

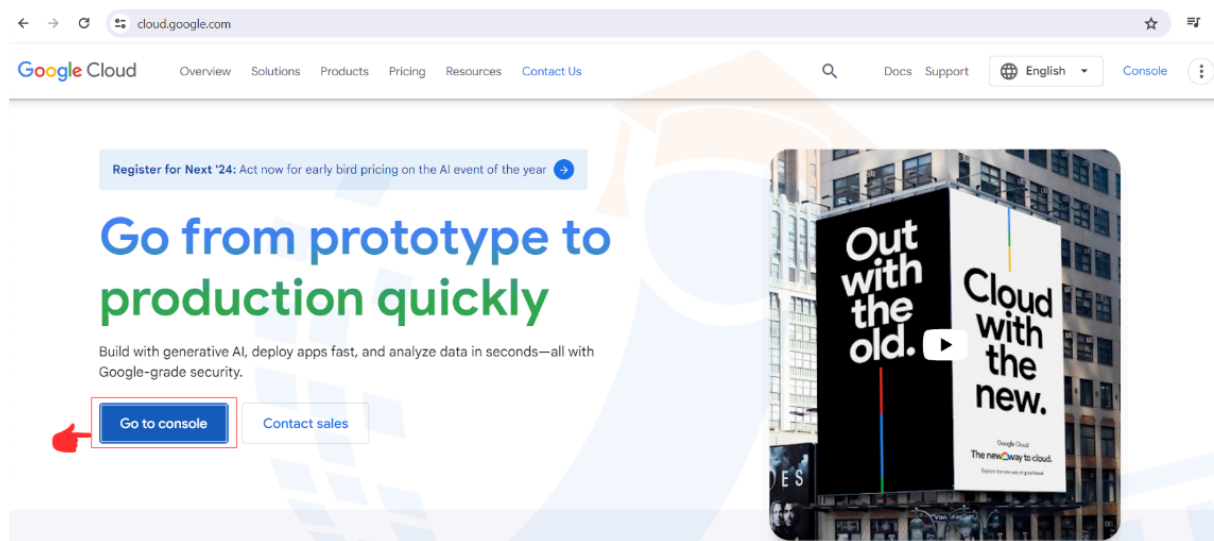
Deploying and Configuring Jenkins Server

(Usually these activities are performed by DevOps Engineers)

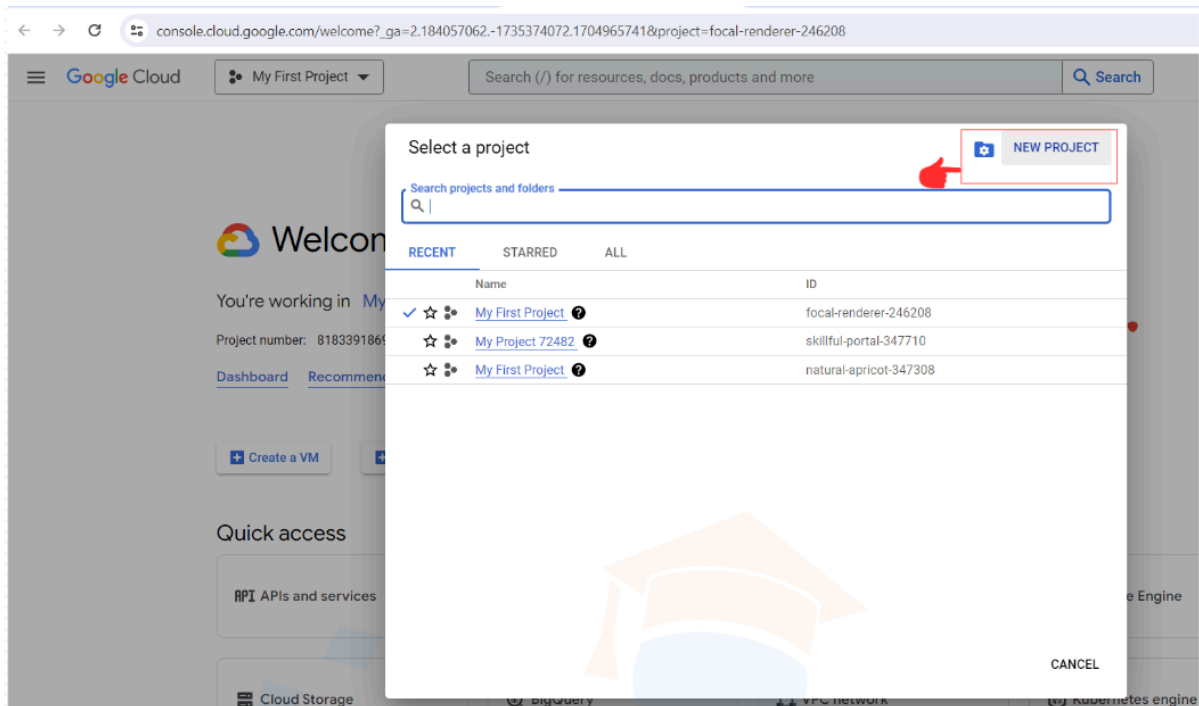
Go to cloud.google.com

Create a GCP account (provides 300\$ free credits that would be valid for 3 months)

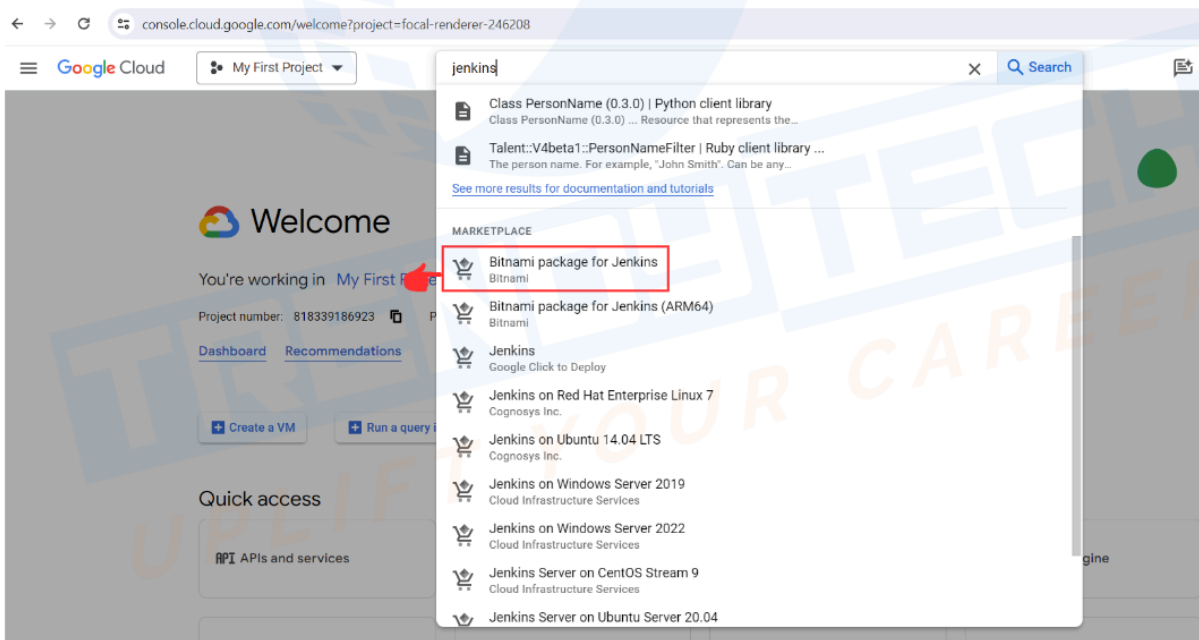
Once you create an account -> Go to Console

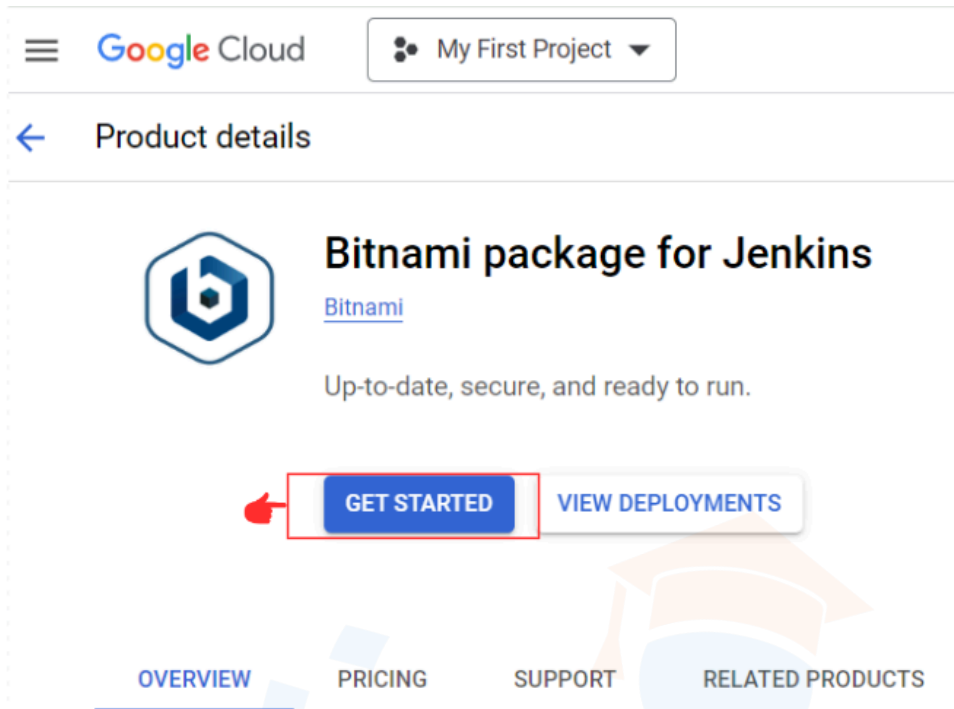


Create a new Project and Deploy Resources (Like - Jenkins)



