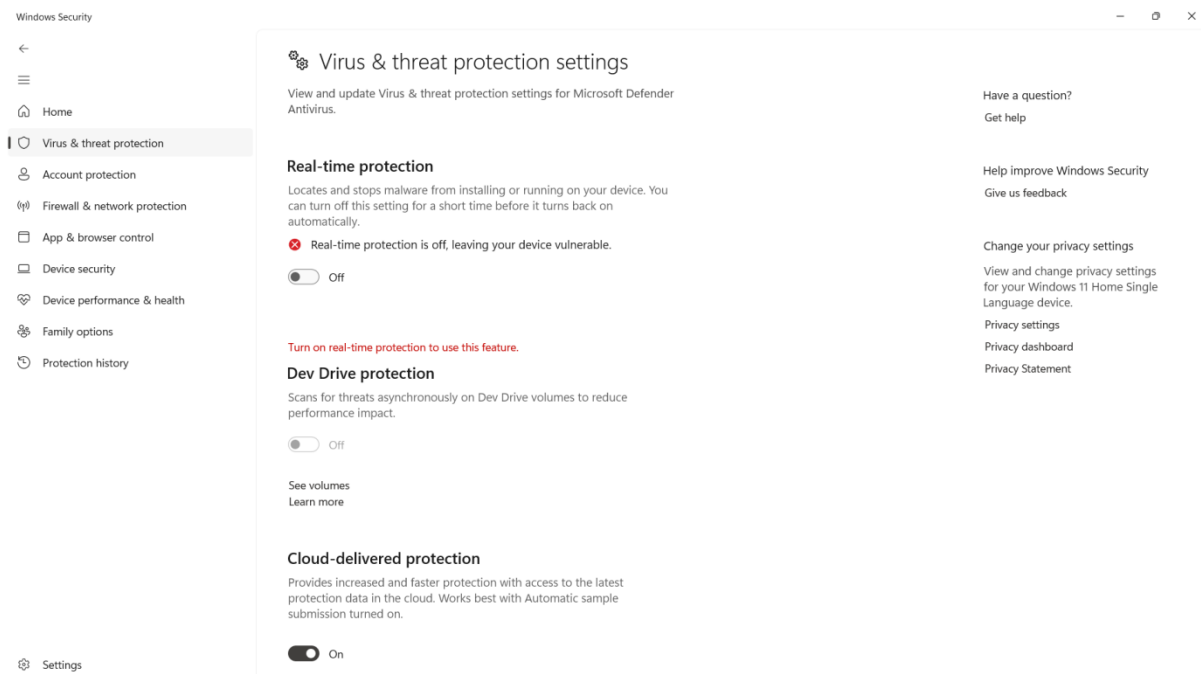


WEEK 11

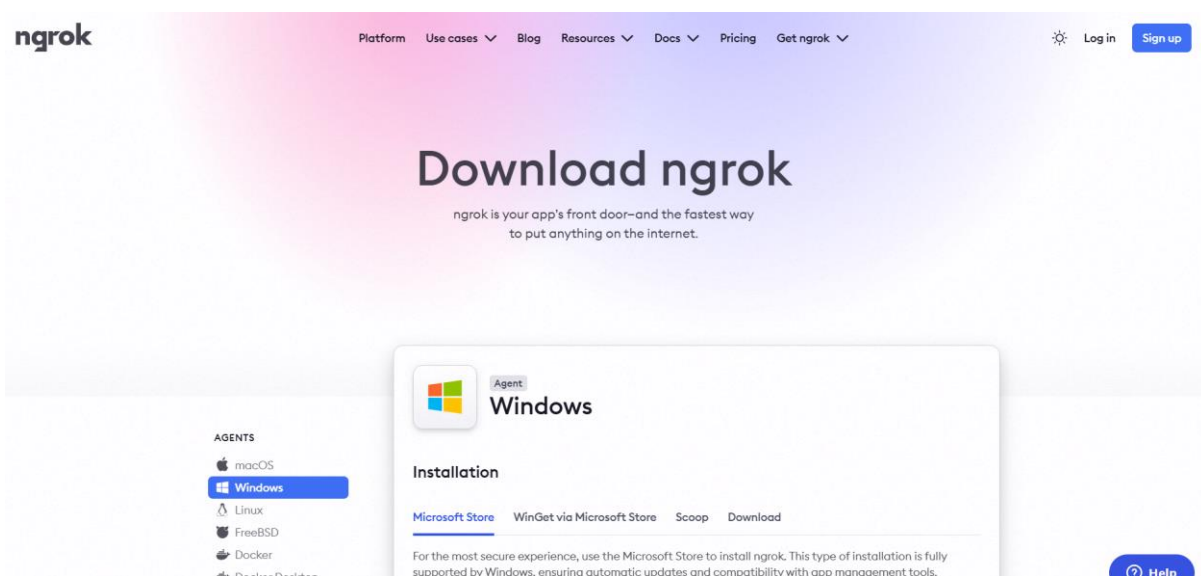
Working on windows 11

Exercise 1: Jenkins CI/CD using Git Webhook

