"Automating CI/CD: Jenkins Integration with GitHub Webhooks"

JENKINS

. Introduction

In modern software development, Continuous Integration (CI) and Continuous Deployment (CD) pipelines play a critical role in automating code build, test, and deployment. Jenkins, a widely used open-source automation server, integrates seamlessly with GitHub webhooks to enable event-driven automation. Instead of polling for changes, Jenkins can be notified by GitHub whenever a new commit, push, or pull request occurs.

What Are Webhooks?

A Webhook is a mechanism where an application (GitHub) sends real-time data to another system (Jenkins) via an HTTP POST request.
In this context, GitHub notifies Jenkins about repository events (e.g., new commits).
Jenkins then automatically triggers the corresponding job or pipeline.

Benefits of Using Webhooks over Polling

- Event-driven: Builds trigger instantly when changes occur.
- Efficient: No constant polling load on Jenkins.
- Scalable: Suitable for large projects with frequent commits.
- Faster feedback: Developers see test/build results immediately.

Steps Involved in Integration

Prerequisites

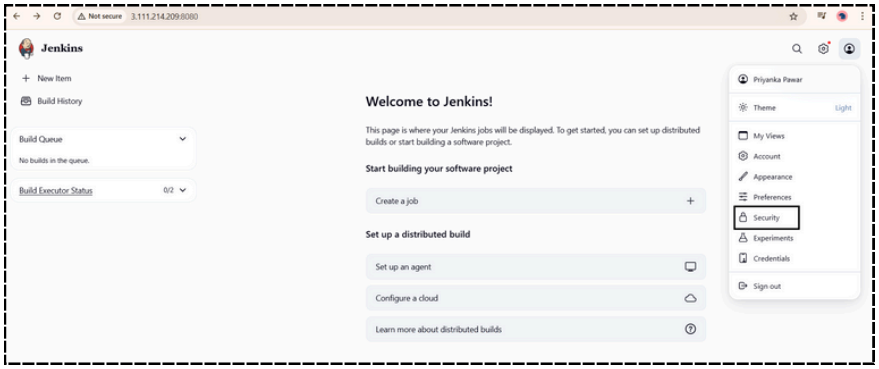
1. Jenkins server installed and accessible (local, cloud, or public IP).
2. GitHub repository available.
3. Jenkins plugins installed: GitHub Plugin, GitHub Integration Plugin, Pipeline Plugin.