Deploy the necessary resources - Like Jenkins

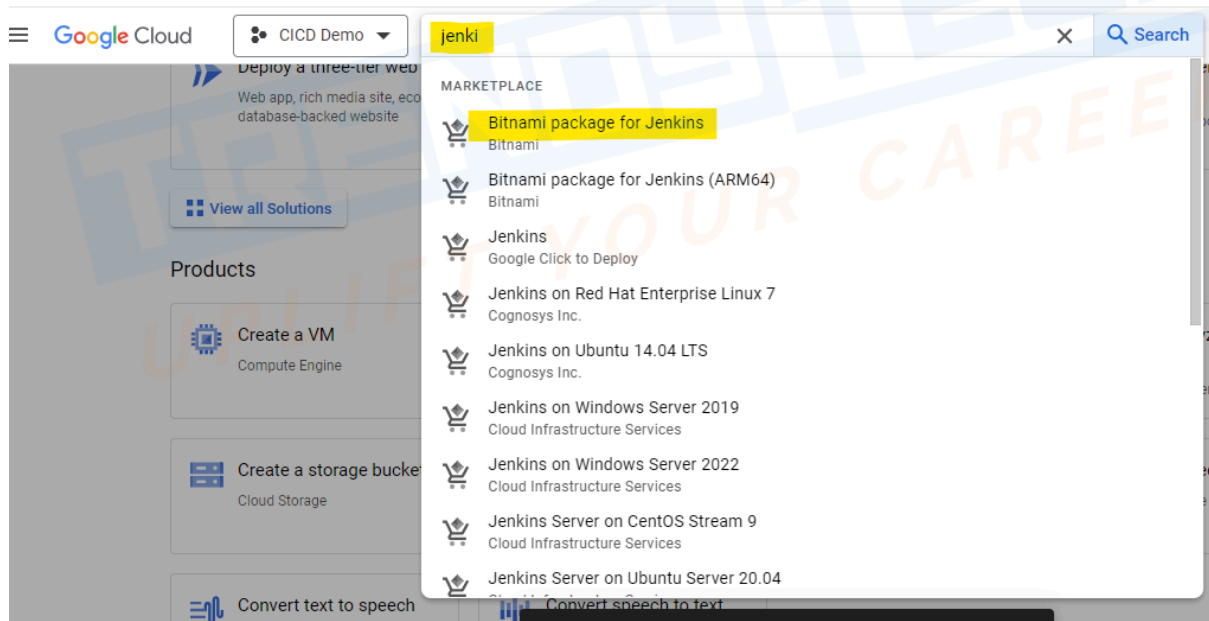





Deploy the necessary resources - Like Jenkins:

To deploy the jenkins follow below steps:


Search for “jenkin” in and in the marketplace you will get the option of “**Bitnami package for Jenkins**” refer attached screenshot.



 Google Cloud

My First Project ▼


← Product details



Bitnami package for Jenkins


[Bitnami](#)

Up-to-date, secure, and ready to run.


[GET STARTED](#) [VIEW DEPLOYMENTS](#)

[OVERVIEW](#) [PRICING](#) [SUPPORT](#) [RELATED PRODUCTS](#)


Allow the installation that it will ask for. Then you will get the option to launch the package as shown in the screenshot below.

 Google Cloud

CICD Demo ▼

 To avoid losing access to Google Cloud services, an administrator must verify this account.

← Product details



Bitnami package for Jenkins

[Bitnami](#)

Up-to-date, secure, and ready to run.

[LAUNCH](#) [VIEW DEPLOYMENTS](#)

[OVERVIEW](#) [PRICING](#) [SUPPORT](#) [RELATED PRODUCTS](#)

Overview

Jenkins is an open source automation server that helps you automate the building, testing, and deployment of any project across multiple platforms.

Additional details

For package deployment, specify the zone and machine type as highlighted in the screenshot. Leave other configurations unchanged, and proceed to click on "Deploy."

Google Cloud

CICD Demo

Search

Help

New Bitnami package for Jenkins deployment

Deployment name *

jenkins-1

Zone

asia-south1-a

Machine type

General purpose

Compute optimized

Memory optimized

Series

N1

Machine type

n1-standard-1 (1 vCPU, 3.75 GB memory)

vCPU

1

Memory

3.75 GB

Additional information

Bitnami package for Jenkins overview

Product provided by Bitnami

Bitnami Jenkins Usage Fee

Bitnami does not charge a usage fee.

INR 0.00/mo

Infrastructure fee

VM instance: 1 vCPU + 3.75 GB memory (n1-standard-1)

Standard Persistent Disk: 10GB

Sustained use discount

INR 3,471.24/mo

INR 40.01/mo

- INR 1,041.37/mo

Estimated monthly total

INR 2,469.87/mo

All products are priced in USD and charged in the currency (INR) specified by your Billing Account. The price for this month is calculated with an exchange rate of 1 USD = 83.35 INR

Note: If you face this issue please try to deploy it again with a different zone.

jenkins-1

DELETE

jenkins-1 failed to deploy

VIEW DETAILS

Overview - jenkins-1

bitnami-package-for bitnami-package-for-jinja

bitnami-package-for-vm-tmpl vm_instance.py

jenkins-1-vm vm instance

generated-password-0 password.py

software-status software_status.py

jenkins-1-config config

jenkins-1-software config waiter

software-status-script software_status_script.py

jenkins-1-tcp-80 firewall

jenkins-1-tcp-443 firewall

bitnami-package-for

bitnami-package-for has resource level errors

jenkins-1-vm:

("ResourceType": "compute.v1.instance", "ResourceErrorCode": "ZONE_RESOURCE_POOL_EXHAUSTED", "ResourceErrorMessage": "The zone 'projects/grand-guru-410310/zones/asia-south1-a' does not have enough resources available to fulfill the request. Try a different zone, or try again later.")

Bitnami package for Jenkins

Solution provided by Bitnami

Site address

Admin user

Admin password (Temporary)

Instance

Instance zone

Instance machine type

user

X9nLGeWskjKv

jenkins-1-vm

asia-south1-a

n1-standard-1

MORE ABOUT THE SOFTWARE

Once it is deployed you will get below screen. Now log into SSH.

The screenshot displays the Google Cloud Platform console for a VM instance named 'jenkins-1'. The left sidebar shows the 'Overview' for 'jenkins-1', with the 'bitnami-package-for' component selected. The main panel shows a warning: 'jenkins-1 has been deployed, but contains warnings' with a 'VIEW DETAILS' button. The right panel shows the 'bitnami-package-for' component details, including the instance name 'jenkins-1-vm', zone 'asia-east1-a', and machine type 'n1-standard-1'. Below this, there are links to 'Get started with Bitnami package for Jenkins' and 'Suggested next steps'.

jenkins-1 [DELETE](#)

jenkins-1 has been deployed, but contains warnings [VIEW DETAILS](#)

Overview - jenkins-1

- bitnami-package-for bitnami-package-for.jinja
 - bitnami-package-for-vm-tmpl vm_instance.py
 - jenkins-1-vm vm instance
 - generated-password-0 password.py
 - software-status software_status.py
 - jenkins-1-config config
 - jenkins-1-software config waiter
 - software-status-script software_status_script.py
 - jenkins-1-tcp-80 firewall
 - jenkins-1-tcp-443 firewall

bitnami-package-for

Instance [jenkins-1-vm](#)

Instance zone asia-east1-a

Instance machine type n1-standard-1

[MORE ABOUT THE SOFTWARE](#)

Get started with Bitnami package for Jenkins

[VISIT THE SITE](#) [SSH](#)

