LAB ACTION PLAN FOR WEEK 11

Jenkins-CI/CD

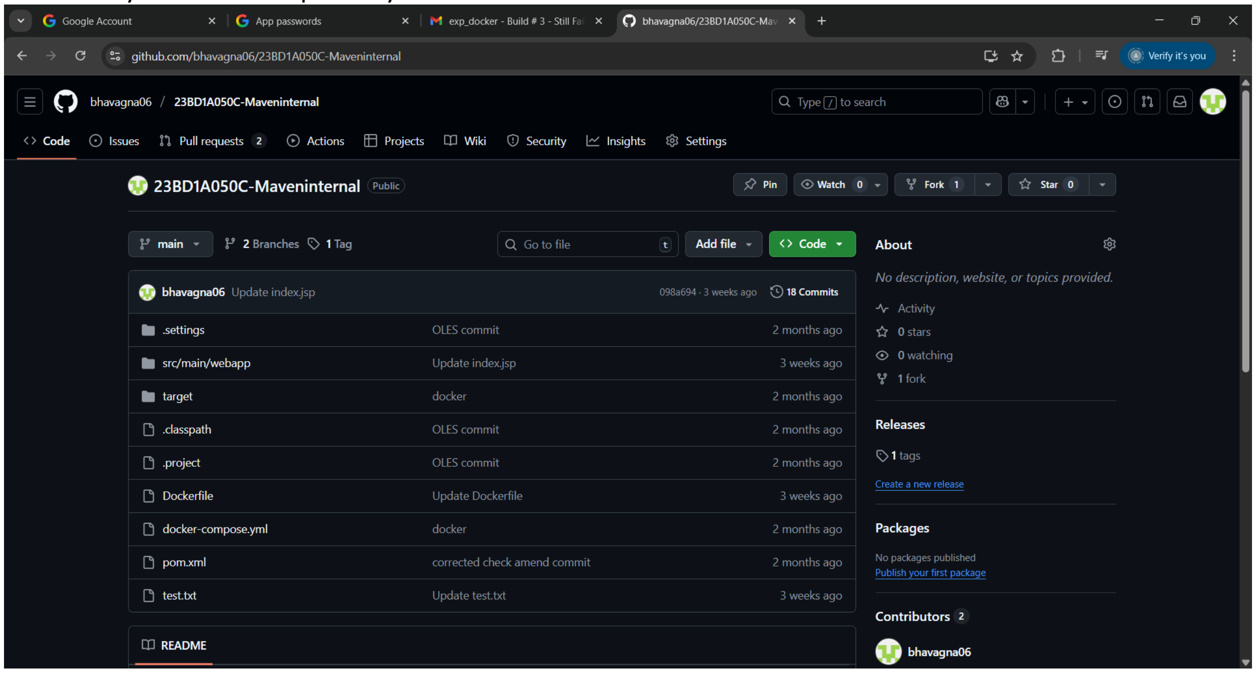
1. CI-Continous Integration using Webhooks .
2. Sending E-mail Notification on Build Failure or success
3. Upload the screenshots for the tasks

Lab

Setting Up Jenkins CI------using GitHub Webhook with Jenkins

Step 1: Configure Webhook in GitHub

1. Go to your GitHub repository.



github.com/bhavagna06/23BD1A050C-Maveninternal

bhavagna06 / 23BD1A050C-Maveninternal

23BD1A050C-Maveninternal Public

main · 2 Branches · 1 Tag

bhavagna06 Update index.jsp · 098a694 · 3 weeks ago · 18 Commits

.settings	OLES commit	2 months ago
src/main/webapp	Update index.jsp	3 weeks ago
target	docker	2 months ago
.classpath	OLES commit	2 months ago
.project	OLES commit	2 months ago
Dockerfile	Update Dockerfile	3 weeks ago
docker-compose.yml	docker	2 months ago
pom.xml	corrected check amend commit	2 months ago
test.txt	Update test.txt	3 weeks ago

README

About

No description, website, or topics provided.

Activity
0 stars
0 watching
1 fork

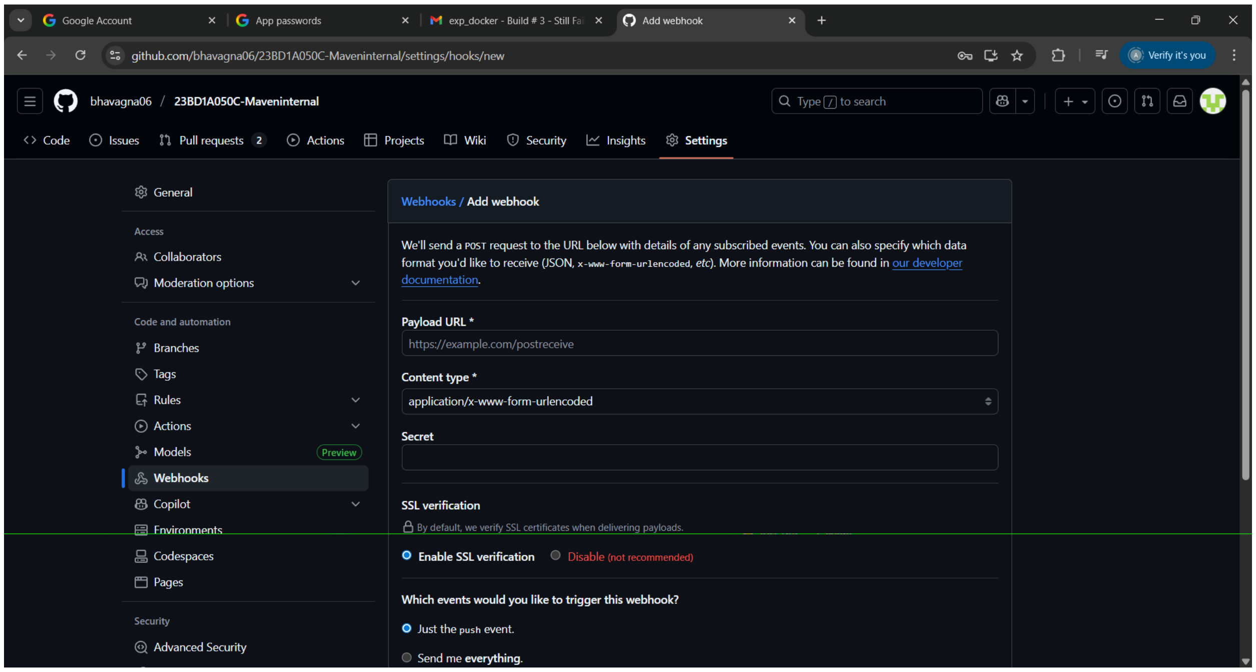
Releases
1 tags
Create a new release

Packages
No packages published
Publish your first package

Contributors 2
bhavagna06

2. Navigate to Settings → **Webhooks.**

3. Click “**Add webhook**”.



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 *

https://example.com/postreceive

Content type *

application/x-www-form-urlencoded

Secret

SSL verification

By default, we verify SSL certificates when delivering payloads.

[✓ Enable SSL verification] Disable (not recommended)

Which events would you like to trigger this webhook?

[✓ Just the `push` event.]

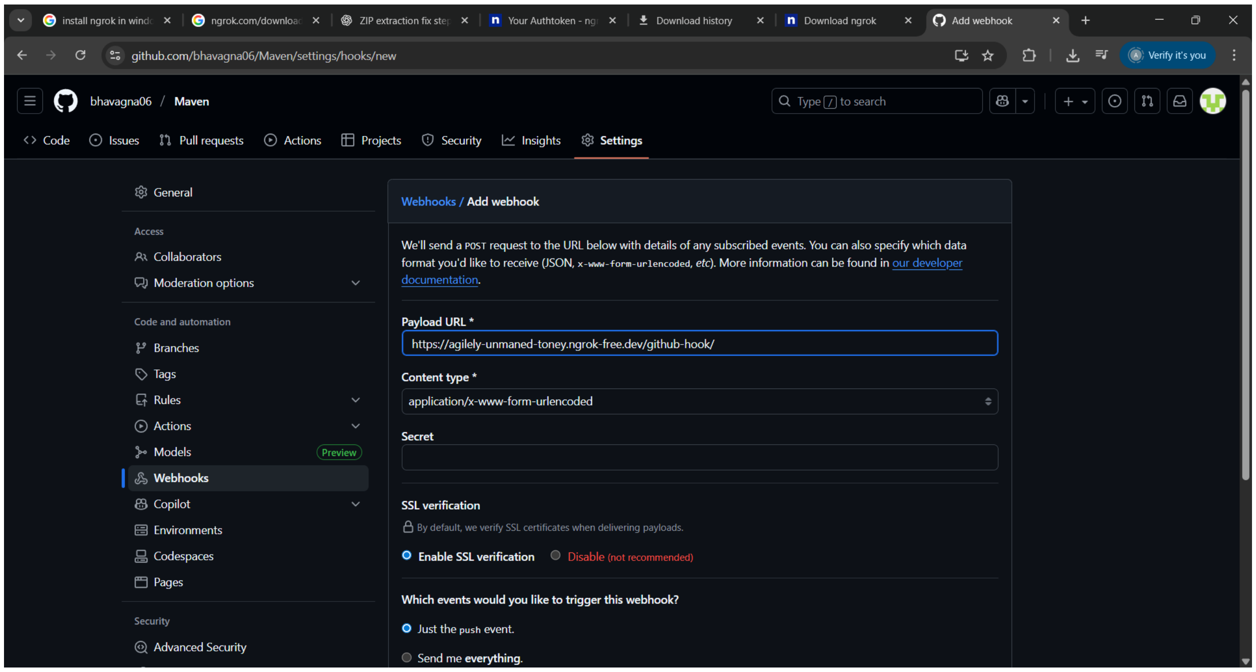
Send me **everything**.

4. In the Payload URL field:
   - Enter the Jenkins webhook URL in the format: http://<**jenkins-server-url**>/github-webhook/

Note: If Jenkins is running on localhost, GitHub cannot access it directly.
Use ngrok to expose your local Jenkins to the internet:

   - ngrok.exe http <**Jenkins local host:8080**>
     - Use the generated ngrok URL, e.g.:
     - http://abc123.ngrok.io/**github-webhook**/
5. Set Content type to:
   application/json
6. Under "Which events would you like to trigger this webhook?", select:
   - Just the push event

7.