JENKINS

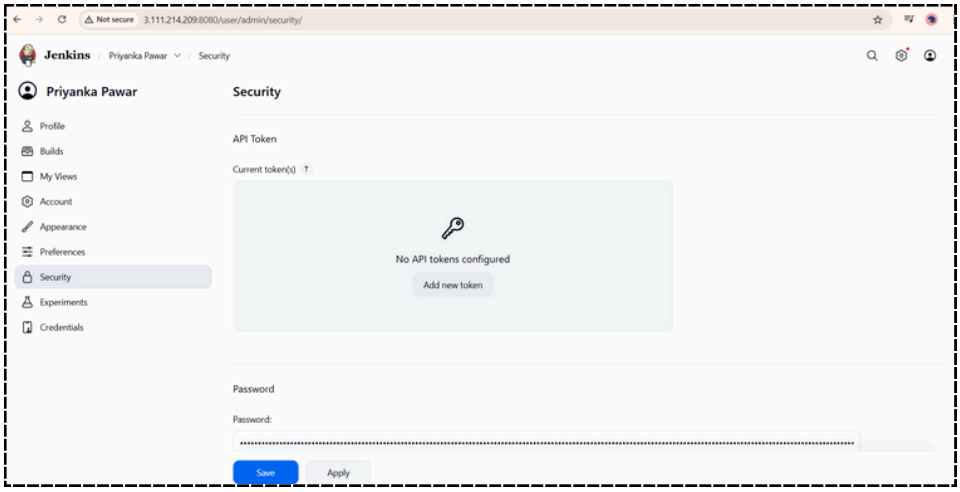
Creating an API Key

Log in to your Jenkins dashboard

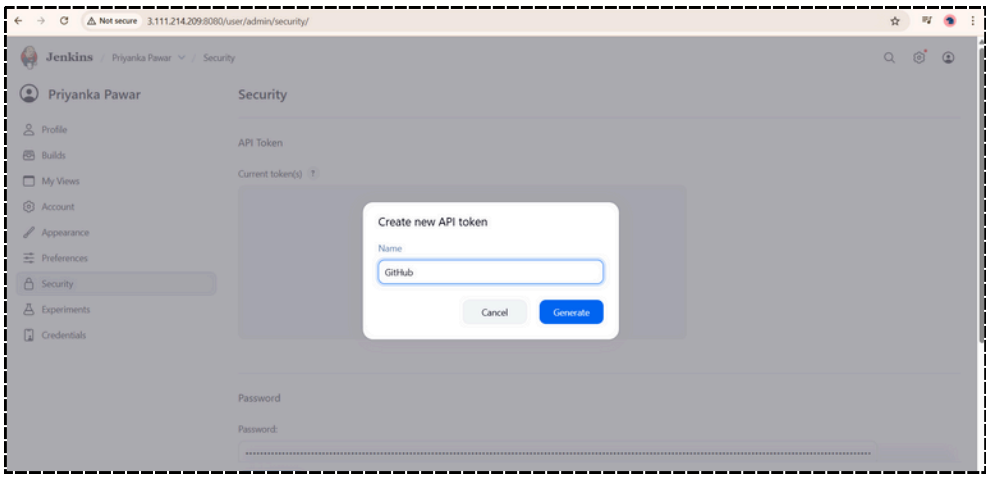
Click on your username click on security



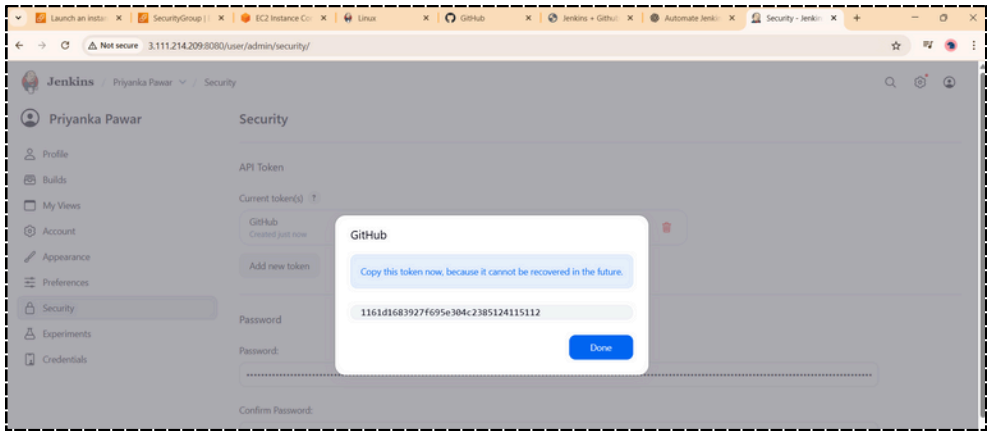
In the "API Token" section, click the "Add new Token" button.



Give token name and Generate token



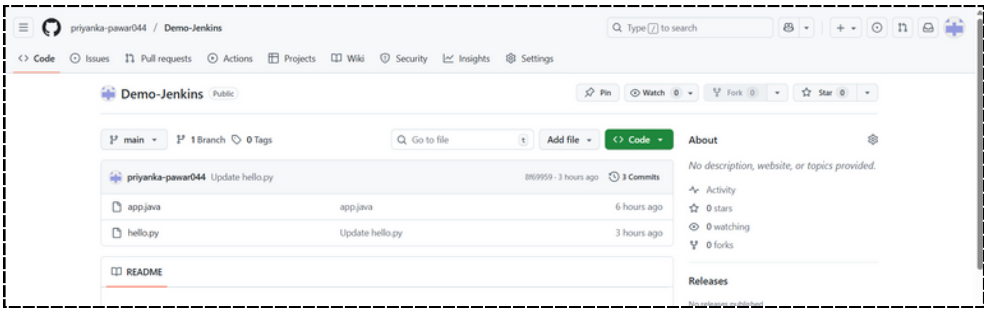
The new token will be displayed.