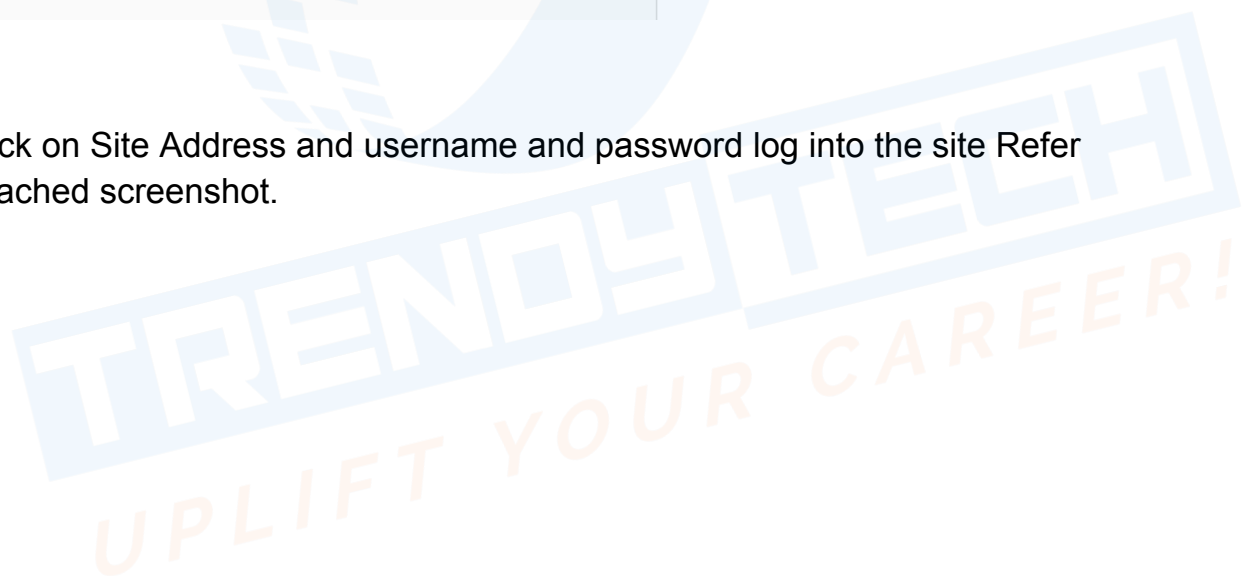
Suggested next steps

- Change the temporary password
For additional security, it is recommended that you change the pass
- Assign a static external IP address to your VM instance
An ephemeral external IP address has been assigned to the VM inst
require a static external IP address, you may promote the address to
[more](#)

Documentation

- [Getting Started](#)
Get started with Bitnami package for Jenkins.


Click on Site Address and username and password log into the site Refer attached screenshot.



×

bitnami-package-for

Close



Bitnami package for Jenkins

Solution provided by Bitnami

Site address	http://35.201.205.191/
Admin user	user
Admin password (Temporary)	JEzgjzZPHKm9
Instance	jenkins-1-vm
Instance zone	asia-east1-a
Instance machine type	n1-standard-1

✓

MORE ABOUT THE SOFTWARE

Get started with Bitnami package for Jenkins

VISIT THE SITE

SSH

Suggested next steps

- Change the temporary password
For additional security, it is recommended that you change the password.

Using the following commands
“sudo apt-get install pip”,
“sudo apt-get install sshpass”
install pip and sshpass respectively in SSH.



```
Linux jenkins-1-vm 5.10.0-26-cloud-amd64 #1 SMP Debian 5.10.197-1 (2023-09-29) x86_64
```

```
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
```

```
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
```

```

  _ _ _ _ _
 | _ | _ | _ |
 | _ | _ | _ |
 | _ | _ | _ |
  _ _ _ _ _
```

```
*** Welcome to the Bitnami package for Jenkins 2.426.2-0 ***
*** Documentation: https://docs.bitnami.com/google/apps/jenkins/ ***
*** https://docs.bitnami.com/google/ ***
*** Bitnami Forums: https://github.com/bitnami/vms/ ***
```

```
aratishatti15@jenkins-1-vm:~$ python --version
```

```
Python 2.7.18
```

```
aratishatti15@jenkins-1-vm:~$ python3 --version
```

```
Python 3.9.2
```

```
aratishatti15@jenkins-1-vm:~$ sudo apt-get install pip
```

```
Reading package lists... Done
```

```
Building dependency tree... Done
```

```
Reading state information... Done
```

```
Note, selecting 'python3-pip' instead of 'pip'
```

```
The following packages were automatically installed and are no longer required:
```

```
libevent-2.1-7 libgnutls-dane0 libunbound8
```

```
Use 'sudo apt autoremove' to remove them.
```

```
The following additional packages will be installed:
```

```
javascript-common libexpat1-dev libjs-jquery libjs-sphinxdoc libjs-underscore
```



```
Preparing to unpack .../14-python3-pip_20.3.4-4+deb11u1_all.deb ...
```

```
Unpacking python3-pip (20.3.4-4+deb11u1) ...
```

```
Setting up javascript-common (11+nmu1) ...
```

```
Setting up python3-wheel (0.34.2-1) ...
```

```
Setting up libexpat1-dev:amd64 (2.2.10-2+deb11u5) ...
```

```
Setting up python-pip-whl (20.3.4-4+deb11u1) ...
```

```
Setting up libjs-jquery (3.5.1+dfsg+~3.5.5-7) ...
```

```
Setting up python3-lib2to3 (3.9.2-1) ...
```

```
Setting up libjs-underscore (1.9.1~dfsg-3) ...
```

```
Setting up python3-distutils (3.9.2-1) ...
```

```
Setting up python3-setuptools (52.0.0-4) ...
```

```
Setting up libpython3.9-dev:amd64 (3.9.2-1) ...
```

```
Setting up python3-pip (20.3.4-4+deb11u1) ...
```

```
Setting up libjs-sphinxdoc (3.4.3-2) ...
```

```
Setting up python3.9-dev (3.9.2-1) ...
```

```
Setting up libpython3-dev:amd64 (3.9.2-3) ...
```

```
Setting up python3-dev (3.9.2-3) ...
```

```
Processing triggers for man-db (2.9.4-2) ...
```

```
aratishatti15@jenkins-1-vm:~$ sudo apt-get install sshpass
```

```
Reading package lists... Done
```

```
Building dependency tree... Done
```

```
Reading state information... Done
```

```
The following packages were automatically installed and are no longer required:
```

```
libevent-2.1-7 libgnutls-dane0 libunbound8
```

```
Use 'sudo apt autoremove' to remove them.
```

```
The following NEW packages will be installed:
```

```
sshpas
```


In UI, install the following plugins

Dashboard view -

This plugin allows you to create a customized dashboard view, providing a summary of information from various jobs or builds. It helps in creating a visual representation of the overall build health and status.

Github branch Source -

The GitHub plugin in Jenkins enables integration with GitHub repositories. It allows Jenkins to trigger builds automatically when changes are pushed to a GitHub repository, and it provides the ability to specify branches for building.

pipeline declarative -

Declarative Pipeline is a feature within the Pipeline plugin that provides a simplified and opinionated syntax for defining build pipelines. It allows you to express pipelines in a more structured and readable format.

pipeline stage view -

The Stage View plugin enhances the visualization of Jenkins Pipelines. It provides a graphical representation of the stages in your pipeline, showing the progress and status of each stage.

To install the plugins follow below steps:

In dashboard select “manage jenkins” => Plugins

Dashboard > Manage Jenkins

+ New Item

People

Build History

Manage Jenkins

My Views

Build Queue

No builds in the queue.

Build Executor Status

1 Idle

Manage Jenkins

Search settings

System Configuration

- System**
Configure global settings and paths.
- Tools**
Configure tools, their locations and automatic installers.
- Plugins**
Add, remove, dis plugins that can functionality of J
- Nodes**
Add, remove, control and monitor the various nodes that Jenkins runs jobs on.
- Clouds**
Add, remove, and configure cloud instances to provision agents on-demand.

Now, go to **Available plugins** => select mentioned plugin => click on **install**

Refer attached screenshot.

Jenkins

Search (CTRL+K)

