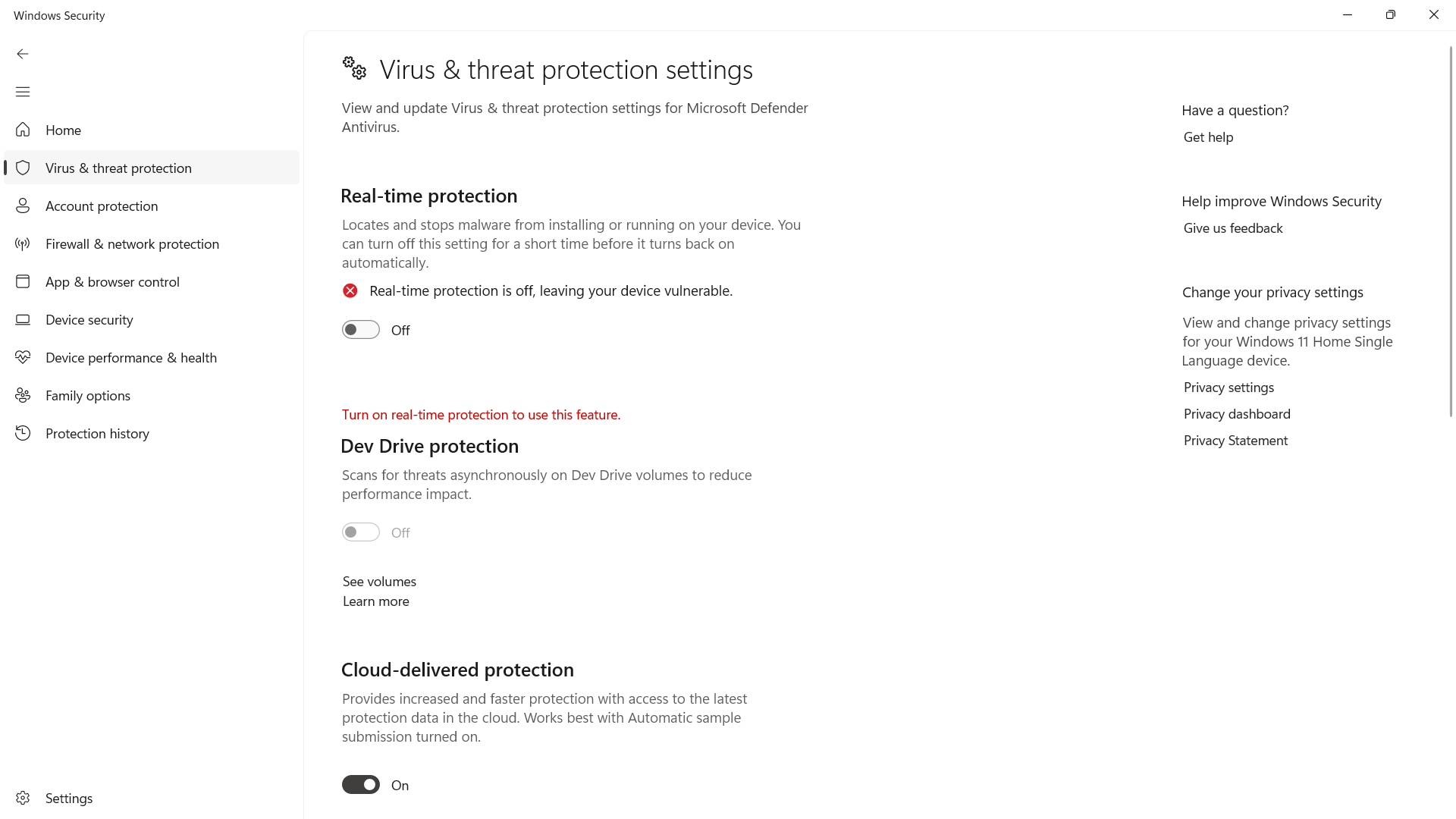
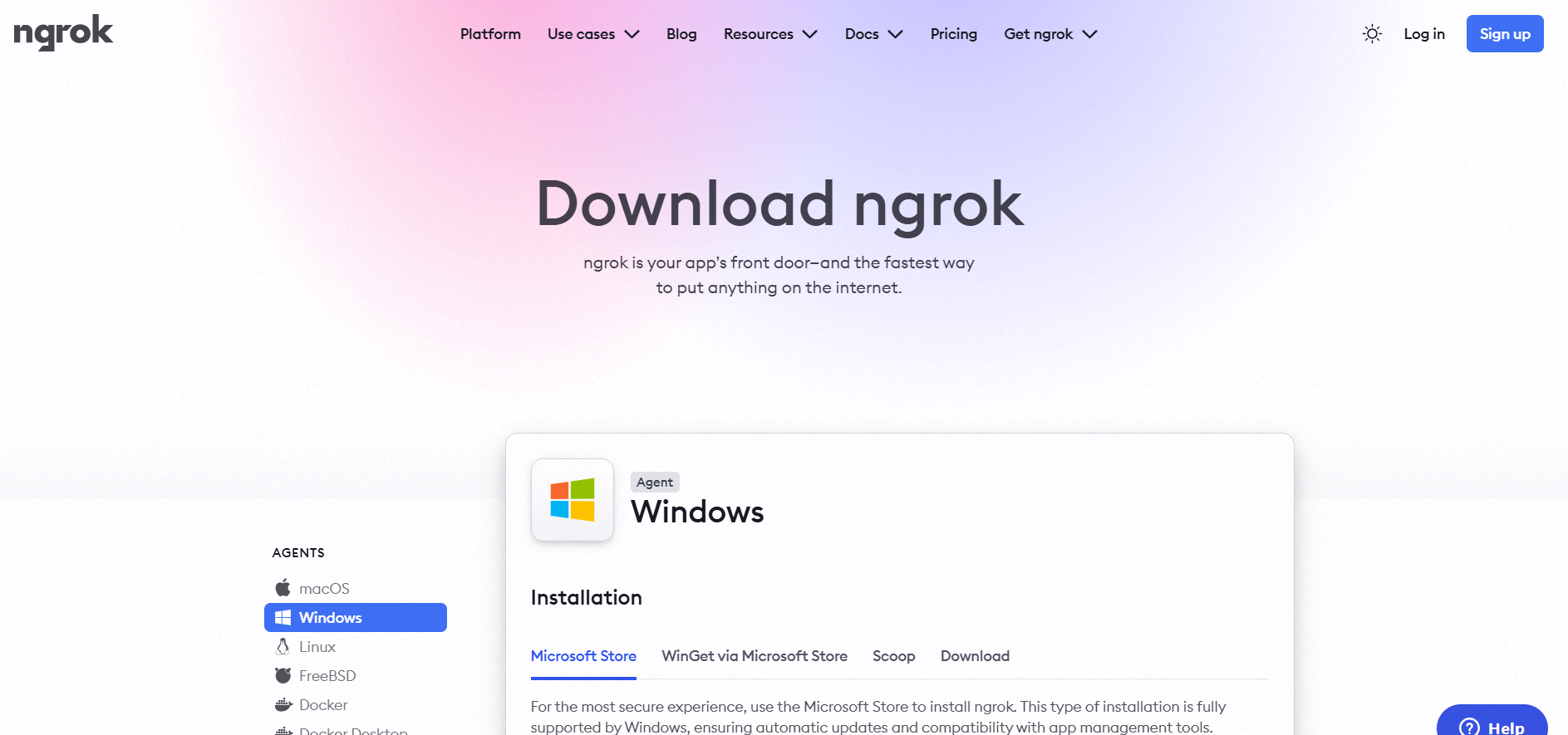
WEEK 11

Working on windows 11

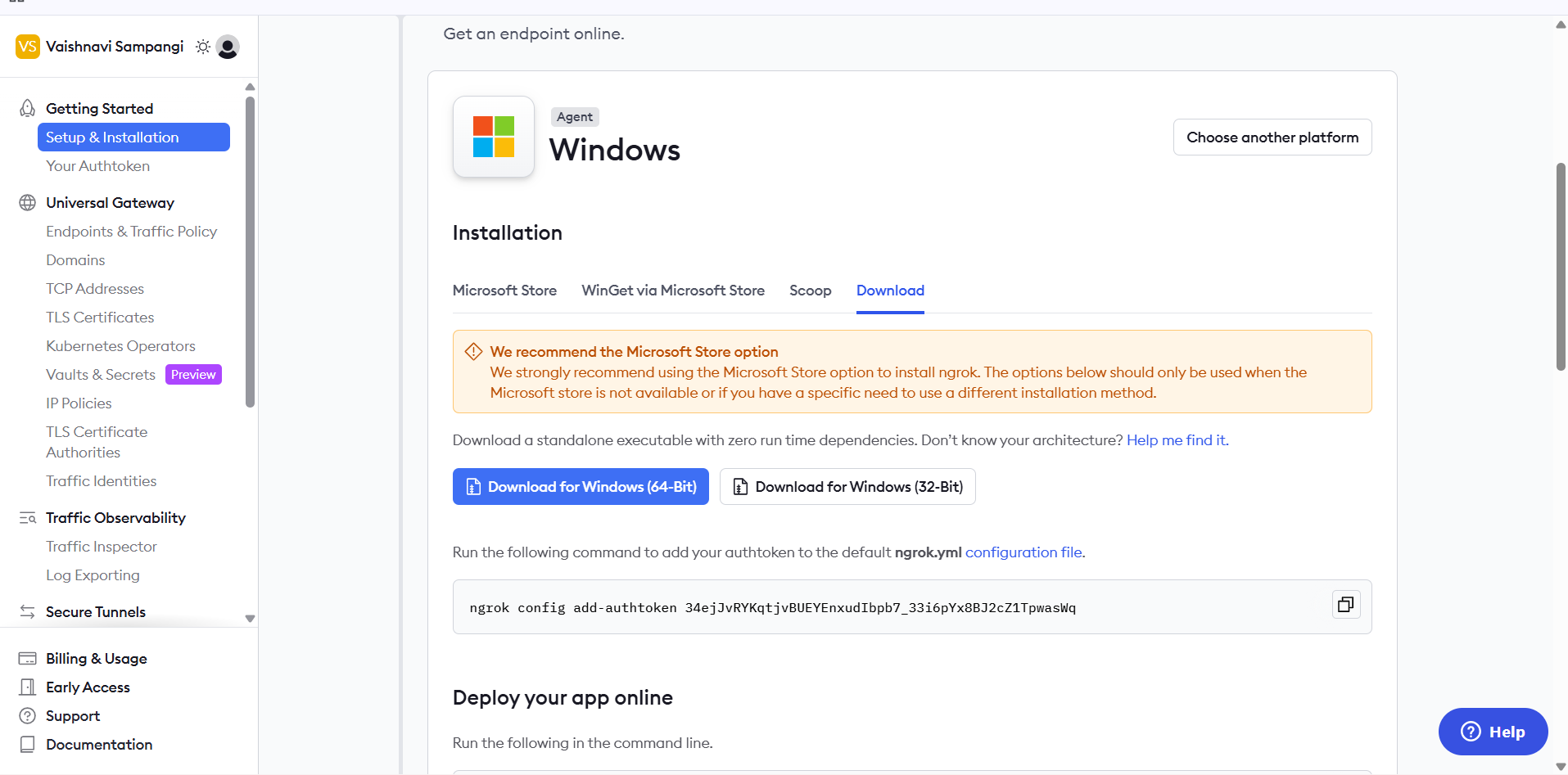
Step-1: To install ngroks Go->settings->privacy and security -> windows security -> off antivirus ->



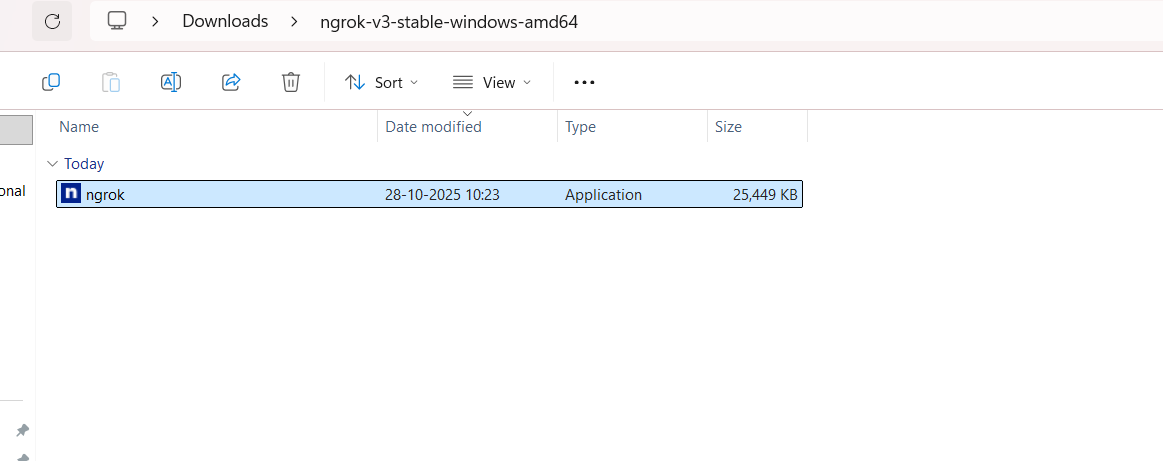
Step-2:Go -> <https://ngrok.com> and signup by giving your name ,email and password of atleast 10 charcaters