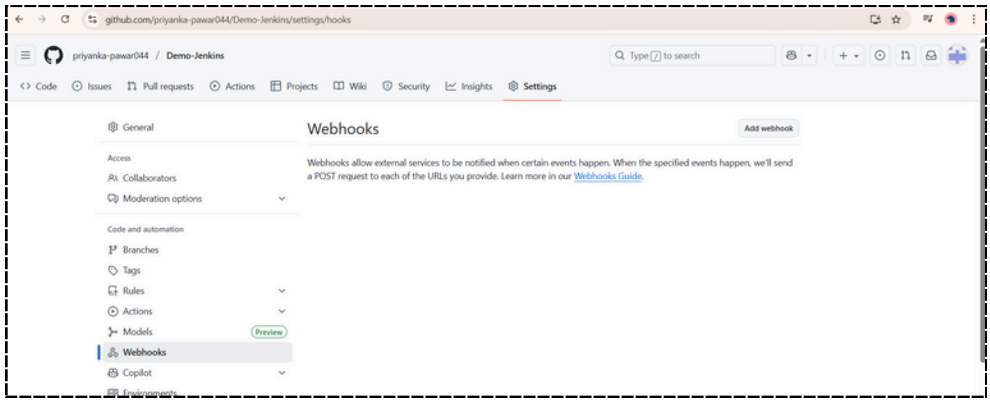
Copy this token immediately and save it in a secure location. For security purposes, Jenkins will not show you this token again.

Configure Your GitHub Repository

Go to your GitHub repository and navigate to Settings > Webhooks.