user

Dashboard > Manage Jenkins > Plugins

Plugins

stage view

Install

Updates

Available plugins

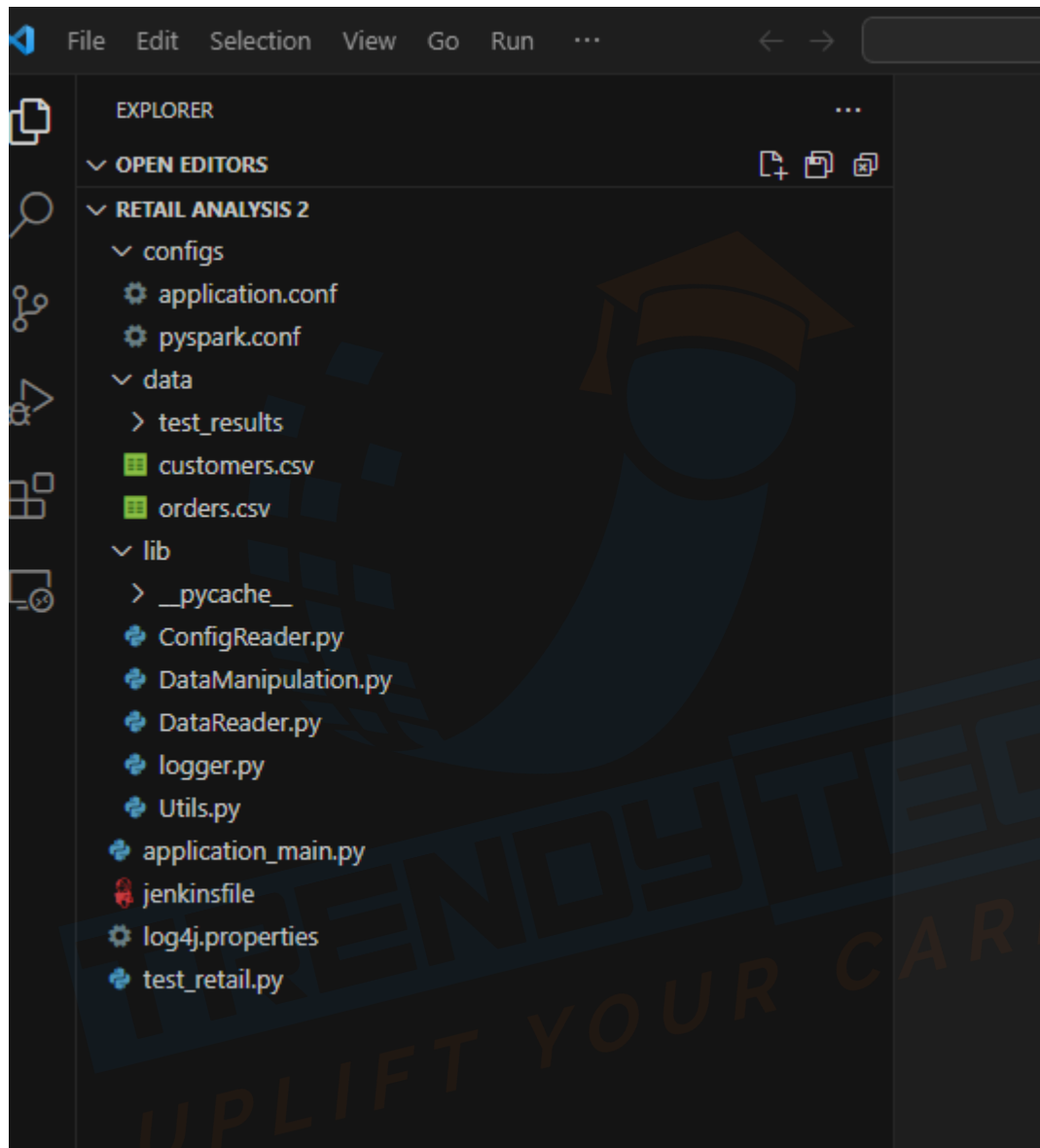
Installed plugins

Advanced settings

Install	Name	Released
<input checked="" type="checkbox"/>	Dashboard View 2.495.v07e81500c3f2 User Interface Customizable dashboard that can present various views of job information.	5 mo 9 days ago
<input checked="" type="checkbox"/>	GitHub Branch Source 1767.va_7d01ea_c7256 pipeline github Source Code Management Multibranch projects and organization folders from GitHub. Maintained by CloudBees, Inc.	23 days ago
<input checked="" type="checkbox"/>	Pipeline: Declarative 2.2168.vf921b_4e72c73 pipeline Miscellaneous An opinionated, declarative Pipeline.	7 days 18 hr ago
<input checked="" type="checkbox"/>	Pipeline: Stage View 2.34 User Interface	2 mo 10 days ago

Steps for creating a project Retail Analysis, initialising Git, and establishing branches on GitHub are as follows:

Create a project Retail Analysis in VS code if you do not have

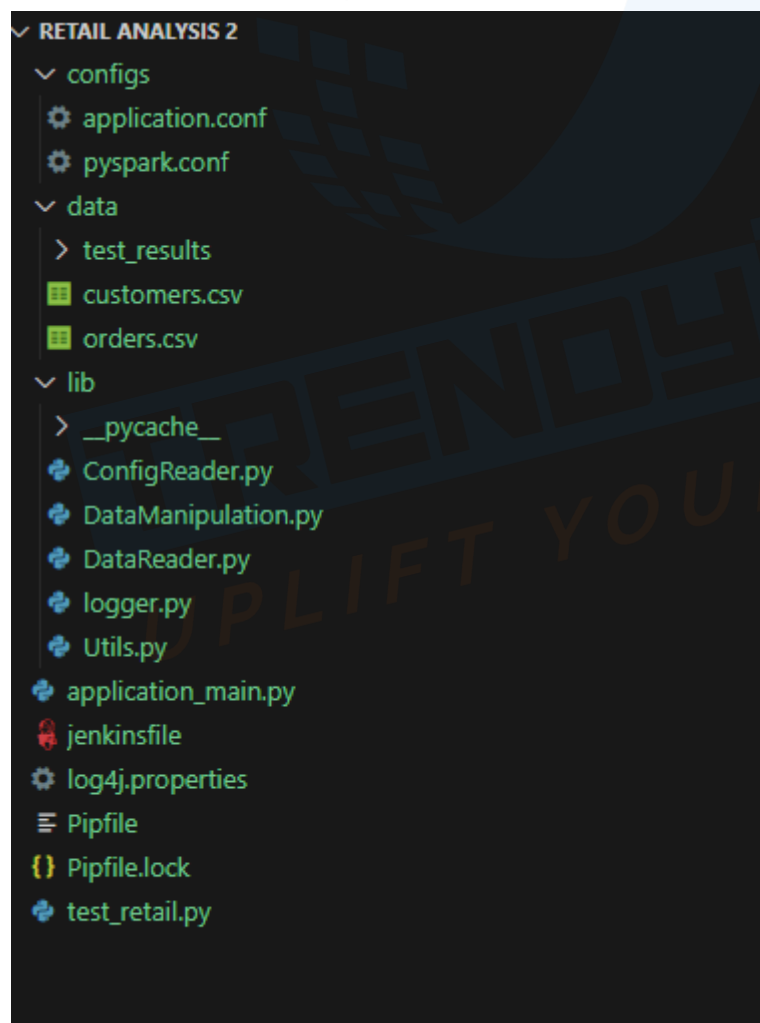


Now **create virtual environment** in it using command : “**pipenv install**”

```
C:\Users\HP\Desktop\Retail Analysis 2>pipenv install
Creating a virtualenv for this project...
Pipfile: C:\Users\HP\Desktop\Retail Analysis 2\Pipfile
Using c:/users/hp/.pyenv/pyenv-win/versions/3.9.2/python3.exe (3.9.2) to create virtualenv...
[=== ] Creating virtual environment...created virtual environment CPython3.9.2.final.0-64 in 10680ms
creator CPython3Windows(dest=C:\Users\HP\.virtualenvs\Retail_Analysis_2-GE1CUMNa, clear=False, no_vcs_i
gnore=False, global=False)
seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle, via=copy, app_data_dir=
C:\Users\HP\AppData\Local\pypa\virtualenv)
added seed packages: pip==23.3.2, setuptools==69.0.3, wheel==0.42.0
activators BashActivator,BatchActivator,FishActivator,MushellActivator,PowerShellActivator,PythonActiva
tor

Successfully created virtual environment!
Virtualenv location: C:\Users\HP\.virtualenvs\Retail_Analysis_2-GE1CUMNa
Pipfile.lock (16c839) out of date, updating to (71871d)...
Locking [packages] dependencies...
Locking [dev-packages] dependencies...
Updated Pipfile.lock (467be4a3aca30b5feab72c4d10f547c063e1c02754d633659e116ca9371871d)!
Installing dependencies from Pipfile.lock (71871d)...
To activate this project's virtualenv, run pipenv shell.
```

Note : It will create pipfile and pipfile.lock



Also install **pytest** using command : “**pipenv install pytest**”

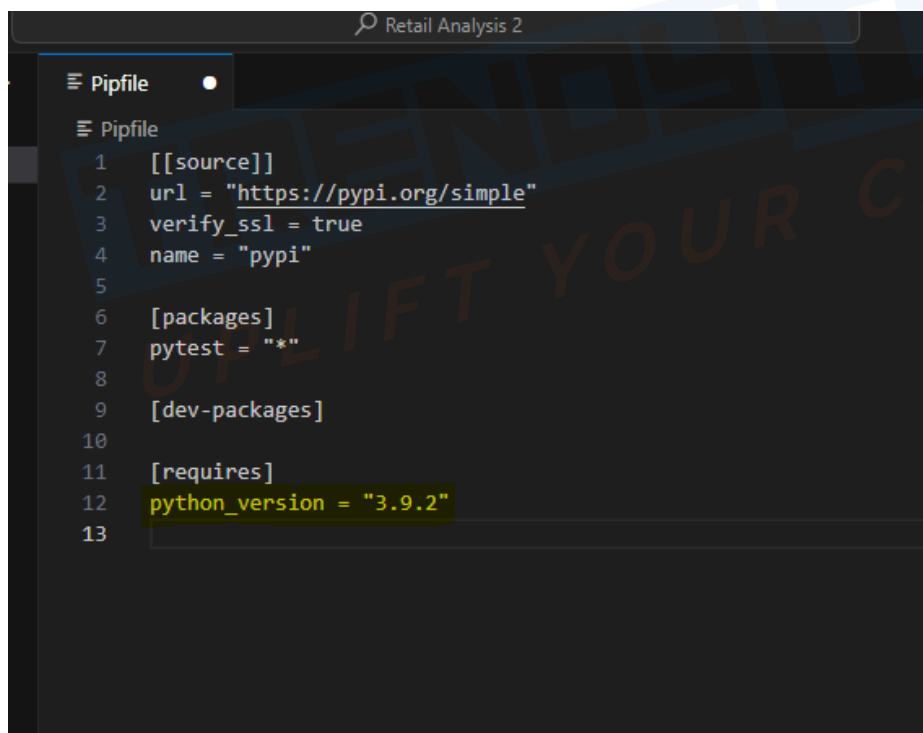
```
C:\Users\HP\Desktop\Retail Analysis 2>pipenv install pytest
Installing pytest...
Resolving pytest...
Added pytest to Pipfile's [packages] ...
Installation Succeeded
Pipfile.lock (71871d) out of date, updating to (f4b28f)...
Locking [packages] dependencies...
Building requirements...
Resolving dependencies...
Success!
Locking [dev-packages] dependencies...
Updated Pipfile.lock (8c4501cc7de24ae9782093e8b563a1a434860751698b668c196074c8e5f4b28f)!
Installing dependencies from Pipfile.lock (f4b28f)...
To activate this project's virtualenv, run pipenv shell.
Alternatively, run a command inside the virtualenv with pipenv run.
```

In the virtual environment install the python version which is present in your jenkins.

