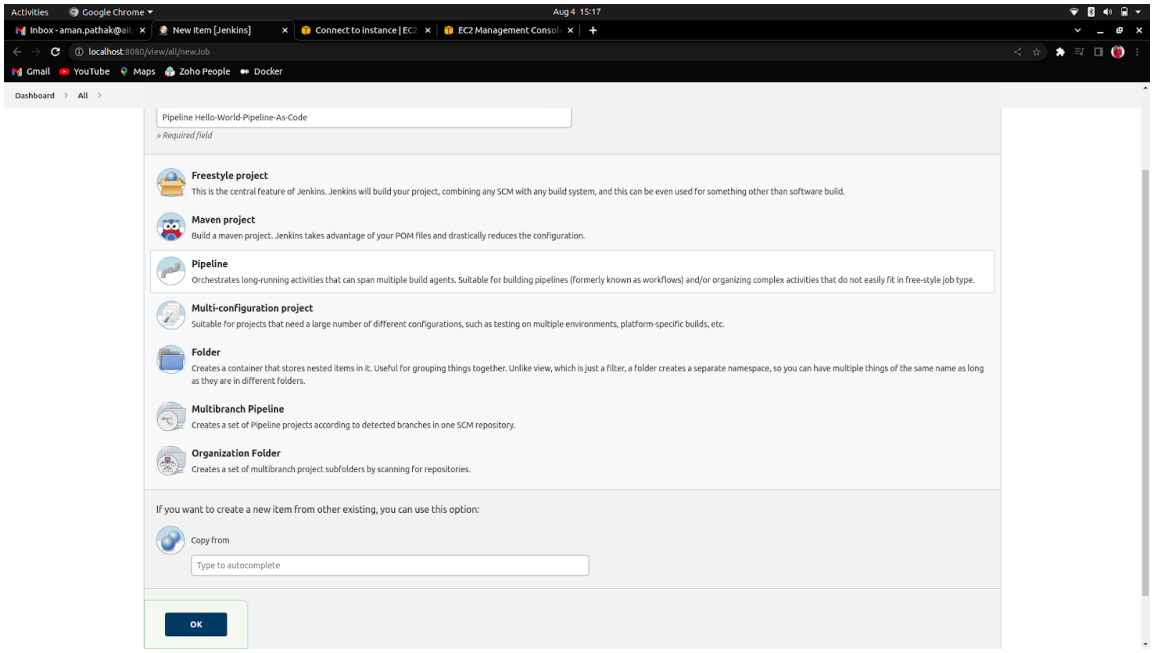


8. You can check on the left side named Linux1.

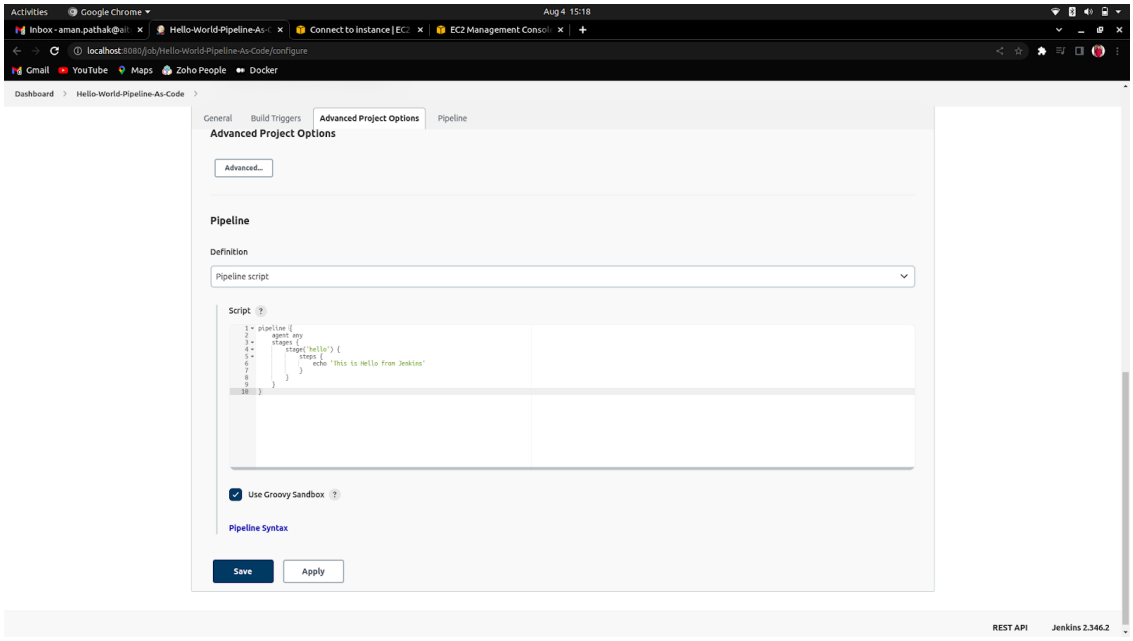


Create Jenkins Pipeline as a Code

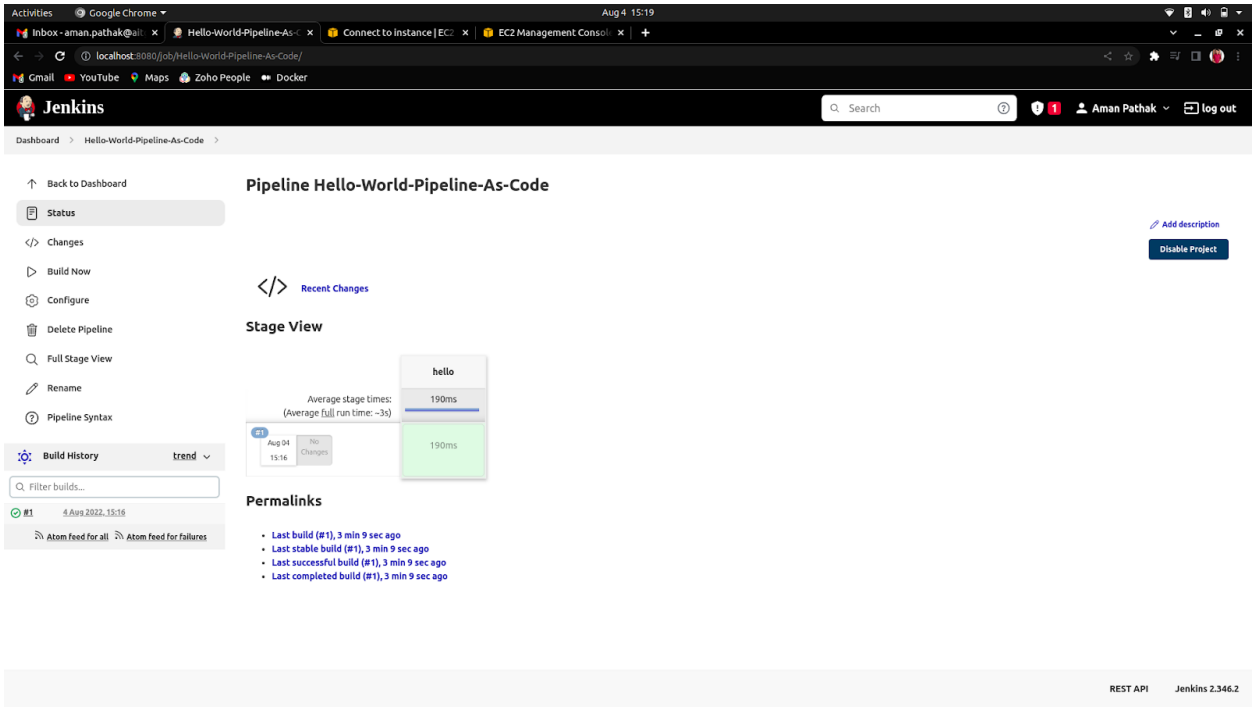
1. Create a Job that will be a Pipeline.



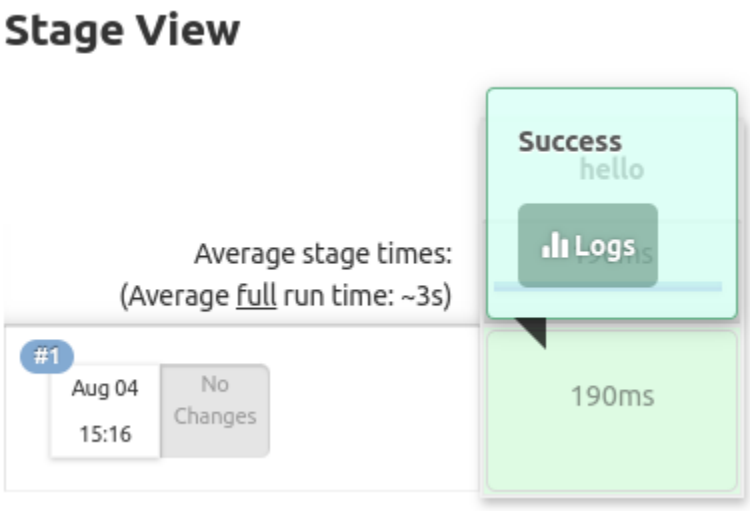
2. Write the Declarative Pipeline to run the job of Hello World.