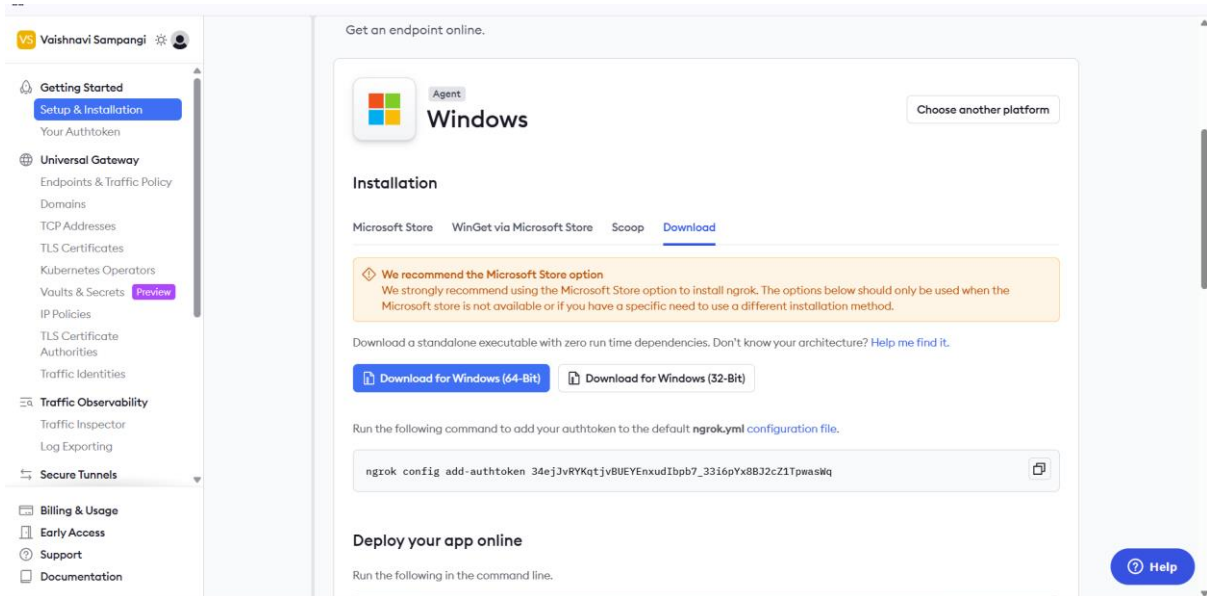
Step-1: To install ngroks Go->settings->privacy and security -> windows security -> off antivirus ->