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 *
https://agilely-unmaned-toney.ngrok-free.dev/github-hook/

Content type *
application/x-www-form-urlencoded

Secret

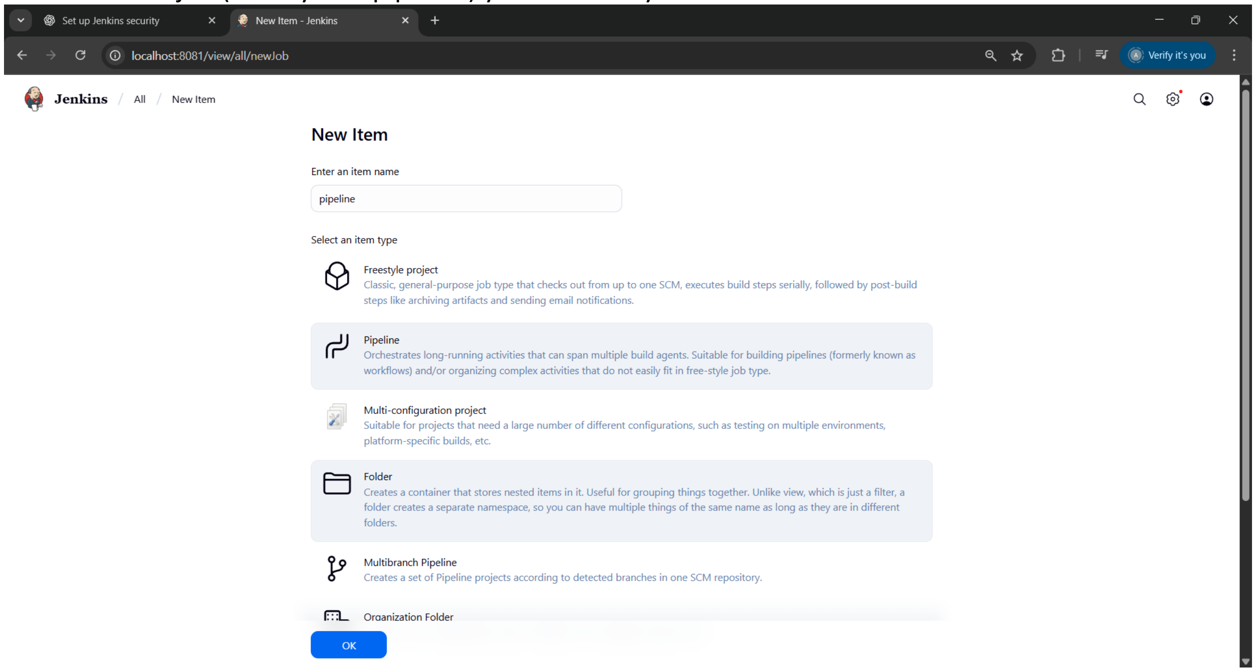
SSL verification
By default, we verify SSL certificates when delivering payloads.
[✓ Enable SSL verification] Disable (not recommended)

Which events would you like to trigger this webhook?
[✓ Just the push event.]
Send me everything.

Click "Add webhook" to save.

Step 2: Configure Jenkins to Accept GitHub Webhooks

1. Open Jenkins Dashboard.
2. Select the job (freestyle or pipeline) you've already created.



New Item

Enter an item name
pipeline

Select an item type

Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

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.

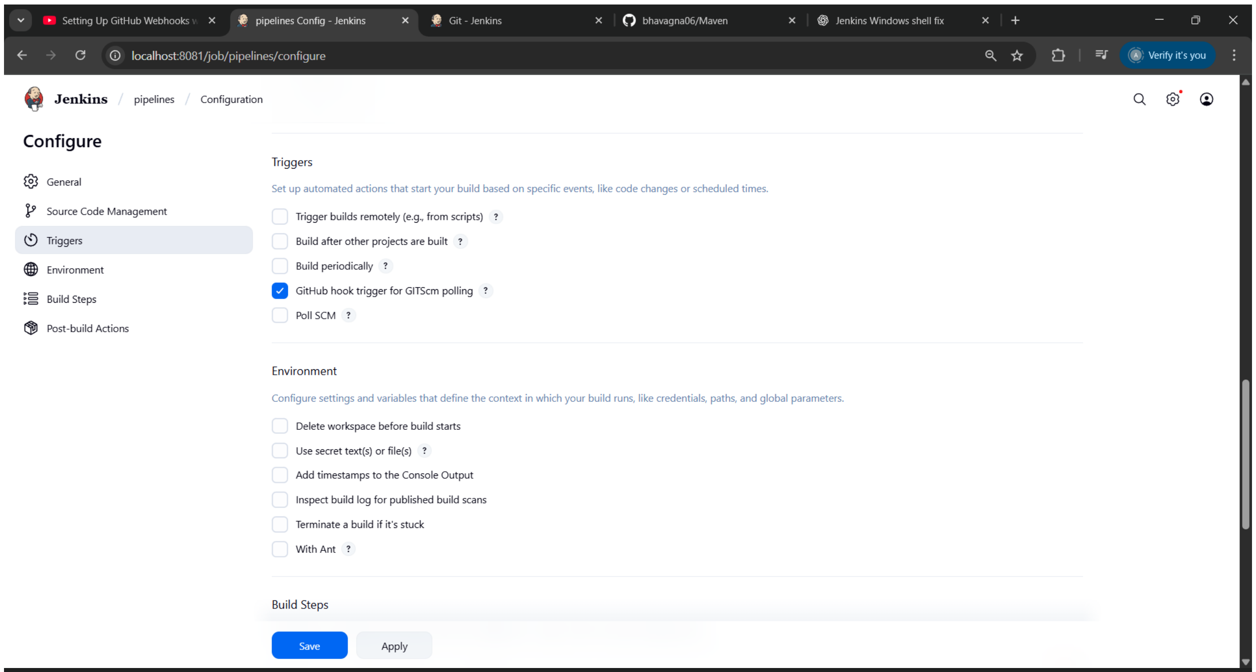
Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

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

3. Click Configure.
4. Scroll down to the Build Triggers section.
5. Check the box: ✅GitHub hook trigger for GITScm polling
6. Click Save.



Triggers

Set up automated actions that start your build based on specific events, like code changes or scheduled times.

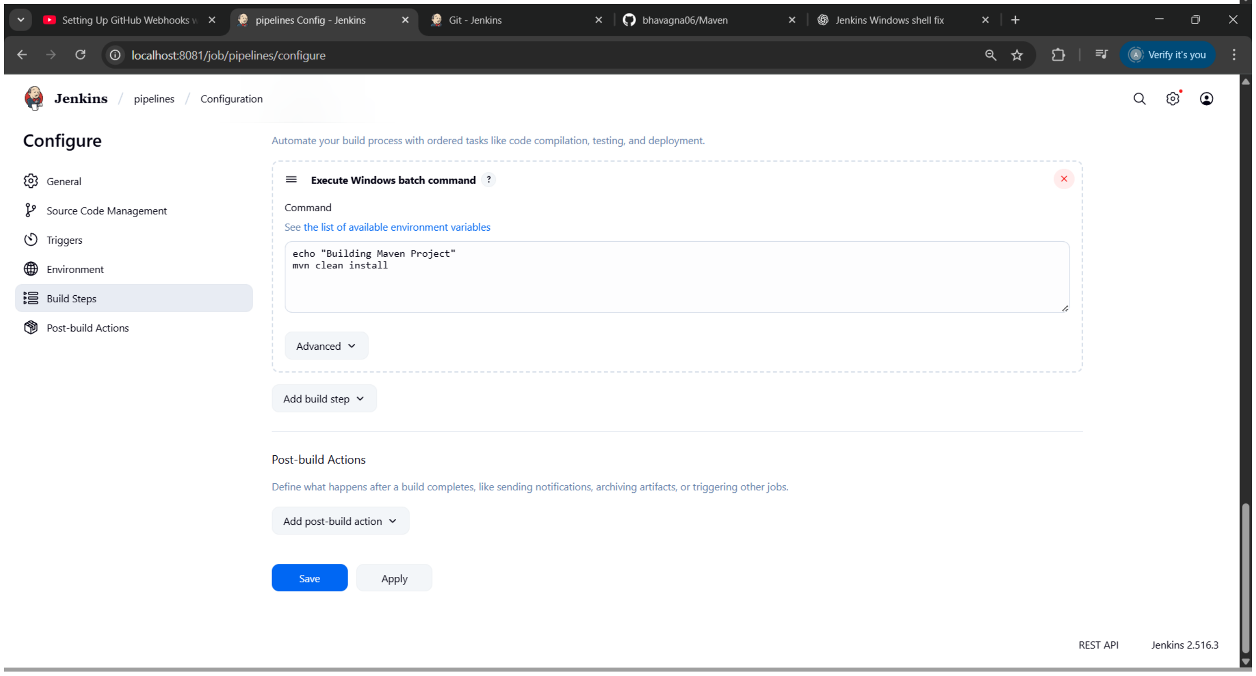
- [ ] Trigger builds remotely (e.g., from scripts)
- [ ] Build after other projects are built
- [ ] Build periodically
- [x] GitHub hook trigger for GITScm polling
- [ ] Poll SCM

Environment

Configure settings and variables that define the context in which your build runs, like credentials, paths, and global parameters.

- [ ] Delete workspace before build starts
- [ ] Use secret text(s) or file(s)
- [ ] Add timestamps to the Console Output
- [ ] Inspect build log for published build scans
- [ ] Terminate a build if it's stuck
- [ ] With Ant

Build Steps



Automate your build process with ordered tasks like code compilation, testing, and deployment.