3. Now, when you build the job you'll see look like this:

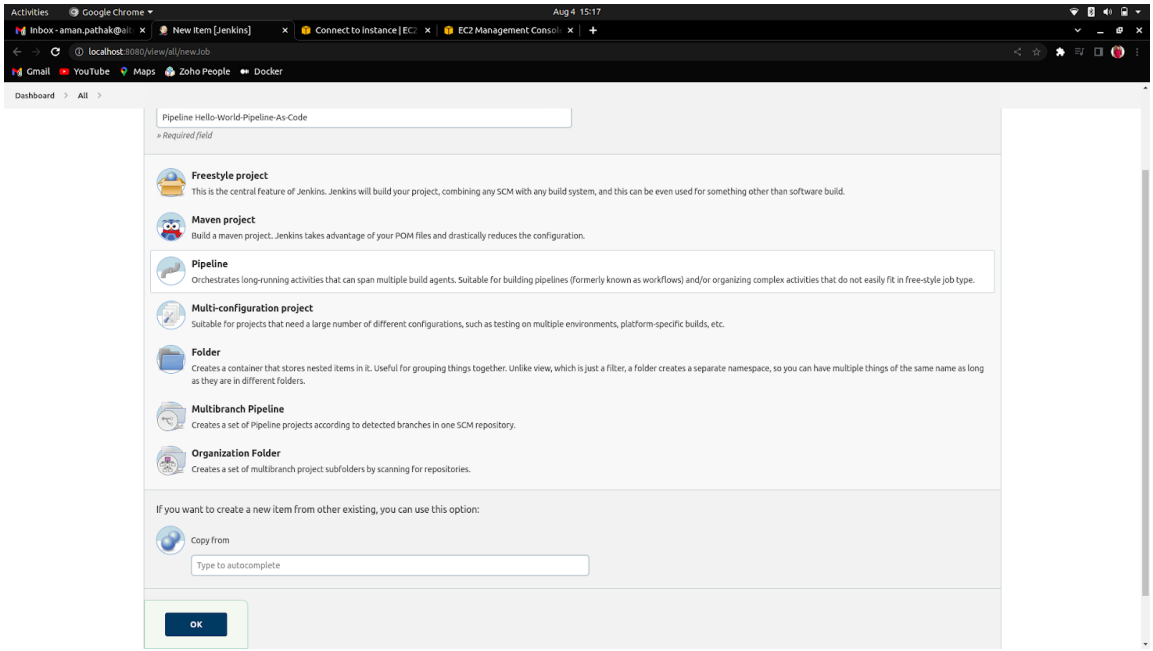


4. To see the output hover on the green box and click on the logs.

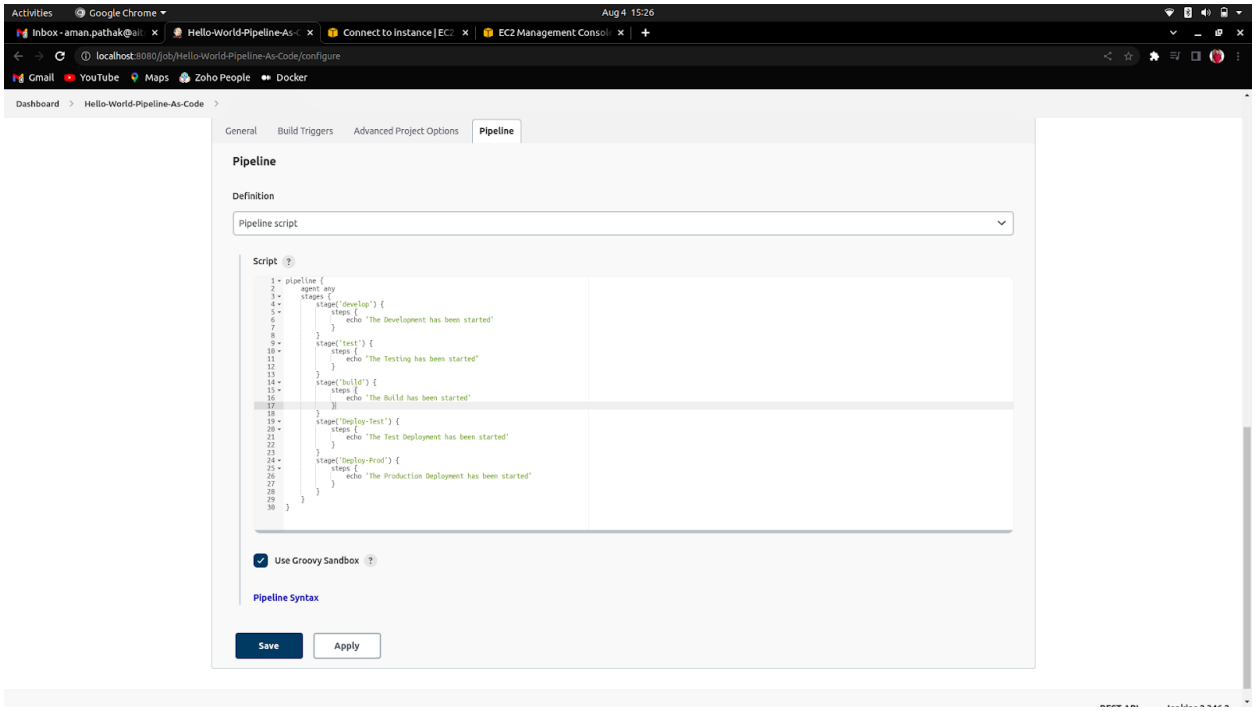


Multi-Staging Pipeline as a Code

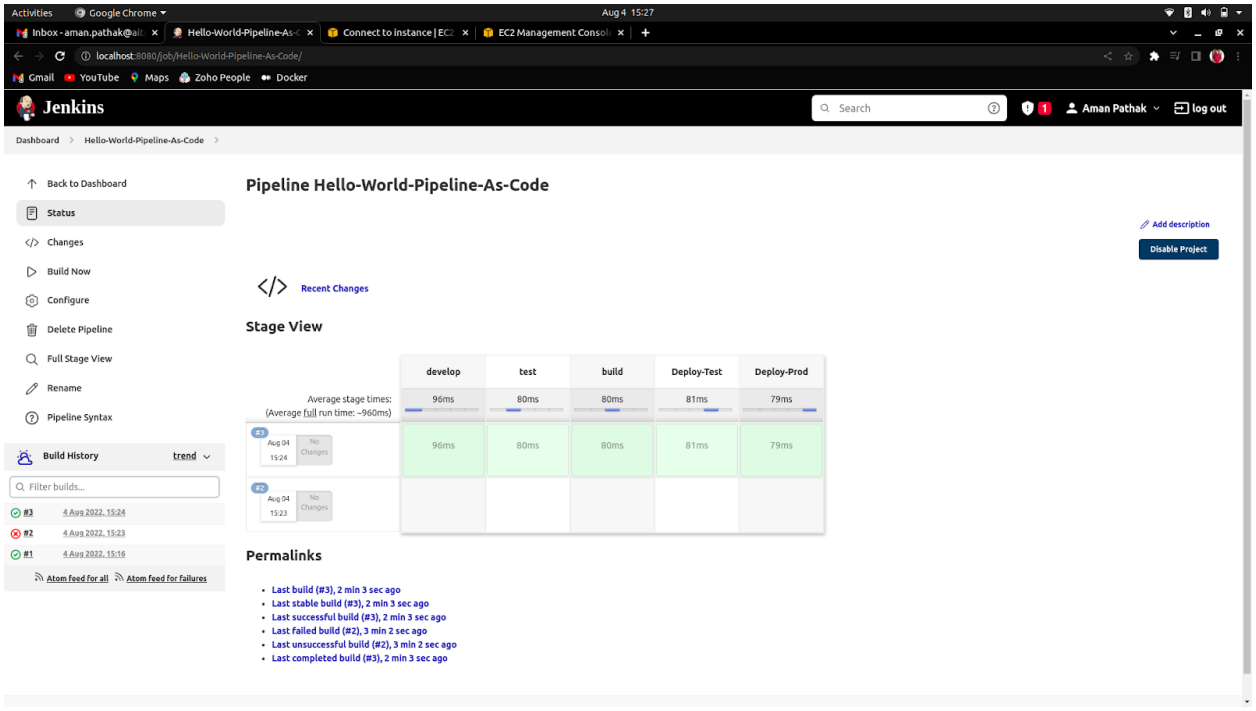
1. Create a Job that will be a Pipeline.