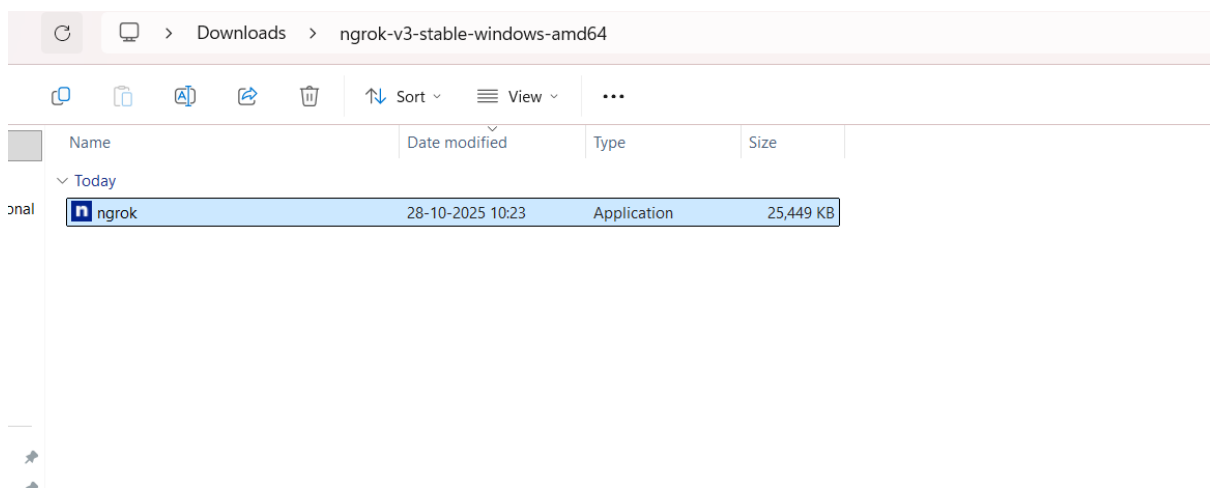
Step-2:Go -> <https://ngrok.com> and signup by giving your name ,email and password of atleast 10 charcaters



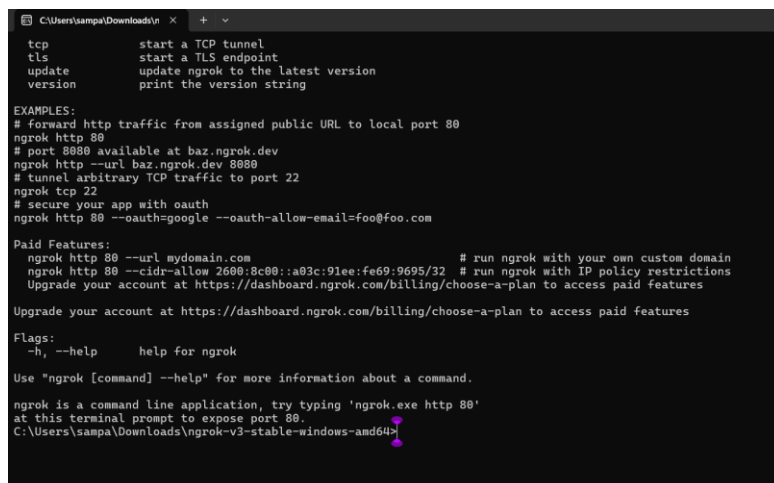
Step-3: After sign in up ,it will show below screen with your name in the top left .now click on download for windows (64Bit) to download ngrok