Execute Windows batch command

Command

See the list of available environment variables

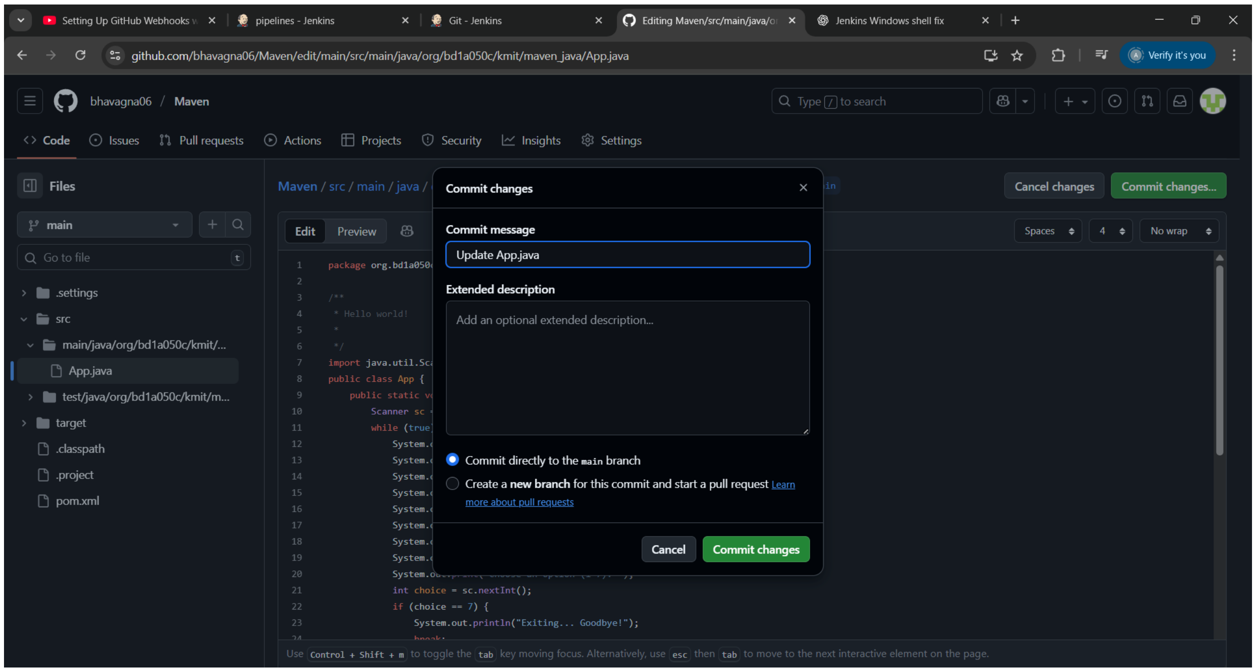
```
echo "Building Maven Project"
mvn clean install
```

Post-build Actions

Define what happens after a build completes, like sending notifications, archiving artifacts, or triggering other jobs.

Step 3: Test the Setup

1. Make any code update in your local repo and push it to GitHub.
2. Once pushed, GitHub will trigger the webhook.
3. Jenkins will automatically detect the change and start the build pipeline.



github.com/bhavagna06/Maven/edit/main/src/main/java/org/bd1a050c/kmit/maven_java/App.java

bhavagna06 / Maven

Maven / src / main / java /

Commit changes

Commit message

Update App.java

Extended description

Add an optional extended description...

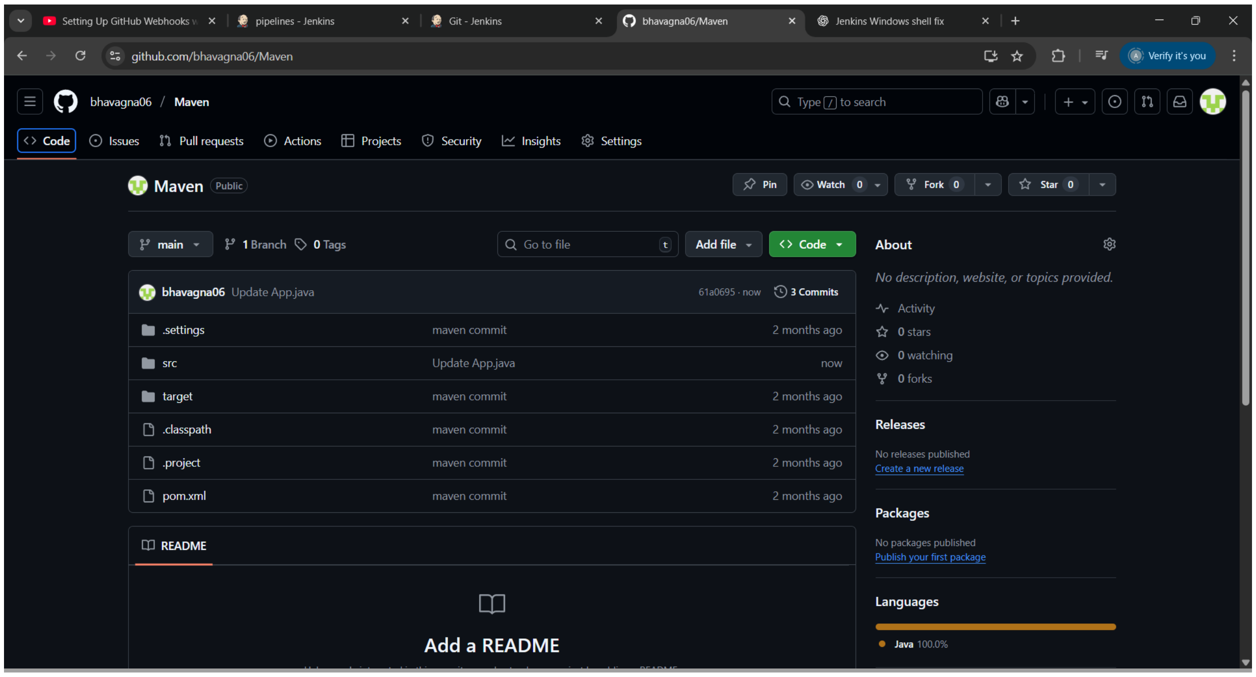
(●) Commit directly to the `main` branch

( ) Create a **new branch** for this commit and start a pull request Learn more about pull requests

Cancel | Commit changes

```
package org.bd1a050c
/**
 * Hello world!
 *
 */
import java.util.Sc
public class App {
    public static v
        Scanner sc
        while (true
            System.
            System.
            System.
            System.
            System.
            System.
            System.
            System.
            System.out.print("Choose an option (1-7): ");
            int choice = sc.nextInt();
            if (choice == 7) {
                System.out.println("Exiting... Goodbye!");
                break;
```

Use `Control + Shift + m` to toggle the `tab` key moving focus. Alternatively, use `esc` then `tab` to move to the next interactive element on the page.



github.com/bhavagna06/Maven

bhavagna06 / Maven

Maven Public

main | 1 Branch | 0 Tags

bhavagna06 Update App.java — 61a0695 · now — 3 Commits

.settings	maven commit	2 months ago
src	Update App.java	now
target	maven commit	2 months ago
.classpath	maven commit	2 months ago
.project	maven commit	2 months ago
pom.xml	maven commit	2 months ago

README

Add a README

About

No description, website, or topics provided.

Activity
0 stars
0 watching
0 forks

Releases

No releases published
Create a new release

Packages

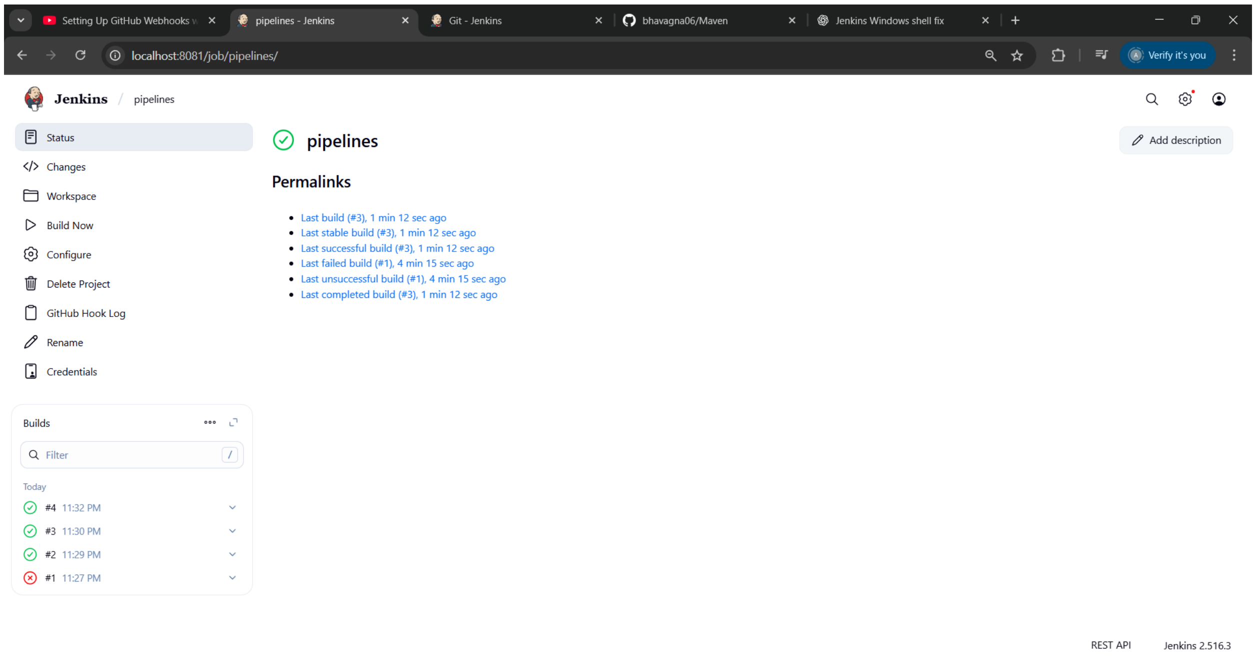
No packages published
Publish your first package

Languages

Java 100.0%

outcome

- You've successfully connected GitHub and Jenkins using webhooks.
- Every time you push code to GitHub, Jenkins will automatically start building your project without manual intervention.