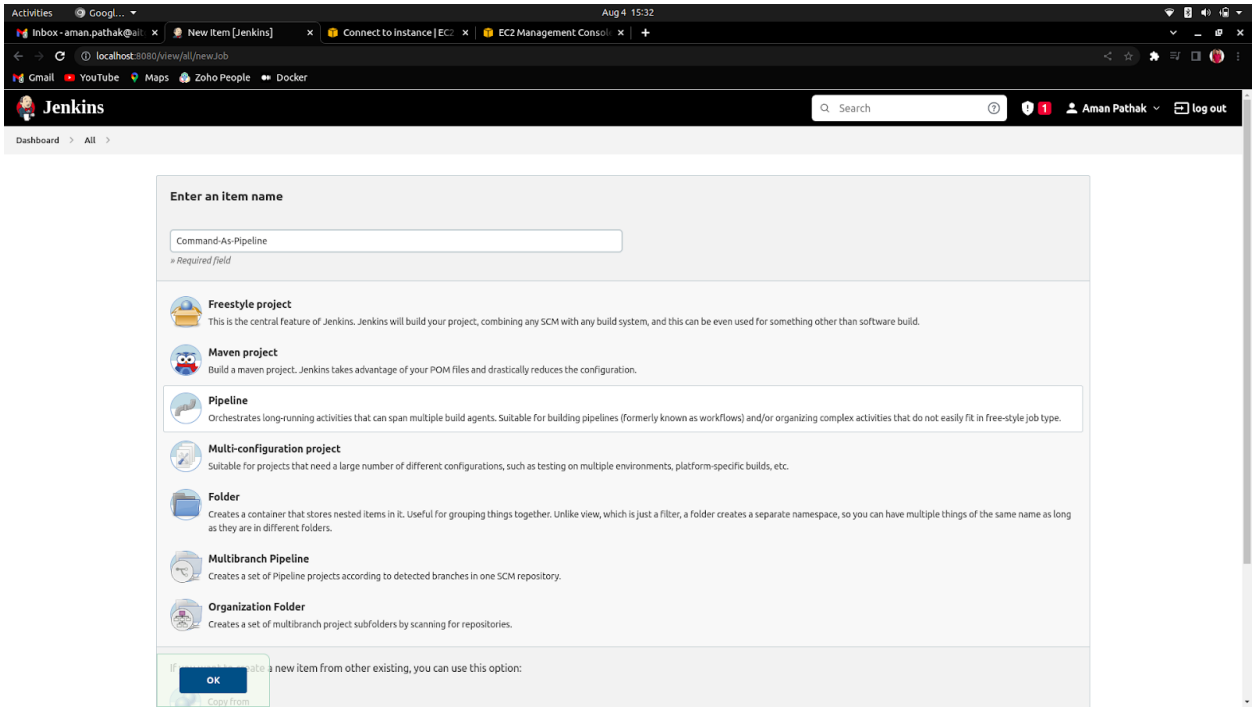
2. Write the Declarative Pipeline.