Step4:After downloading ,Extract the file and click on ngrok.exe



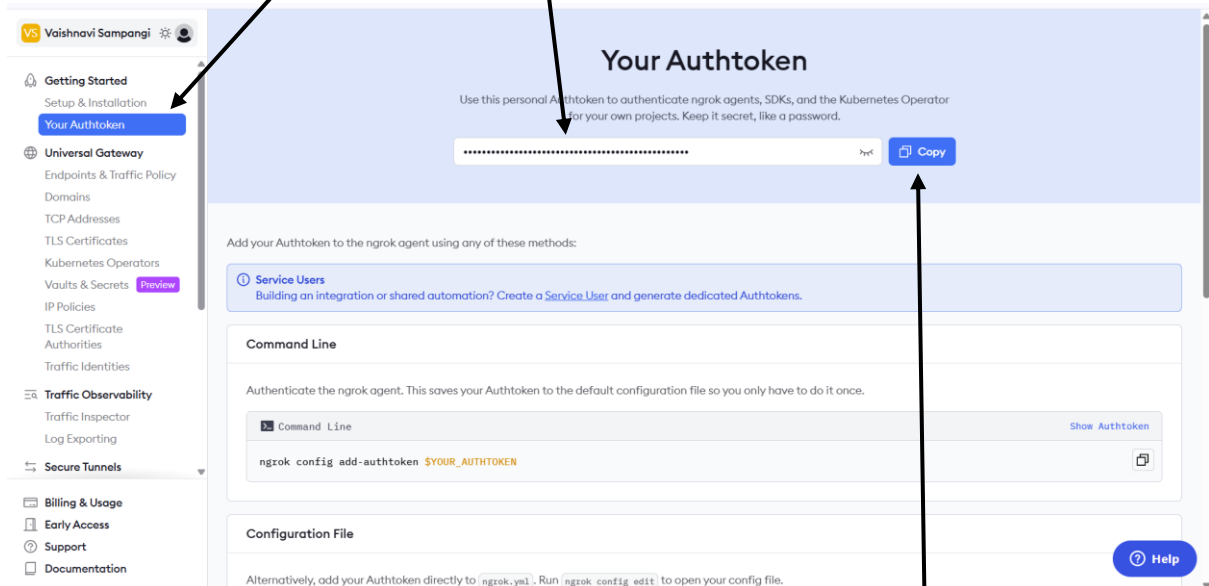
Ngrok command prompt appears as below



Step-5: Connect Your ngrok Account (optional but useful)

- Go to ngrok gives you an auth token.
- Then go to your Authtoken click here
-

Copy your Authtoken



CREATE AUTHENTICATOR [<https://dashboard.ngrok.com/get-started/your-authtoken>]
Run this command in ngrok command prompt:(replace <your_token> with yours):

ngrok config add-authtoken <your_token> // syntax:

Example command

ngrok config add-authtoken 34ejJvRYKqtjvBUEYEnxudIbpb7_33i6pYx8BJ2cZ1TpwasWq

```
C:\Users\sampa\Downloads\ngrok-v3-stable-windows-amd64>ngrok config add-authtoken 34ejJvRYKqtjvBUEYEnxudIbpb7_33i6pYx8BJ2cZ1TpwasWq
Authtoken saved to configuration file: C:\Users\sampa\AppData\Local\ngrok\ngrok.yml

C:\Users\sampa\Downloads\ngrok-v3-stable-windows-amd64>
```

Step-6

Start a Tunnel for Jenkins

- Check on which port is your Jenkins running . for this give in browsers or url localhost:8081
For me Jenkins is running on 8081
- Go to ngrok command prompt and type below command
- ngrok http 8081 //Always use this command to start a tunnel for jenkins .

Type in ngrok command prompt:

```
C:\Users\sampa\Downloads\ngrok-v3-stable-windows-amd64>ngrok http 8081
```

Next it shows this public jenkins URL generated by ngrok that can be pasted into github repo for Webhooks.

```
ngrok - tunnel local ports to public URLs and inspect traffic

USAGE:
  ngrok [command] [flags]

COMMANDS:
  api          CLI to api.ngrok.com
  completion   generates shell completion code for bash or zsh
  config       update or migrate ngrok's configuration file
  credits      prints author and licensing information
  help         help about any command
  http         start an HTTP tunnel
  service      run and control ngrok as a background service
  start        start endpoints in the config file by name
  tcp          start a TCP tunnel
  tls          start a TLS endpoint
  update       update ngrok to the latest version
  version      print the version string

ngrok

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

Session Status      online
Account             Vaishnavi Sampangi (Plan: Free)
Version             3.32.0
Region              India (in)
Latency              63ms
Web Interface        http://127.0.0.1:4040
Forwarding            https://unhired-stormily-alaine.ngrok-free.dev -> http://localhost:8081

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

Copy this URL only highlighted part

```
ngrok - tunnel local ports to public URLs and inspect traffic

USAGE:
  ngrok [command] [flags]

COMMANDS:
  api          CLI to api.ngrok.com
  completion   generates shell completion code for bash or zsh
  config       update or migrate ngrok's configuration file
  credits      prints author and licensing information
  help         help about any command
  http         start an HTTP tunnel
  service      run and control ngrok as a background service
  start        start endpoints in the config file by name
  tcp          start a TCP tunnel
  tls          start a TLS endpoint
  update       update ngrok to the latest version
  version      print the version string

ngrok

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

Session Status      online
Account             Vaishnavi Sampangi (Plan: Free)
Version             3.32.0
Region              India (in)
Latency              88ms
Web Interface        http://127.0.0.1:4040
Forwarding            https://unhired-stormily-alaine.ngrok-free.dev -> http://localhost:8081