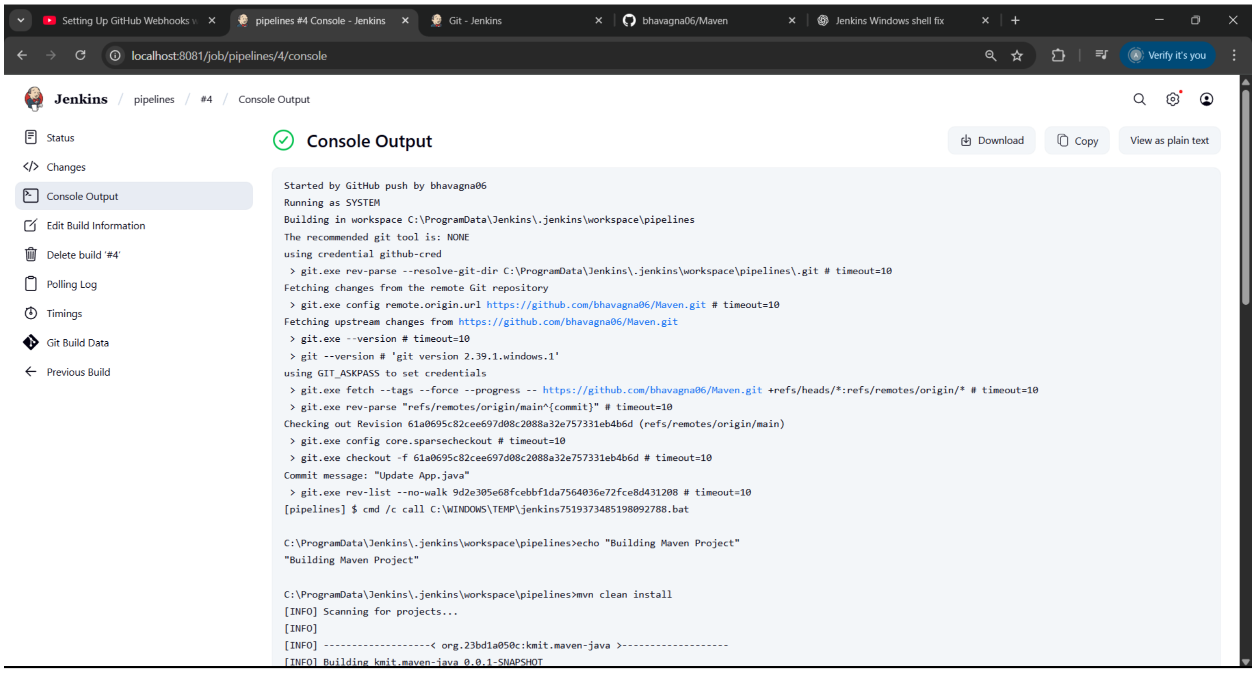


pipelines

Permalinks

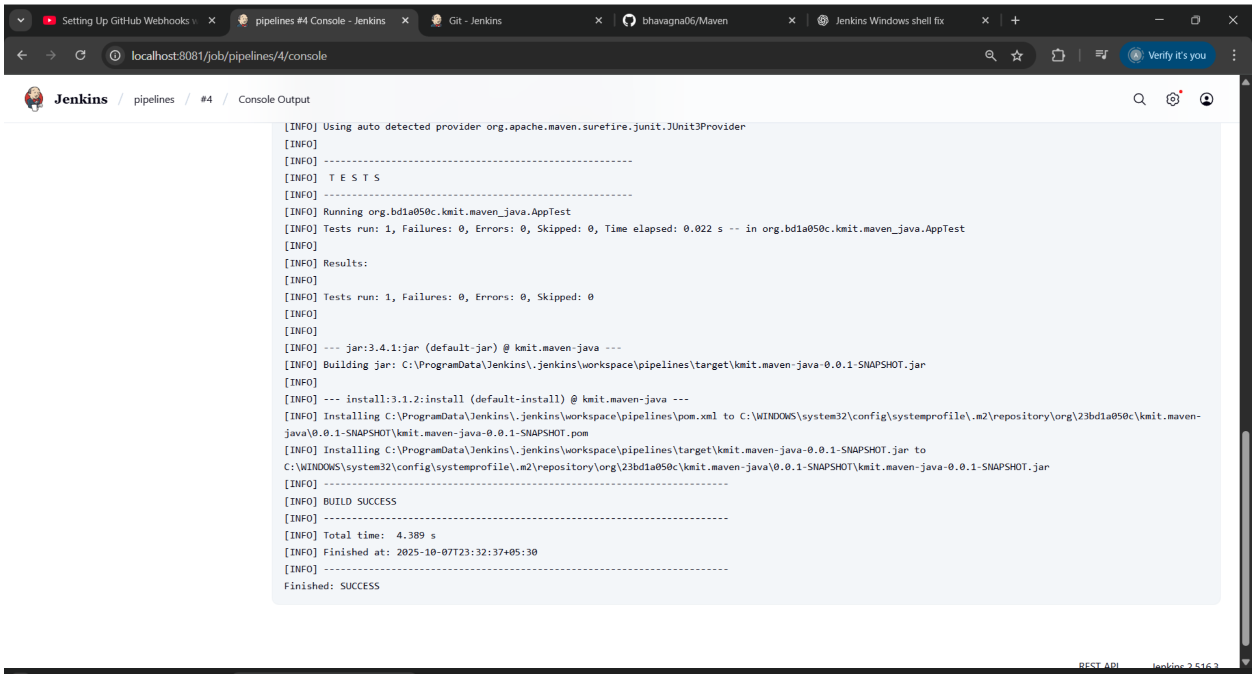
- Last build (#3), 1 min 12 sec ago
- Last stable build (#3), 1 min 12 sec ago
- Last successful build (#3), 1 min 12 sec ago
- Last failed build (#1), 4 min 15 sec ago
- Last unsuccessful build (#1), 4 min 15 sec ago
- Last completed build (#3), 1 min 12 sec ago

```
#4  11:32 PM
#3  11:30 PM
#2  11:29 PM
#1  11:27 PM
```

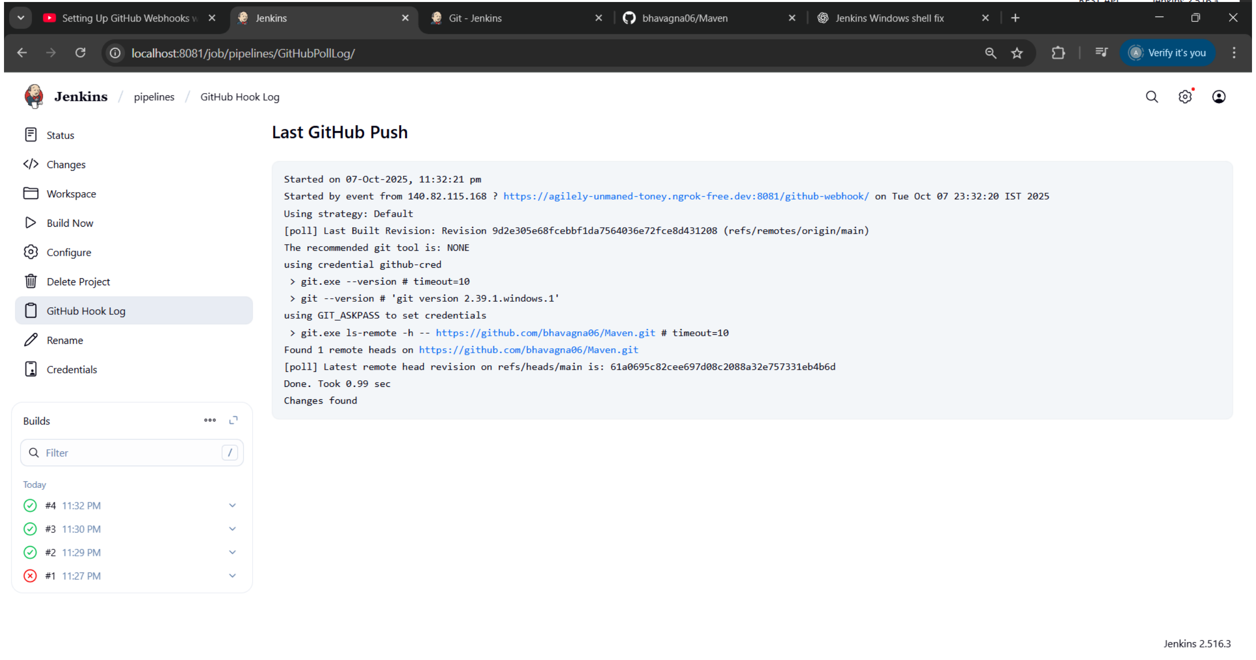


Console Output

```
Started by GitHub push by bhavagna06
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\.jenkins\workspace\pipelines
The recommended git tool is: NONE
using credential github-cred
 > git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\pipelines\.git # timeout=10
Fetching changes from the remote Git repository
 > git.exe config remote.origin.url https://github.com/bhavagna06/Maven.git # timeout=10
Fetching upstream changes from https://github.com/bhavagna06/Maven.git
 > git.exe --version # timeout=10
 > git --version # 'git version 2.39.1.windows.1'
using GIT_ASKPASS to set credentials
 > git.exe fetch --tags --force --progress -- https://github.com/bhavagna06/Maven.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git.exe rev-parse "refs/remotes/origin/main^{commit}" # timeout=10
Checking out Revision 61a0695c82cee697d08c2088a32e757331eb4b6d (refs/remotes/origin/main)
 > git.exe config core.sparsecheckout # timeout=10
 > git.exe checkout -f 61a0695c82cee697d08c2088a32e757331eb4b6d # timeout=10
Commit message: "Update App.java"
 > git.exe rev-list --no-walk 9d2e305e68fcebbf1da7564036e72fce8d431208 # timeout=10
[pipelines] $ cmd /c call C:\WINDOWS\TEMP\jenkins7519373485198092788.bat

C:\ProgramData\Jenkins\.jenkins\workspace\pipelines>echo "Building Maven Project"
"Building Maven Project"

C:\ProgramData\Jenkins\.jenkins\workspace\pipelines>mvn clean install
[INFO] Scanning for projects...
[INFO]
[INFO] ------------------< org.23bd1a050c:kmit.maven-java >-------------------
[INFO] Building kmit.maven-java 0.0.1-SNAPSHOT
```


Jenkins / pipelines / #4 / Console Output
localhost:8081/job/pipelines/4/console

```
[INFO] Using auto detected provider org.apache.maven.surefire.junit.JUnit3Provider
[INFO]
[INFO] -------------------------------------------------------
[INFO]  T E S T S
[INFO] -------------------------------------------------------
[INFO] Running org.bd1a050c.kmit.maven_java.AppTest
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.022 s -- in org.bd1a050c.kmit.maven_java.AppTest
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO]
[INFO] --- jar:3.4.1:jar (default-jar) @ kmit.maven-java ---
[INFO] Building jar: C:\ProgramData\Jenkins\.jenkins\workspace\pipelines\target\kmit.maven-java-0.0.1-SNAPSHOT.jar
[INFO]
[INFO] --- install:3.1.2:install (default-install) @ kmit.maven-java ---
[INFO] Installing C:\ProgramData\Jenkins\.jenkins\workspace\pipelines\pom.xml to C:\WINDOWS\system32\config\systemprofile\.m2\repository\org\23bd1a050c\kmit.maven-
java\0.0.1-SNAPSHOT\kmit.maven-java-0.0.1-SNAPSHOT.pom
[INFO] Installing C:\ProgramData\Jenkins\.jenkins\workspace\pipelines\target\kmit.maven-java-0.0.1-SNAPSHOT.jar to
C:\WINDOWS\system32\config\systemprofile\.m2\repository\org\23bd1a050c\kmit.maven-java\0.0.1-SNAPSHOT\kmit.maven-java-0.0.1-SNAPSHOT.jar
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  4.389 s
[INFO] Finished at: 2025-10-07T23:32:37+05:30
[INFO] ------------------------------------------------------------------------
Finished: SUCCESS
```