Add Webhooks

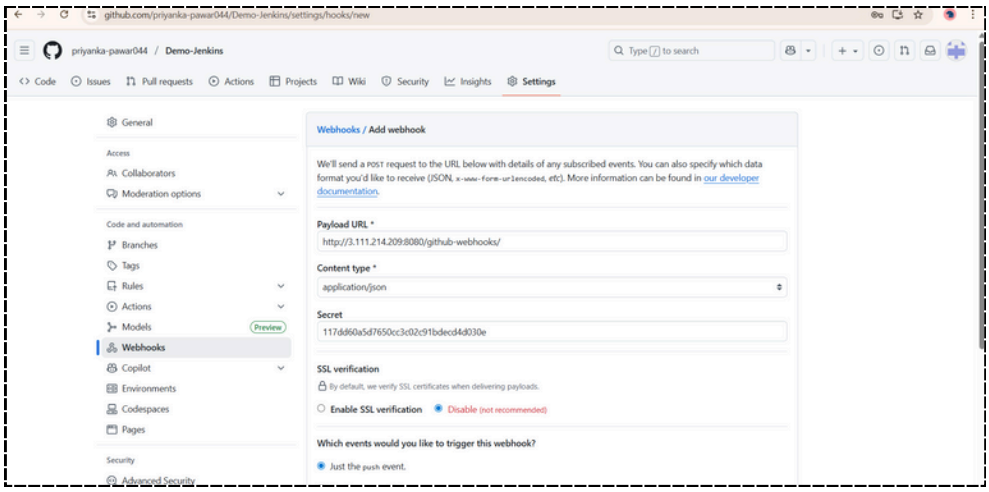


JENKINS

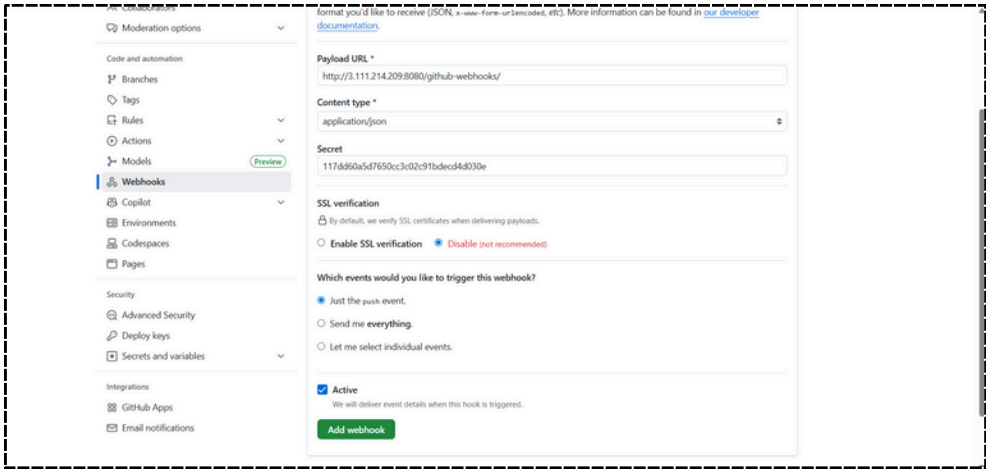
Enter Payload URL:
http://<your-jenkins-server>:8080/github-webhook/
(Make sure /github-webhook/ is added)

Content type → application/json

Secret add jenkins tocken



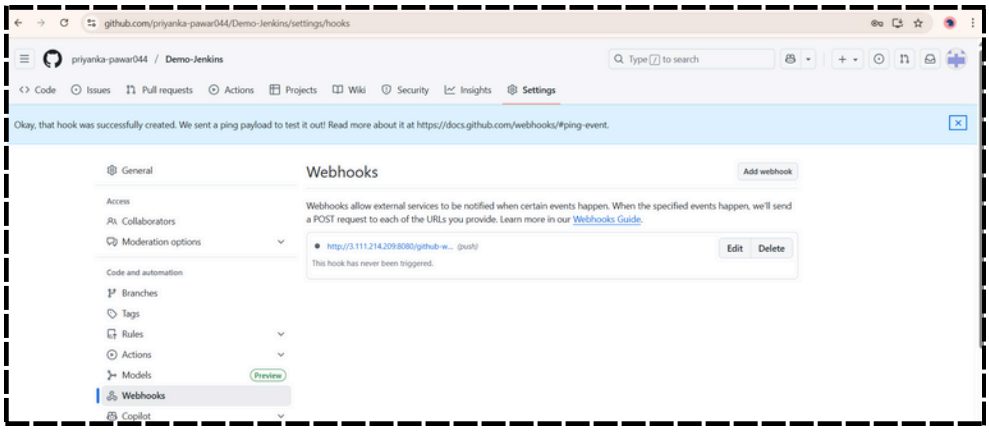
Choose when to trigger:
"Just the push event" (common)
Or "Send me everything" (for PRs, issues, etc.)