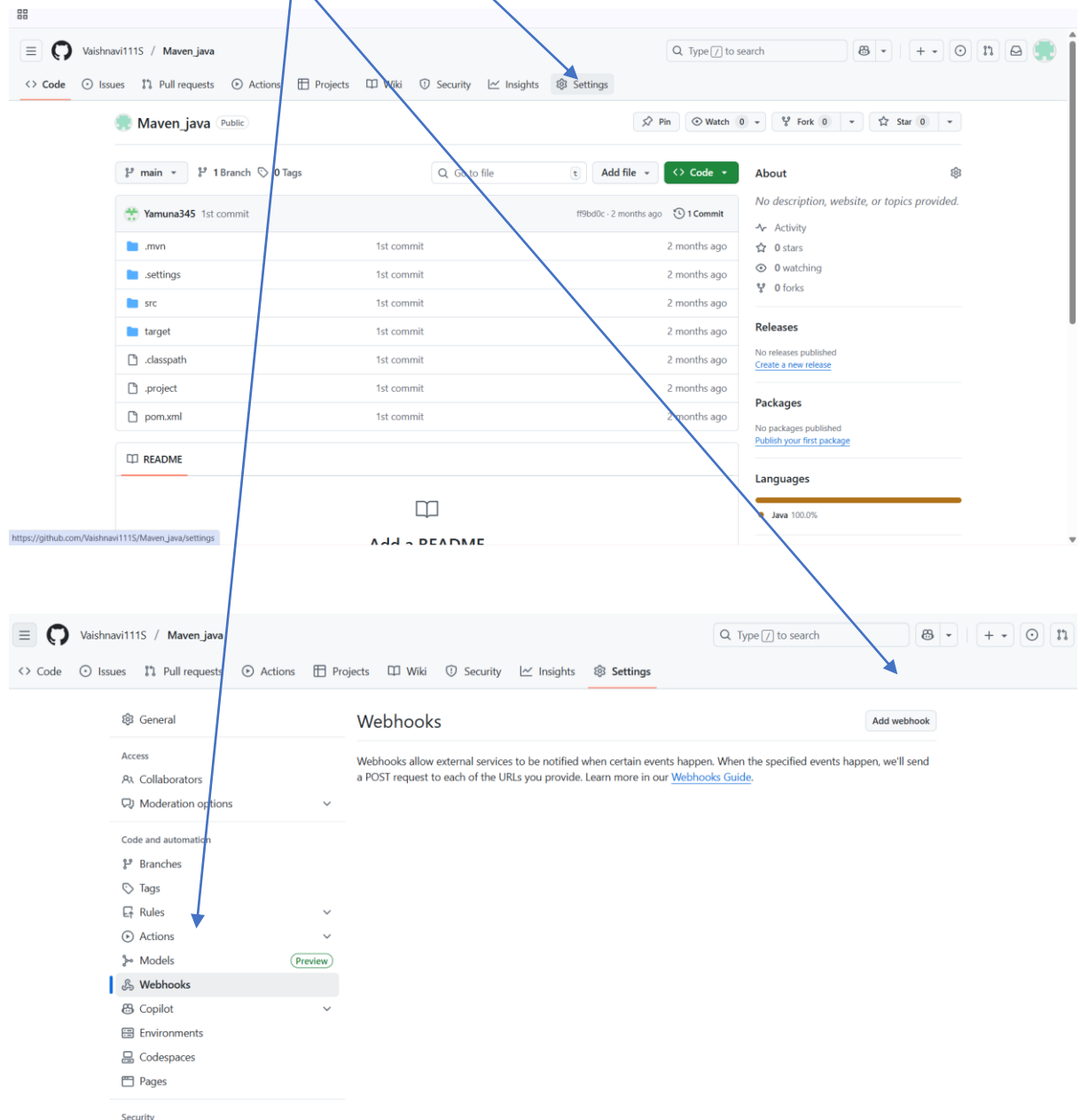
Connections          ttl    opn    rt1    rt5    p50    p90
                     0      0      0.00   0.00   0.00   0.00
```

Step-7: Configure Webhook in GitHub

1. Go to your GitHub repository.

2. Navigate to Settings → **Webhooks**.
3. Click “**Add webhook**”.
4. In the Payload URL field:
 - Enter the Jenkins webhook URL in the format:
`http://<jenkins-server-url>/github-webhook/`
Ex: `https://unhired-stormily-alaine.ngrok-free.dev/github-webhook/`

Note: If Jenkins is running on localhost, GitHub cannot access it directly



Step-8:

- Add url <https://unhired-stormily-alaine.ngrok-free.dev/github-webhook/>
- Set content Type to application/json
- Under “Which events would you like to trigger this webhook?”, select:
Just the push event.
 - Click “Add webhook” to save.

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://unhired-stormily-alaine.ngrok-free.dev/github-webhook/>

Content type *

application/json

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.

☐ Let me select individual events.

☒ Active

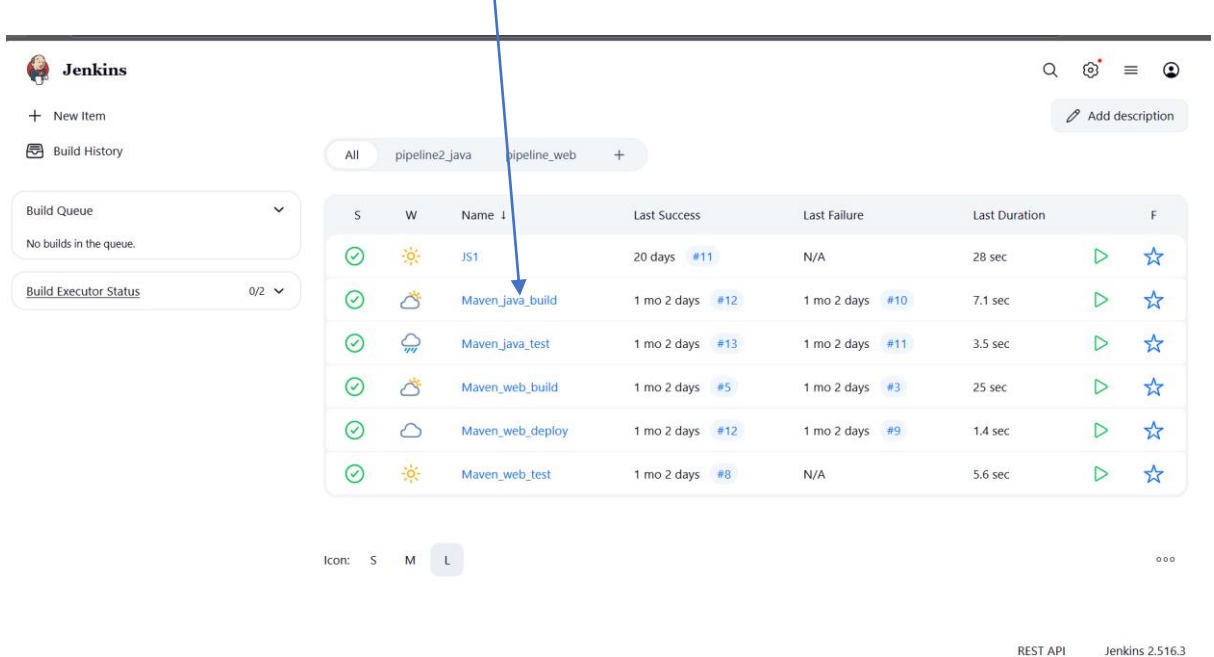
We will deliver event details when this hook is triggered.

Add webhook

Step 10: Configure Jenkins to Accept GitHub Webhooks

1. Open Jenkins Dashboard.

2. Select the job (freestyle or pipeline) you've already created.



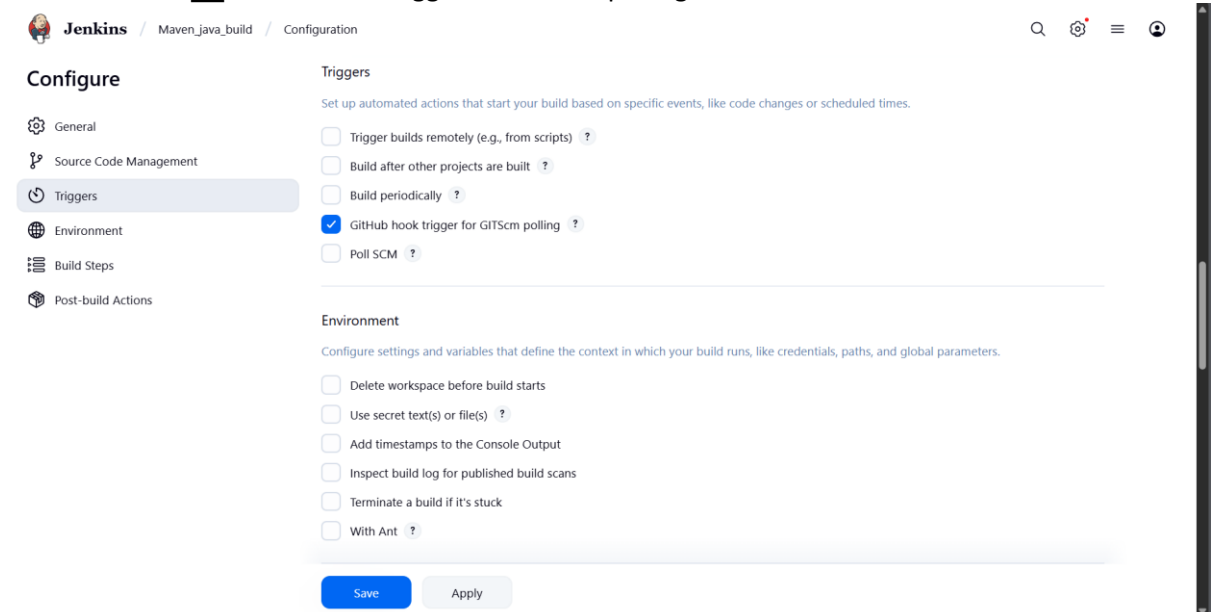
The screenshot shows the Jenkins dashboard. On the left, there's a sidebar with 'New Item', 'Build History', 'Build Queue' (No builds in the queue), and 'Build Executor Status' (0/2). The main area displays a table of jobs. A blue arrow points to the 'Maven_java_build' job. The table has columns: S (Status), W (Webhook), Name, Last Success, Last Failure, Last Duration, and F (Favorite). The jobs listed are JS1, Maven_java_build, Maven_java_test, Maven_web_build, Maven_web_deploy, and Maven_web_test.