Last GitHub Push

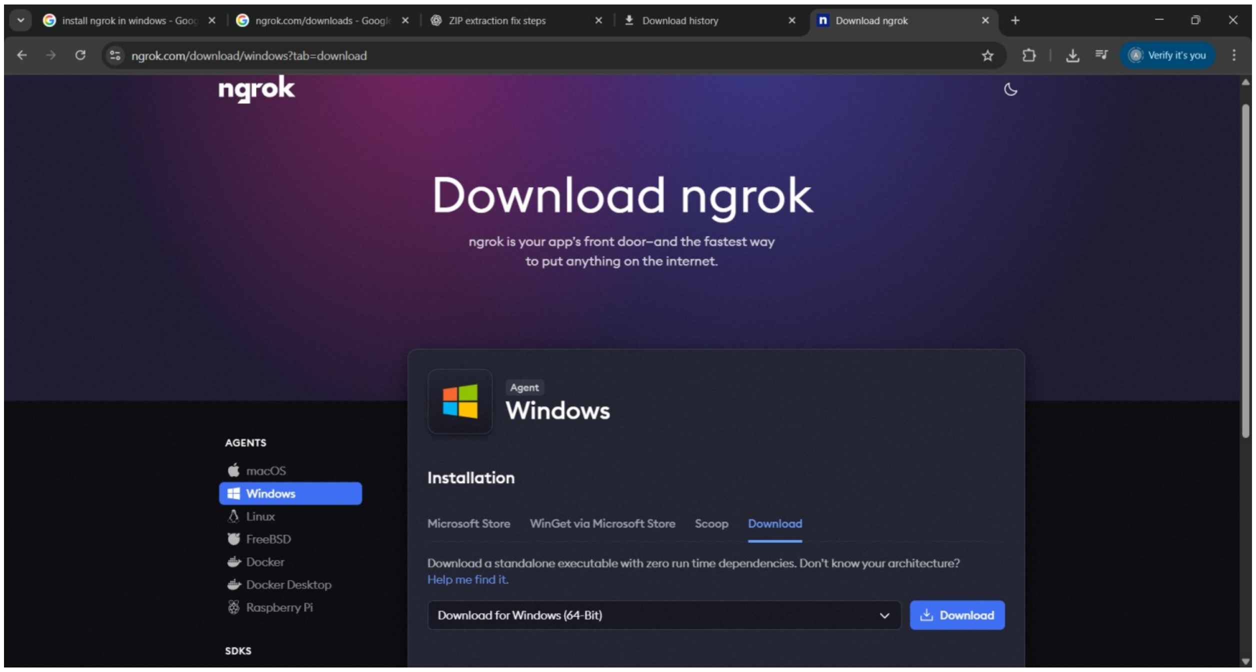
```
Started on 07-Oct-2025, 11:32:21 pm
Started by event from 140.82.115.168 ? https://agilely-unmaned-toney.ngrok-free.dev:8081/github-webhook/ on Tue Oct 07 23:32:20 IST 2025
Using strategy: Default
[poll] Last Built Revision: Revision 9d2e305e68fcebbf1da7564036e72fce8d431208 (refs/remotes/origin/main)
The recommended git tool is: NONE
using credential github-cred
 > git.exe --version # timeout=10
 > git --version # 'git version 2.39.1.windows.1'
using GIT_ASKPASS to set credentials
 > git.exe ls-remote -h -- https://github.com/bhavagna06/Maven.git # timeout=10
Found 1 remote heads on https://github.com/bhavagna06/Maven.git
[poll] Latest remote head revision on refs/heads/main is: 61a0695c82cee697d08c2088a32e757331eb4b6d
Done. Took 0.99 sec
Changes found
```

Set-uping the ngrok

How to Install and Use ngrok

Step 1. Download ngrok

https://ngrok.com/download
Download and extract it for your OS (Windows, macOS, or Linux).

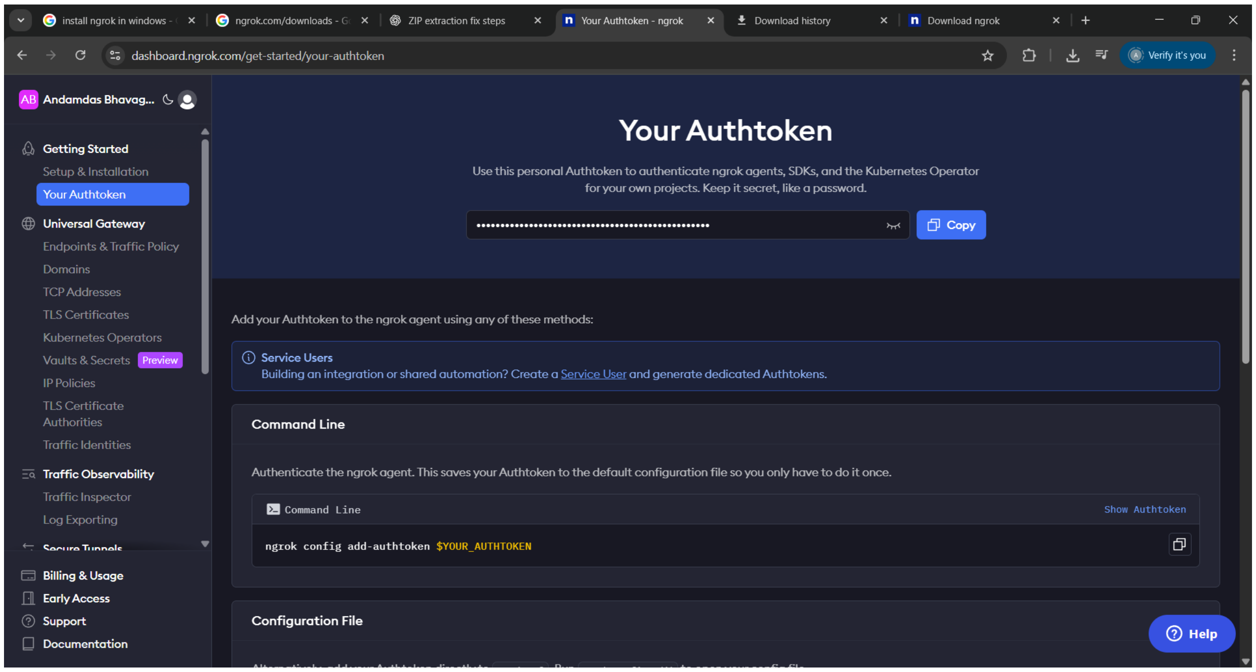


Download ngrok

ngrok is your app's front door—and the fastest way to put anything on the internet.

Download a standalone executable with zero run time dependencies. Don't know your architecture? Help me find it.

Step 2. Connect Your ngrok Account (optional but useful)
After you sign up (free), ngrok gives you an auth token.
CREATE AUTHENTICATOR [https://dashboard.ngrok.com/get-started/your-authtoken]
Run this command (replace <your_token> with yours):
ngrok config add-authtoken <your_token>
This ensures stable sessions and more control.



Your Authtoken

Use this personal Authtoken to authenticate ngrok agents, SDKs, and the Kubernetes Operator for your own projects. Keep it secret, like a password.

Add your Authtoken to the ngrok agent using any of these methods:

Service Users
Building an integration or shared automation? Create a Service User and generate dedicated Authtokens.

Command Line

Authenticate the ngrok agent. This saves your Authtoken to the default configuration file so you only have to do it once.

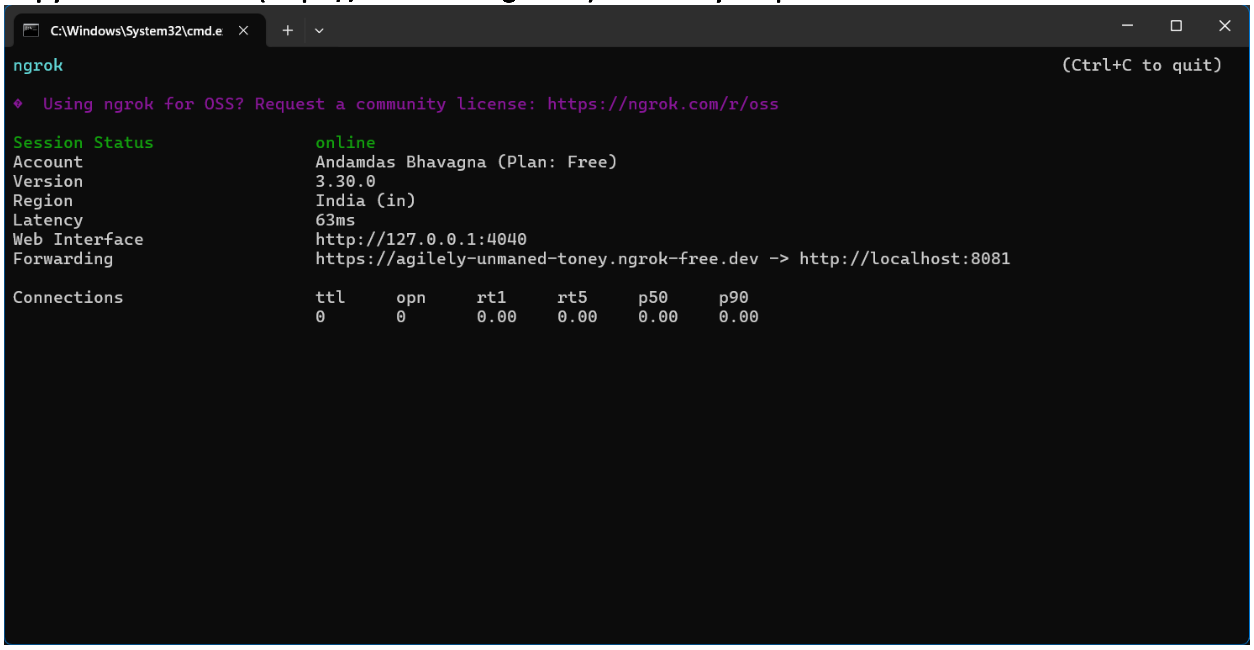
```
Command Line
ngrok config add-authtoken $YOUR_AUTHTOKEN
```

Configuration File

Step 3. Start a Tunnel for Jenkins

Assuming Jenkins runs locally on port 8085:
ngrok http 8085
You'll see output like:

Session Status online
Forwarding https://1234abcd.ngrok.io -> http://localhost:8080
Copy the HTTPS URL (https://1234abcd.ngrok.io) — this is your public Jenkins URL for webhooks.