Here it is 3.9.2 so creating virtual environment with python 3.9.2

Note: Before removing the virtual environment, first **exit** it using the “**exit()**” command.

Now you can remove it using the command you can “**pipenv -rm**” and **recreate it** with the required **python version**.



```
Pipfile
[[source]]
url = "https://pypi.org/simple"
verify_ssl = true
name = "pypi"

[packages]
pytest = "*"

[dev-packages]

[requires]
python_version = "3.9.2"
```

If your existing project is not under version control, use **"git init"** to start tracking changes.

```
C:\Users\HP\Desktop\Retail Analysis 2>git init
Initialized empty Git repository in C:/Users/HP/Desktop/Retail Analysis 2/.git/
```

To **rename** branch **master** to **main** use command : **"git branch -M main"**

Using the commands **"git add ."** add all the changes

```
C:\Users\HP\Desktop\Retail Analysis 2>git add .
warning: in the working copy of 'Pipfile', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'Pipfile.lock', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'data/customers.csv', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'data/orders.csv', LF will be replaced by CRLF the next time Git touches it
```

And using command **"git commit -m "performing initial commit""** commit all the changes

```
C:\Users\HP\Desktop\Retail Analysis 2>git commit -m "performing initial commit"
[main (root-commit) b499b8b] performing initial commit
31 files changed, 81720 insertions(+)
create mode 100644 Pipfile
create mode 100644 Pipfile.lock
create mode 100644 application_main.py
create mode 100644 configs/application.conf
create mode 100644 configs/pyspark.conf
create mode 100644 data/customers.csv
create mode 100644 data/orders.csv
create mode 100644 data/test_results/state_aggregate.csv
create mode 100644 jenkinsfile
create mode 100644 lib/ConfigReader.py
```

Go to github and create a repo with the name **Retail Project**


Refer attached screenshot.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk (*).



Owner * Repository name *

 aratishatti15 / **Retail_analysis2**

✔ Retail_analysis2 is available.

Great repository names are short and memorable. Need inspiration? How about **automatic-disco** ?

Description (optional)

- ☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.
- ☐  **Private**
You choose who can see and commit to this repository.

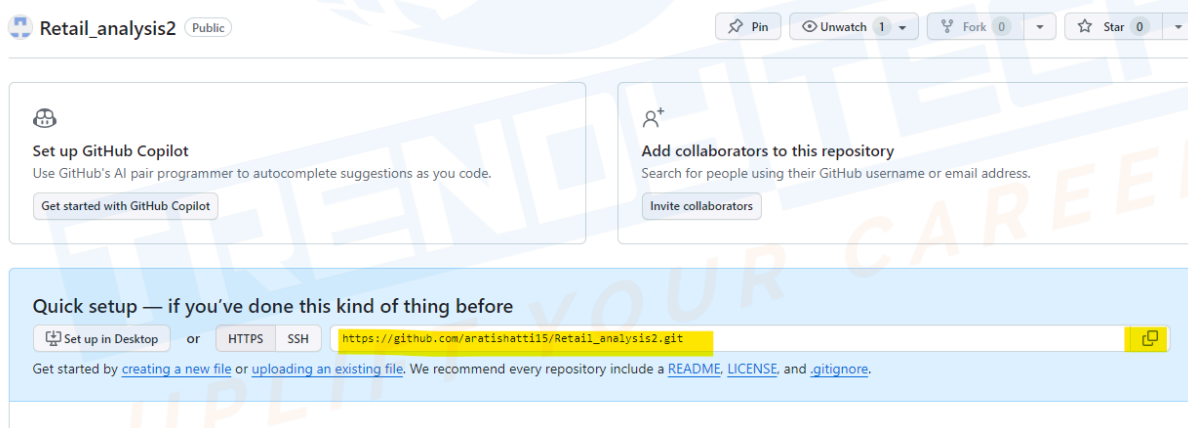
Initialize this repository with:

- ☐ **Add a README file**
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: None ▾

Also copy the ssh link as highlighted in the screenshot.



The screenshot shows the GitHub repository page for 'Retail_analysis2' by user 'aratishatti15'. The repository is public. At the top, there are buttons for Pin, Unwatch (1), Fork (0), and Star (0). Below this, there are two cards: 'Set up GitHub Copilot' and 'Add collaborators to this repository'. A 'Quick setup' section is visible, showing options for 'Set up in Desktop', 'HTTPS', and 'SSH'. The SSH link is highlighted in yellow: `https://github.com/aratishatti15/Retail_analysis2.git`. Below the SSH link, there is a note: 'Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).'

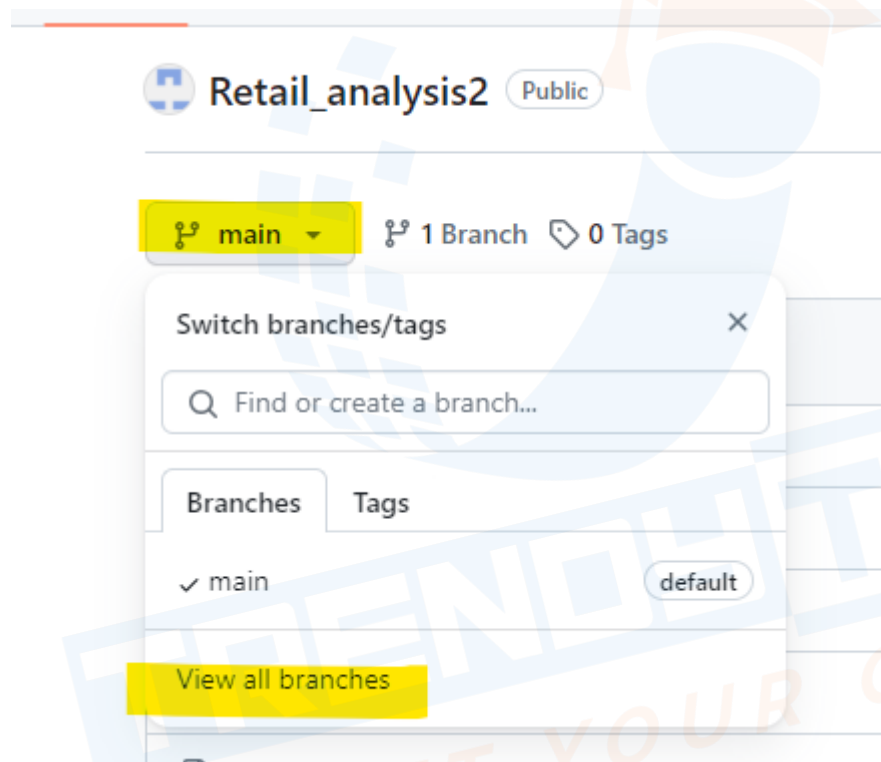
`git remote add origin <url of project>`

```
C:\Users\HP\Desktop\Retail Analysis 2>git remote add origin https://github.com/aratishatti15/Retail_analysis2.git
```

To push the code to the origin main run the command: **“git push origin main”**

```
C:\Users\HP\Desktop\Retail Analysis 2>git push origin main
Enumerating objects: 38, done.
Counting objects: 100% (38/38), done.
Delta compression using up to 2 threads
Compressing objects: 100% (36/36), done.
Writing objects: 100% (38/38), 715.50 KiB | 3.22 MiB/s, done.
Total 38 (delta 8), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (8/8), done.
To https://github.com/aratishatti15/Retail_analysis2.git
* [new branch]      main -> main
```

Click on view all branches and create the new branches : **main**, **dev**, **uat**, **test**



Branches

Overview

Yours

Active

Stale

All

Q

Search branches...

Default

Branch	Updated	Check
main	<div><div></div>3 minutes ago</div>	

Your branches

Branch	Updated	Check
test	<div><div></div>now</div>	
uat	<div><div></div>now</div>	
dev	<div><div></div>now</div>	

All have the same code

Create and switch to a new branch in a Git repository on your local machine using the command :

git checkout -b feature-rp-50001

```
C:\Users\HP\Desktop\Retail Analysis 2>git checkout -b feature-rp-50001
Switched to a new branch 'feature-rp-50001'
```

After making modifications, commit all the changes, and then push the changes to the remote repository using the command

git push origin feature-rp-50001

```
C:\Users\HP\Desktop\Retail Analysis 2>git add .

C:\Users\HP\Desktop\Retail Analysis 2>git commit -m "initial commit in feature"
On branch feature-rp-50001
nothing to commit, working tree clean

C:\Users\HP\Desktop\Retail Analysis 2>git push origin feature-rp-50001
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'feature-rp-50001' on GitHub by visiting:
remote:   https://github.com/aratishatti15/Retail_analysis2/pull/new/feature-rp-50001
remote:
To https://github.com/aratishatti15/Retail_analysis2.git
 * [new branch]      feature-rp-50001 -> feature-rp-50001
```

when you execute the above a new feature branch will be created in github

Configuring Jenkins to interpret GitHub events.