S	W	Name	Last Success	Last Failure	Last Duration	F
✓	☀	JS1	20 days #11	N/A	28 sec	▶ ☆
✓	☁	Maven_java_build	1 mo 2 days #12	1 mo 2 days #10	7.1 sec	▶ ☆
✓	☁	Maven_java_test	1 mo 2 days #13	1 mo 2 days #11	3.5 sec	▶ ☆
✓	☁	Maven_web_build	1 mo 2 days #5	1 mo 2 days #3	25 sec	▶ ☆
✓	☁	Maven_web_deploy	1 mo 2 days #12	1 mo 2 days #9	1.4 sec	▶ ☆
✓	☀	Maven_web_test	1 mo 2 days #8	N/A	5.6 sec	▶ ☆

Icon: S M L

REST API Jenkins 2.516.3

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



The screenshot shows the Jenkins configuration page for 'Maven_java_build'. The left sidebar has 'Configure' selected. The main area is divided into 'Triggers' and 'Environment' sections. In the 'Triggers' section, the 'GitHub hook trigger for GITScm polling' checkbox is checked. The 'Environment' section has several unchecked checkboxes.

Configure

- General
- Source Code Management
- Triggers
- Environment
- Build Steps
- Post-build Actions

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 ?
- ☒ 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 ?

Save Apply

6. Click Save.

Step 11: 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.

Jenkins

Maven_java_build

Configuration

Search

Settings

Menu

Profile

Configure

General

Source Code Management

Triggers

Environment

Build Steps

Post-build Actions

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 ?

☒ 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 ?