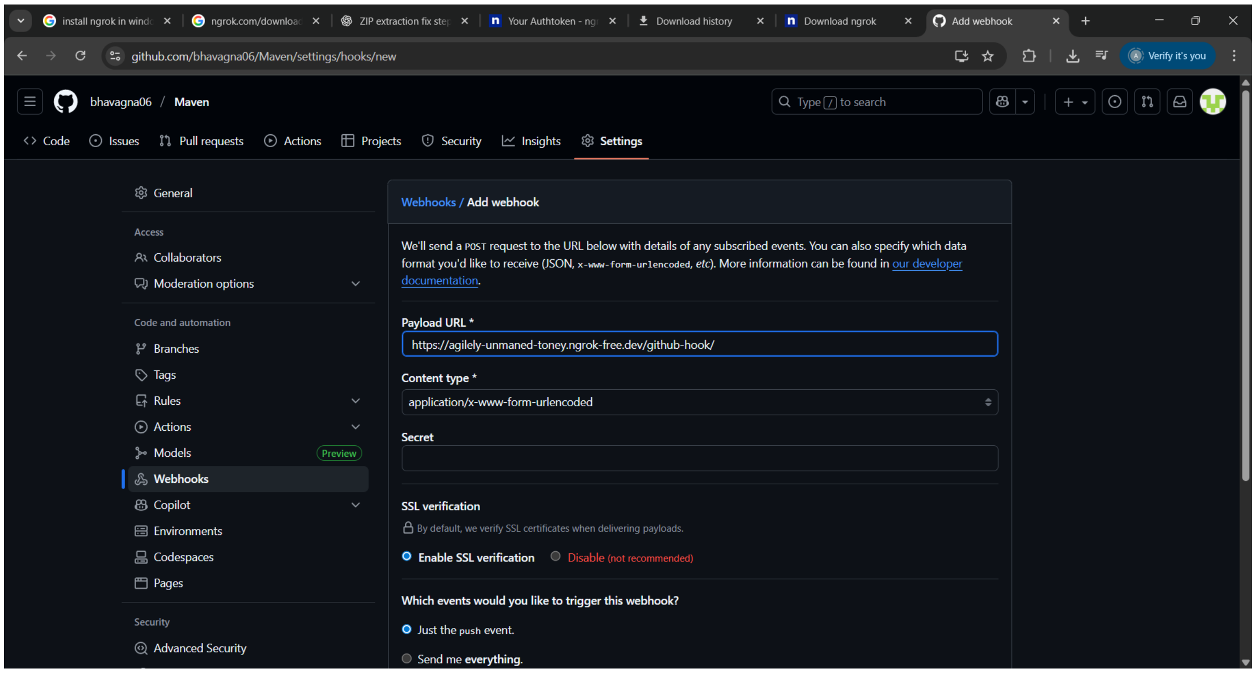
```
ngrok                                                                   (Ctrl+C to quit)

❖ Using ngrok for OSS? Request a community license: https://ngrok.com/r/oss

Session Status                online
Account                       Andamdas Bhavagna (Plan: Free)
Version                       3.30.0
Region                        India (in)
Latency                       63ms
Web Interface                 http://127.0.0.1:4040
Forwarding                    https://agilely-unmaned-toney.ngrok-free.dev -> http://localhost:8081

Connections                   ttl     opn     rt1     rt5     p50     p90
                              0       0       0.00    0.00    0.00    0.00
```

Step 4. Use it in GitHub Webhook

In your GitHub repo → Settings → Webhooks:

- **Payload URL:** ***[paste the url generated by ngrok]***

https://1234abcd.ngrok.io/github-webhook/ [please include this – remaining all default]
Now, whenever you push code, GitHub sends an event to that URL, which ngrok forwards to your local Jenkins.



github.com/bhavagna06/Maven/settings/hooks/new

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 *
https://agilely-unmaned-toney.ngrok-free.dev/github-hook/

Content type *
application/x-www-form-urlencoded

Secret

SSL verification
By default, we verify SSL certificates when delivering payloads.
[✓ Enable SSL verification] Disable (not recommended)

Which events would you like to trigger this webhook?
[✓ Just the push event.]
Send me everything.

Setting Up Jenkins Email Notification Setup (Using Gmail with App Password)

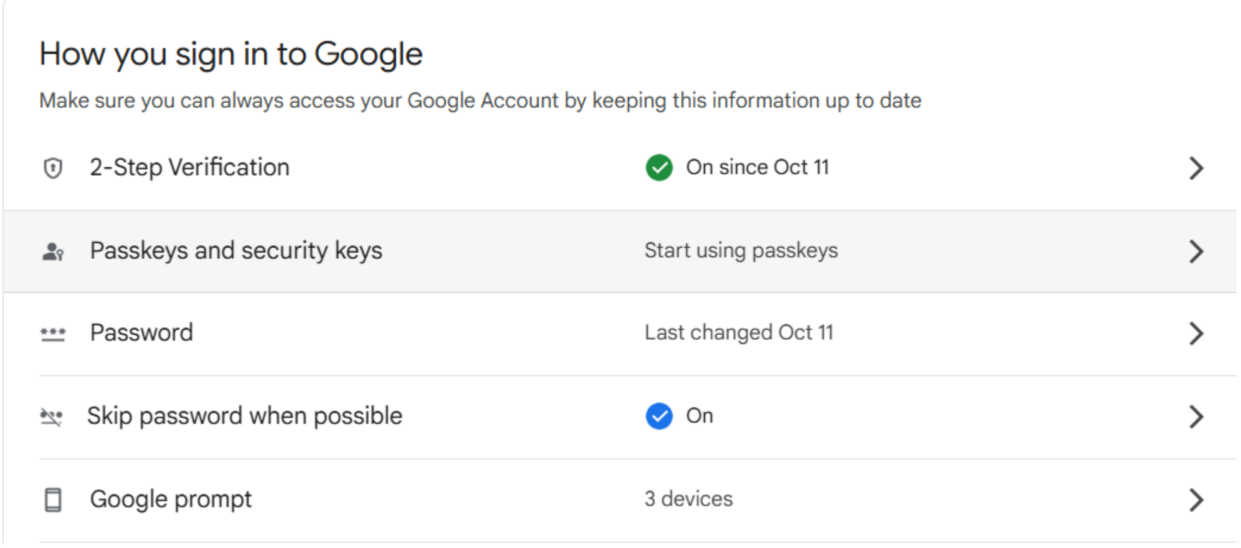
Creation of app password

1. Gmail: Enable App Password (for 2-Step Verification)

i. Go to: https://myaccount.google.com

ii. Enable 2-Step Verification

- Navigate to:
  - Security → 2-Step Verification
  - Turn it **ON**
  - Complete the OTP verification process (via phone/email)
  -



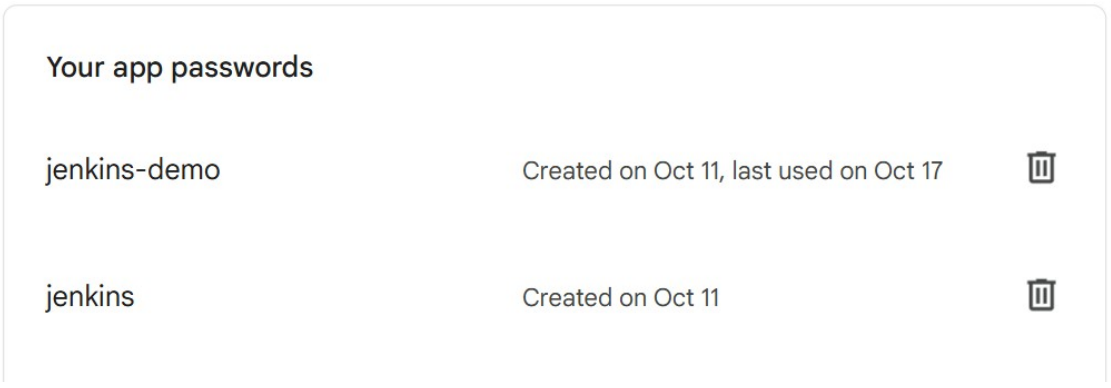
How you sign in to Google

Make sure you can always access your Google Account by keeping this information up to date

2-Step Verification	✓ On since Oct 11
Passkeys and security keys	Start using passkeys
Password	Last changed Oct 11
Skip password when possible	✓ On
Google prompt	3 devices

iii. Generate App Password for Jenkins

- Go to:
  - Security → App passwords
- Select:
  - **App**: Other (Custom name)
  - **Name**: Jenkins-Demo
- Click **Generate**
- Copy the **16-digit app password**
  - Save it in a secure location (e.g., Notepad)



Your app passwords

jenkins-demo	Created on Oct 11, last used on Oct 17
jenkins	Created on Oct 11

2. Jenkins Plugin Installation

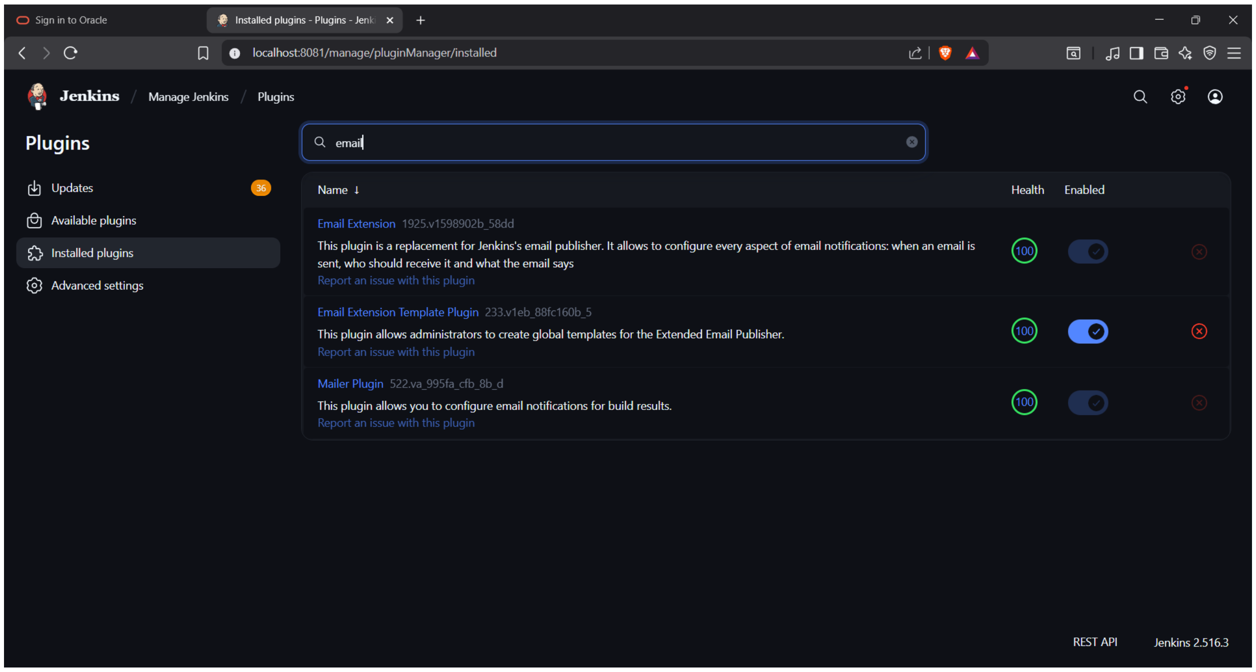
i. Open Jenkins Dashboard

ii. Navigate to:

- Manage Jenkins → Manage Plugins

iii. Install Plugin:

- Search for and install:
    - Email Extension Plugin



localhost:8081/manage/pluginManager/installed

Jenkins / Manage Jenkins / Plugins

Search: email

Name ↓	Health	Enabled
Email Extension 1925.v1598902b_58dd This plugin is a replacement for Jenkins's email publisher. It allows to configure every aspect of email notifications: when an email is sent, who should receive it and what the email says Report an issue with this plugin	100	✓
Email Extension Template Plugin 233.v1eb_88fc160b_5 This plugin allows administrators to create global templates for the Extended Email Publisher. Report an issue with this plugin	100	✓
Mailer Plugin 522.va_995fa_cfb_8b_d This plugin allows you to configure email notifications for build results. Report an issue with this plugin	100	✓

3. Configure Jenkins Global Email Settings

i. Go to:

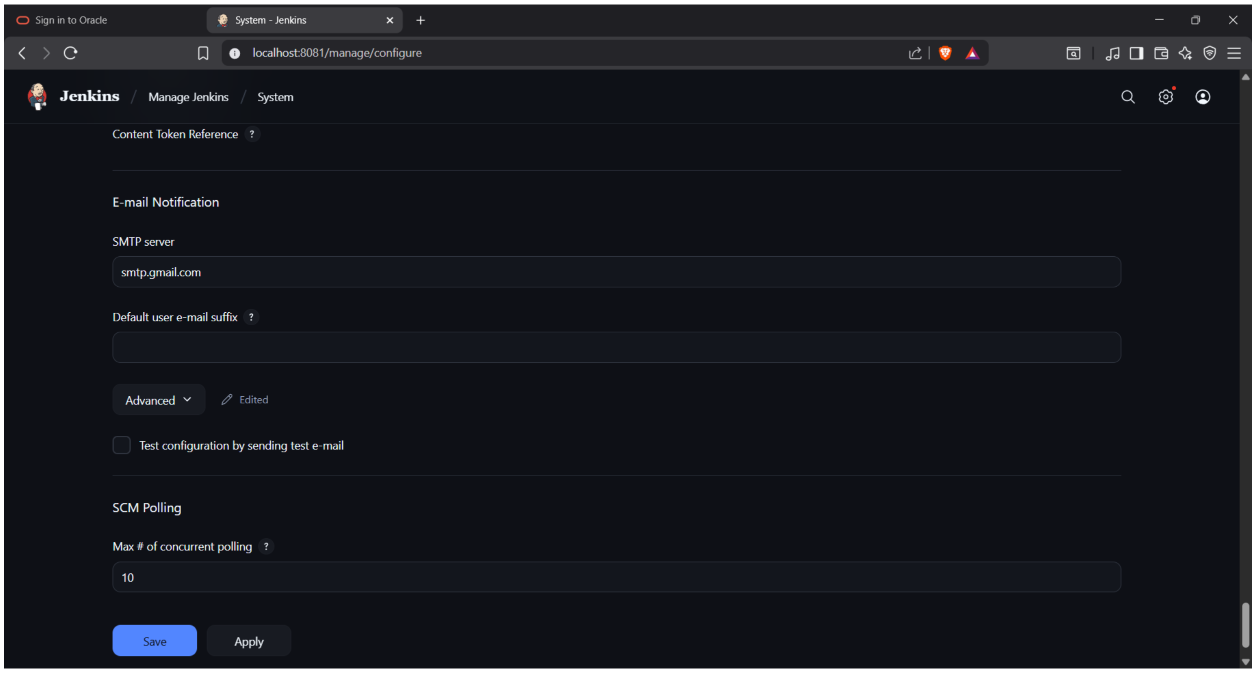
- Manage Jenkins → Configure System

A. E-mail Notification Section

Field	**Value**
SMTP Server	smtp.gmail.com
Use SMTP Auth	✅ Enabled
User Name	Your Gmail ID (e.g., archanareddykmit@gmail.com)
Password	Paste the 16-digit App Password
Use SSL	✅ Enabled
SMTP Port	465
Reply-To Address	Your Gmail ID (same as above)

➤ **Test Configuration**

- Click: Test configuration by sending test e-mail
- Provide a valid email address to receive a test mail
- ✅ Should receive email from Jenkins



E-mail Notification

SMTP server

smtp.gmail.com

Default user e-mail suffix

Test configuration by sending test e-mail

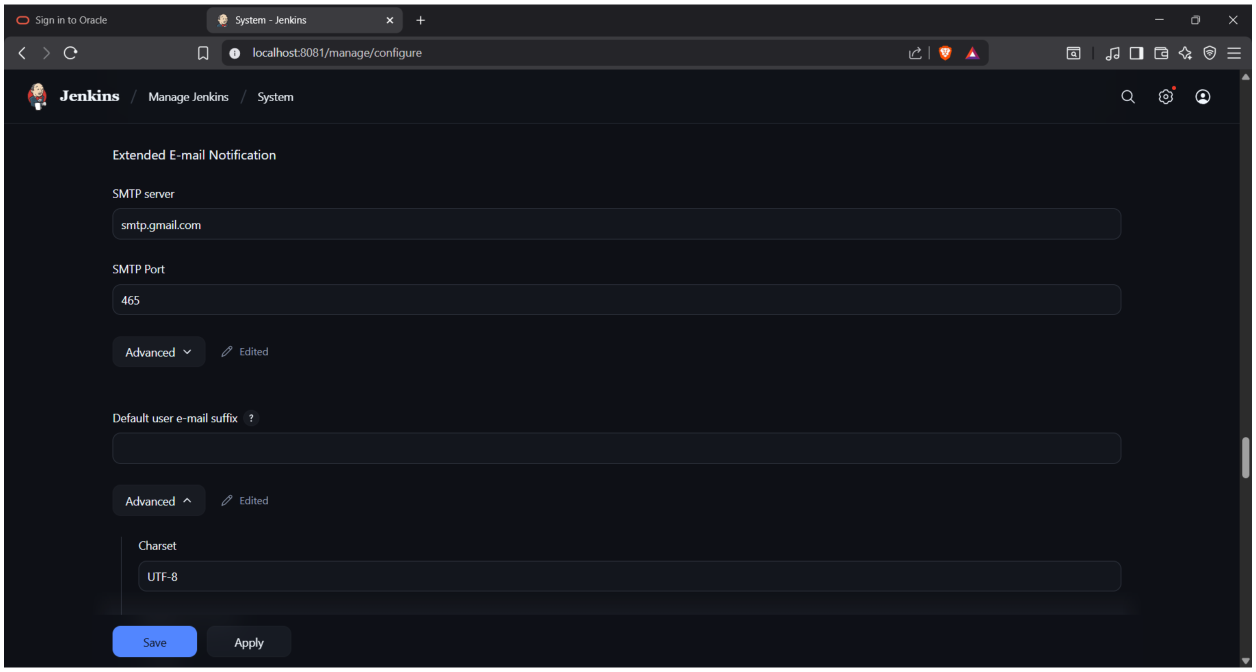
SCM Polling

Max # of concurrent polling

10

B. Extended E-mail Notification Section

Field	Value
SMTP Server	smtp.gmail.com
SMTP Port	465
Use SSL	✅ Enabled
Credentials	Add Gmail ID and App Password as Jenkins credentials
Default Content Type	text/html or leave default
Default Recipients	Leave empty or provide default emails
Triggers	Select as per needs (e.g., Failure)


Sign in to Oracle
System - Jenkins
localhost:8081/manage/configure
Jenkins / Manage Jenkins / System

Extended E-mail Notification

SMTP server: smtp.gmail.com

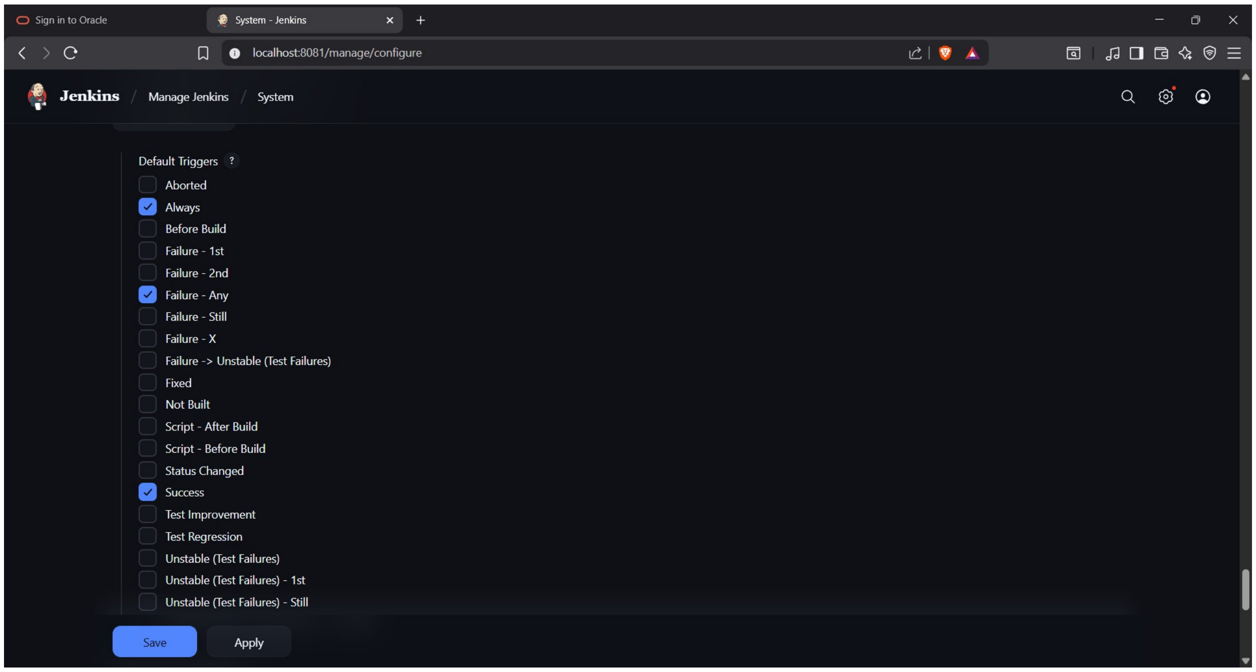
SMTP Port: 465

Advanced — Edited

Default user e-mail suffix ?