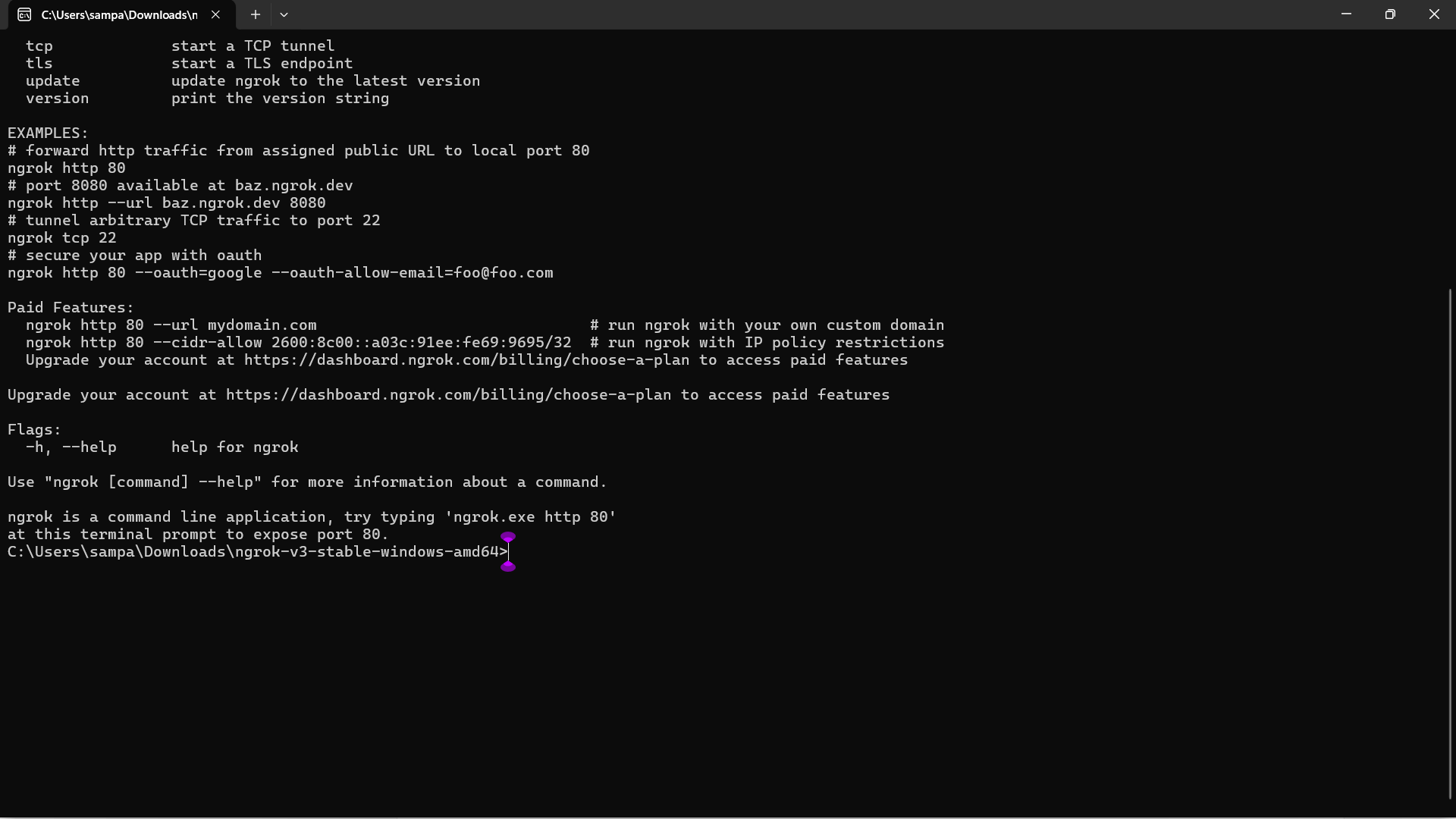
Step-3: After sign in up ,it will show below screen with your name in the top left .now click on download for windows (64Bit) to download ngork



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

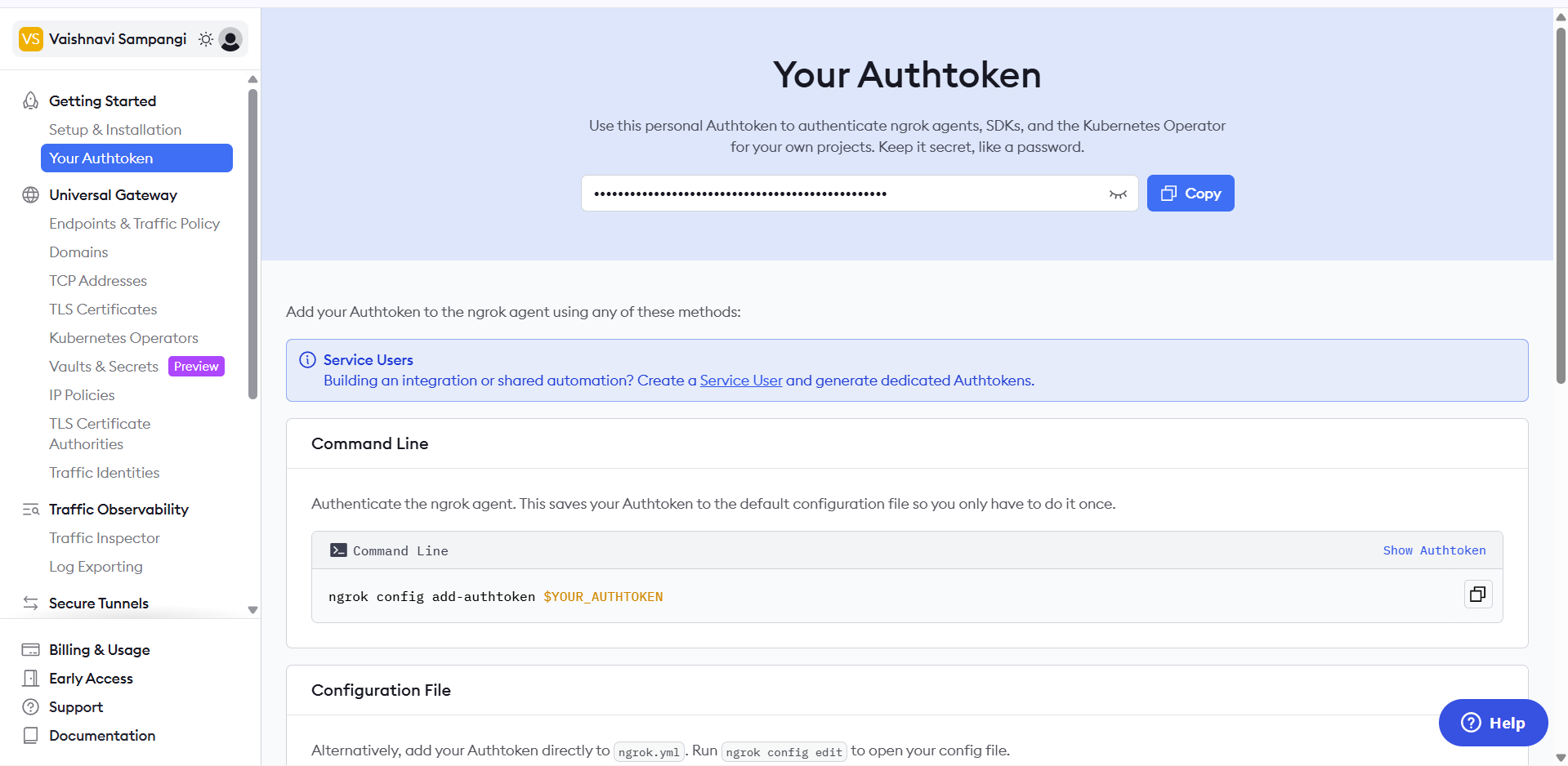


Ngrok command prompt appears as below



**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

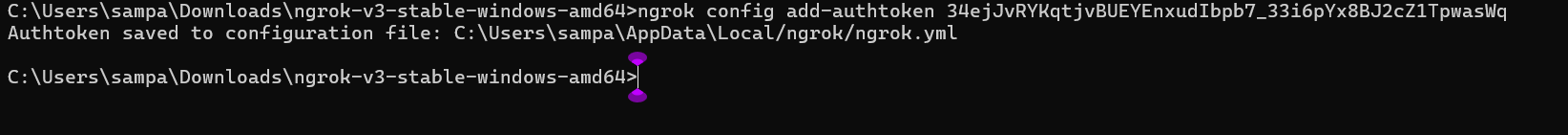


CREATE AUTHENTICATOR [https://dashboard.ngrok.com/get-started/your-authtoken]  
Run this command in ngrok command prompt:(replace <your\_token> with yours):

**ngrok config add-authtoken <your\_token> // syntax:**

**Example command**

**ngrok config add-authtoken 34ejJvRYKqtjvBUEYEnxudIbpb7\_33i6pYx8BJ2cZ1TpwasWq**

****

**Step-6**

**Start a Tunnel for Jenkins**

* Check on which port is your Jenkins running . for this give in browers or url localhost:8081

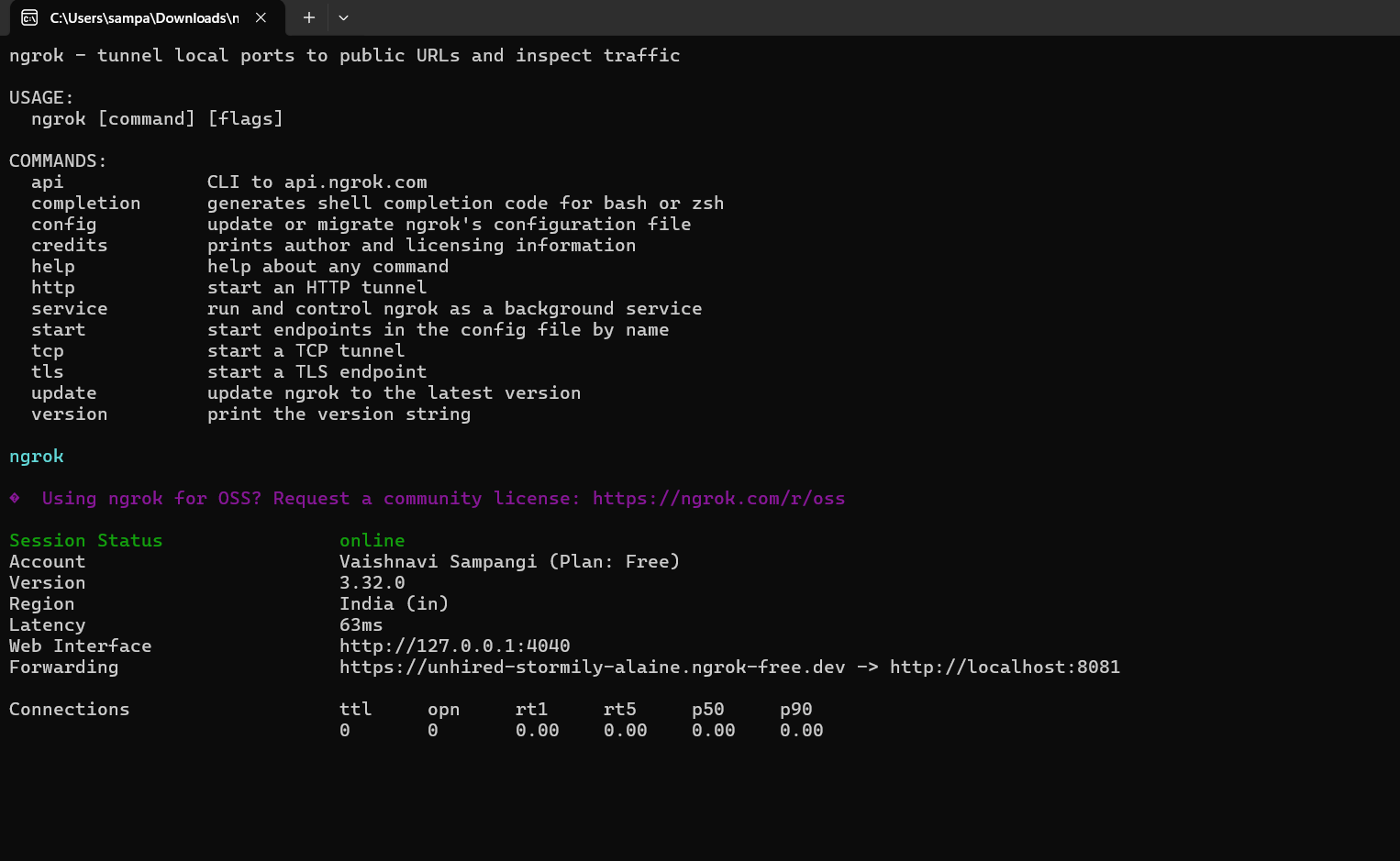
For me Jenkins is running on 8081

* Go to ngrok command prompt and type below command
* ngork http 8081 //Always use this command to start a tunnel for jenkins .