Save

Apply

Jenkins

+ New Item

Build History

Build Queue

No builds in the queue.

Build Executor Status

0/2

All

pipeline2_java

pipeline_web

+

S	W	Name ↓	Last Success	Last Failure	Last Duration	F
✓	☀	JS1	20 days #11	N/A	28 sec	▶ ☆
✓	☁	Maven_java_build	1 mo 2 days #12	1 mo 2 days #10	7.1 sec	▶ ☆
✓	☁	Maven_java_test	1 mo 2 days #13	1 mo 2 days #11	3.5 sec	▶ ☆
✓	☁	Maven_web_build	1 mo 2 days #5	1 mo 2 days #3	25 sec	▶ ☆
✓	☁	Maven_web_deploy	1 mo 2 days #12	1 mo 2 days #9	1.4 sec	▶ ☆
✓	☀	Maven_web_test	1 mo 2 days #8	N/A	5.6 sec	▶ ☆

Icon: S M L

Vaishnavi1115 / Maven_java

REST API

Jenkins 2.516.3

<> Code

Issues

Pull requests

Actions

Projects

Wiki

Security

Insights

Settings

Maven_java /

Name your file...

in main

Cancel changes

Commit changes...

Edit

Preview

Spaces 2 No wrap

1

demo of webhook

Vaishnavi1115 / Maven_java

REST API

Jenkins 2.516.3

<> Code

Issues

Pull requests

Actions

Projects

Wiki

Security

Insights

Settings

Maven_java /

DemoFile

in main

Cancel changes

Commit changes...

Edit

Preview

Spaces 2 No wrap

1

Webhook

Commit changes

Commit message

commit for webhook

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

Vaishnavi1115 / Maven_Java

Q Type to search

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main Maven_Java /

Go to file Add file

Vaishnavi1115 commit for webhook

3686ba · now History

Name	Last commit message	Last commit date
.mvn	1st commit	2 months ago
.settings	1st commit	2 months ago
src	1st commit	2 months ago
target	1st commit	2 months ago
.classpath	1st commit	2 months ago
.project	1st commit	2 months ago
DemoFile	commit for webhook	now