Advanced — Edited

Charset: UTF-8


Sign in to Oracle
System - Jenkins
localhost:8081/manage/configure
Jenkins / Manage Jenkins / System

Default Triggers ?

- [ ] Aborted
- [x] Always
- [ ] Before Build
- [ ] Failure - 1st
- [ ] Failure - 2nd
- [x] Failure - Any
- [ ] Failure - Still
- [ ] Failure - X
- [ ] Failure -> Unstable (Test Failures)
- [ ] Fixed
- [ ] Not Built
- [ ] Script - After Build
- [ ] Script - Before Build
- [ ] Status Changed
- [x] Success
- [ ] Test Improvement
- [ ] Test Regression
- [ ] Unstable (Test Failures)
- [ ] Unstable (Test Failures) - 1st
- [ ] Unstable (Test Failures) - Still

4. Configure Email Notifications for a Jenkins Job

i. Go to:

- Jenkins → Select a Job → Configure

---

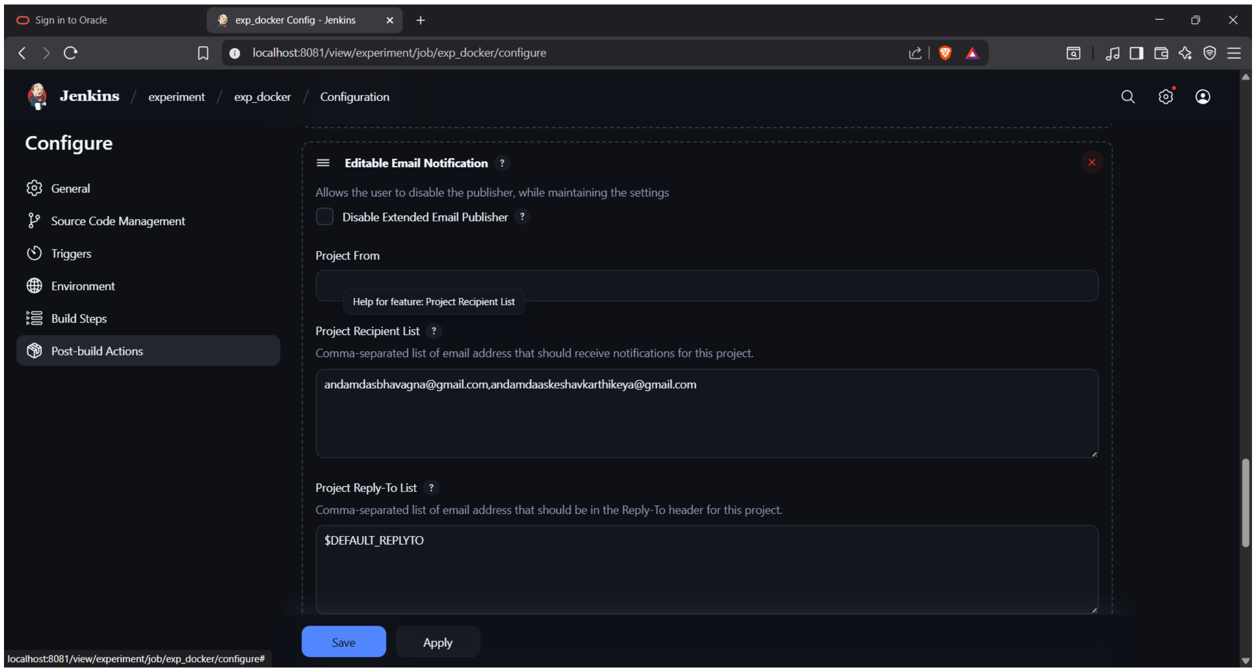
ii. In the Post-build Actions section:

- Click: Add post-build action → **Editable Email Notification**

A. Fill in the fields:

Field	Value
Project Recipient List	Add recipient email addresses (comma-separated)
Content Type	Default (text/plain) or text/html
Triggers	Select events (e.g., Failure, Success, etc.)
Attachments	(Optional) Add logs, reports, etc.

iii. Click Save



Editable Email Notification

Allows the user to disable the publisher, while maintaining the settings

Disable Extended Email Publisher

Project From

Project Recipient List

Comma-separated list of email address that should receive notifications for this project.

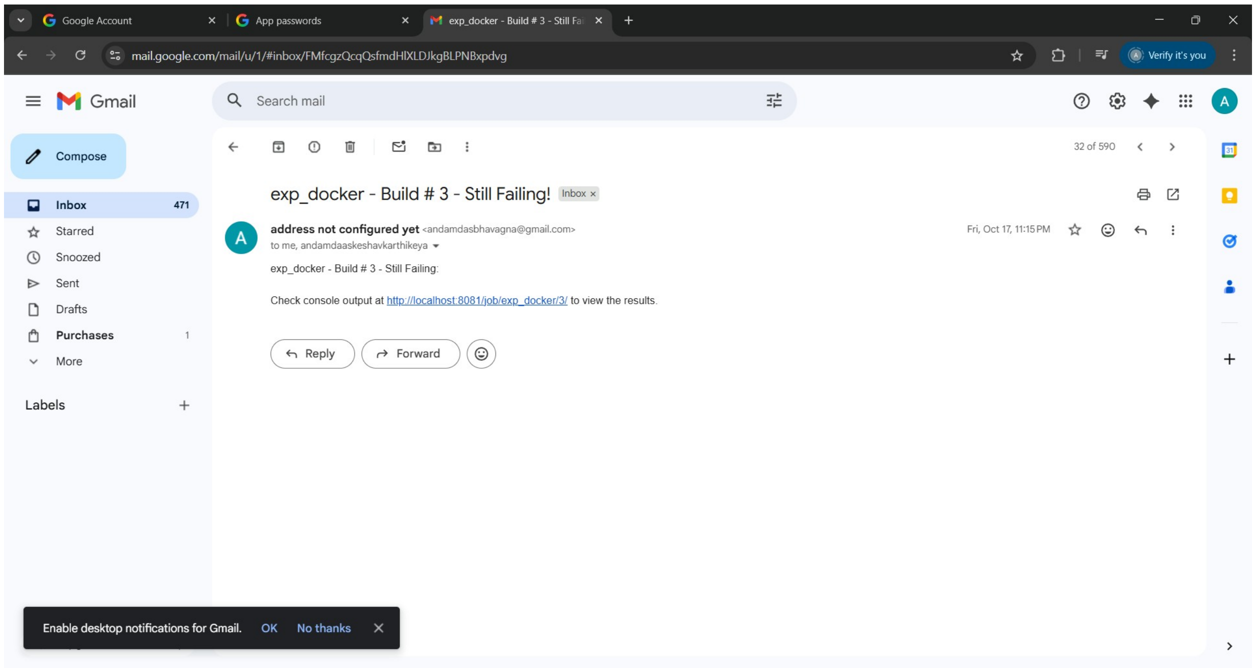
andamdasbhavagna@gmail.com,andamdaaskeshavkarthikeya@gmail.com

Project Reply-To List

Comma-separated list of email address that should be in the Reply-To header for this project.

$DEFAULT_REPLYTO

Now your Jenkins job is set up to send email notifications based on the build status!


exp_docker - Build # 3 - Still Failing!

address not configured yet <andamdasbhavagna@gmail.com>

to me, andamdaaskeshavkarthikeya

Fri, Oct 17, 11:15 PM

exp_docker - Build # 3 - Still Failing:

Check console output at http://localhost:8081/job/exp_docker/3/ to view the results.

●

Takeaway :
Students learned how to integrate Jenkins with GitHub using webhooks to automate build triggers and configure email notifications to monitor build success or failure effectively.

Viva Questions

1. What is Continuous Integration (CI)?

Continuous Integration (CI) is a development practice where developers frequently merge their code changes into a shared repository, triggering automated builds and tests. This ensures that integration issues are detected early and the codebase remains stable.

2. What is Continuous Deployment or Continuous Delivery (CD)?

Continuous Deployment, an extension of CD, automatically deploys every validated change to production without manual approval

3. What is the role of Jenkins in a CI/CD pipeline?

Jenkins automates the entire CI/CD process. It pulls code from repositories, builds it, runs tests, and deploys the application automatically based on triggers. Jenkins helps in continuous integration, delivery, and deployment through pipelines

4. What is a webhook in GitHub?

A webhook in GitHub is a way for GitHub to send real-time notifications to an external server whenever specific events like push

5. Why are webhooks used in Jenkins integration?

Webhooks are used so that Jenkins can be automatically build the pipeline if can changes in git hub repo

6. What are the different types of build triggers available in Jenkins?

Jenkins has five types of triggers

Build periodically

Poll SCM

Build after other projects are built

GitHub hook trigger for GITScm polling

Manual build trigger

7. What is the difference between polling and webhook triggers?

Polling trigger the build for fixed intervals ,webhooks trigger the build when git hub repo as any changes or commit

8. What is ngrok and why is it used in Jenkins–GitHub integration?

ngrok is a tunneling tool that provide public url, because GitHub webhooks need a publicly accessible URL to send notifications

9. How does ngrok help in setting up webhooks for Jenkins running on a local machine?

ngrok creates a public HTTPS URL that forwards incoming requests to your local Jenkins server.It allows GitHub to communicate with Jenkins even if it is hosted on your local machine

10. Why do we configure email notifications in Jenkins and how are they useful for monitoring build results?

Email notifications inform developers automatically about build success or failure.So that they can fix the issue and track the build