Making configurations in jenkins so that it can read the github events like branch creation, git push, pull request ..etc

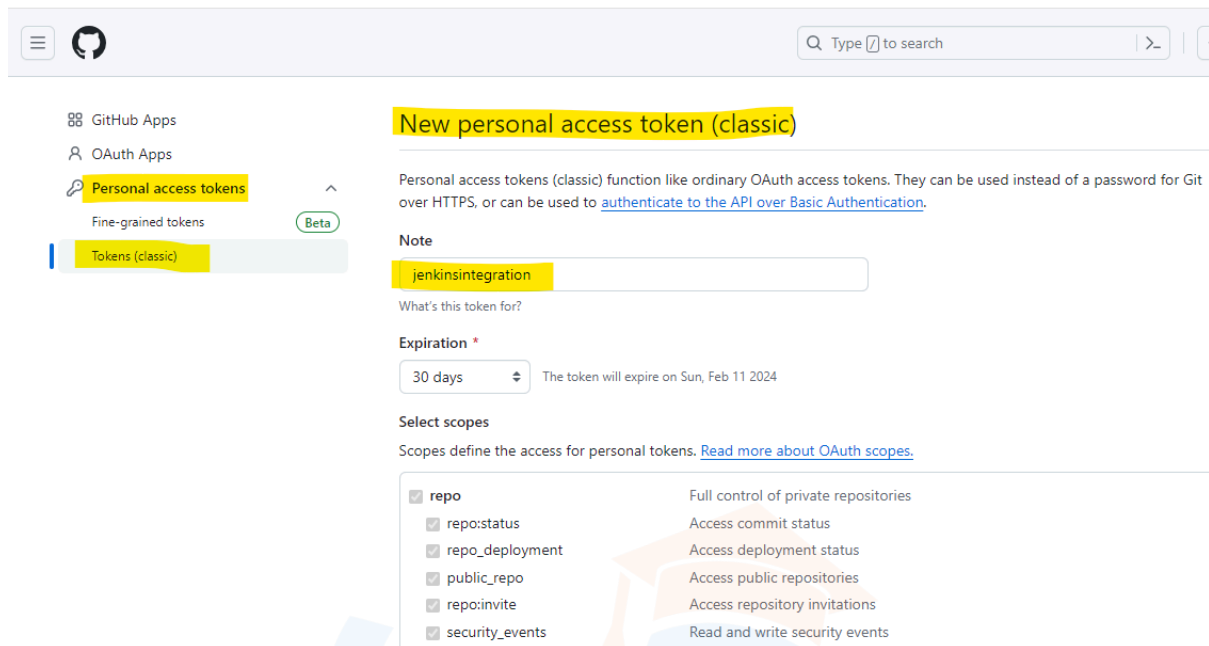
To establish the connection between github and jenkins, it has to be done from both ways.

Steps to follow in Jenkins:

Pre-step:

For creation of token in Github:

Go to Github => Settings => Developer Settings => Personal Access token => Tokens (Classic) => select all the scopes => generate token => Copy the token



GitHub Apps

OAuth Apps

Personal access tokens

Fine-grained tokens

Tokens (classic)

New personal access token (classic)

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

Note

jenkinsintegration

What's this token for?

Expiration *

30 days The token will expire on Sun, Feb 11 2024

Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes.](#)

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input checked="" type="checkbox"/> repo:status	Access commit status
<input checked="" type="checkbox"/> repo_deployment	Access deployment status
<input checked="" type="checkbox"/> public_repo	Access public repositories
<input checked="" type="checkbox"/> repo:invite	Access repository invitations
<input checked="" type="checkbox"/> security_events	Read and write security events

Go to Jenkins

Dashboard => Manage Jenkins => System

Mention the github token while creating secret text.



Domain

Global credentials (unrestricted)

Kind

Secret text

Scope ?

Global (Jenkins, nodes, items, all child items, etc)

Secret

.....

ID ?

Dashboard > Manage Jenkins > System >

GitHub Servers ?

≡ **GitHub Server** ?

Name ?

API URL ?

Credentials ?

Secret text

+ Add

Credentials verified for user aratishatti15, rate limit: 4999

Test connection

Save Apply

Also set Github Api usage limit strategy as shown below and then save it.

Dashboard > Manage Jenkins > System >

Add GitHub Server

Advanced Edited

GitHub API usage

GitHub API usage rate limiting strategy ?

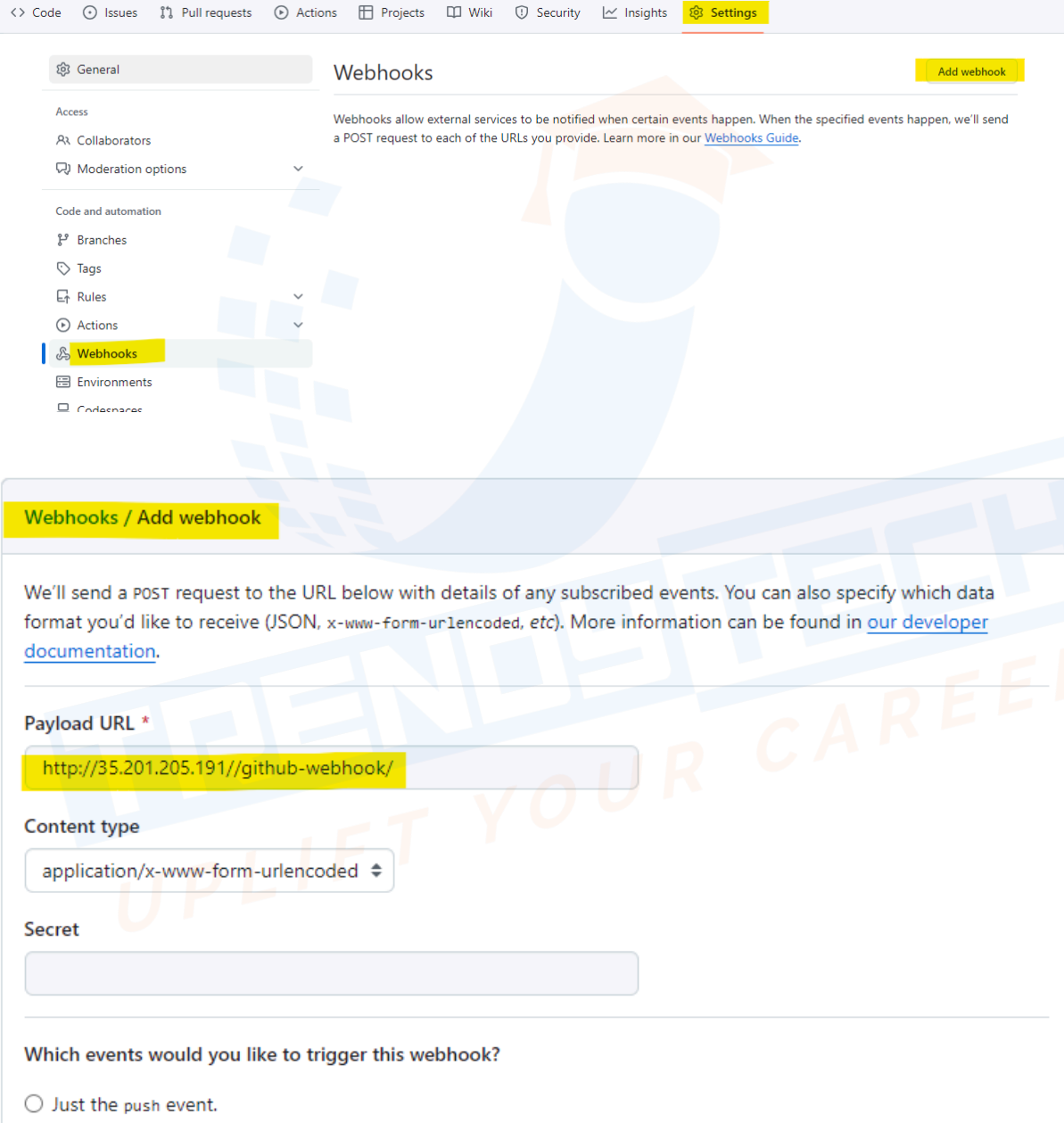
Never check rate limit (NOT RECOMMENDED)

In Github

Go to the Repository for Retail Analysis => Setting => Webhooks

Now create a new webhook in which you have to mention jenkins URL and “/github-webhook/” as shown below.

<jenkinsurl>/github-webhook/



The screenshot shows the GitHub repository settings page for 'Webhooks'. The left sidebar has a 'Webhooks' tab selected. The main content area is titled 'Webhooks' and includes an 'Add webhook' button. Below this, there is a section for configuring a new webhook. The 'Payload URL' field is filled with 'http://35.201.205.191//github-webhook/'. The 'Content type' is set to 'application/x-www-form-urlencoded'. The 'Secret' field is empty. The 'Which events would you like to trigger this webhook?' section shows the 'Just the push event' option selected.