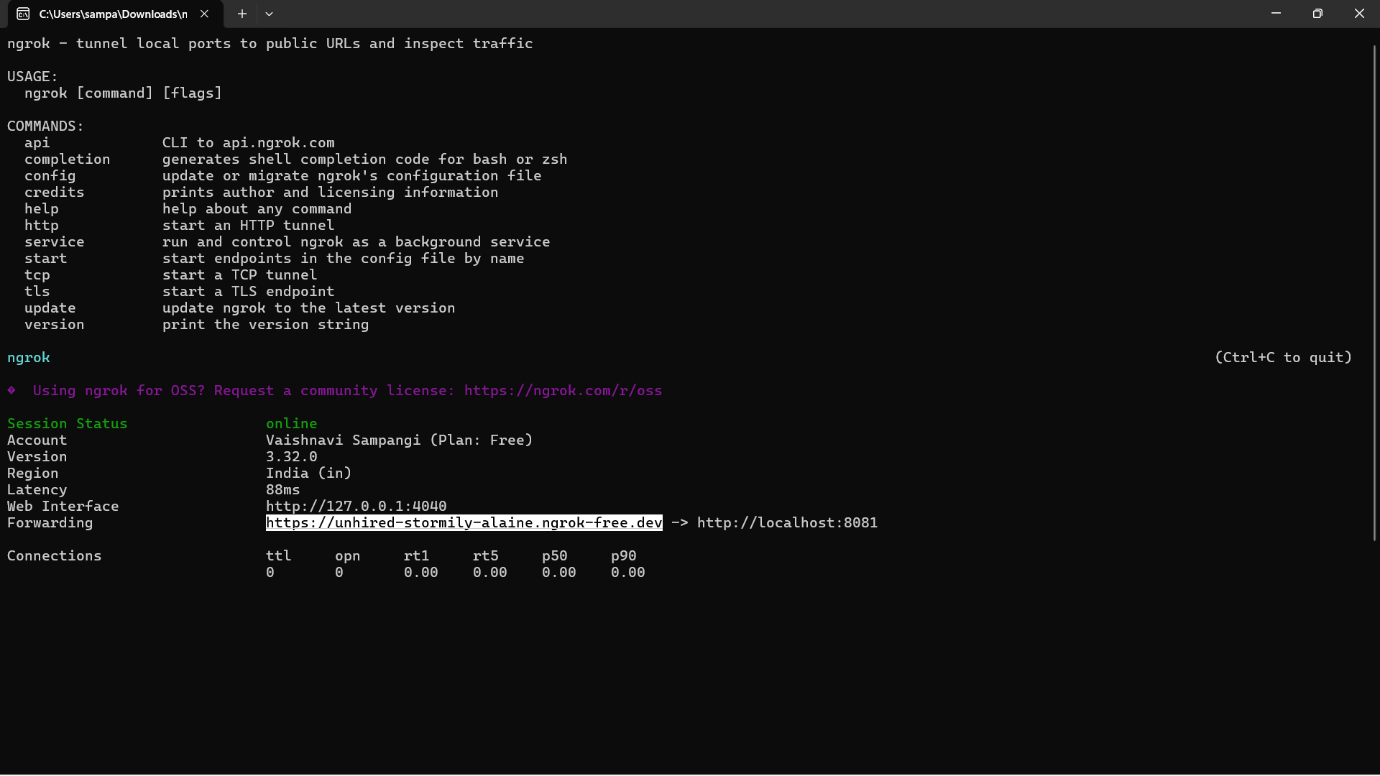
Type in ngrok command prompt:



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



Copy this URL only highlighted part

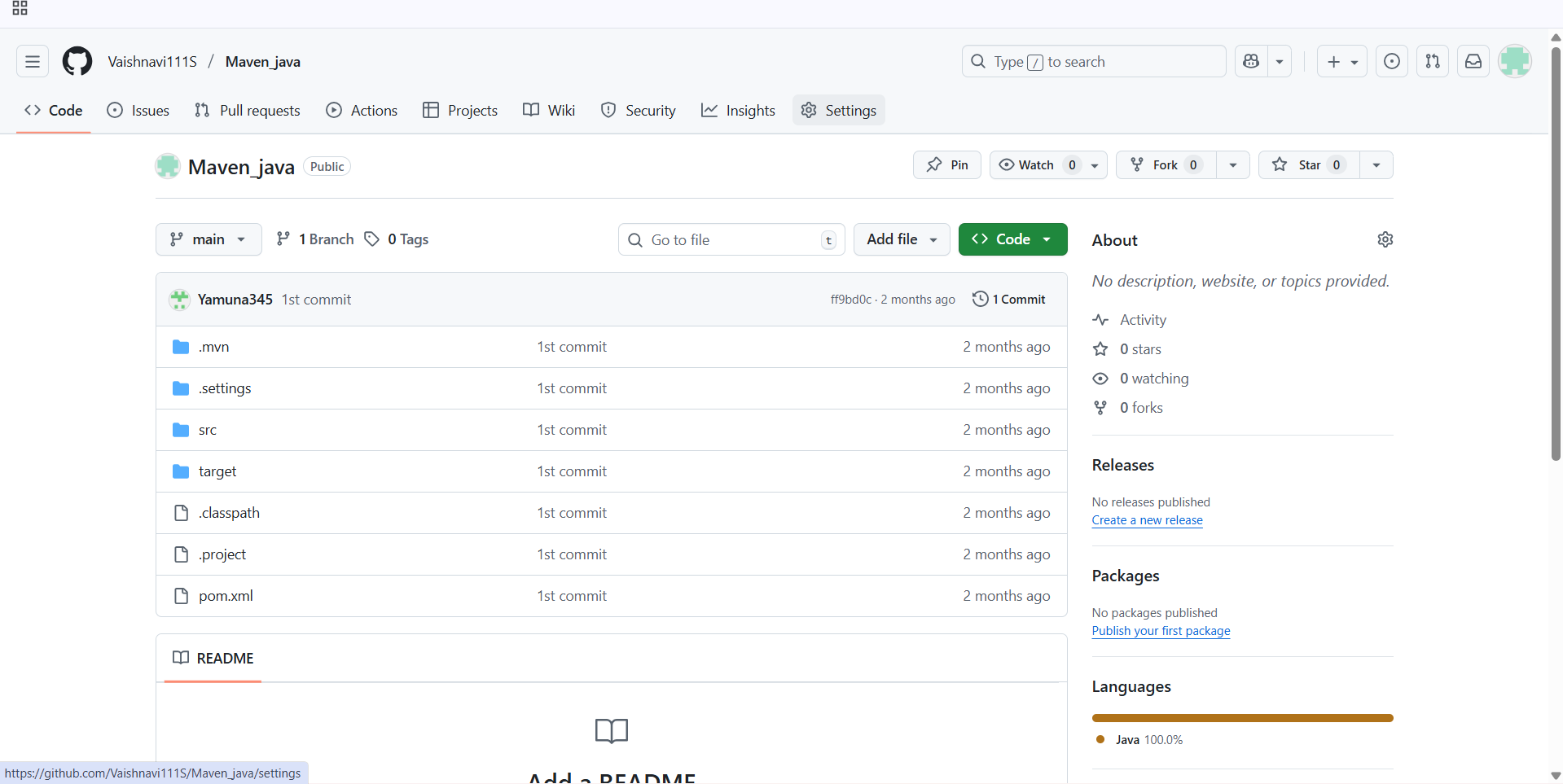


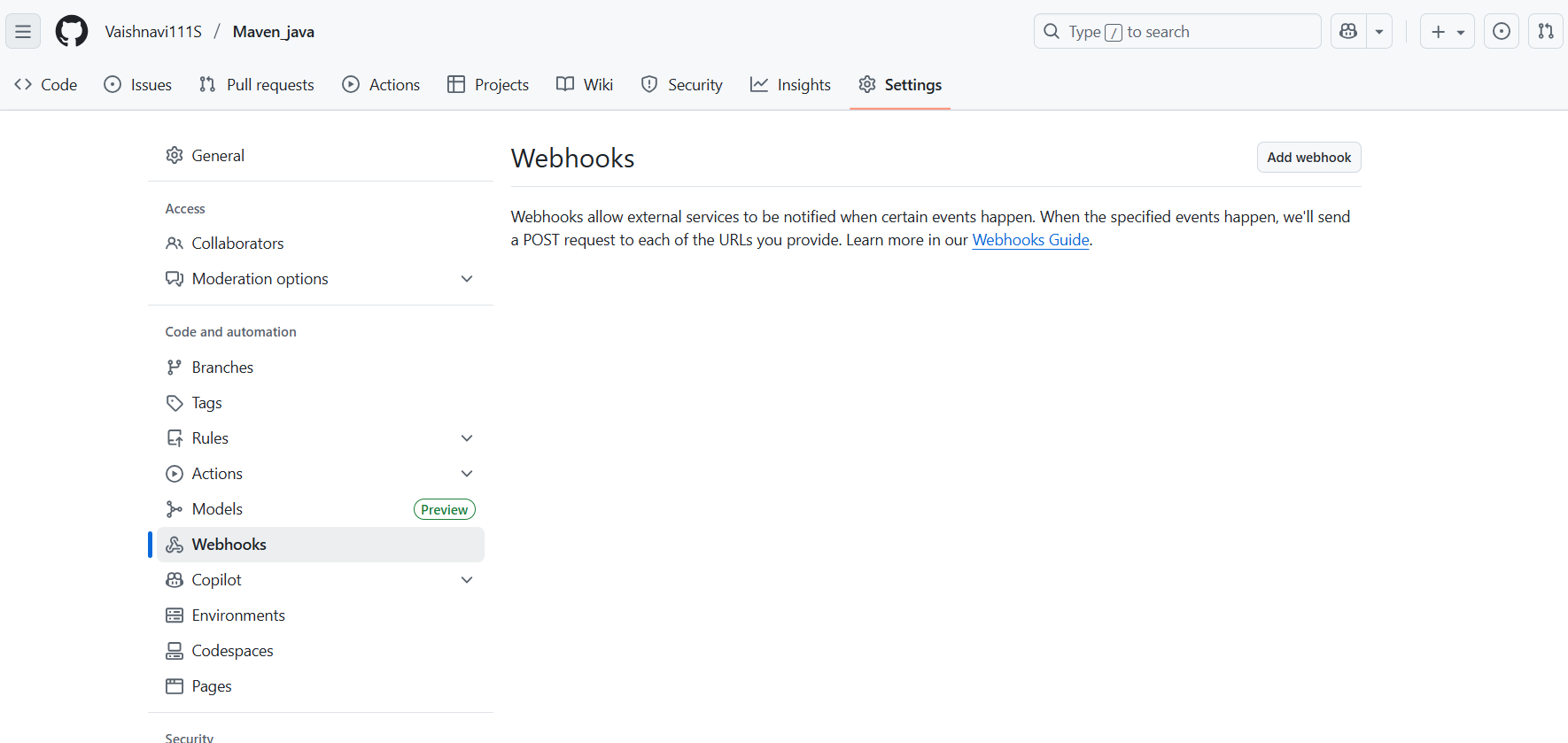
Step-7: **Configure Webhook in GitHub**

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



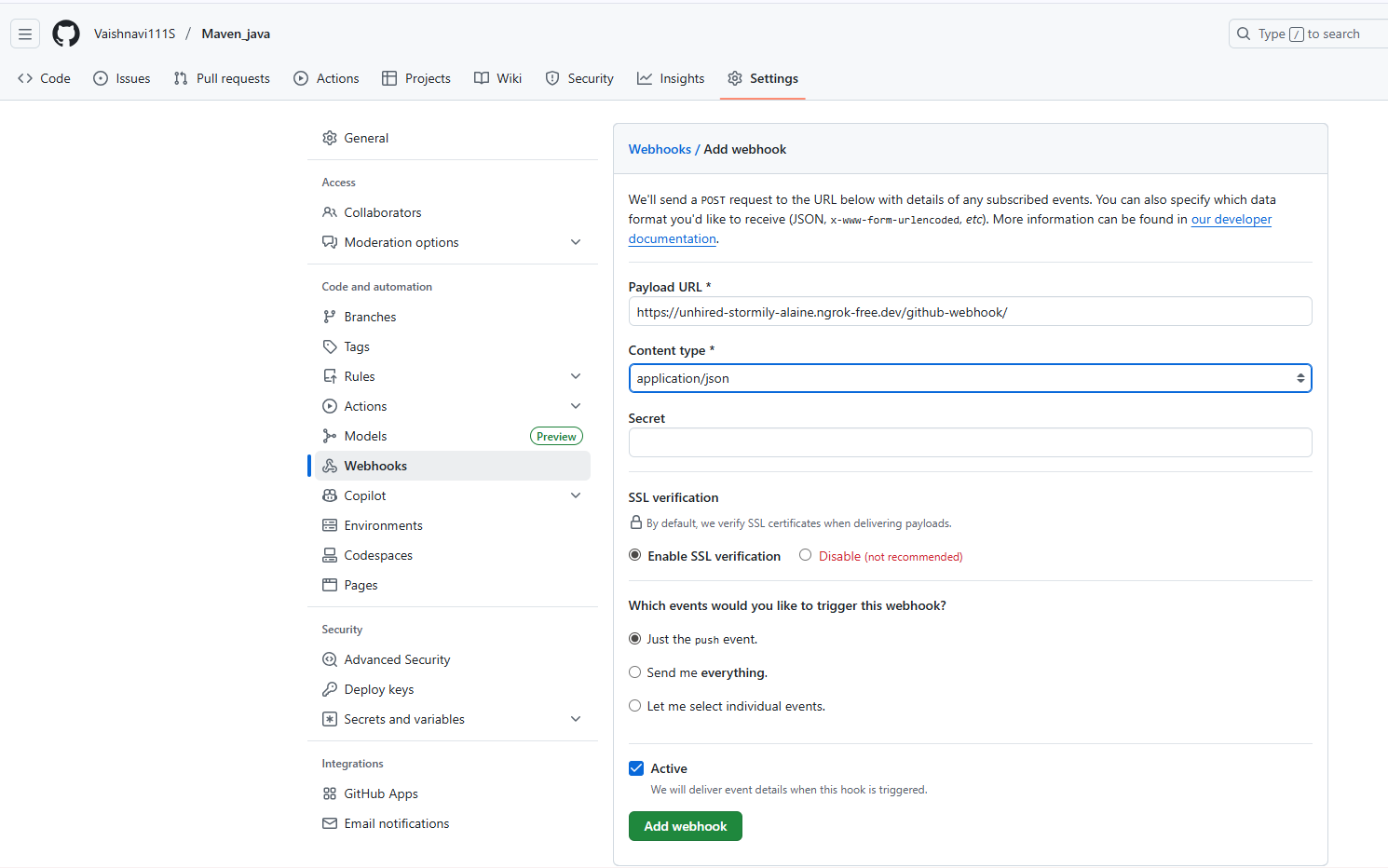


Step-8:

* + - Add url [**https://unhired-stormily-alaine.ngrok-free.dev**/github-webhook/](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:

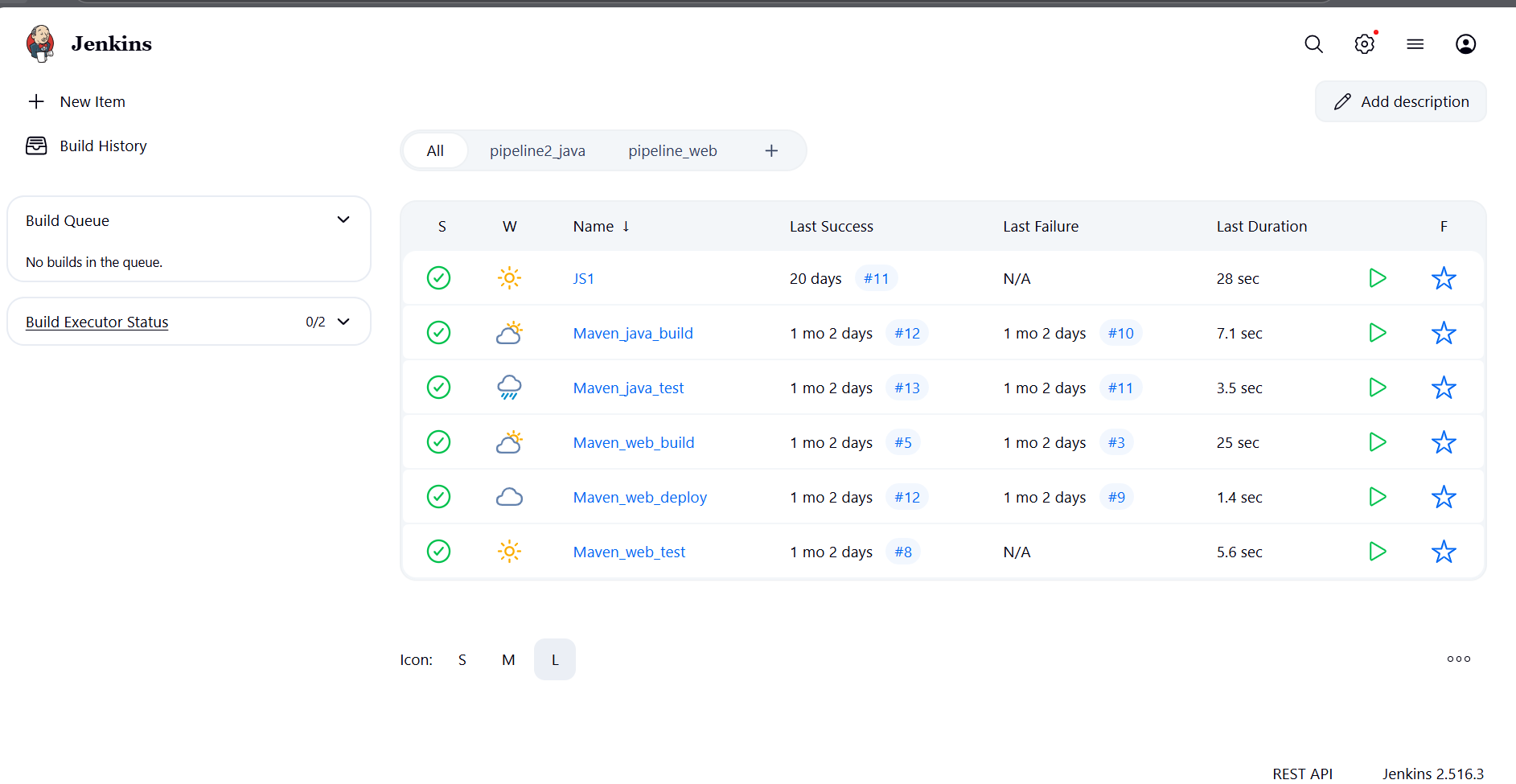
Just the push event.

* Click “Add webhook” to save.



**Step 10: Configure Jenkins to Accept GitHub Webhooks**

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



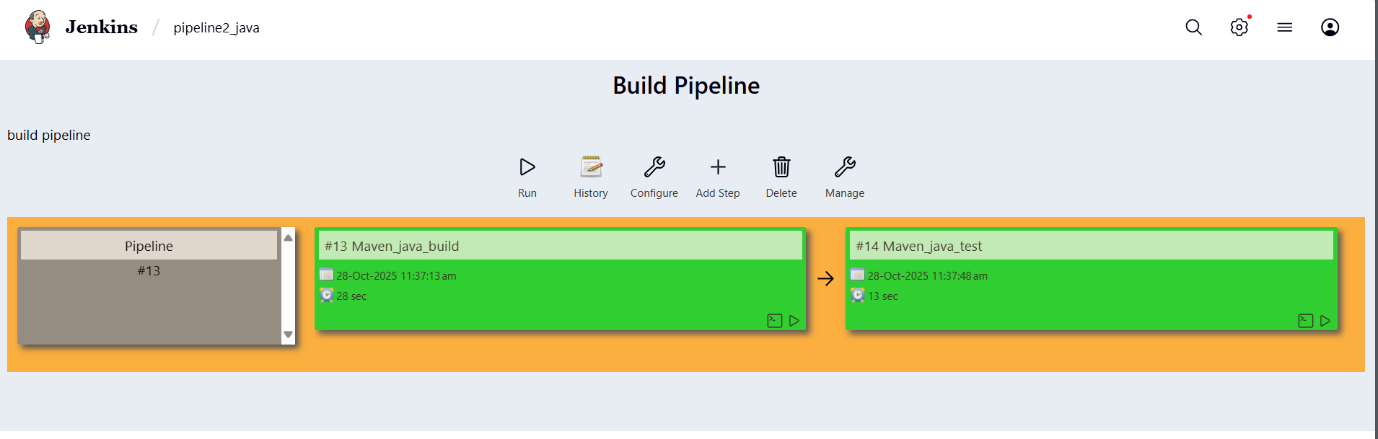
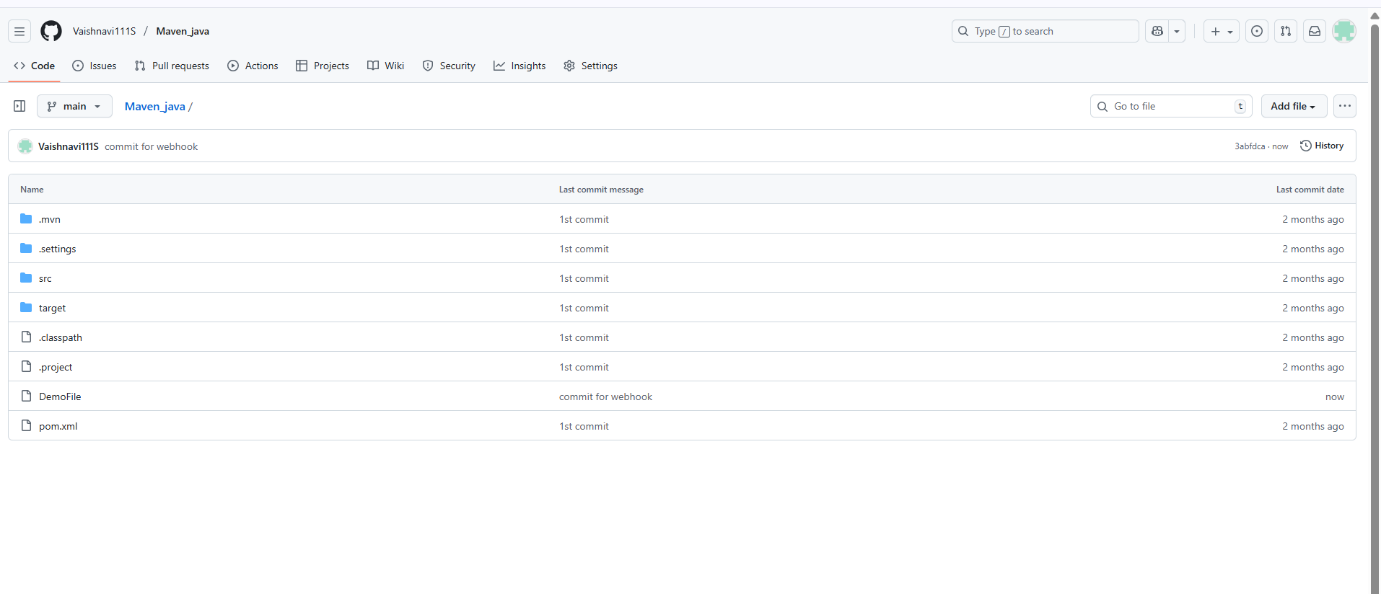
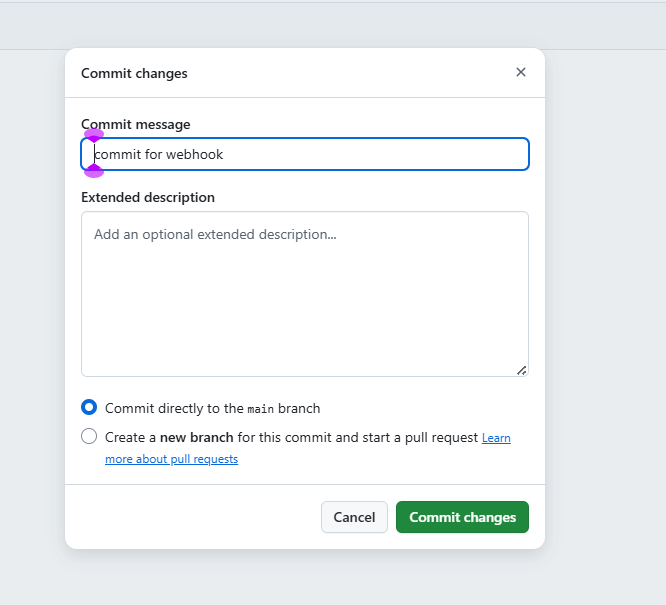
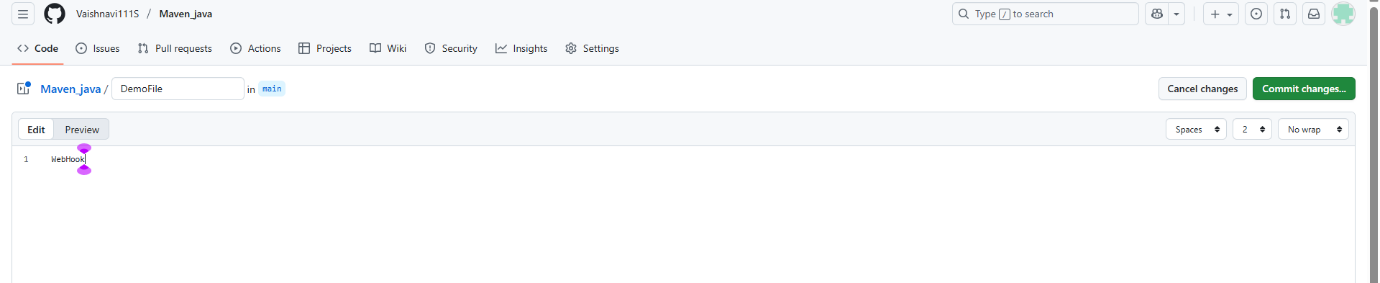
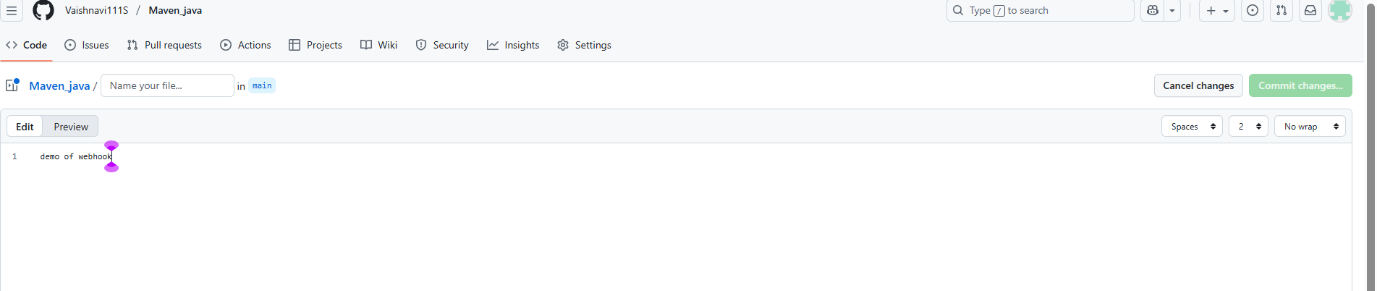
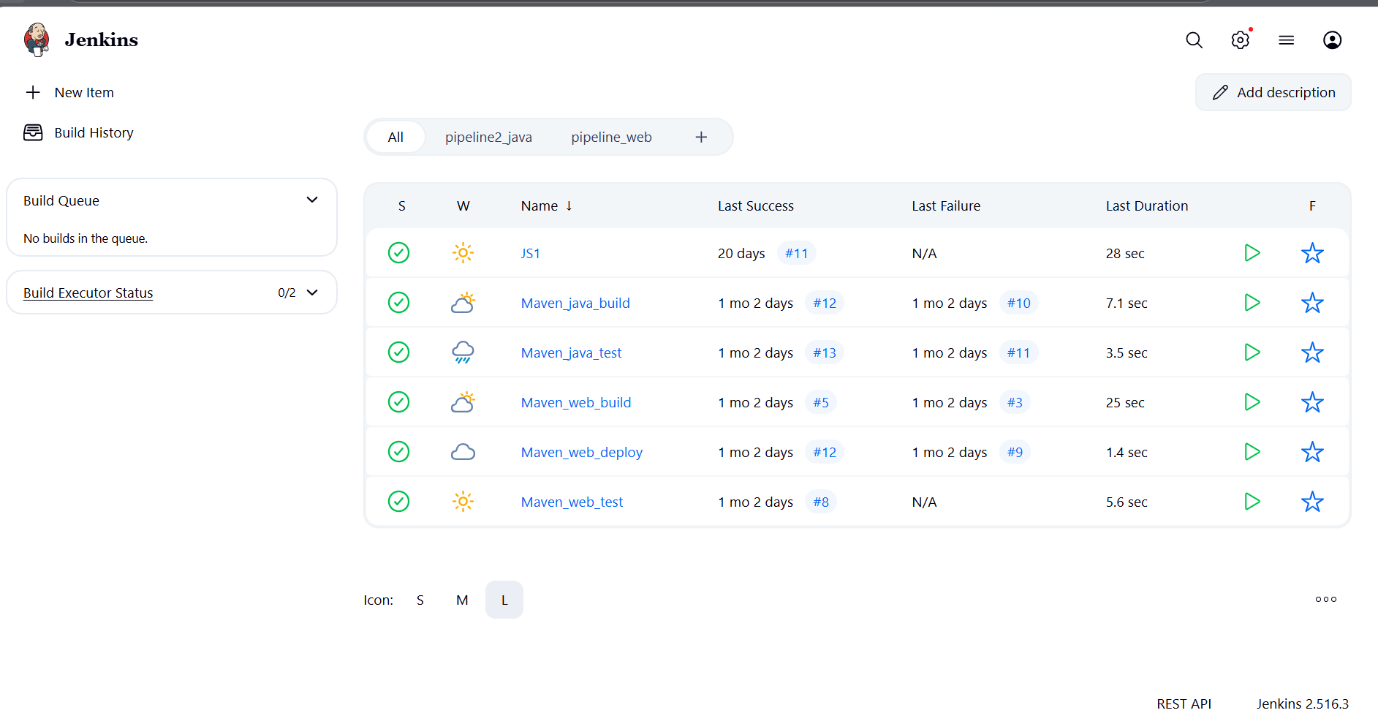
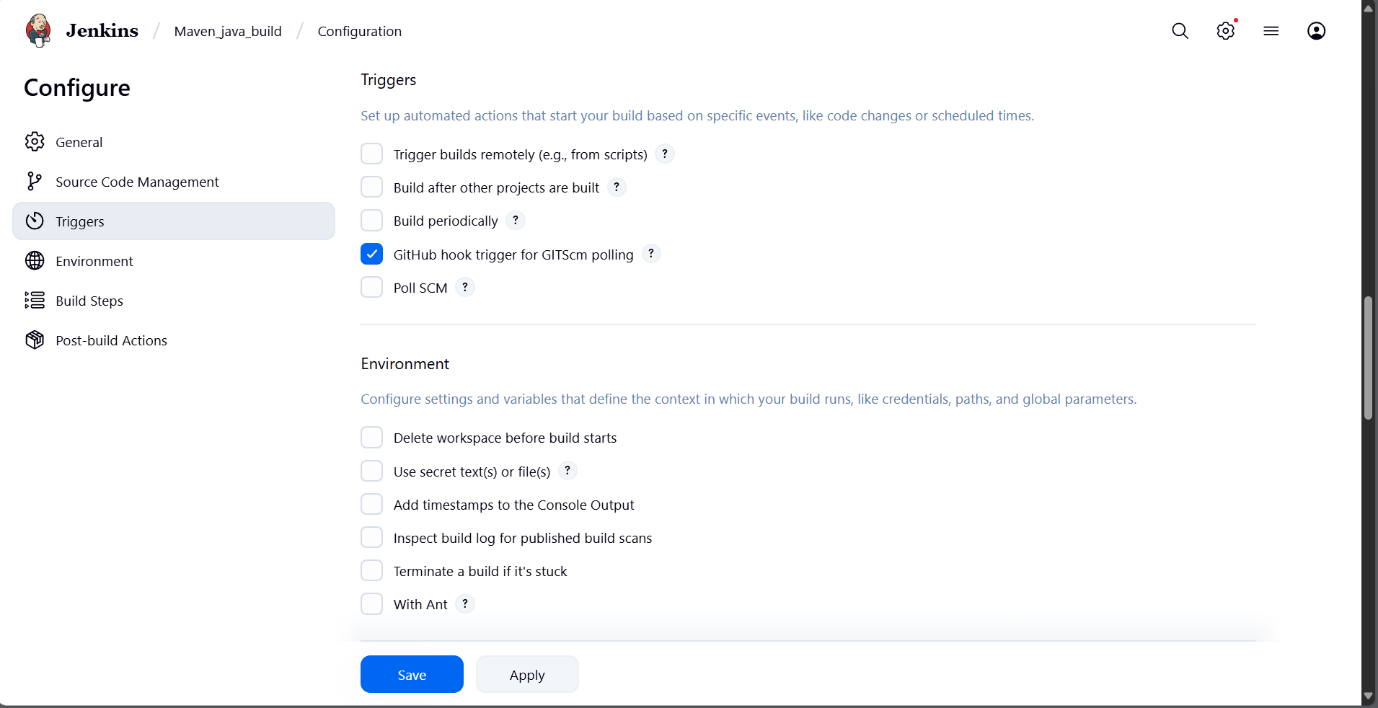
1. Click Configure.
2. Scroll down to the Build Triggers section.
3. Check the box: ✅GitHub hook trigger for GITScm polling