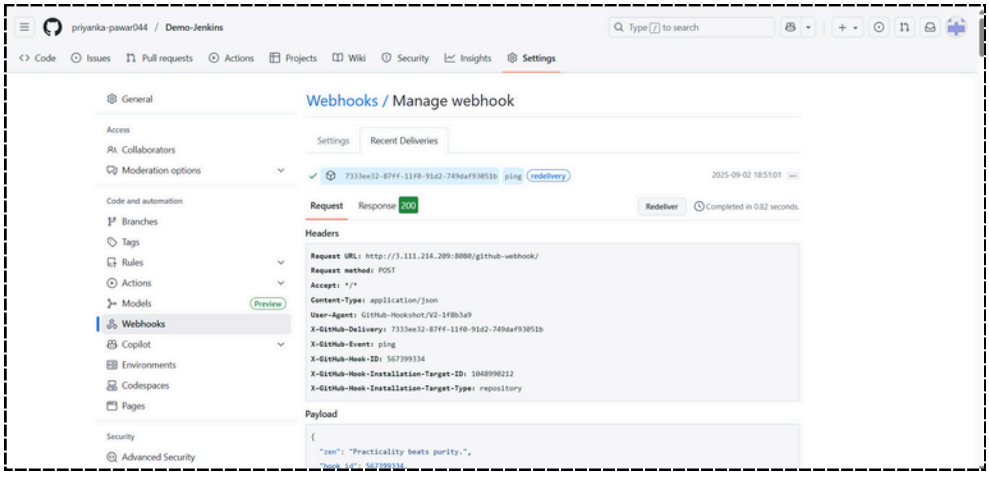


Click Add Webhook

JENKINS

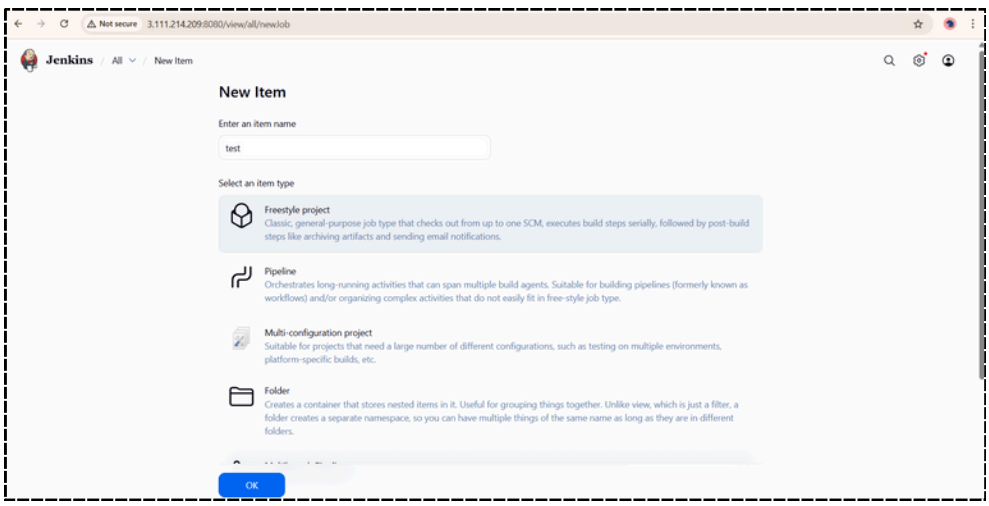
That Green Shows The Webhook is configured successfully.