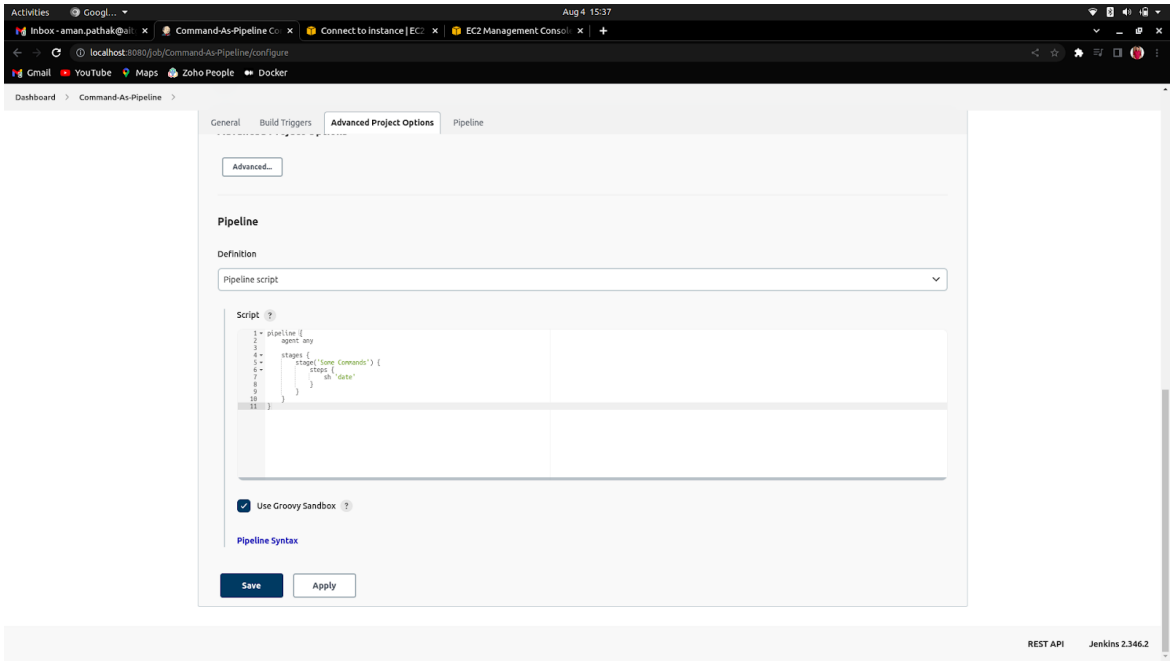


3. The Output:

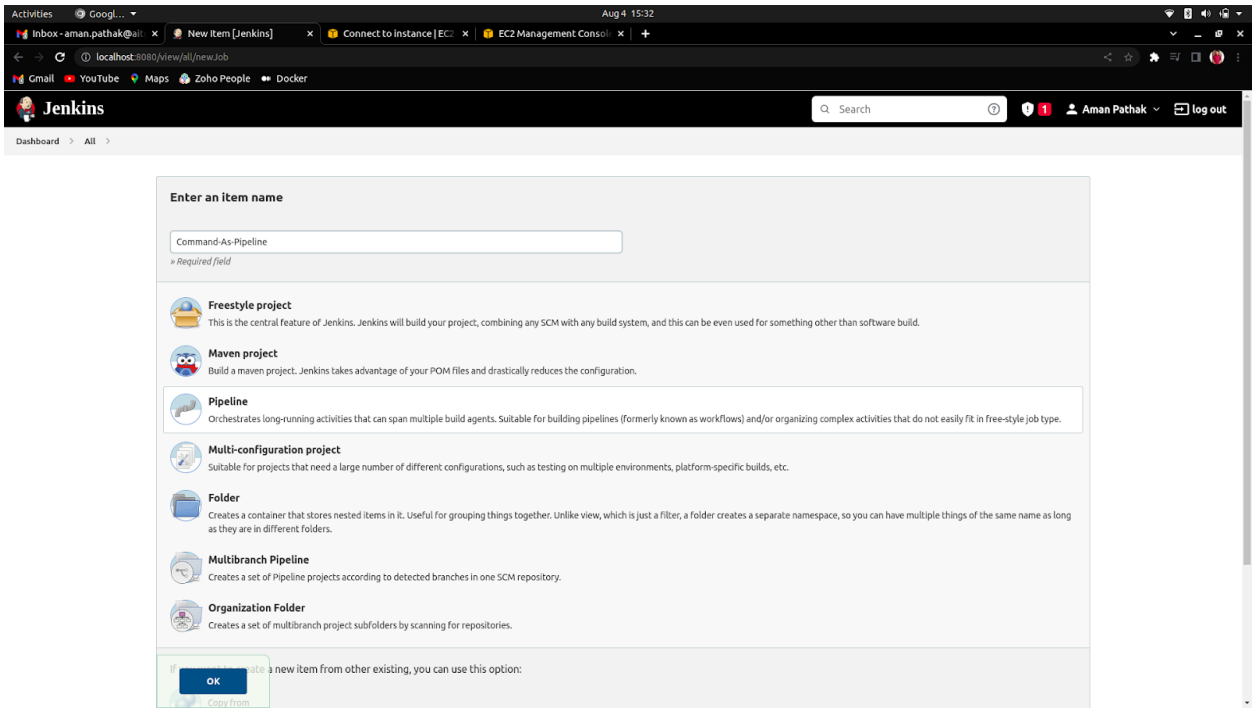

Activities
Google Chrome
Aug 4 15:27
Inbox - aman.pathak@ait
Hello-World-Pipeline-As-C
Connect to instance | EC2
EC2 Management Consol
localhost:8080/job/Hello-World-Pipeline-As-Code/
Gmail
YouTube
Maps
Zoho People
Docker
Jenkins
Search
Aman Pathak
log out
Dashboard
Hello-World-Pipeline-As-Code
Back to Dashboard
Status
Changes
Build Now
Configure
Delete Pipeline
Full Stage View
Rename
Pipeline Syntax
Build History
trend
Filter builds...
#3 4 Aug 2022, 15:24
#2 4 Aug 2022, 15:23
#1 4 Aug 2022, 15:16
Atom feed for all
Atom feed for failures