1. 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**.**

****

**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**](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** | ✅ 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

**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) |

**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.

1.What is Continuous Integration (CI)?

Continuous Integration (CI) is a DevOps practice where developers frequently merge their code changes into a shared repository. Each integration triggers an automated build and test process to detect errors early, ensure compatibility, and maintain software quality.

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

Continuous Delivery (CD) prepares code changes for release automatically, but deployment to production may require manual approval. Continuous Deployment goes further, automatically deploying every change that passes tests to production.

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

Jenkins is an open-source automation server that automates the build, test, and deployment stages of a CI/CD pipeline. It integrates with version control systems and triggers builds automatically.

4. What is a webhook in GitHub?

A webhook in GitHub is an automated message sent to an external service (like Jenkins) when specific events occur, such as a push or pull request. It notifies Jenkins immediately to trigger builds.

5. Why are webhooks used in Jenkins integration?

Webhooks enable GitHub to instantly notify Jenkins when a commit or pull request occurs, triggering builds immediately without waiting for Jenkins to poll for changes.

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

1. Build periodically

2. Poll SCM

3. GitHub webhook trigger

4. Manual trigger

5. Upstream/Downstream triggers

6. Build after other projects are built.

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

Polling: Jenkins periodically checks for changes, which can be delayed and inefficient.

Webhook: GitHub instantly notifies Jenkins when changes occur, making it faster and more efficient.

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

ngrok is a tool that creates a secure public URL for a local machine. It’s used so GitHub webhooks can reach a locally running Jenkins server.

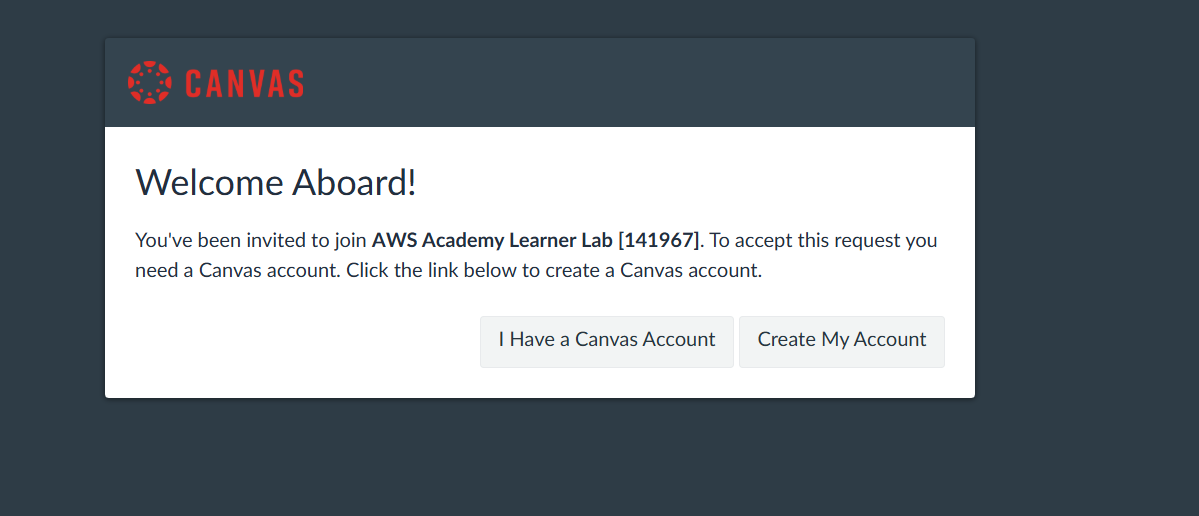
9. How does ngrok help in setting up webhooks for Jenkins running on a local machine?