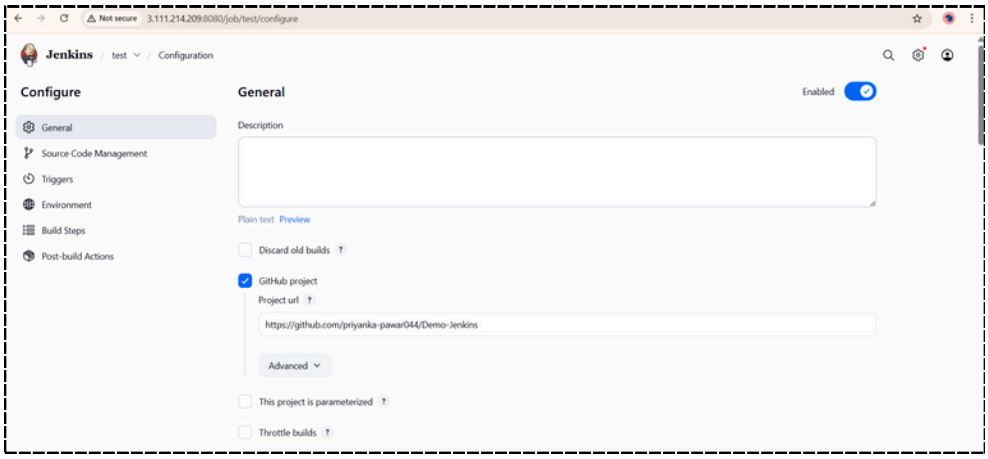
Configure Your Jenkins Job

create a new Freestyle project



JENKINS

Under Source Code Management, select Git and enter repo URL



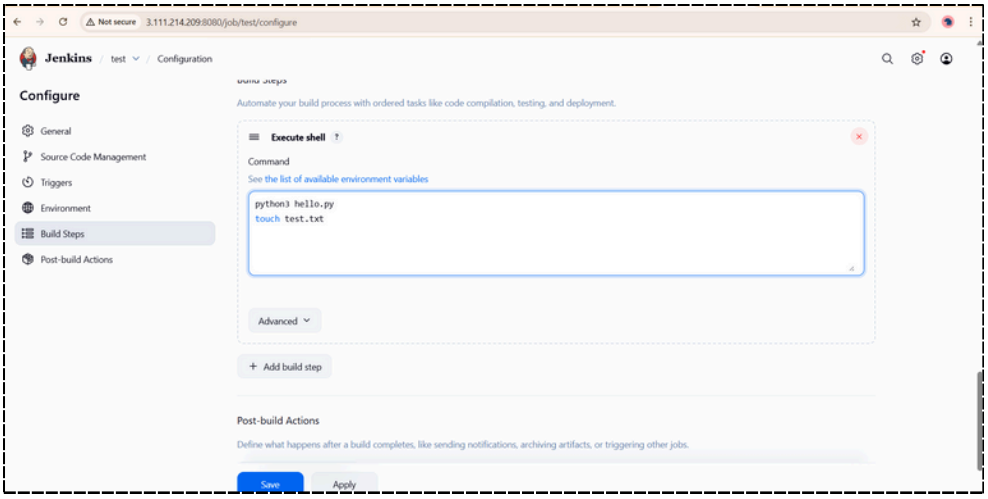
Source Code Manager add Git Repository URL

Select Branches to build

JENKINS

Go to Build Triggers → check GitHub hook trigger for GITScm polling

Add your build steps (Maven, Gradle, Shell, etc.)



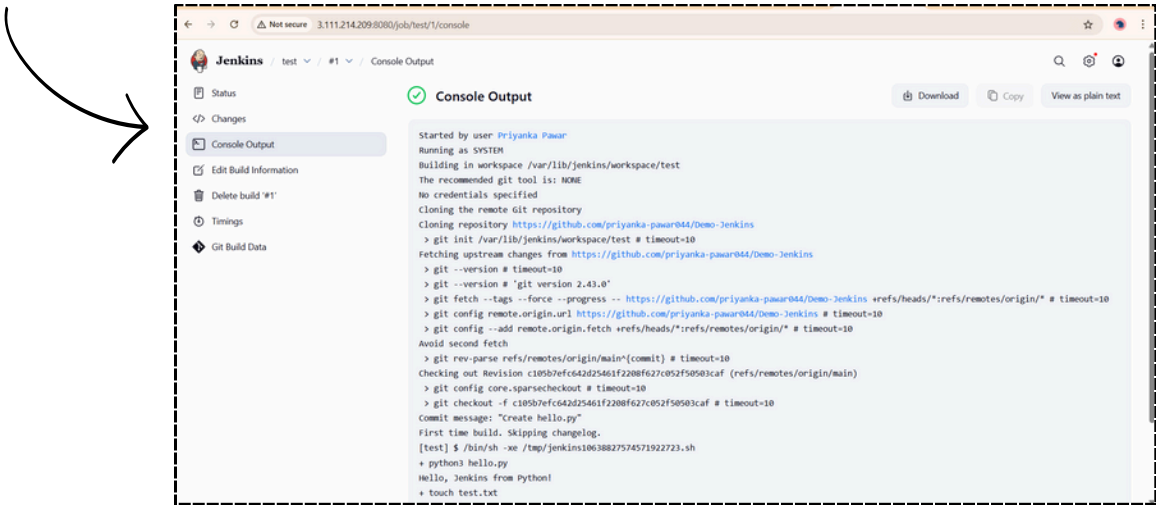
Save the job configuration.