Webhooks / Add webhook

We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, x-www-form-urlencoded, etc). More information can be found in [our developer documentation](#).

Payload URL *

http://35.201.205.191//github-webhook/

Content type

application/x-www-form-urlencoded

Secret

Which events would you like to trigger this webhook?

☐ Just the push event.

Which events would you like to trigger this webhook? => Let me select individual events.

=> Branch or tag creation, Pull requests, Pushes

Refer attached screenshot.

Which events would you like to trigger this webhook?

☐ Just the push event.

☐ Send me everything.

☒ Let me select individual events.

☒ **Branch or tag creation**

Branch or tag created.

☐ **Branch protection configurations**

All branch protections disabled or enabled for a repository.

☐ **Check runs**

Check run is created, requested, rerequested, or

unresolved.

☒ **Pull requests**

Pull request assigned, auto merge disabled, auto merge enabled, closed, converted to draft, demilestoned, dequeued, edited, enqueued, labeled, locked, milestoned, opened, ready for review, reopened, review request removed, review requested, synchronized, unassigned, unlabeled, or unlocked.

☐ **Registry packages**

Registry package published or updated in a repository.

☐ **Secrets**

Secret created, updated, or deleted.

☐ **Branch or tag deletion**

Branch or tag deleted.

☐ **Branch protection rules**

Branch protection rule created, deleted or edited.

☐ **Check suites**

Check suite is created, requested, rerequested, or

Pull request review submitted, edited, or dismissed.

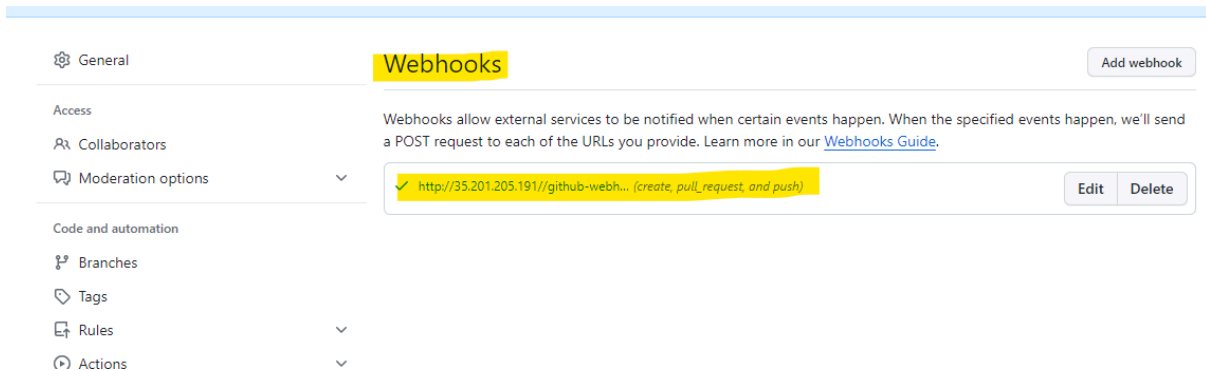
☒ **Pushes**

Git push to a repository.

☐ **Releases**

Release created, edited, published, unpublished, or deleted.

And click on “Add Webhook”. The webhook will be listed as shown below.



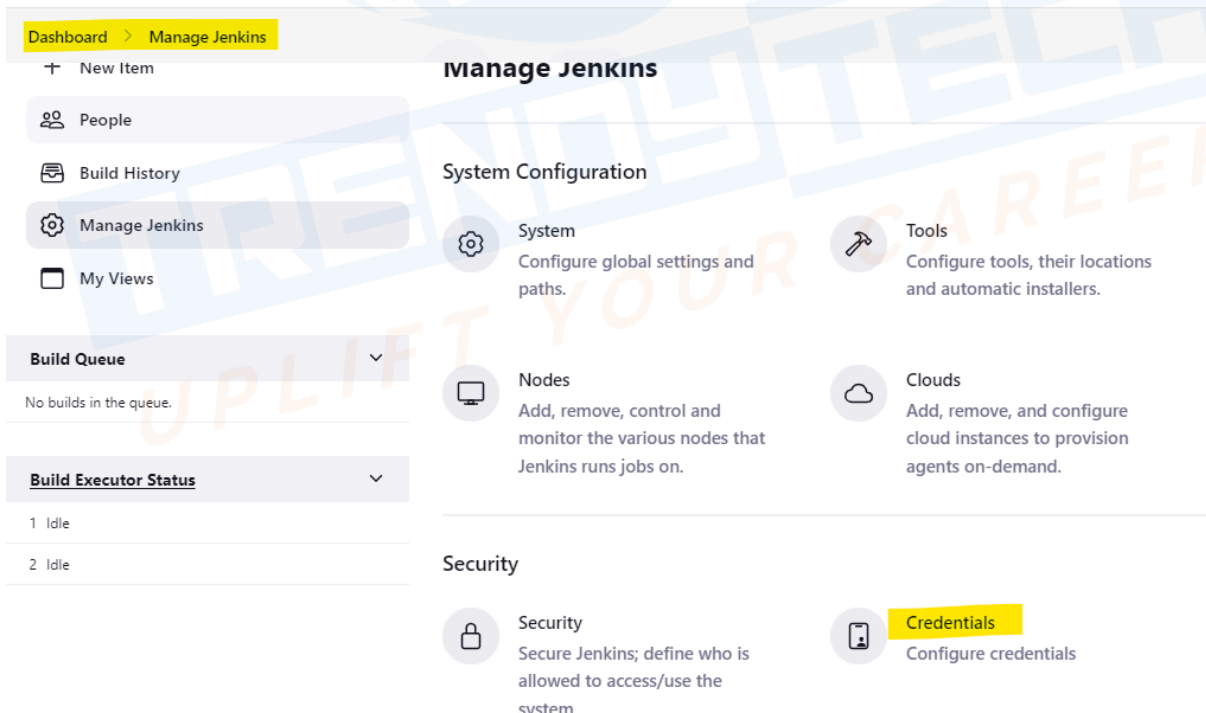
The screenshot shows the Jenkins 'Webhooks' configuration page. On the left, a sidebar contains links for 'General', 'Access', 'Collaborators', 'Moderation options', 'Code and automation', 'Branches', 'Tags', 'Rules', and 'Actions'. The 'Webhooks' tab is highlighted in yellow. The main content area explains that webhooks allow external services to be notified of events and provides a link to the 'Webhooks Guide'. Below this, a table lists configured webhooks. One webhook is shown with a green checkmark, the URL 'http://35.201.205.191/github-webh...', and the event types '(create, pull_request, and push)'. To the right of the table are 'Edit' and 'Delete' buttons. An 'Add webhook' button is located in the top right corner.

Webhook	Events	Actions
✓ http://35.201.205.191/github-webh...	(create, pull_request, and push)	Edit Delete

With these steps Both ways connectivity has been done.

Also set the credentials for your multi lab cluster.

Go to jenkins => Manage jenkins => Credentials => System => Global Credentials => Add Credentials



The screenshot shows the 'Manage Jenkins' page. The breadcrumb trail at the top is 'Dashboard > Manage Jenkins', with 'Manage Jenkins' highlighted in yellow. The left sidebar contains links for 'New Item', 'People', 'Build History', 'Manage Jenkins' (highlighted), and 'My Views'. Below the sidebar, there are sections for 'Build Queue' (showing 'No builds in the queue.') and 'Build Executor Status' (showing two 'Idle' executors). The main content area is titled 'Manage Jenkins' and is divided into two sections: 'System Configuration' and 'Security'. The 'System Configuration' section includes links for 'System' (Configure global settings and paths), 'Tools' (Configure tools, their locations and automatic installers), 'Nodes' (Add, remove, control and monitor the various nodes that Jenkins runs jobs on), and 'Clouds' (Add, remove, and configure cloud instances to provision agents on-demand). The 'Security' section includes links for 'Security' (Secure Jenkins; define who is allowed to access/use the system) and 'Credentials' (Configure credentials), with 'Credentials' highlighted in yellow.

Dashboard > Manage Jenkins > Credentials

Credentials

T	P	Store	Domain	ID	Name
		System	(global)	5e2c236d-1e3a-442b-b4de-0eb524ee26fc	Secret text

Stores scoped to Jenkins

P	Store	Domains
	System	(global)

Icon: S M L

Dashboard > Manage Jenkins > Credentials > System

System

+ Add domain

Domain	Description
Global credentials (unrestricted)	Credentials that should be available irrespective of domain specification to requirements matching.

Icon: S M L

Provide the username, password and id for your "Multi Lab Cluster," and the system will generate the corresponding credentials as depicted below.

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted)

Username with password

Scope ?
Global (Jenkins, nodes, items, all child items, etc)

Username ?
itv006753

☐ Treat username as secret ?

Password ?
.....