Pipeline Hello-World-Pipeline-As-Code
Add description
Disable Project
Recent Changes
Stage View
develop
test
build
Deploy-Test
Deploy-Prod
Average stage times:
(Average full run time: ~960ms)
96ms
80ms
80ms
81ms
79ms
#3
Aug 04
15:24
No Changes
96ms
80ms
80ms
81ms
79ms
#2
Aug 04
15:23
No Changes
Permalinks
Last build (#3), 2 min 3 sec ago
Last stable build (#3), 2 min 3 sec ago
Last successful build (#3), 2 min 3 sec ago
Last failed build (#2), 3 min 2 sec ago
Last unsuccessful build (#2), 3 min 2 sec ago
Last completed build (#3), 2 min 3 sec ago

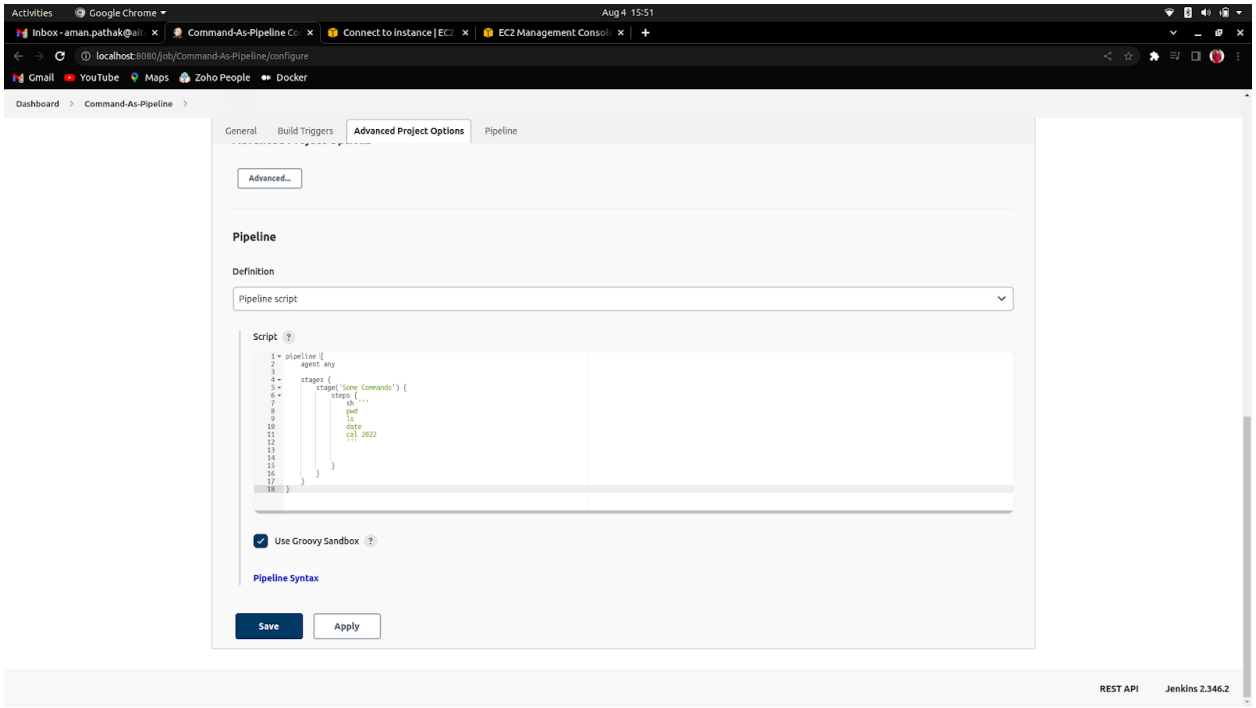
How to run Commands in Pipeline as a Code