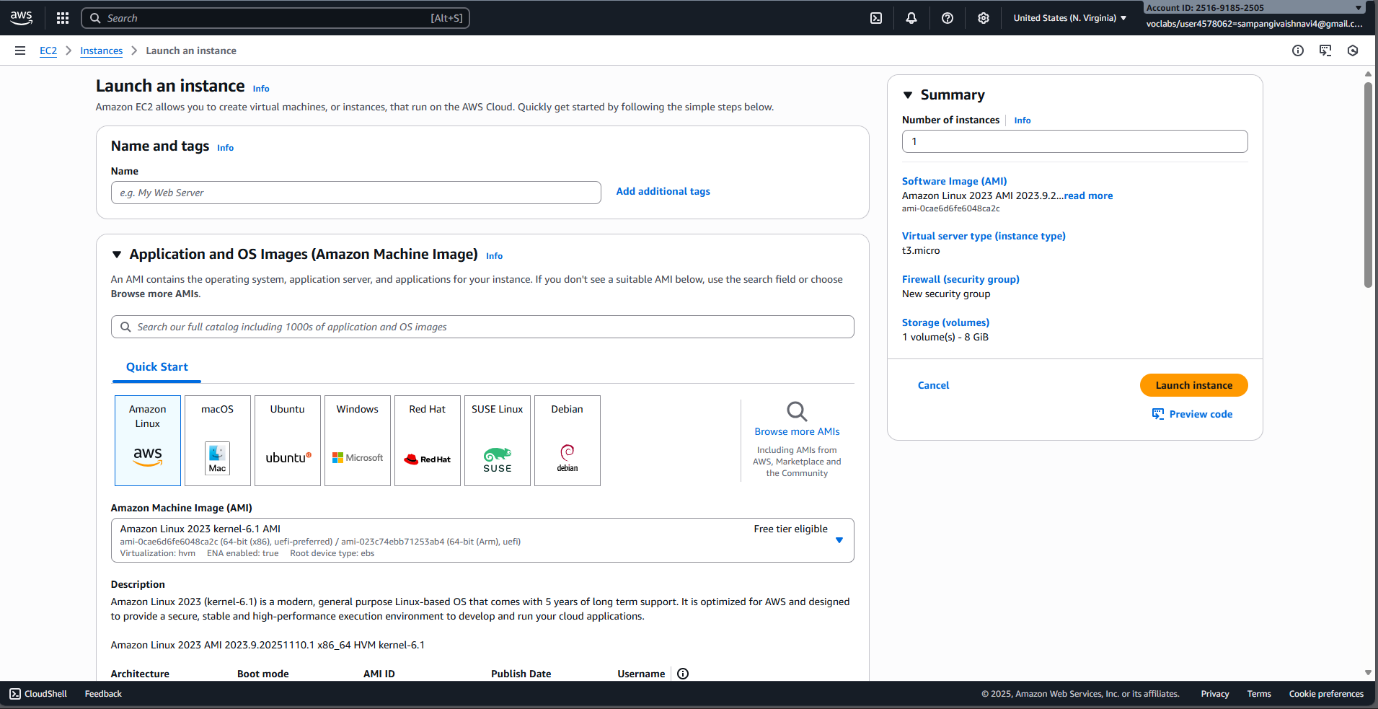
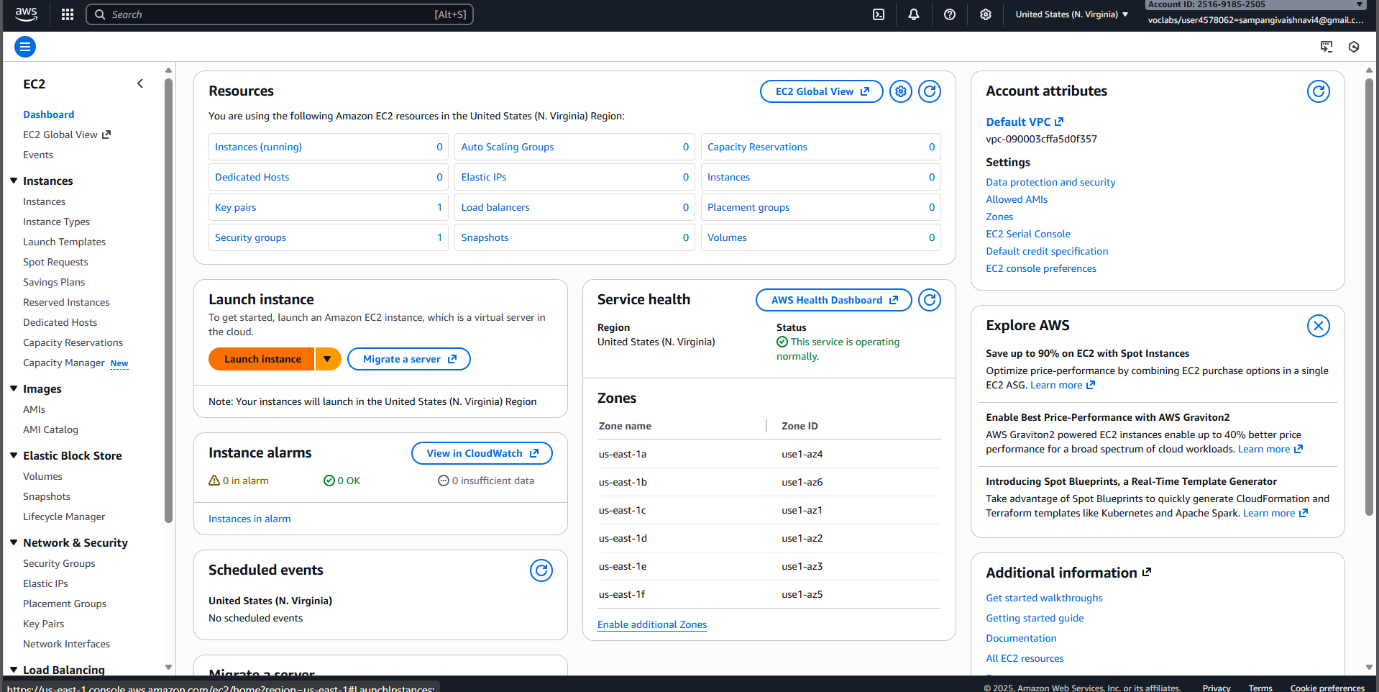
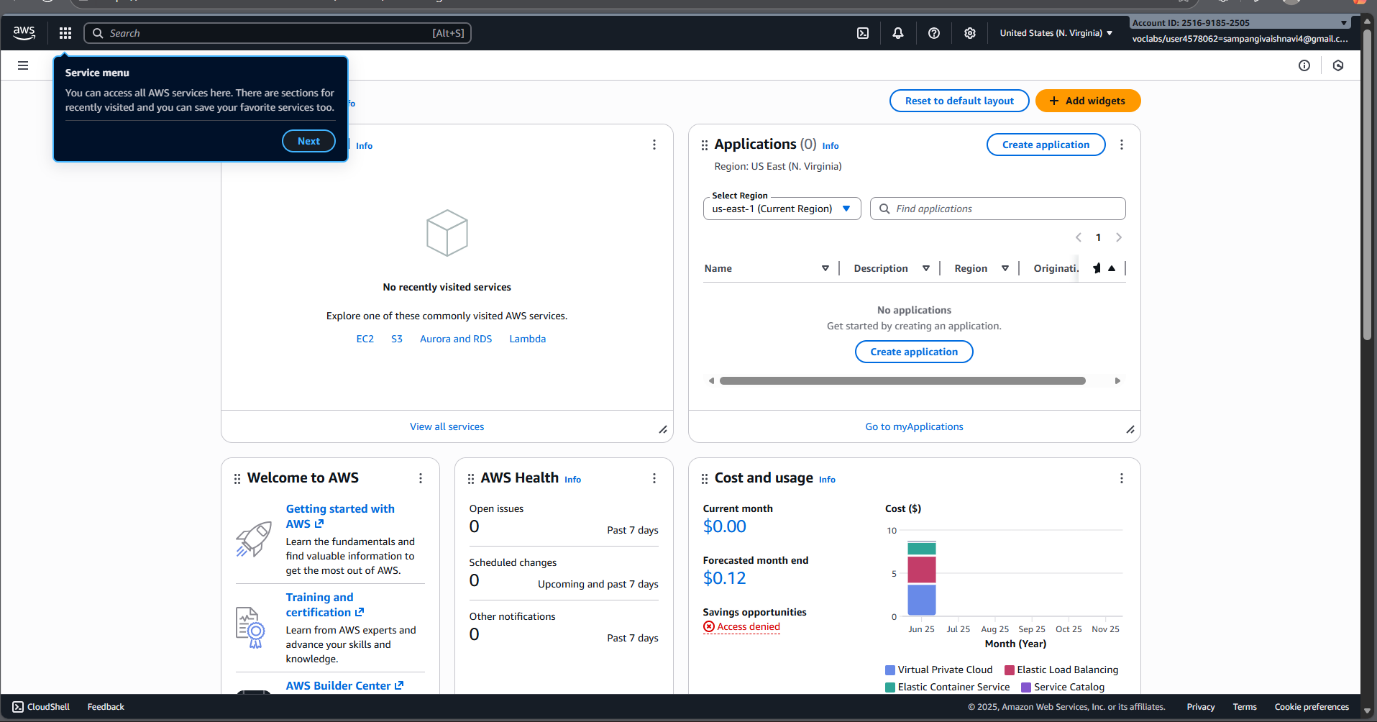
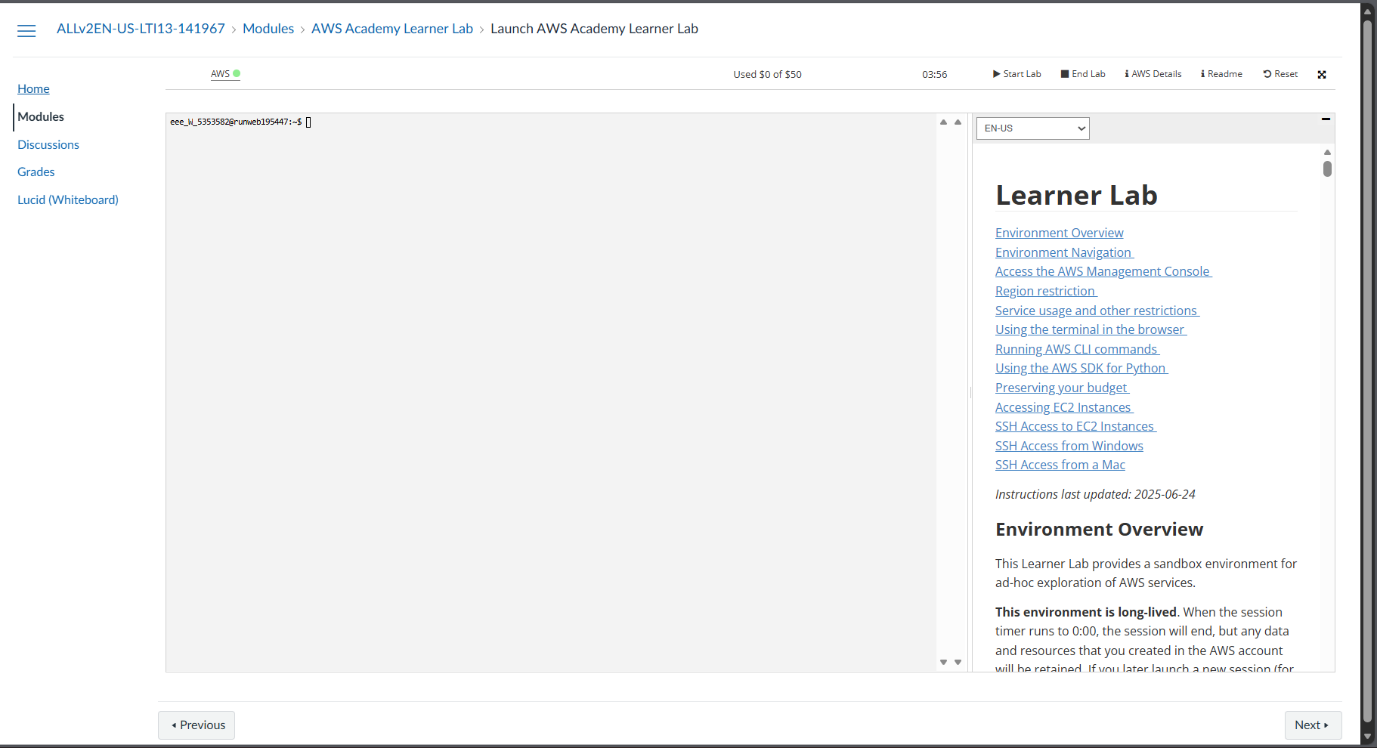
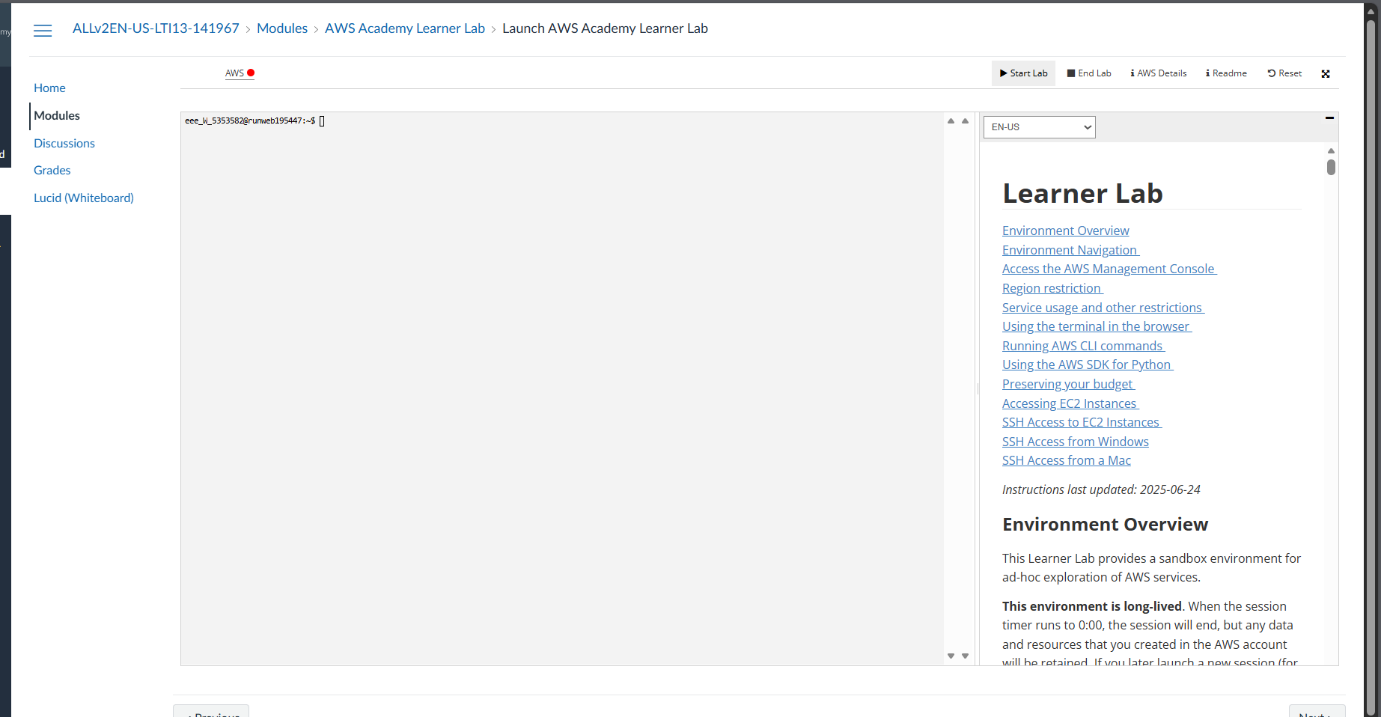
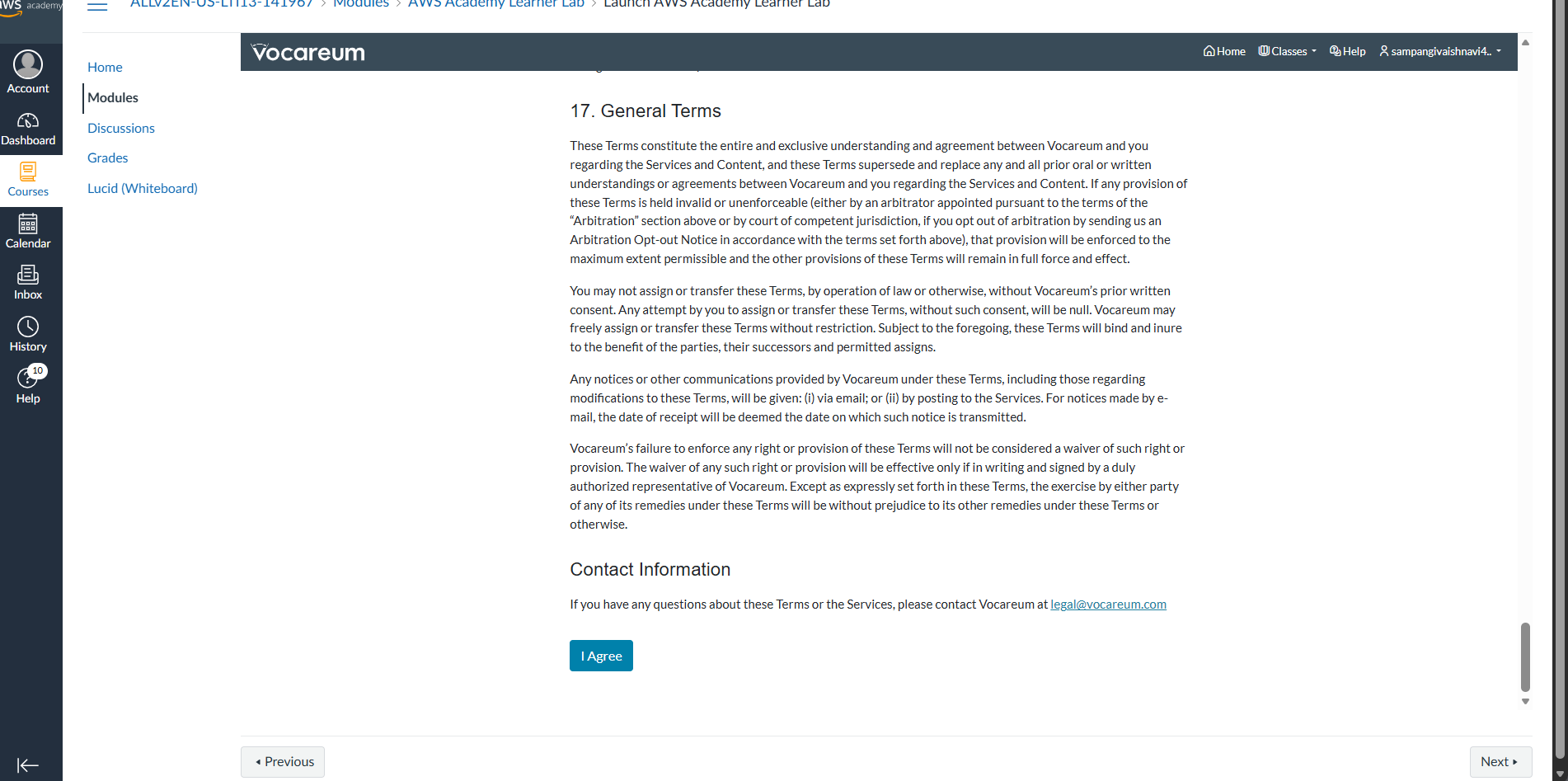
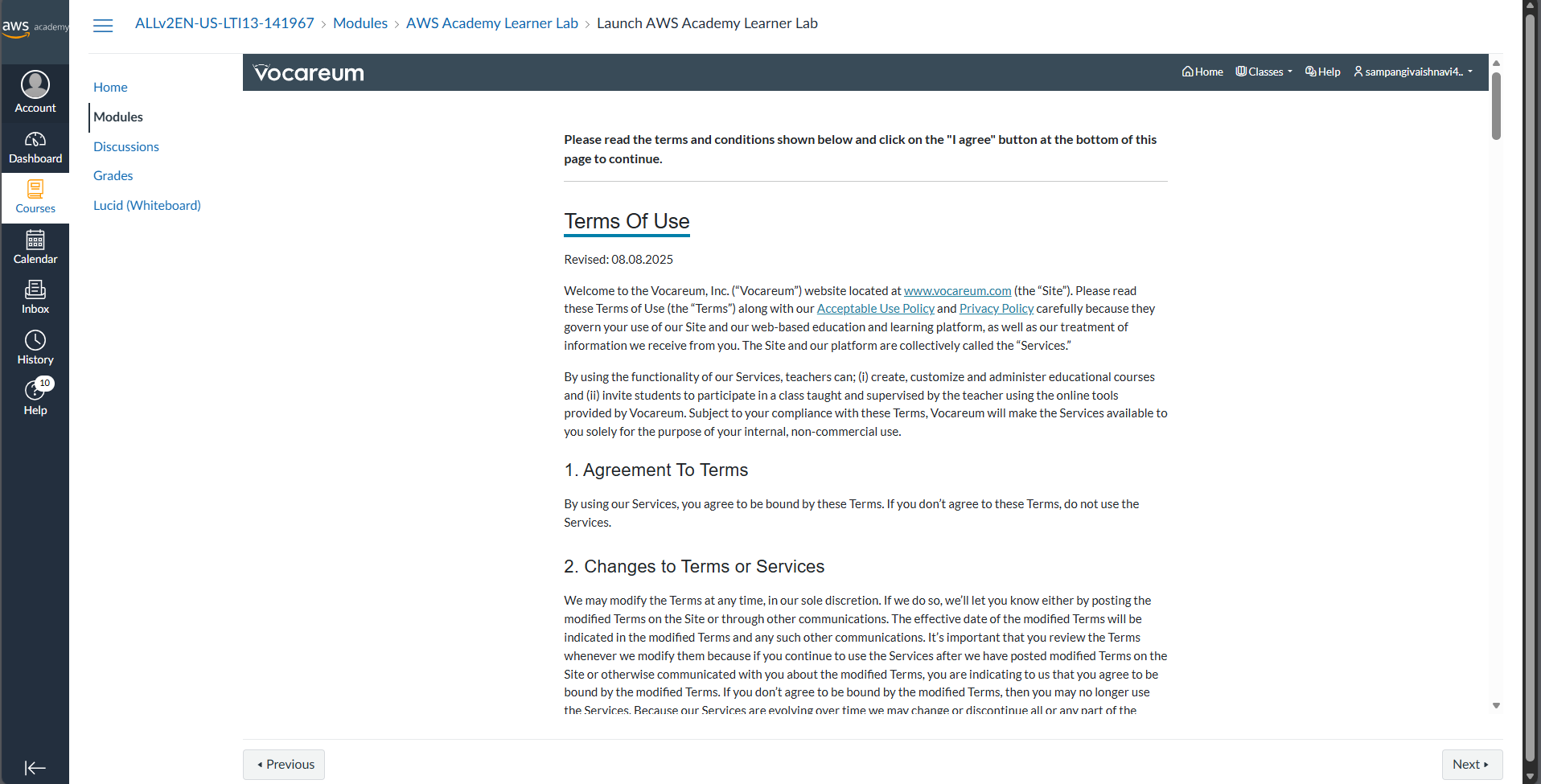
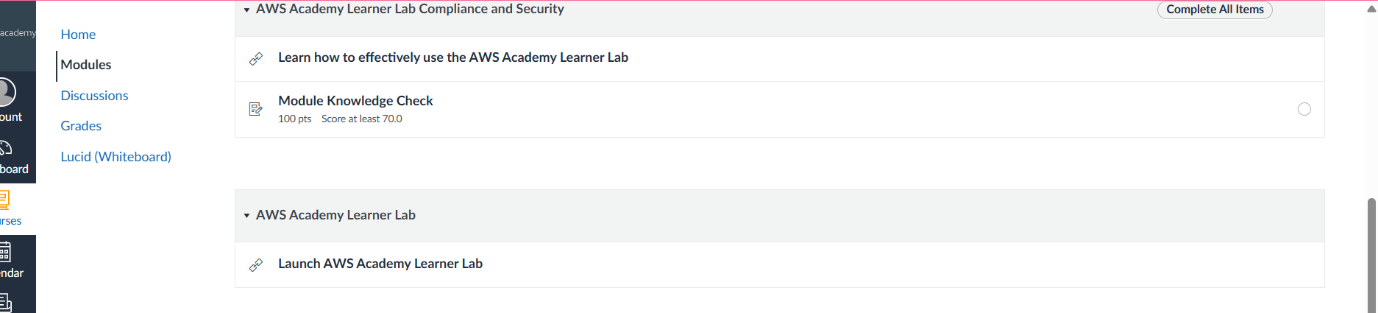
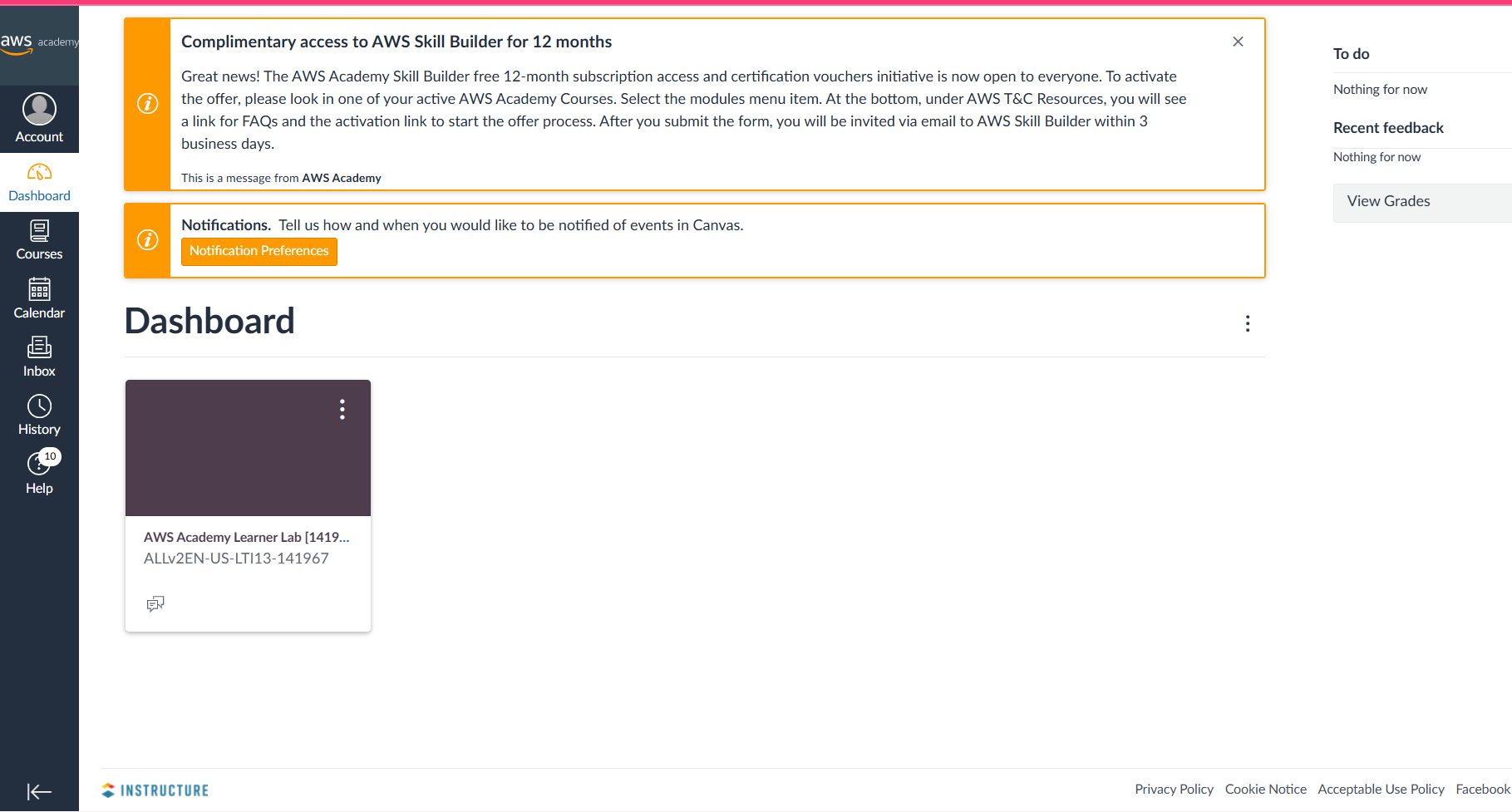