ID ?
labcreds





Create

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >

Global credentials (unrestricted)

[+ Add Credentials](#)

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
 5e2c236d-1e3a-442b-b4de-0eb524ee26fc	Secret text	Secret text	
 ff534522-6cee-4e28-aca8-a6eb74be6b0c	itv006753/*****	Username with password	

Icon: ☒ S ☐ M ☐ L

Note: Modify the Jenkinsfile with the provided code, and proceed to add, commit and push the changes.

```
C:\Users\HP\Desktop\Retail Analysis 2>git add .

C:\Users\HP\Desktop\Retail Analysis 2>git commit -m "Edited jenkins file"
[feature-rp-50001 ba153a1] Edited jenkins file
1 file changed, 27 insertions(+)

C:\Users\HP\Desktop\Retail Analysis 2>git push origin feature-rp-50001
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 2 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 395 bytes | 395.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/aratishatti15/Retail_analysis2.git
b499b8b..ba153a1 feature-rp-50001 -> feature-rp-50001
```

Creation of Multibranch pipeline in Jenkins


Go to the Jenkins UI (Dashboard) => New items => Create the multibranch pipeline


Dashboard > All >


Enter an item name

Retail_analysis_pipeline

» Required field


**Freestyle project**
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even use for something other than software build.


**Pipeline**
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.


**Folder**
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.


OK

Dashboard > All >

**Pipeline**
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

**Folder**
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

**Multibranch Pipeline**
Creates a set of Pipeline projects according to detected branches in one SCM repository.

**Organization Folder**
Creates a set of multibranch project subfolders by scanning for repositories.

OK

Dashboard > Retail_analysis_pipeline > Configuration

Configuration

General

Branch Sources

Build Configuration

Scan Multibranch Pipeline Triggers

Orphaned Item Strategy

Appearance

Health metrics

Properties

General

Enabled

Display Name ?

Retail pipeline

Description

Plain text [Preview](#)

Branch Sources

[Save](#) [Apply](#)

Add the URL of Retail Analysis repository and in discover branches option mention “All branches”

Dashboard > Retail_analysis_pipeline > Configuration

Configuration

General

Branch Sources

Build Configuration

Scan Multibranch Pipeline Triggers

Orphaned Item Strategy

Appearance

Health metrics

Properties

Repository HTTPS URL

Repository HTTPS URL ?

[https://github.com/aratishatti15/Retail_analysis2.git](#)

Credentials ok. Connected to [https://github.com/aratishatti15/Retail_analysis2](#). [Validate](#)

☐ Repository Scan - Deprecated Visualization

Behaviors

[Discover branches](#) ?

Strategy ?

All branches

[Save](#) [Apply](#)

Note: Mention the name of jenkins file as you have created as it is case sensitive

Dashboard > Retail_analysis_pipeline > Configuration

Configuration

- General
- Branch Sources
- Build Configuration**
- Scan Multibranch Pipeline Triggers
- Orphaned Item Strategy
- Appearance
- Health metrics
- Properties

Build Configuration

Mode
by Jenkinsfile

Script Path ?
jenkinsfile

Scan Multibranch Pipeline Triggers
☐ Periodically if not otherwise run ?

Orphaned Item Strategy

Save Apply

After the pipeline is triggered and successfully executed, you'll observe the following output in the Jenkins UI.

Dashboard > Retail pipeline >

Retail pipeline

Folder name: Retail_analysis_pipeline

Disable Multibranch Pipeline

Branches (5) Pull Requests (0)

S	W	Name ↓	Last Success	Last Failure	Last Duration
✓	☀	dev	18 min #1	N/A	2.3 sec
✓	☁	feature-rp-50001	5 min 31 sec #4	9 min 21 sec #3	10 sec
✓	☀	main	18 min #1	N/A	2.3 sec
✓	☀	test	18 min #1	N/A	2.3 sec
✓	☀	uat	18 min #1	N/A	2.3 sec

This indicates that the pipeline is successfully triggered upon the git push.

Now create the new branch “**git checkout -b feature-rp-50001**” and push changes to remote using the command “**git push origin feature-rp-50001**”.

```

C:\Users\HP\Desktop\Retail Analysis 2>git checkout -b feature-rp-50002
Switched to a new branch 'feature-rp-50002'

C:\Users\HP\Desktop\Retail Analysis 2>git push origin feature-rp-50002
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'feature-rp-50002' on GitHub by visiting:
remote:   https://github.com/aratishatti15/Retail_analysis2/pull/new/feature-rp-50002
remote:
To https://github.com/aratishatti15/Retail_analysis2.git
 * [new branch]      feature-rp-50002 -> feature-rp-50002

C:\Users\HP\Desktop\Retail Analysis 2>

```

After the pipeline is triggered and successfully executed, you'll observe the following output in the Jenkins UI.

Dashboard > Retail pipeline >

Scan Repository Log

Multibranch Pipeline Events

Delete Multibranch Pipeline

People

Build History

GitHub

Rename

Pipeline Syntax

Credentials

Builds (6) Pull Requests (0)

S	W	Name ↓	Last Success	Last Failure
✓	☀	dev	23 min #1	N/A
✓	☁	feature-rp-50001	11 min #4	14 min
✓	☀	feature-rp-50002	2 min 9 sec #1	N/A
✓	☀	main	23 min #1	N/A
✓	☀	test	23 min #1	N/A
✓	☀	uat	23 min #1	N/A

Build Queue

No builds in the queue.

Icon: S M L

Icon legend

Atom feed for all

Atom feed for

Stage Logs (Deploy)

Print Message -- deploy completed successful (self time 9ms)

deploy completed successful

</> Changes Full project name: Retail_analysis_pipeline/feature-rp-50001

Build Now

View Configuration

Full Stage View

GitHub

Pipeline Syntax

Build History trend

Stage View

Average stage times:
(Average full run time: ~10s)

	Declarative: Checkout SCM	Build	Test	Package	Deploy
	5s	194ms	130ms	122ms	119ms
#4 Jan 12 01:38 No Changes	5s	194ms	130ms	122ms	119ms
#3					

This indicates that the pipeline is successfully triggered upon the creation of a new branch.

Now create and approve the pull request for merging changes of branch “feature-rp-50001” into “dev” branch.

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#) or [learn more about diff compari](#)

base: dev ← compare: feature-rp-50001 ✓ Able to merge. These branches can be automatically merged.

Add a title

Feature rp 50001

Add a description

Write Preview H B I

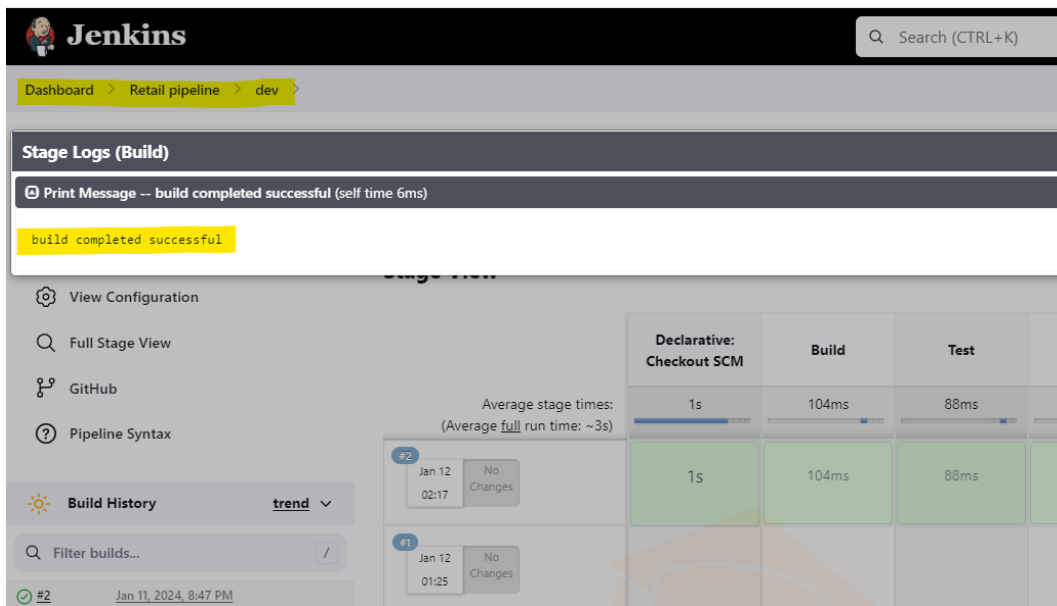
Add your description here...

Reviewer
No review

Assignee
No one assigned

Labels
None yet

Projects
None yet



This indicates that the pipeline is successfully triggered upon the creation of a new branch.

To facilitate the deployment of the Retail Project code from our local system to the edge node within our Multi-node lab, we'll craft a Jenkins file tailored to this specific use case.

Modify the Jenkinsfile with the provided code in feature-rp-50002, and proceed to add, commit and push the changes.

```

• PS C:\Users\HP\Desktop\Retail Analysis 2> git add .
• PS C:\Users\HP\Desktop\Retail Analysis 2> git commit -m "Modified jenkins file"
[feature-rp-50002 9934cfd] Modified jenkins file
1 file changed, 14 insertions(+), 9 deletions(-)
• PS C:\Users\HP\Desktop\Retail Analysis 2> git push origin feature-rp-50002
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 2 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 579 bytes | 13.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/aratishatti15/Retail_analysis2.git
e462ae2..9934cfd feature-rp-50002 -> feature-rp-50002
PS C:\Users\HP\Desktop\Retail Analysis 2>

```

Now the pipeline will run and it will zip all the files in the Retail project folder as per code and using scp command it will copy files to the mentioned path in the code of edge node and you can see the retail project folder in edge node of multi node lab.