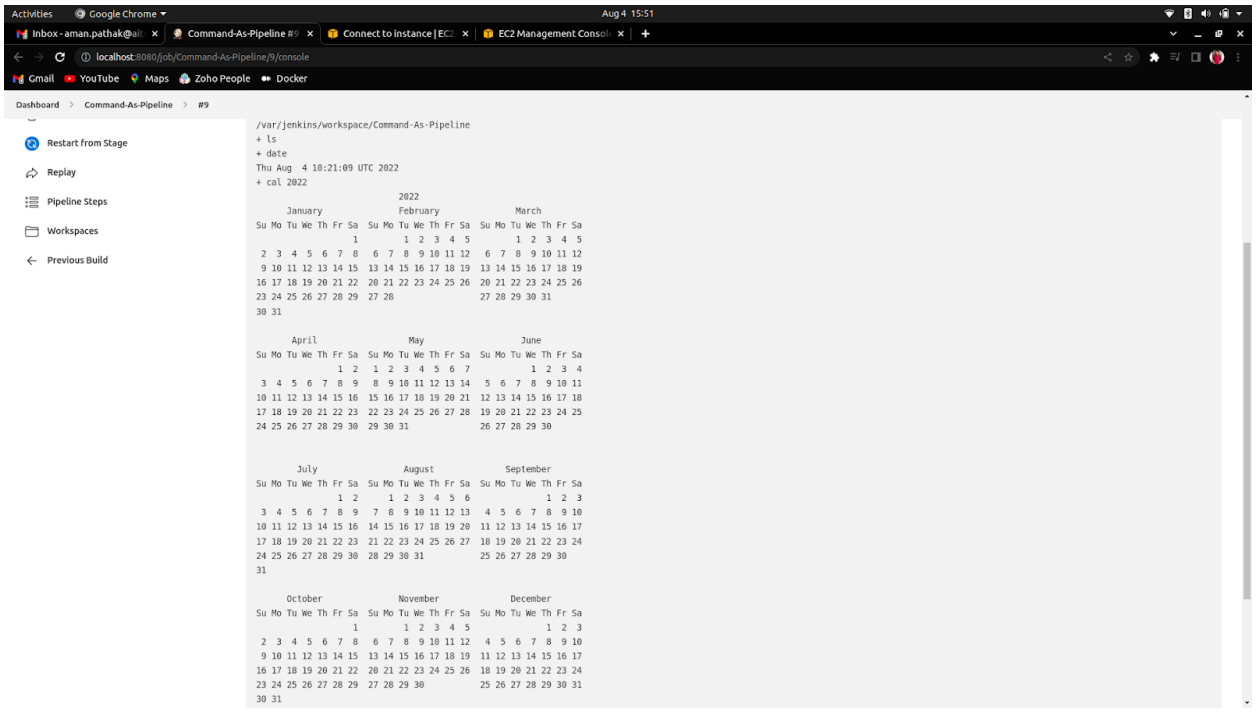


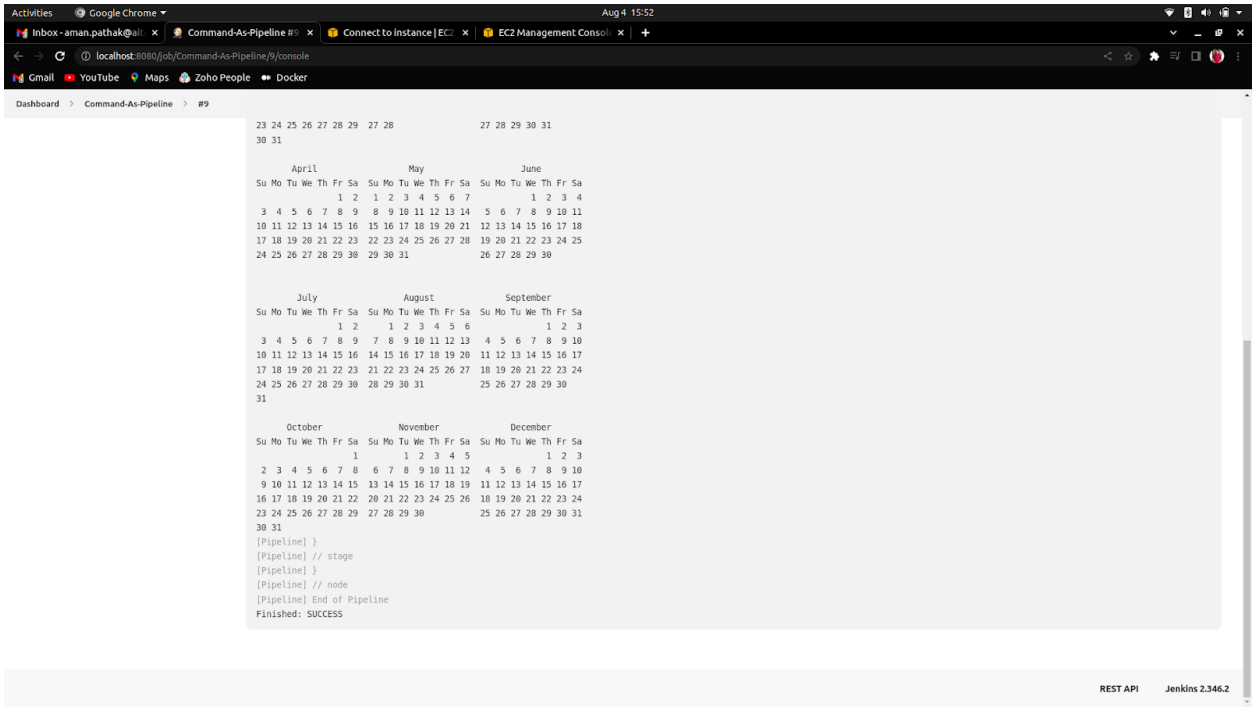
How to run multiple commands in Pipeline as a code