pom.xml	1st commit	2 months ago

+ New Item

Build History

Build Queue

No builds in the queue.

Build Executor Status

Maven_java_build #13

All pipeline2_java pipeline_web +

S	W	Name	Last Success	Last Failure	Last Duration	F
✓	☀	JS1	20 days #11	N/A	28 sec	▶ ☆
✓	☁	Maven_java_build	1 mo 2 days #12	1 mo 2 days #10	7.1 sec	▶ ☆
✓	☁	Maven_java_test	1 mo 2 days #13	1 mo 2 days #11	3.5 sec	▶ ☆
✓	☁	Maven_web_build	1 mo 2 days #5	1 mo 2 days #3	25 sec	▶ ☆
✓	☁	Maven_web_deploy	1 mo 2 days #12	1 mo 2 days #9	1.4 sec	▶ ☆
✓	☀	Maven_web_test	1 mo 2 days #8	N/A	5.6 sec	▶ ☆

Icons: S M L

Jenkins / pipeline2_java

Build Pipeline

build pipeline

Run History Configure Add Step Delete Manage

Pipeline #13

#13 Maven_java_build

#14 Maven_java_test

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.

EXCERCISE-2

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)

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)

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

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	<input checked="" type="checkbox"/> Enabled
User Name	Your Gmail ID (e.g., archanareddykmit@gmail.com)
Password	Paste the 16-digit App Password
Use SSL	<input checked="" type="checkbox"/> 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

B. Extended E-mail Notification Section

Field	Value
SMTP Server	smtp.gmail.com
SMTP Port	465
Use SSL	<input checked="" type="checkbox"/> Enabled

Field	Value
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)

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

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

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.