JENKINS

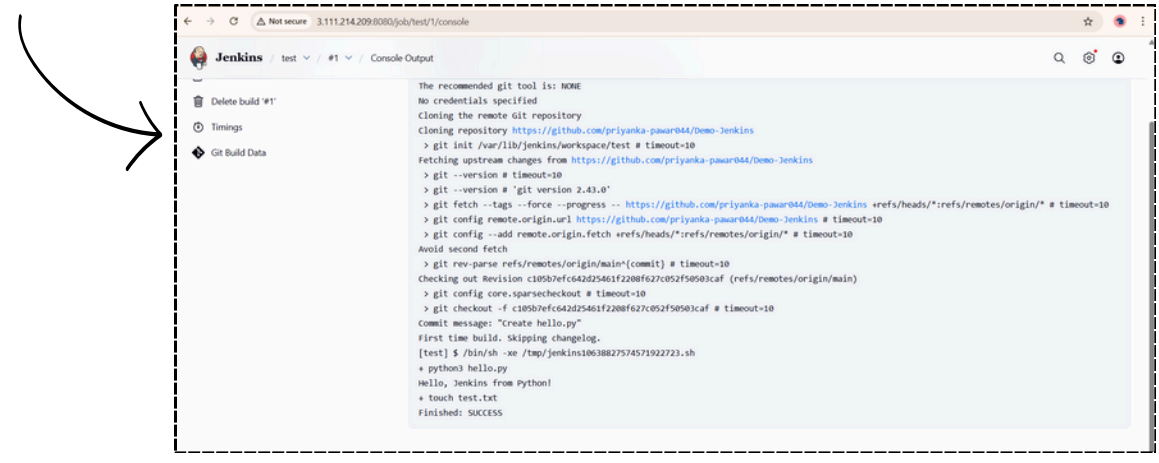
Build the first job manually

Click on build now

Check Console Output

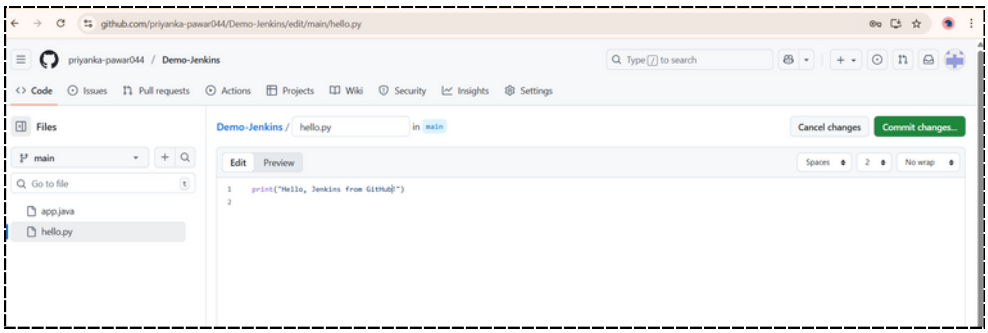


Build is Successfully Finished

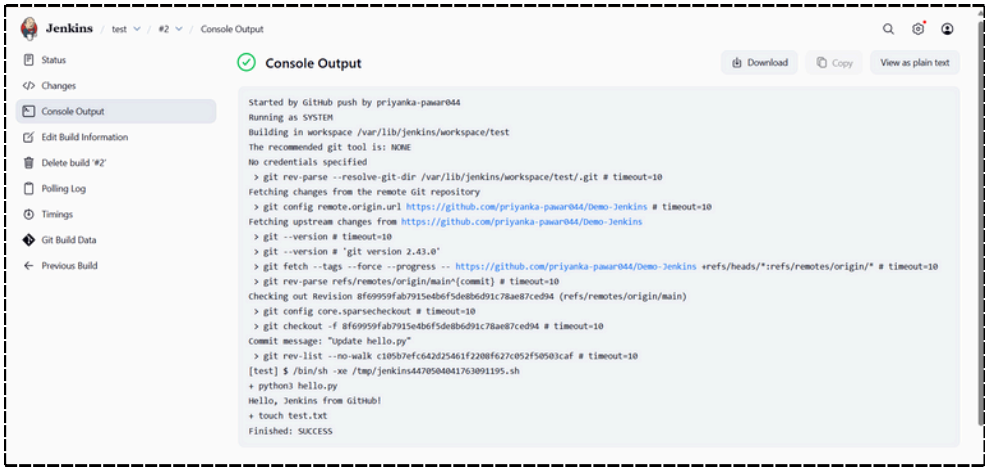


JENKINS

Test make changes in repo and commit changes



It will trigger changes and build job again



Build is Successfully Finished

✅ Conclusion

Integrating Jenkins with GitHub webhooks establishes an automated, event-driven CI/CD pipeline. Instead of Jenkins constantly polling GitHub, webhooks notify Jenkins of any changes in real time. This reduces resource usage, speeds up feedback, and ensures reliable, automated builds.
Such integration is a fundamental step towards achieving DevOps best practices, where collaboration, automation, and continuous delivery accelerate the software development lifecycle.


Thank
You