The Output:


Activities
Google Chrome
Aug 4 15:51
Inbox - aman.pathak@alt
Command-As-Pipeline #9
Connect to instance | EC2
EC2 Management Console
localhost:8080/job/Command-As-Pipeline/9/console
Gmail
YouTube
Maps
Zoho People
Docker
Dashboard
Command-As-Pipeline
#9
Restart from Stage
Replay
Pipeline Steps
Workspaces
Previous Build
/var/jenkins/workspace/Command-As-Pipeline
+ ls
+ date
Thu Aug 4 10:21:09 UTC 2022
+ cal 2022
2022
January
February
March
April
May
June
July
August
September
October
November
December


Activities
Google Chrome
Aug 4 15:52
Inbox - aman.pathak@alt
Command-As-Pipeline #9
Connect to instance | EC2
EC2 Management Console
localhost:8080/job/Command-As-Pipeline/9/console
Gmail
YouTube
Maps
Zoho People
Docker
Dashboard
Command-As-Pipeline
#9
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
REST API
Jenkins 2.346.2

How to set the Environment variable Globally in the Code Pipeline

Pipeline

Definition

Pipeline script

Script

```
pipeline {
    agent any
    environment {
        name = "jenkins_user"
    }
    stages {
        stage('Global Variable') {
            steps {
                sh 'echo $name'
            }
        }
    }
}
```

Use Groovy Sandbox

How to set the Environment variable Locally in Code Pipeline

Definition

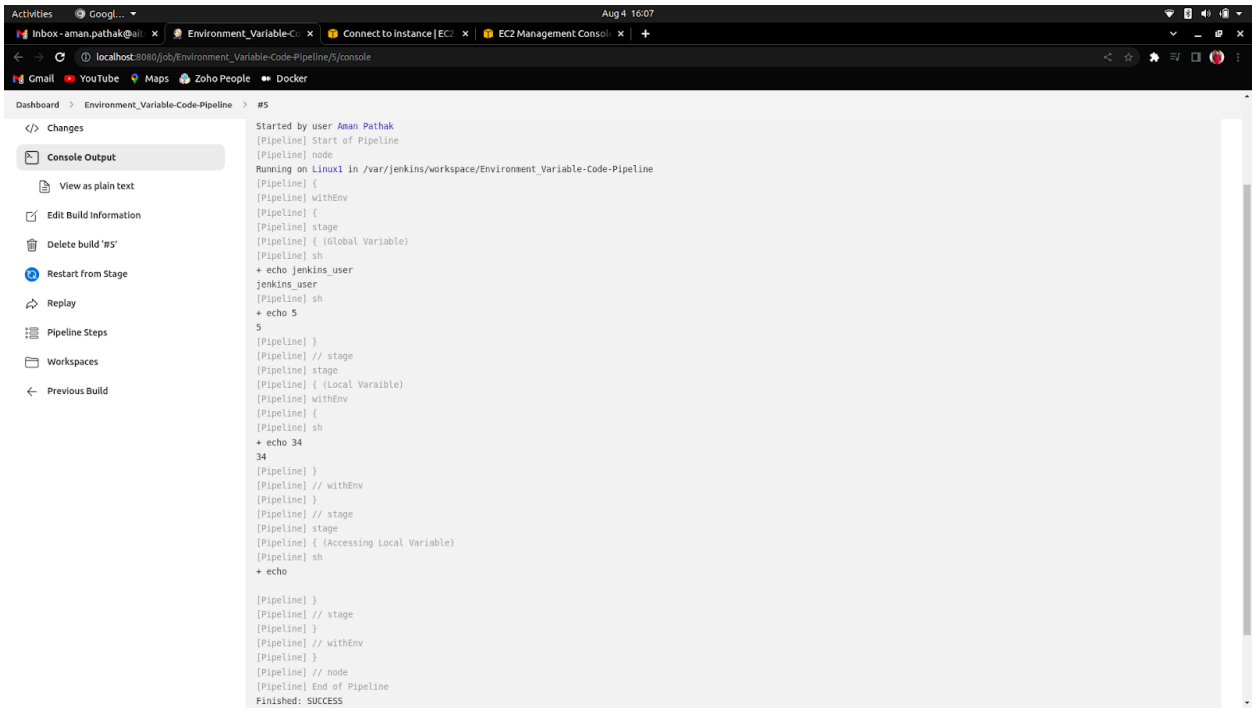
Pipeline script

Script

```
pipeline {
    agent any
    environment {
        name = "jenkins_user"
    }
    stages {
        stage('Global Variable') {
            steps {
                sh 'echo $name'
                sh 'echo $BUILD_ID'
            }
        }
        stage('Local Varaible') {
            environment {
                user_id = '34'
            }
            steps {
                sh 'echo $user_id'
            }
        }
        stage('Accessing Local Variable') {
            steps {
                sh 'echo $user_id'
            }
        }
    }
}
```