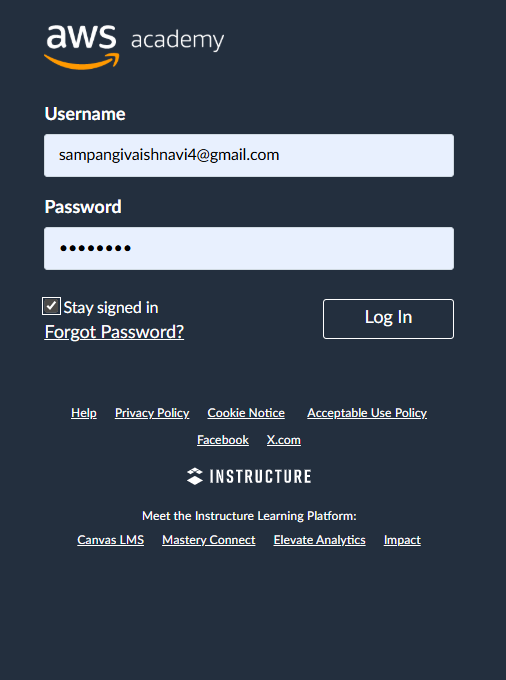
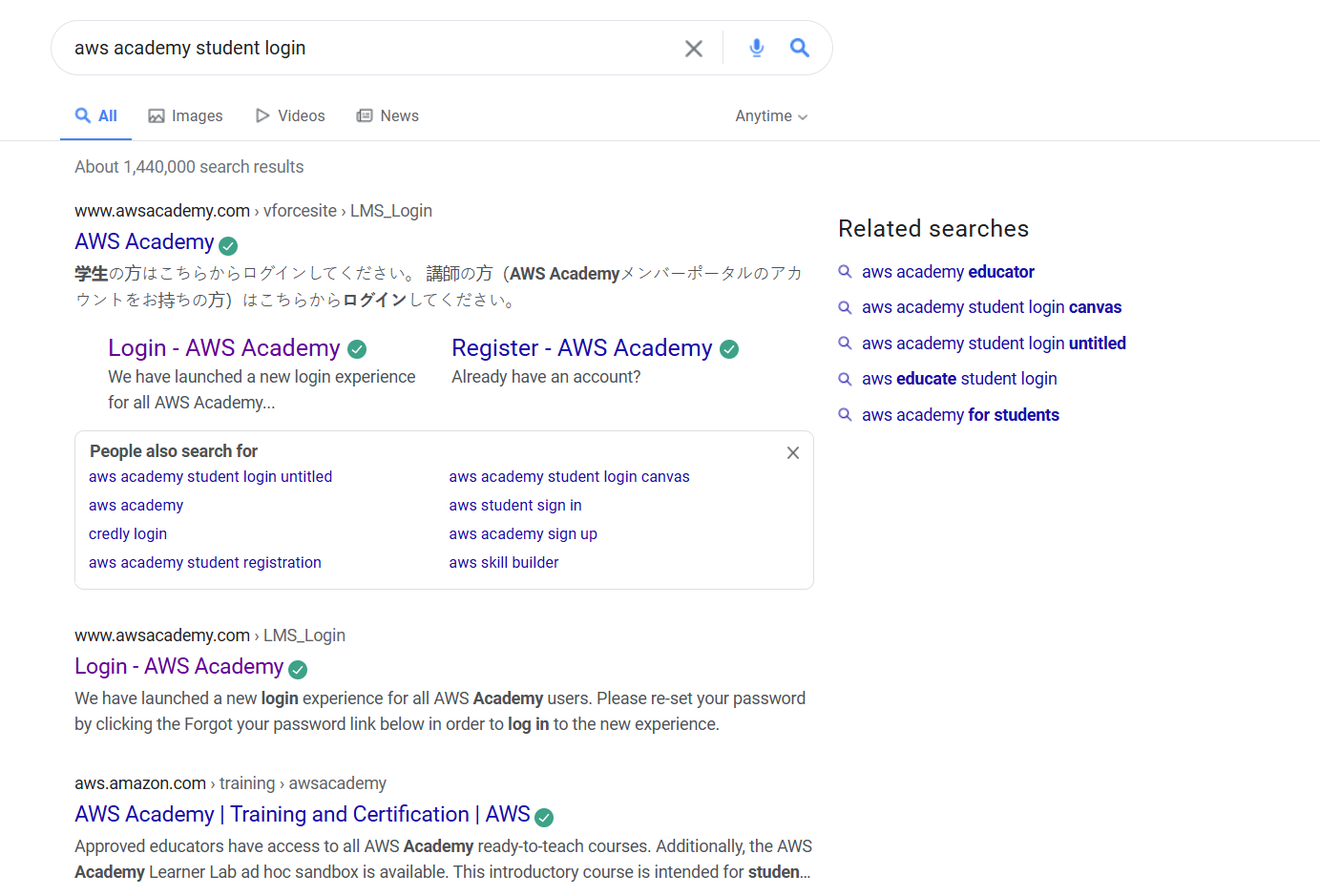
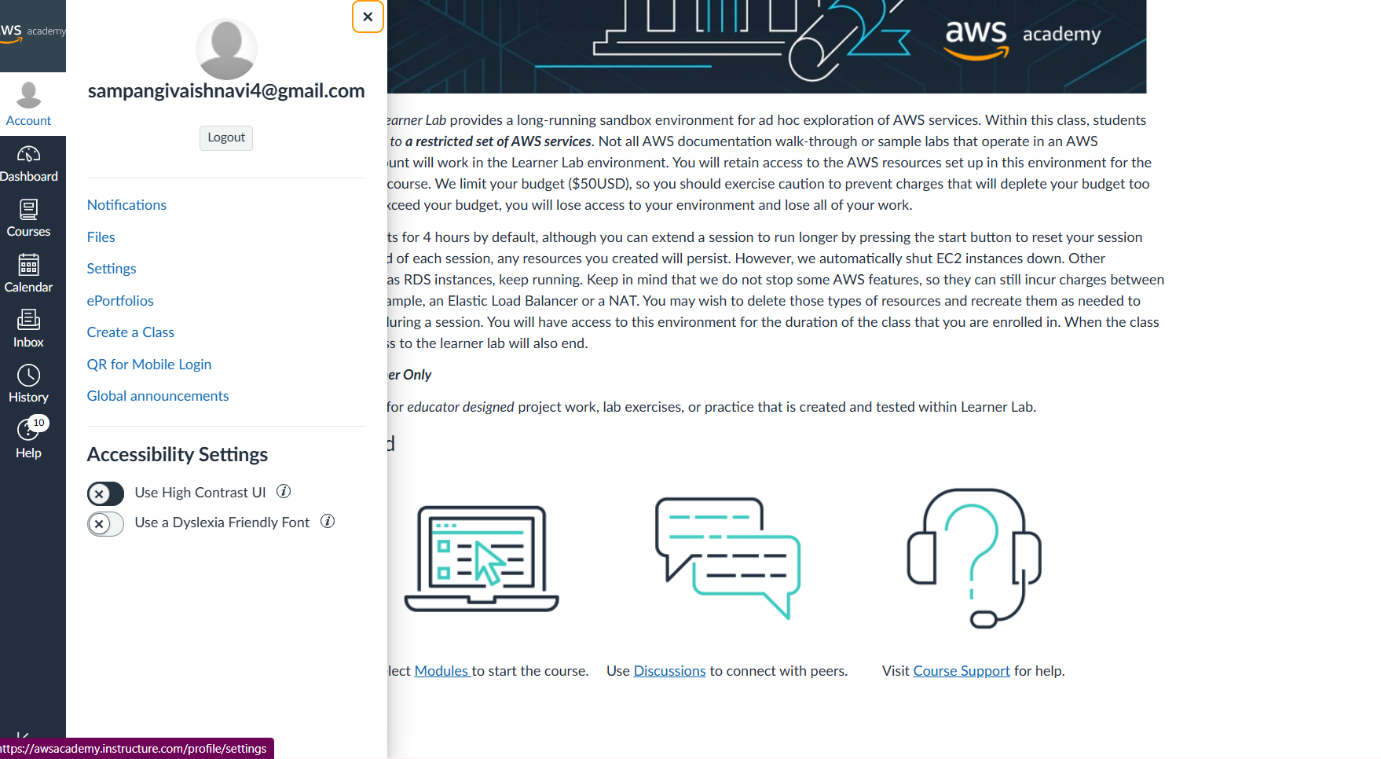
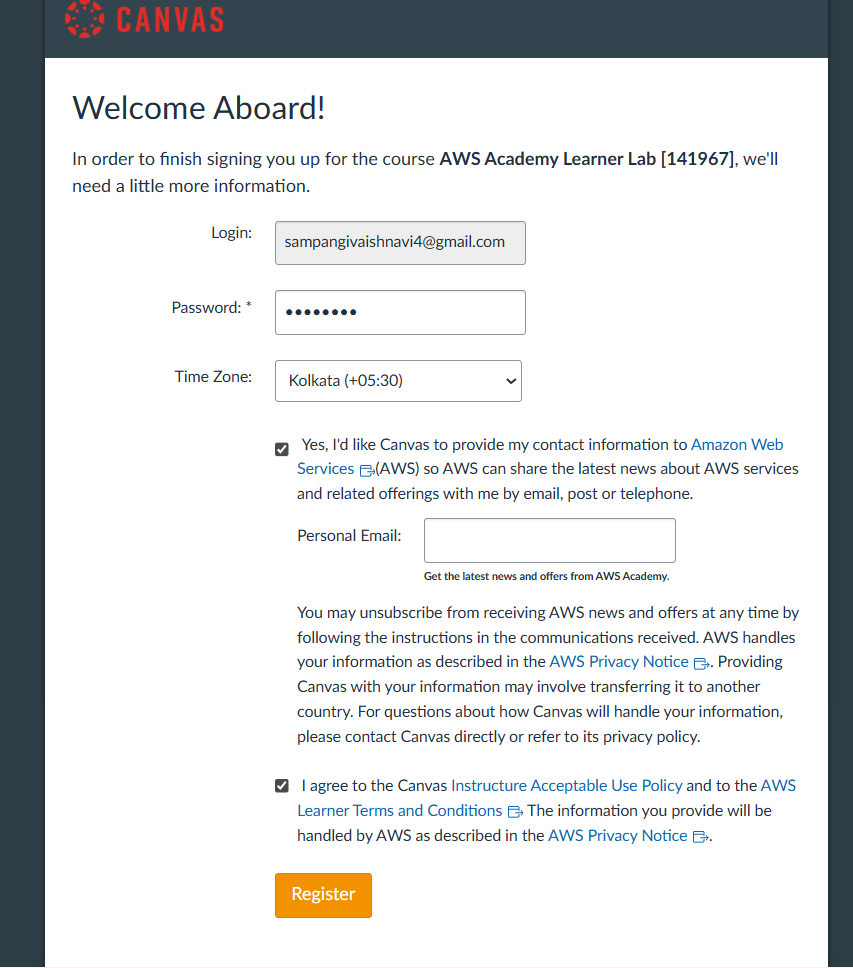
When Jenkins runs locally (e.g., on http://localhost:8080), GitHub can’t reach it. Running 'ngrok http 8080' creates a public URL (e.g., https://abcd1234.ngrok.io), which GitHub uses to send webhook events to Jenkins.