Use Groovy Sandbox

The Output



How to take input from the users in the Code Pipeline

Pipeline

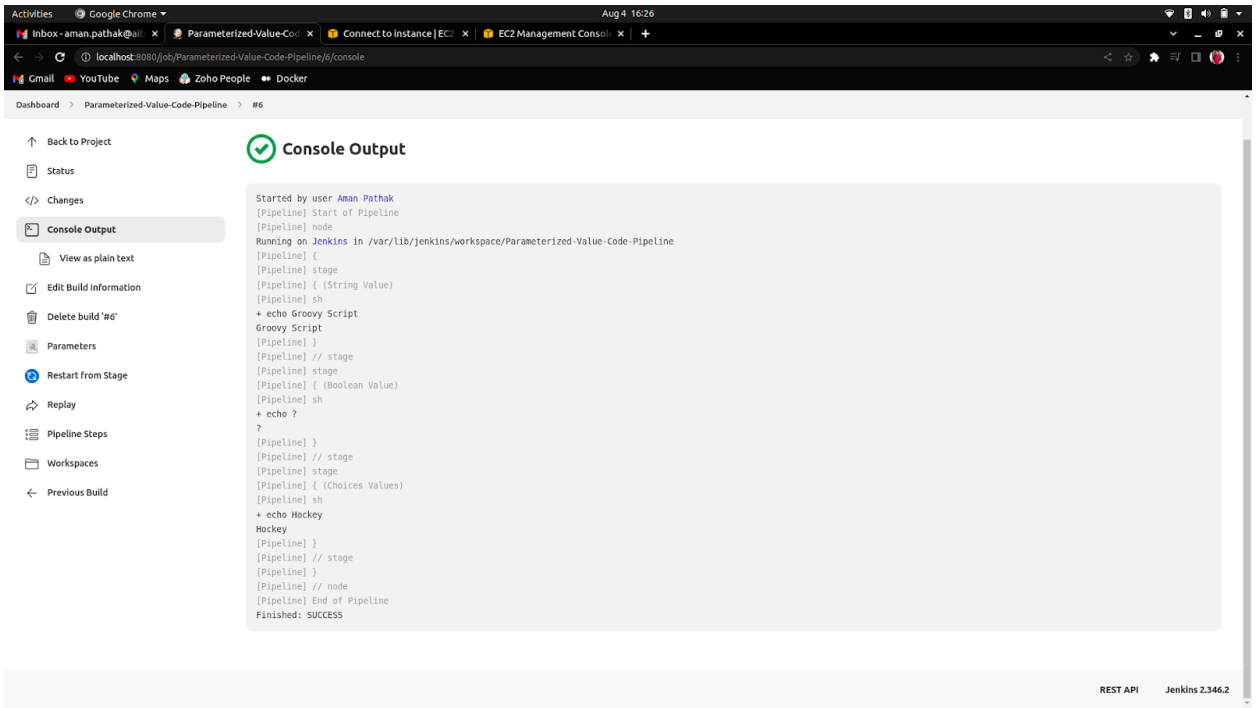
Definition

Pipeline script

Script ?

```
pipeline {
    agent any
    parameters {
        string(name: 'username', defaultValue: 'Jenkins', description: 'Who is the Current User')
        booleanParam(name: 'Graduated?', defaultValue: 'true', description: 'Are you Graduated')
        choice(name: 'hobby', choices: ['Cricket','Chess','Football','Hockey'], description: 'Favourite Sport')
    }
    stages {
        stage('String Value') {
            steps {
                sh 'echo $username'
            }
        }
        stage('Boolean Value') {
            steps {
                sh 'echo $Graduated?'
            }
        }
        stage('Choices Values') {
            steps {
                sh 'echo $hobby'
            }
        }
    }
}
```

The Output:



Pipeline Script of the Last Job

```
pipeline {
    agent any
    parameters {
        string(name: 'username', defaultValue: 'Jenkins', description: 'Who is the Current User')
        booleanParam(name: 'Graduated?', defaultValue: 'true', description: 'Are you Graduated')
        choice(name: 'hobby', choices: ['Cricket','Chess','Football','Hockey'], description: 'Favourite Sport')
    }
    stages {
        stage('String Value') {
            input {
                message "Do we start?"
                ok "Yes, Let's do this"
            }
            steps {
                sh 'echo $username'
            }
        }
        stage('Boolean Value') {
            steps {
                sh 'echo $Graduated?'
            }
        }
        stage('Choices Values') {
            input {
                message "Want to Continue?"
                ok "Yeah! Definitely"
            }
            steps {
                sh 'echo $hobby'
            }
        }
    }
    post {
        always {
            echo 'The Jenkins!!!!'
        }
        success {
            echo 'The Final Acheivement is Here!!!'
        }
        failure {
            echo 'Seriouslly! We got the Failure from the other End!!!!'
        }
    }
}
```

In **conclusion**, the above notes provide a beginner-level introduction to Jenkins and its role in enabling Continuous Integration and Continuous Delivery (CI/CD) practices. By understanding the fundamental concepts and features of Jenkins, beginners can kickstart their journey towards automating their software development processes and achieving faster, more reliable project deliveries.

For those seeking to advance their knowledge and skills in Jenkins, there is a world of possibilities to explore. Jenkins offers a vast array of plugins and integrations with various tools, allowing advanced users to tailor their CI/CD pipelines to meet specific project requirements. By delving deeper into Jenkins' capabilities, experienced users can optimize their workflows, integrate advanced testing and deployment strategies, and harness the full potential of CI/CD for continuous improvement.

Moreover, Jenkins Pipelines offers an exciting avenue for advanced users to adopt a "pipeline-as-code" approach, enabling better version control, collaboration, and scalability. With declarative pipeline scripts, developers can encapsulate complex build and deployment logic, making it easier to manage and maintain their CI/CD workflows effectively.

While the notes provide a solid foundation for beginners, it is essential to continue exploring Jenkins and other related technologies to stay up-to-date with the latest advancements in CI/CD practices. Online tutorials, documentation, forums, and community resources are valuable sources to deepen one's expertise and stay ahead in the rapidly evolving landscape of DevOps and CI/CD.

Whether at the beginner or advanced level, embracing Jenkins as a pivotal tool in the software development lifecycle empowers teams to automate, collaborate, and deliver high-quality software efficiently. By leveraging Jenkins' capabilities and staying curious, developers can continually refine their CI/CD processes and drive innovation in the dynamic world of modern software development.

Did you find it Useful?

Leave a comment!

ABD UR REHMAN NAWAZ

@ABD-UR-REHMAN-NAWAZ

REHMANWARRAICH8282@GMAIL.OCM

FOLLOW FOR MORE

Like Comment Repost