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

Email notifications inform developers about build results (success, failure, unstable). They help monitor CI/CD progress and allow quick response to issues.

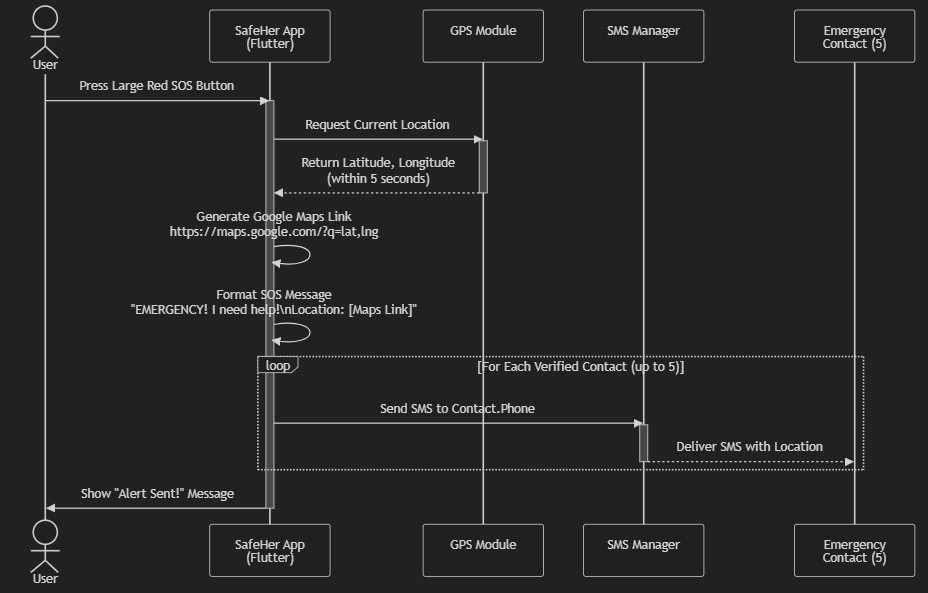
# AWS ACCOUNT :





## Sequence Diagram:

PROJECT NAME: Safeher- women safety application



## Class diagram:

