**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.
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/

Note: If Jenkins is running on localhost, GitHub cannot access it directly.  
**Use** [**ngrok**](https://ngrok.com/) **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**/

1. Set Content type to:  
   application/json
2. Under “Which events would you like to trigger this webhook?”, select:
   * Just the push event
3. Click “Add webhook” to save.

**Step 2: Configure Jenkins to Accept GitHub Webhooks**

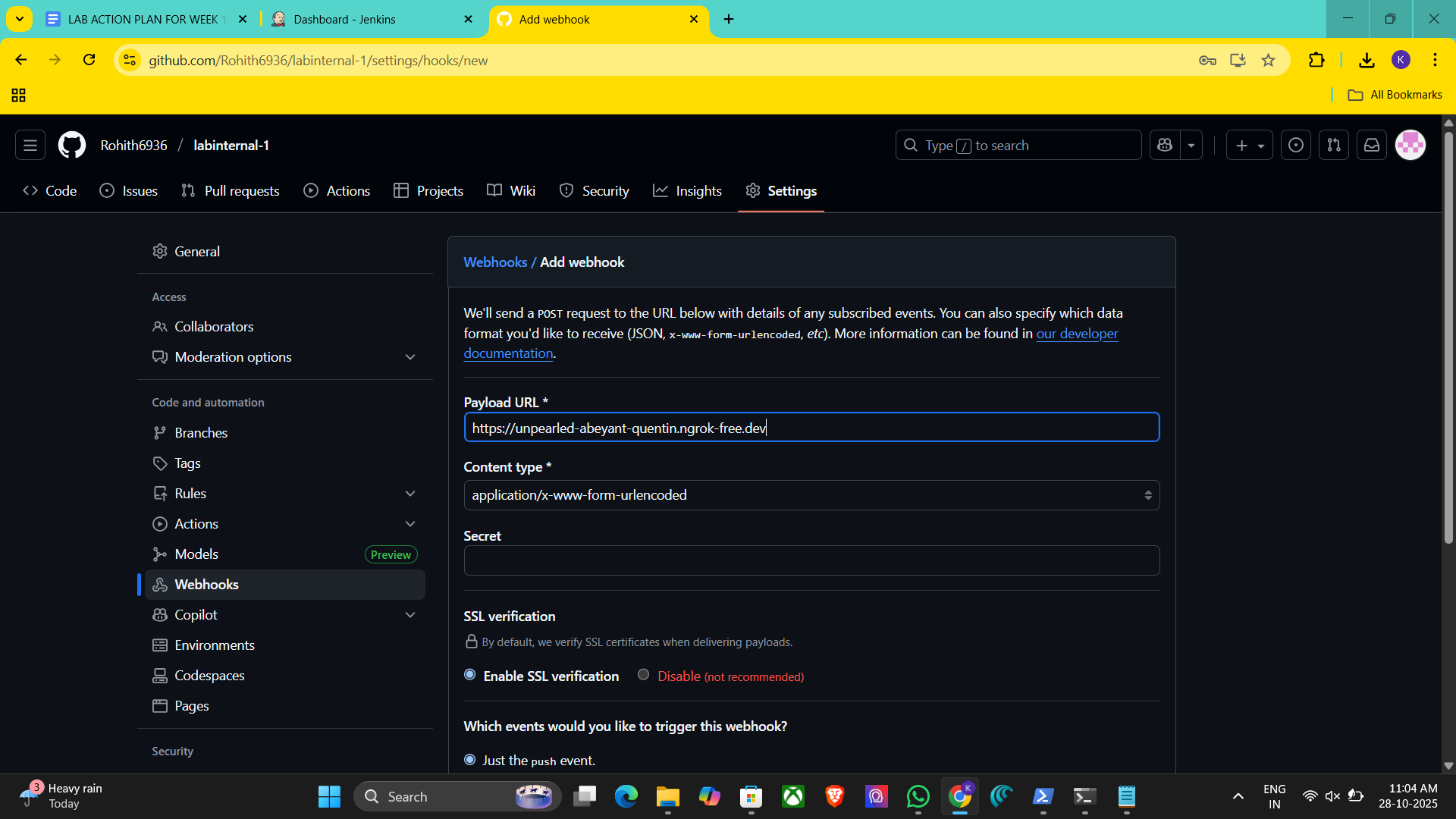
1. Open Jenkins Dashboard.
2. Select the job (freestyle or pipeline) you’ve already created.
3. Click Configure.
4. Scroll down to the Build Triggers section.
5. Check the box: ✅GitHub hook trigger for GITScm polling
6. Click Save**.**

**Step 3: Test the Setup**

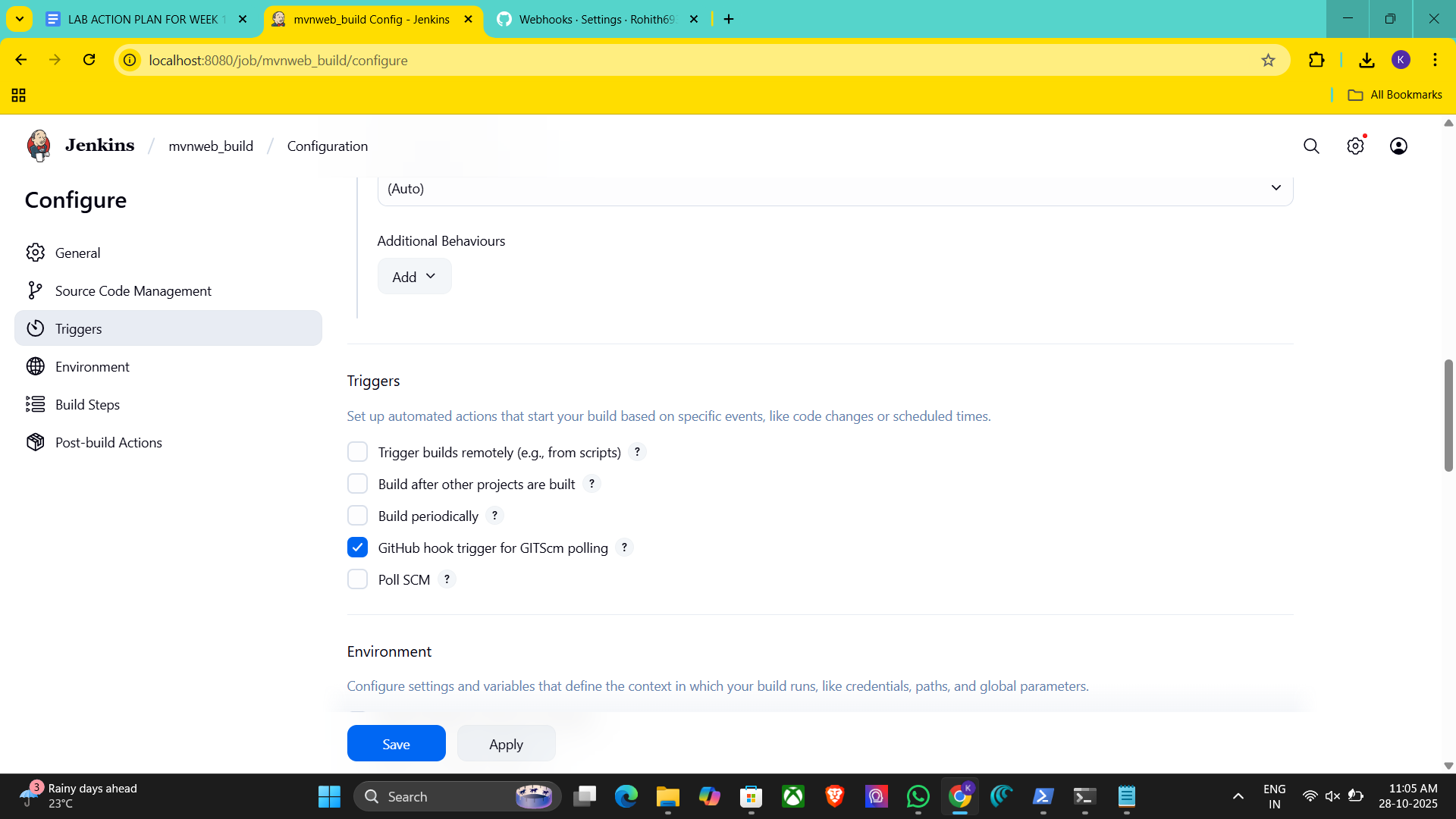
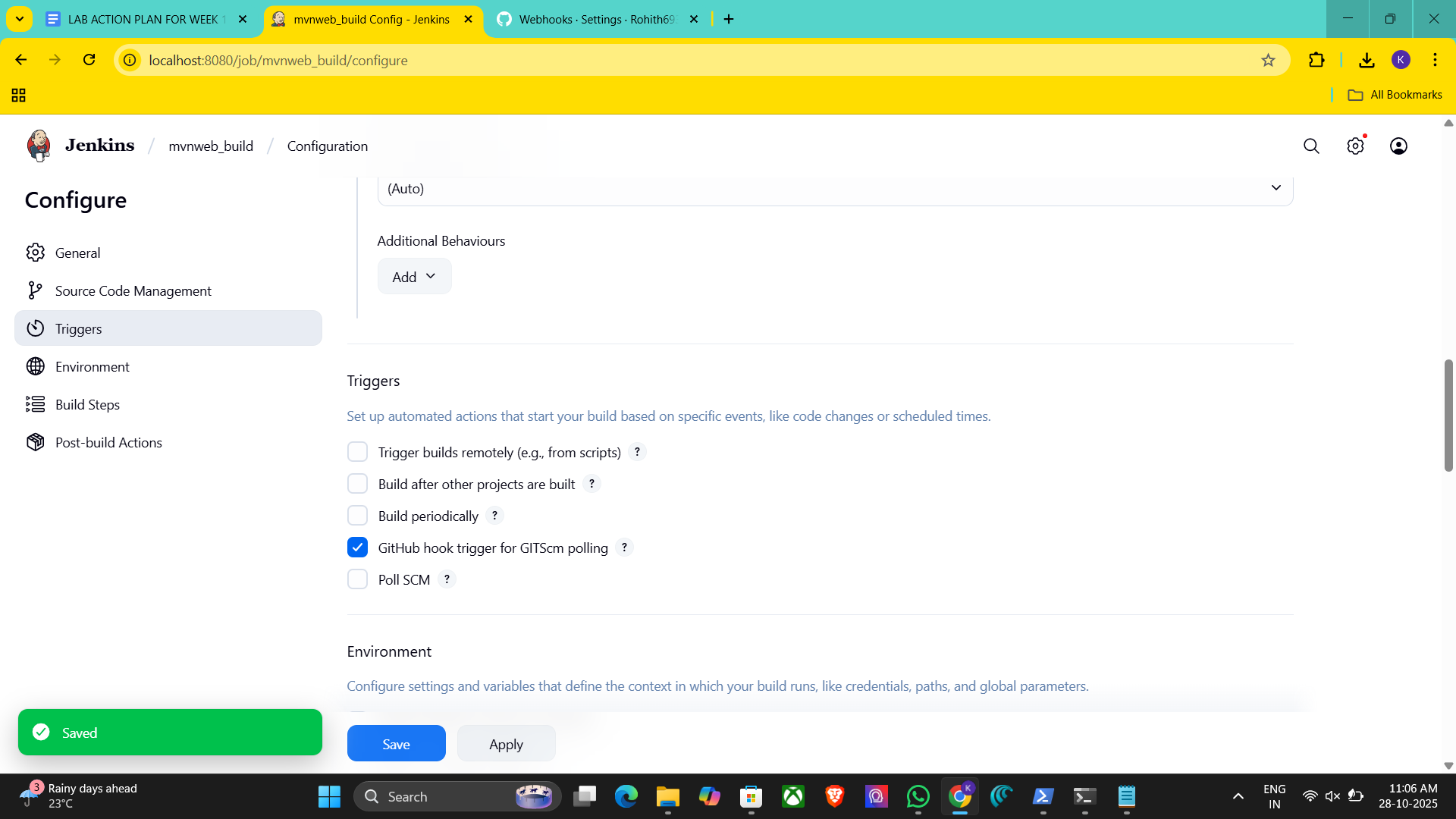
1. Make any code update in your local repo and push it to GitHub.
2. Once pushed, GitHub will trigger the webhook.
3. Jenkins will automatically detect the change and start the build pipeline**.**

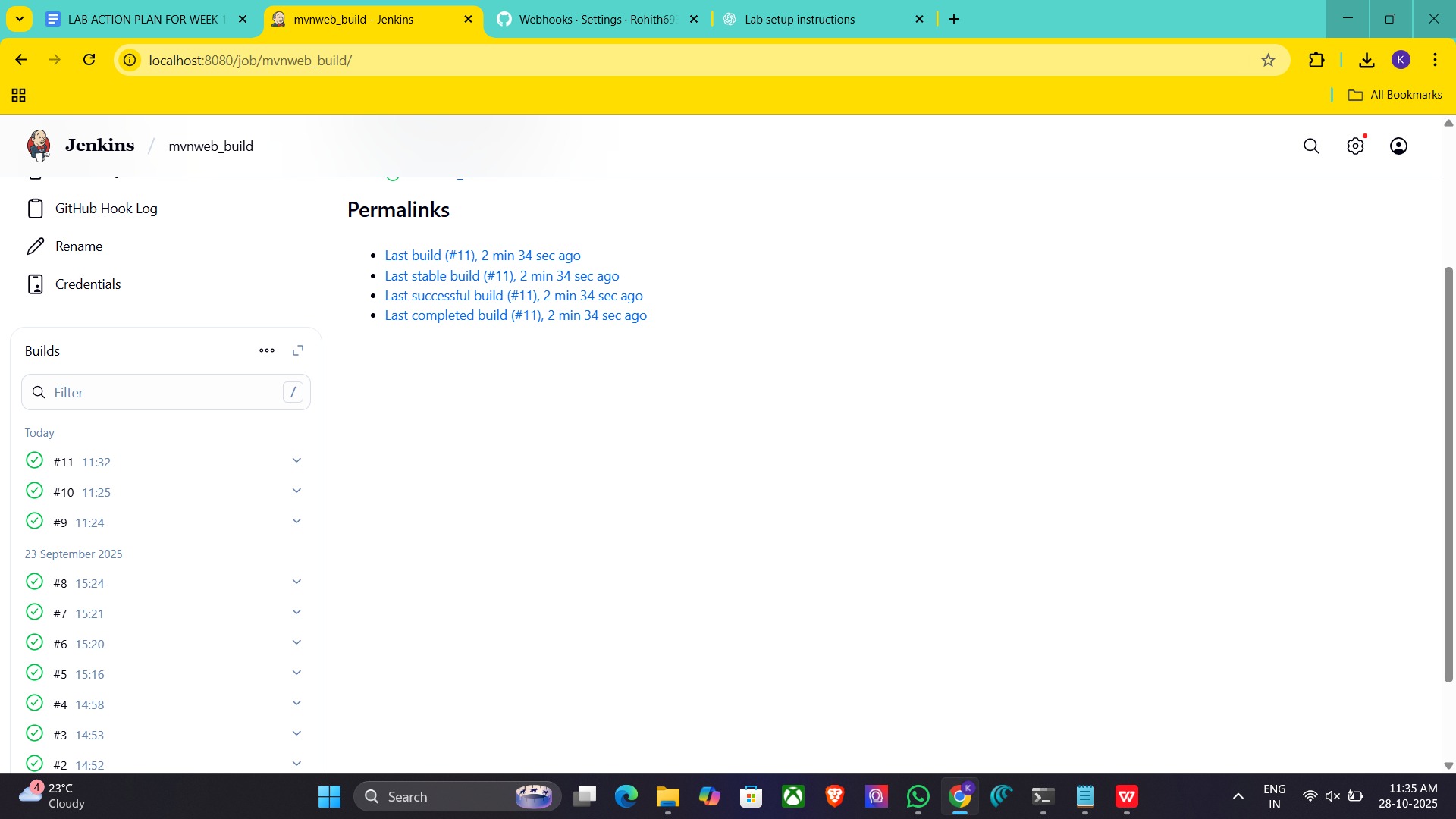
**outcome**

* 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.

****

****

****

****

**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).

**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.**

**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.**

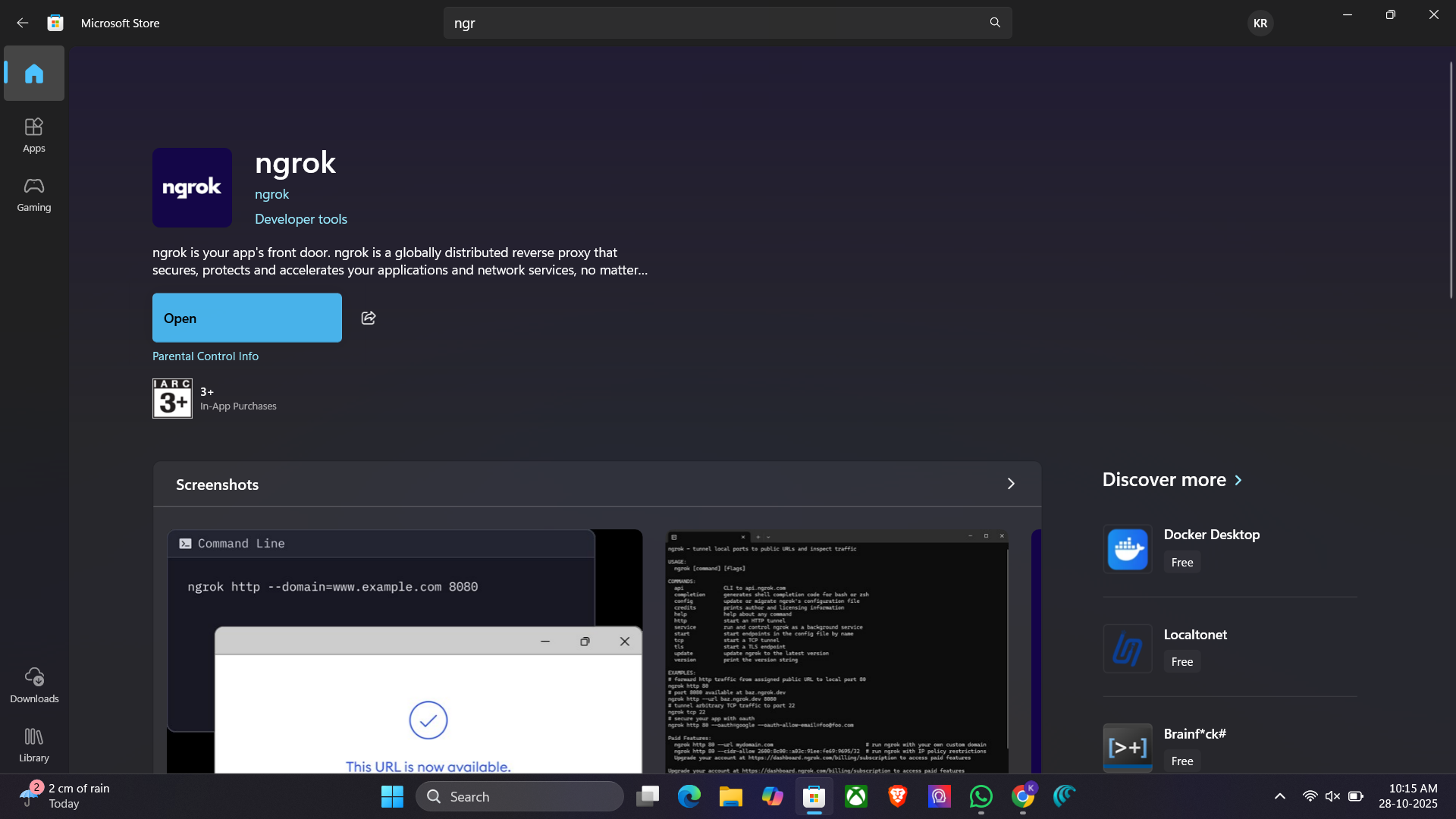
**Step 4. Use it in GitHub Webhook**

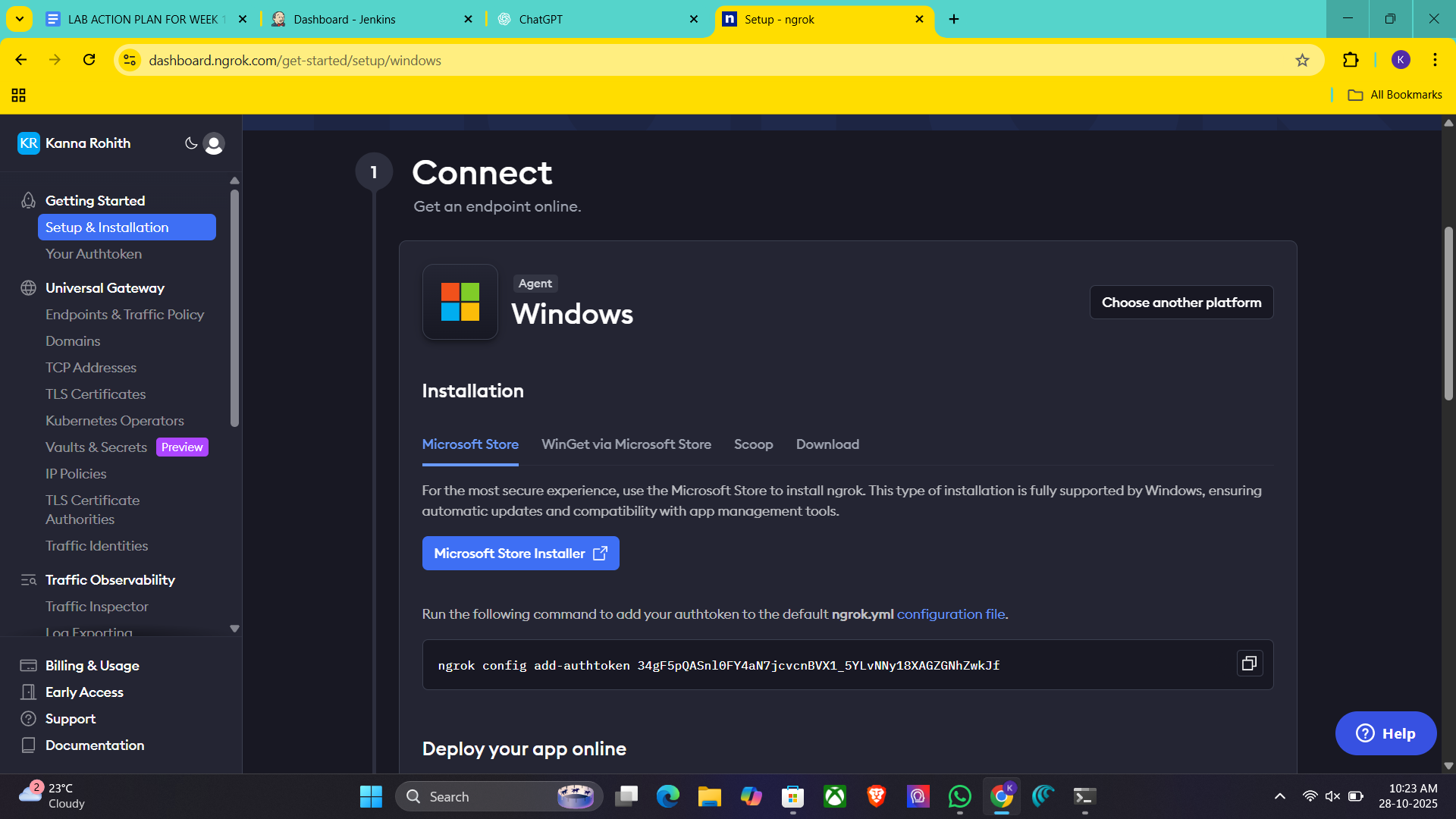
**In your GitHub repo → Settings → Webhooks:**

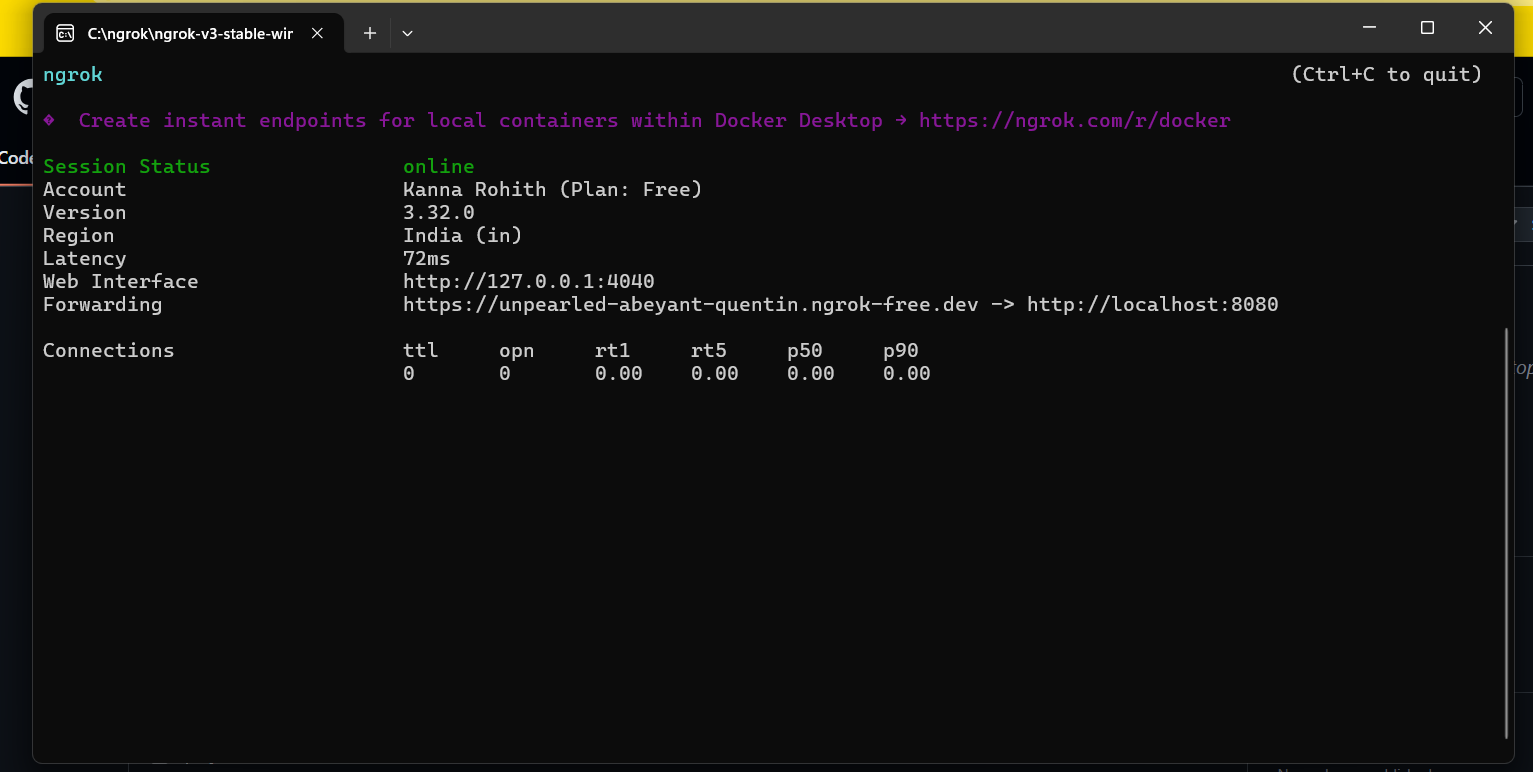
* **Payload URL: *[paste the url generated by ngrok]***

**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.**

****

****

****

**------------------------------------------------------------------------------------------------------------------------------------------**

**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**

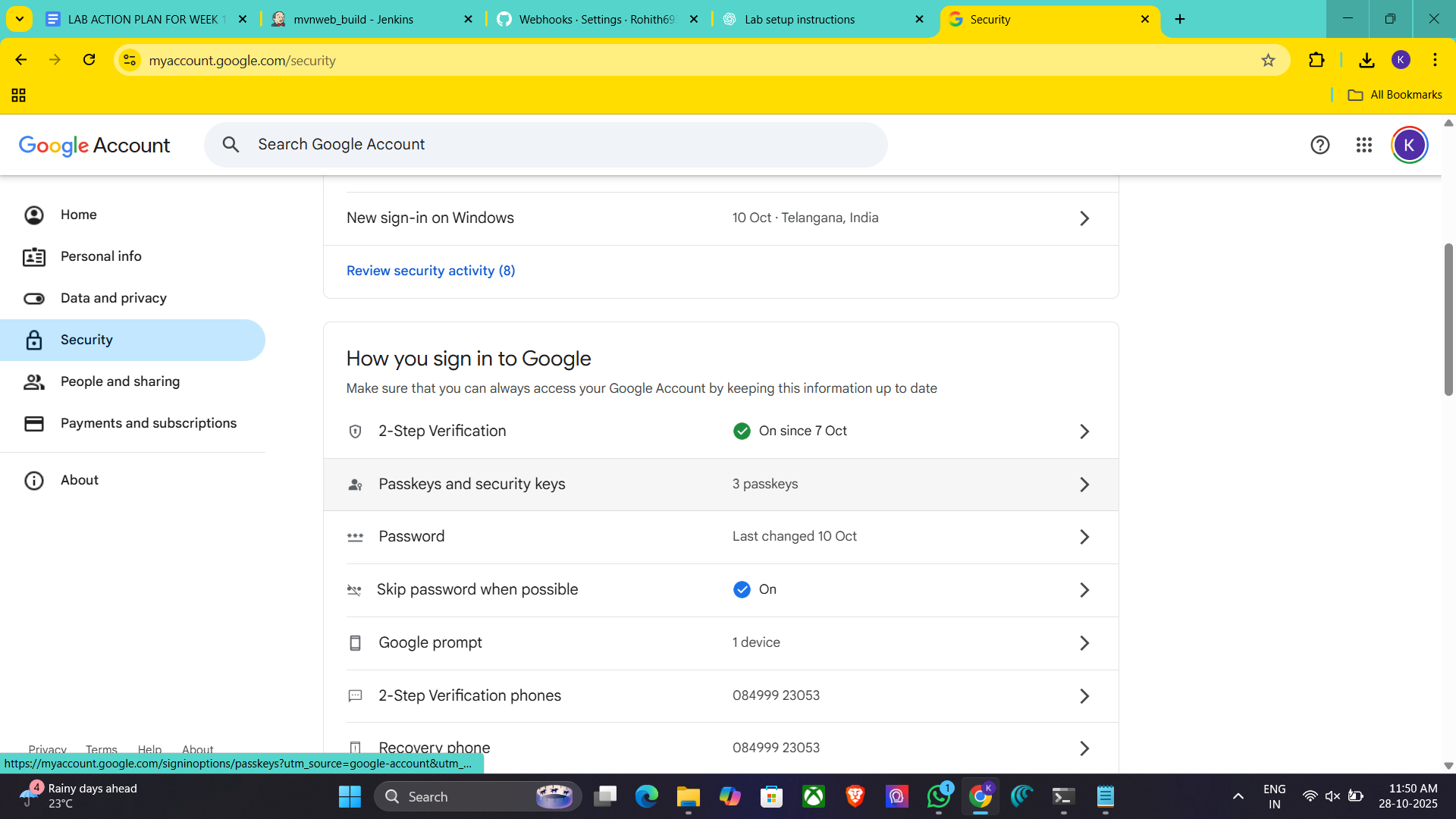
**A. Fill in the fields:**

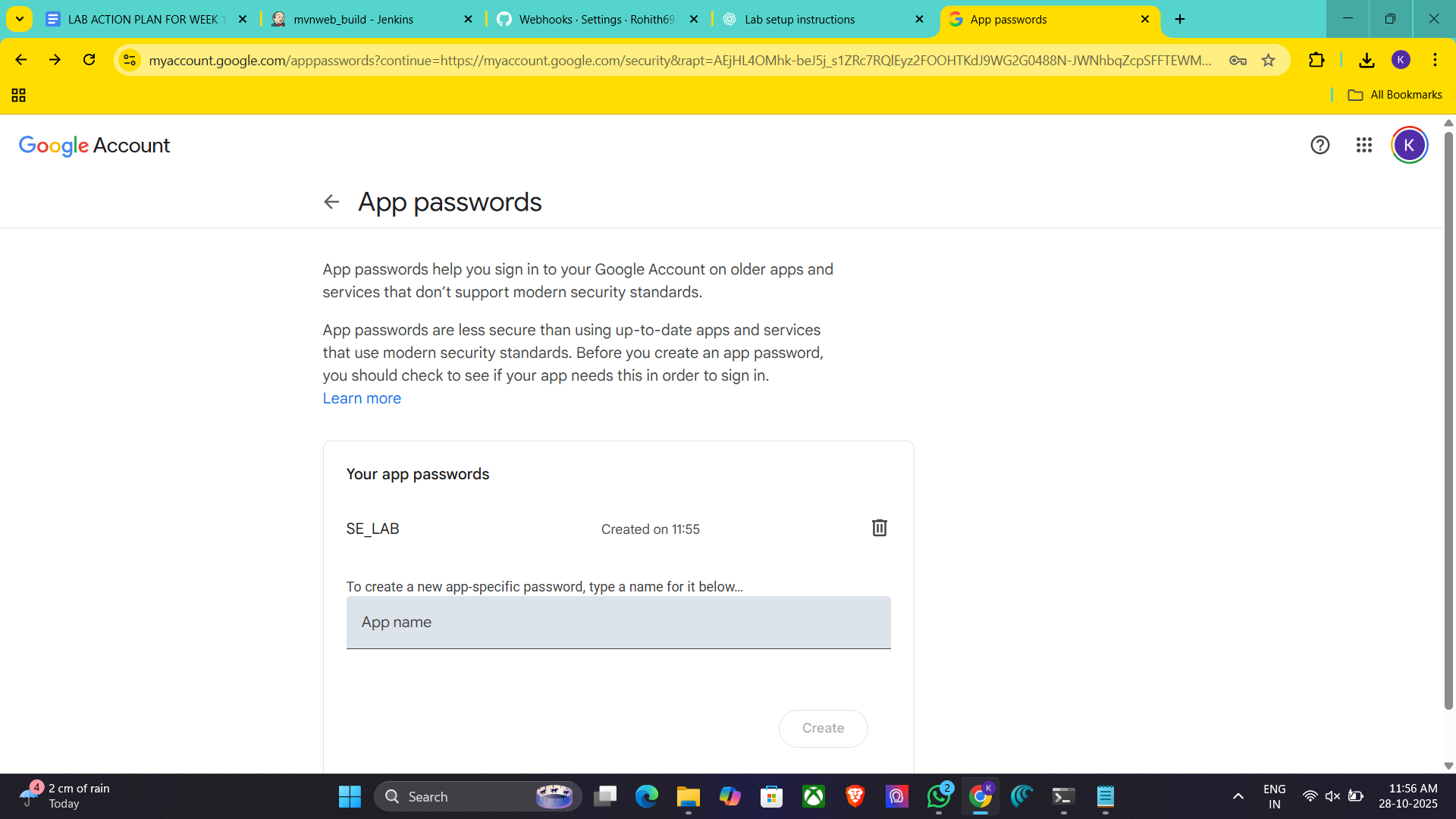
| **Field** | **Value** |
| --- | --- |
| **Project Recipient List** | Add recipient email addresses (comma-separated) |
| **Content Type** | Default (text/plain) or text/html |
| **Triggers** | Select events (e.g., Failure, Success, etc.) |
| **Attachments** | (Optional) Add logs, reports, etc. |

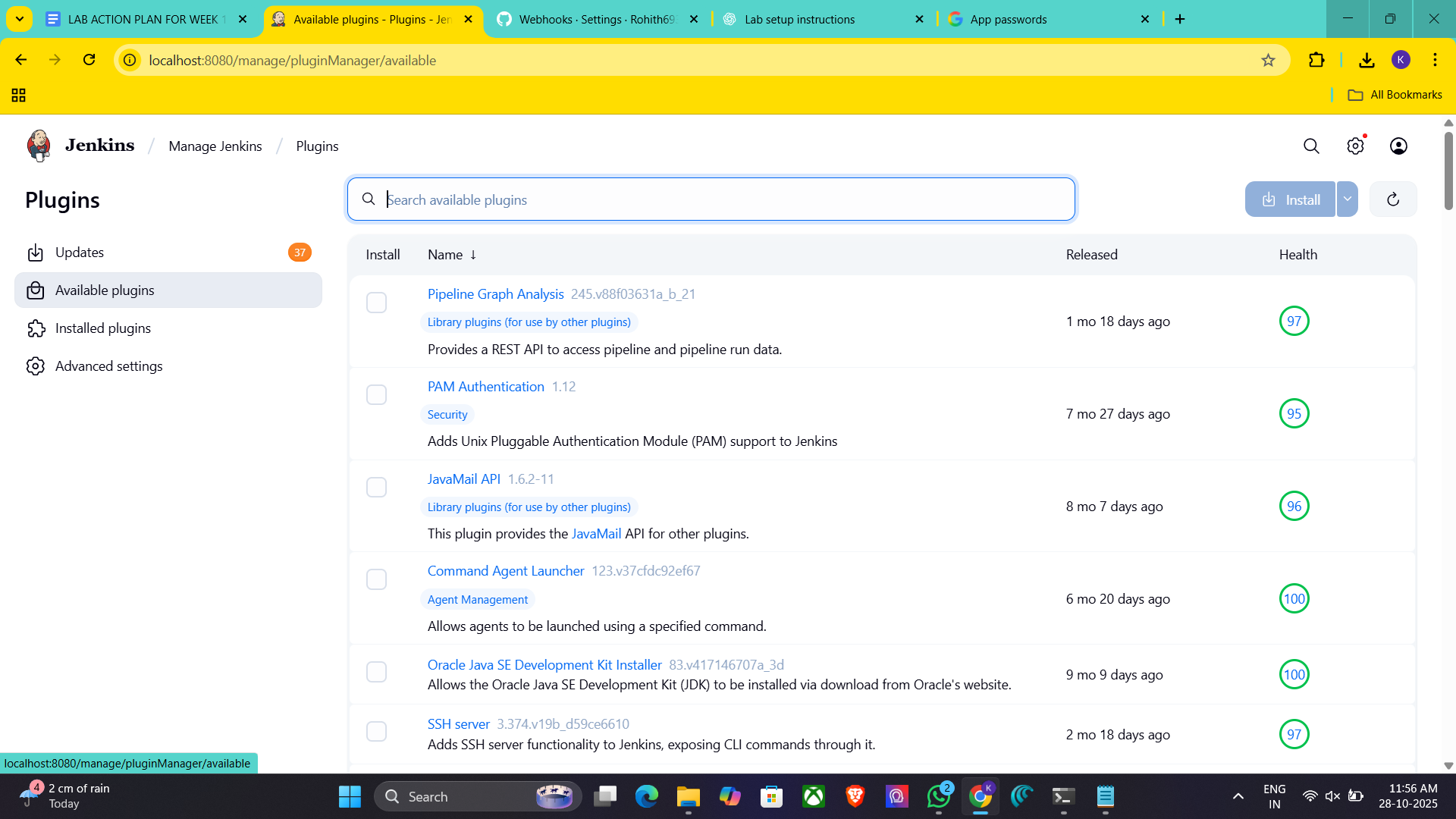
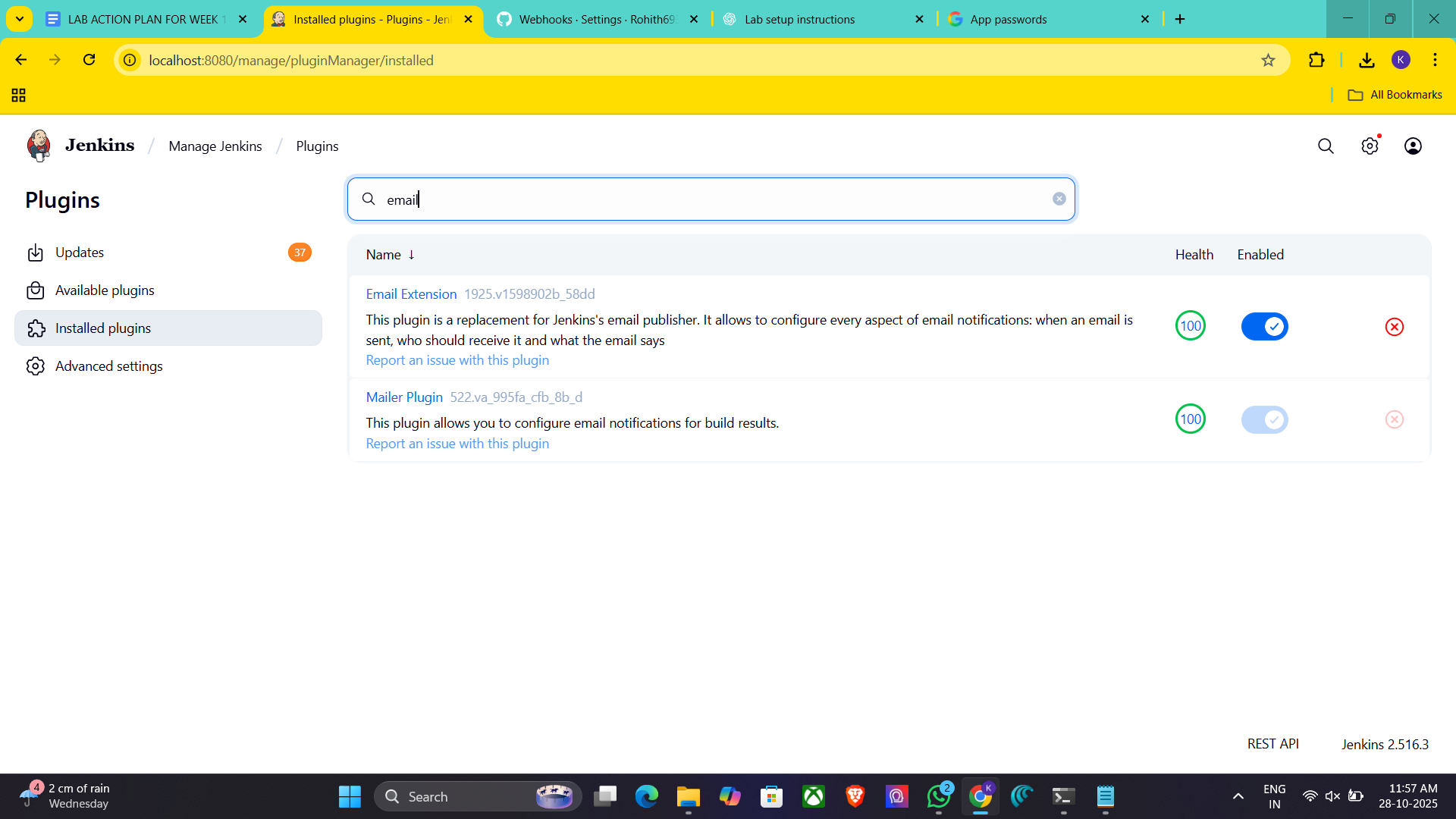
**iii. Click Save**

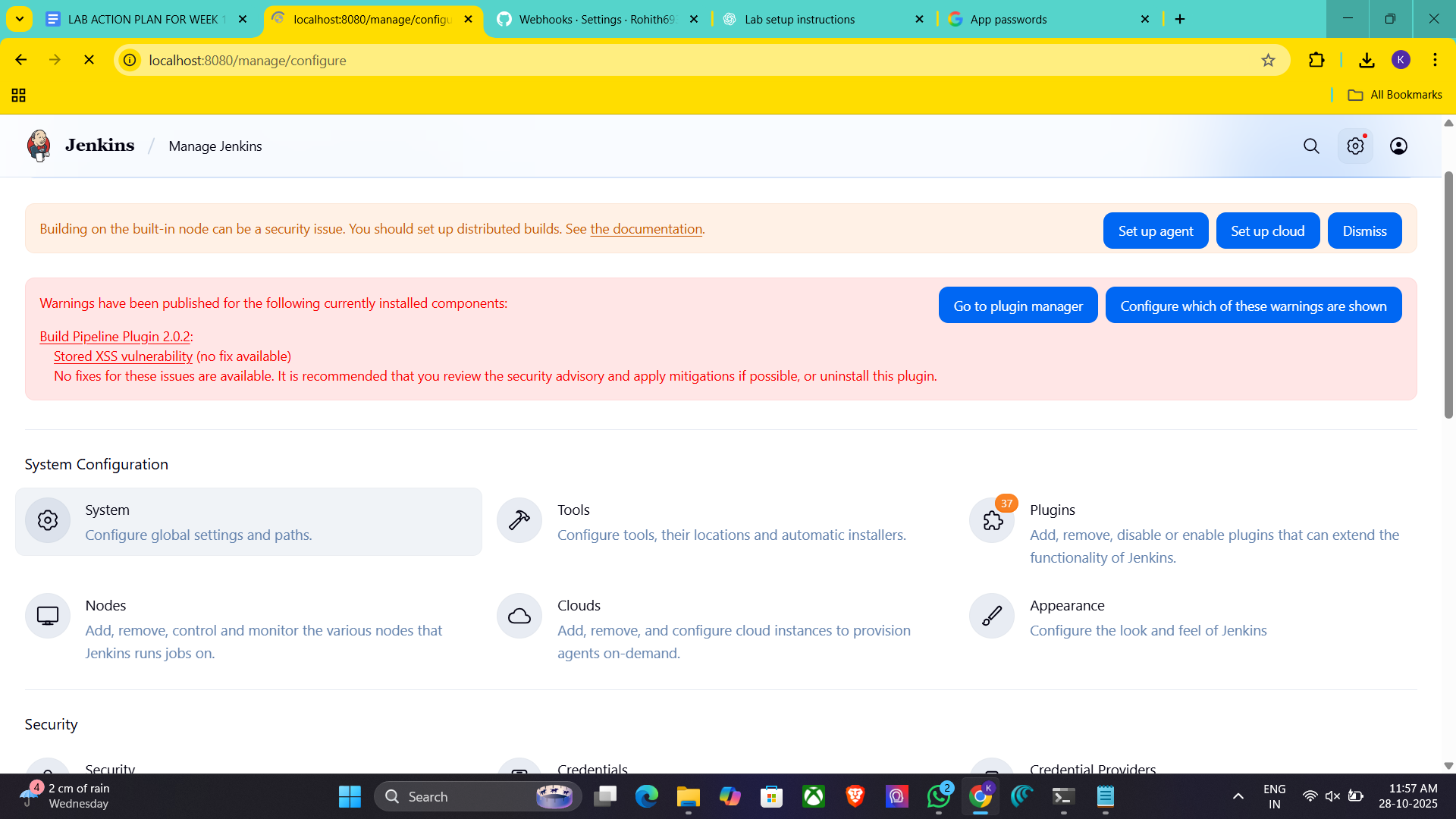
**Now your Jenkins job is set up to send email notifications based on the build status!**

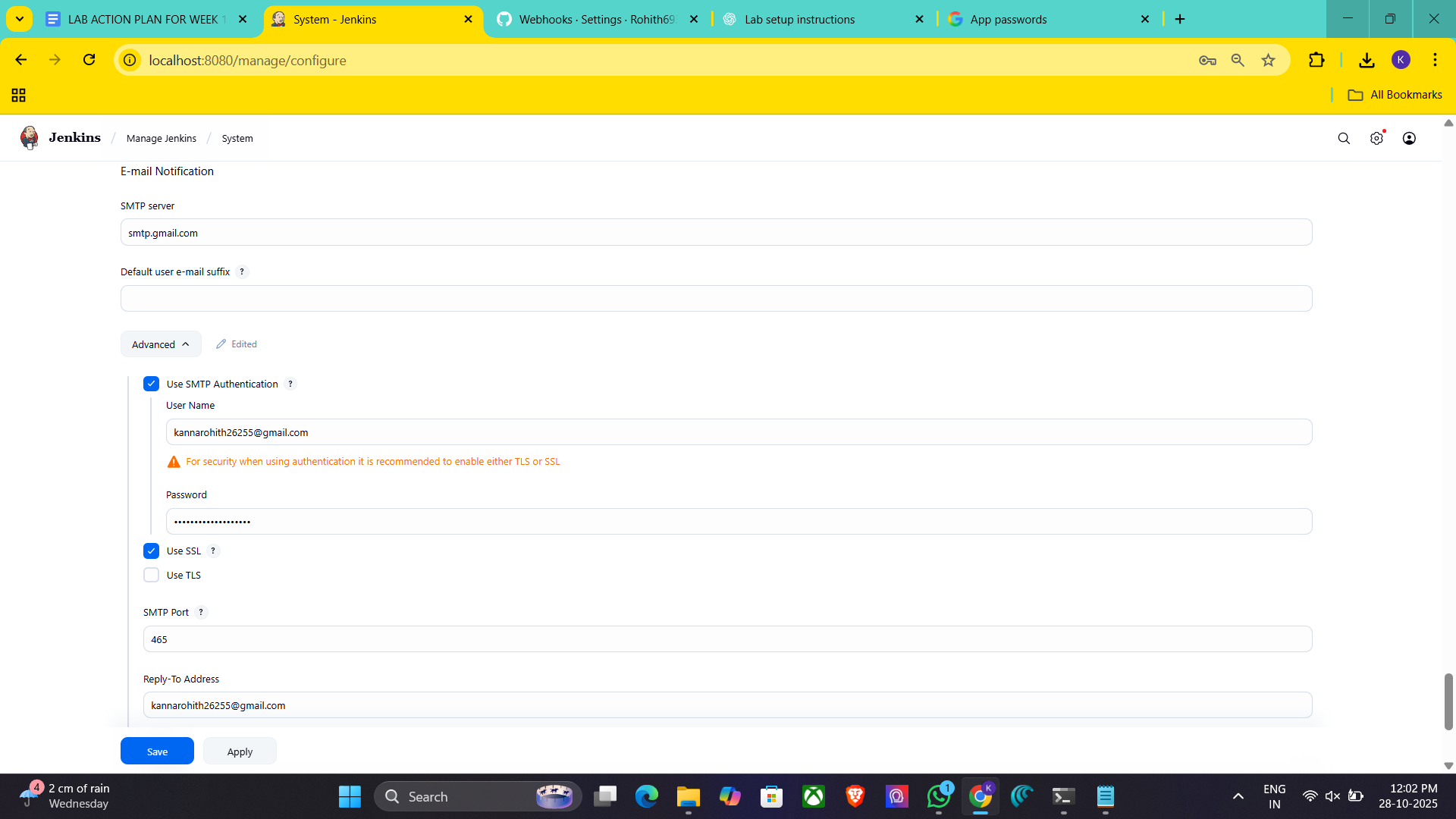
**Takeaway :**  
Students learned how to integrate Jenkins with GitHub using webhooks to automate build triggers and configure email notifications to monitor build success or failure effectively.

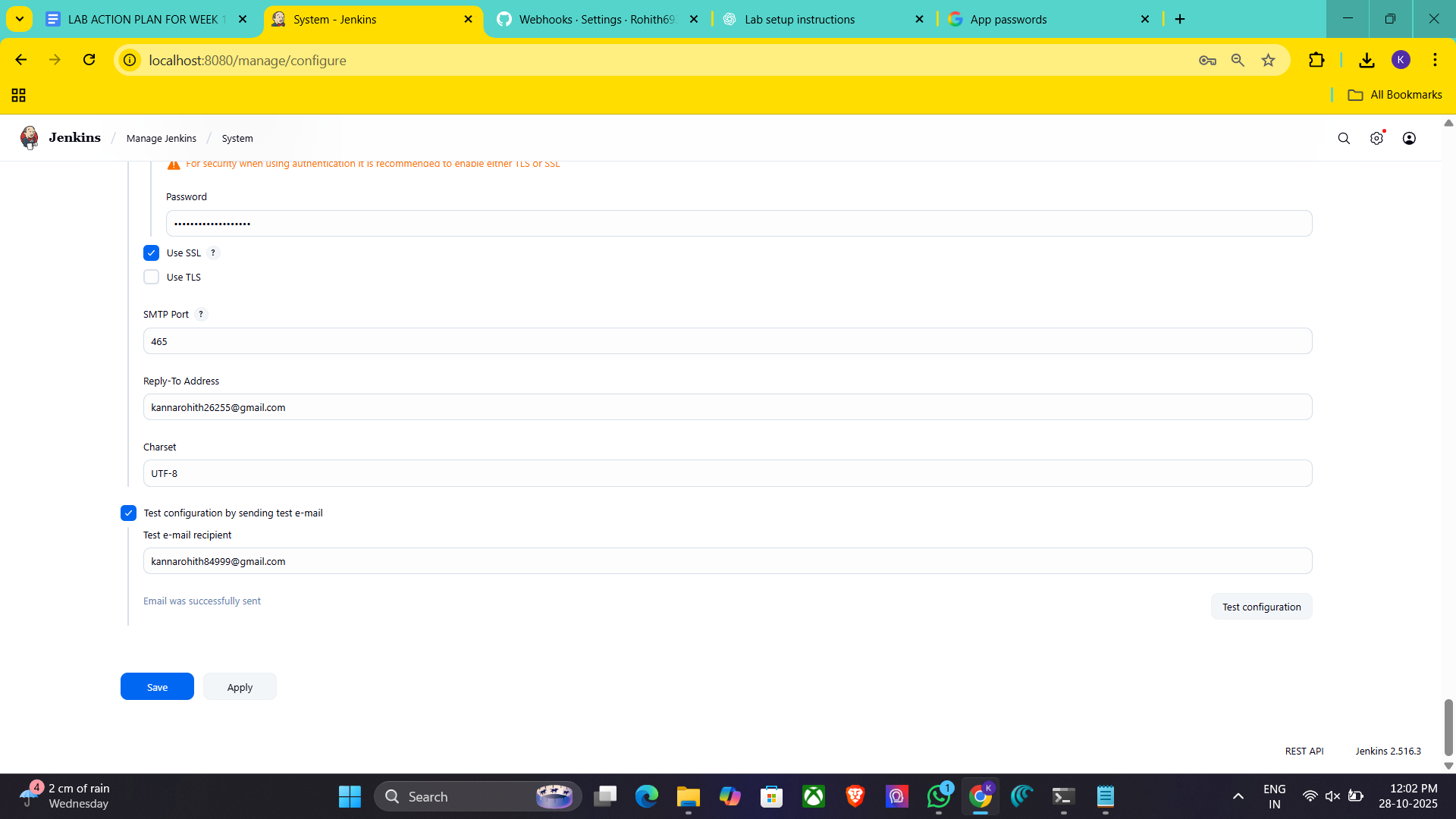


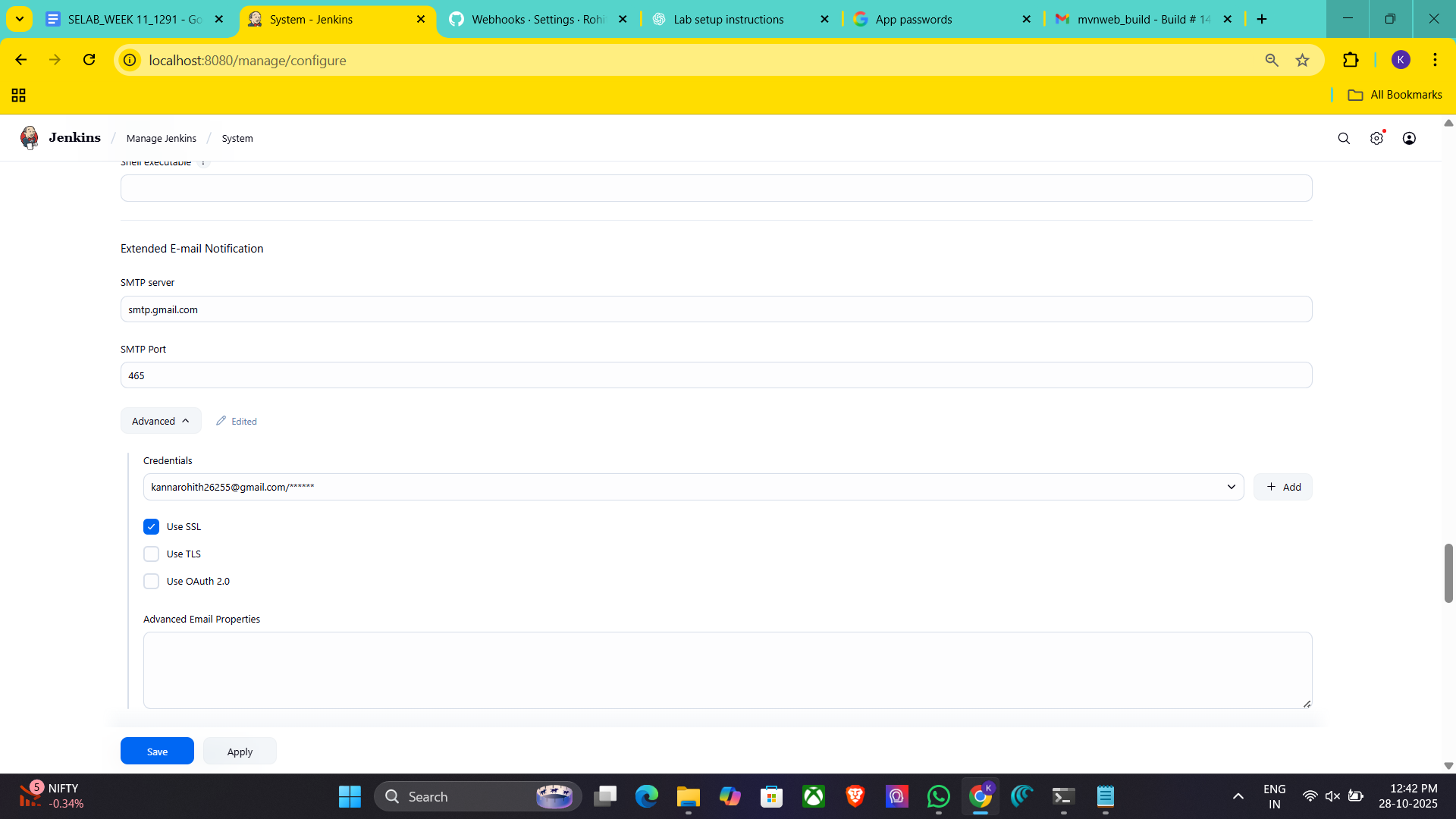


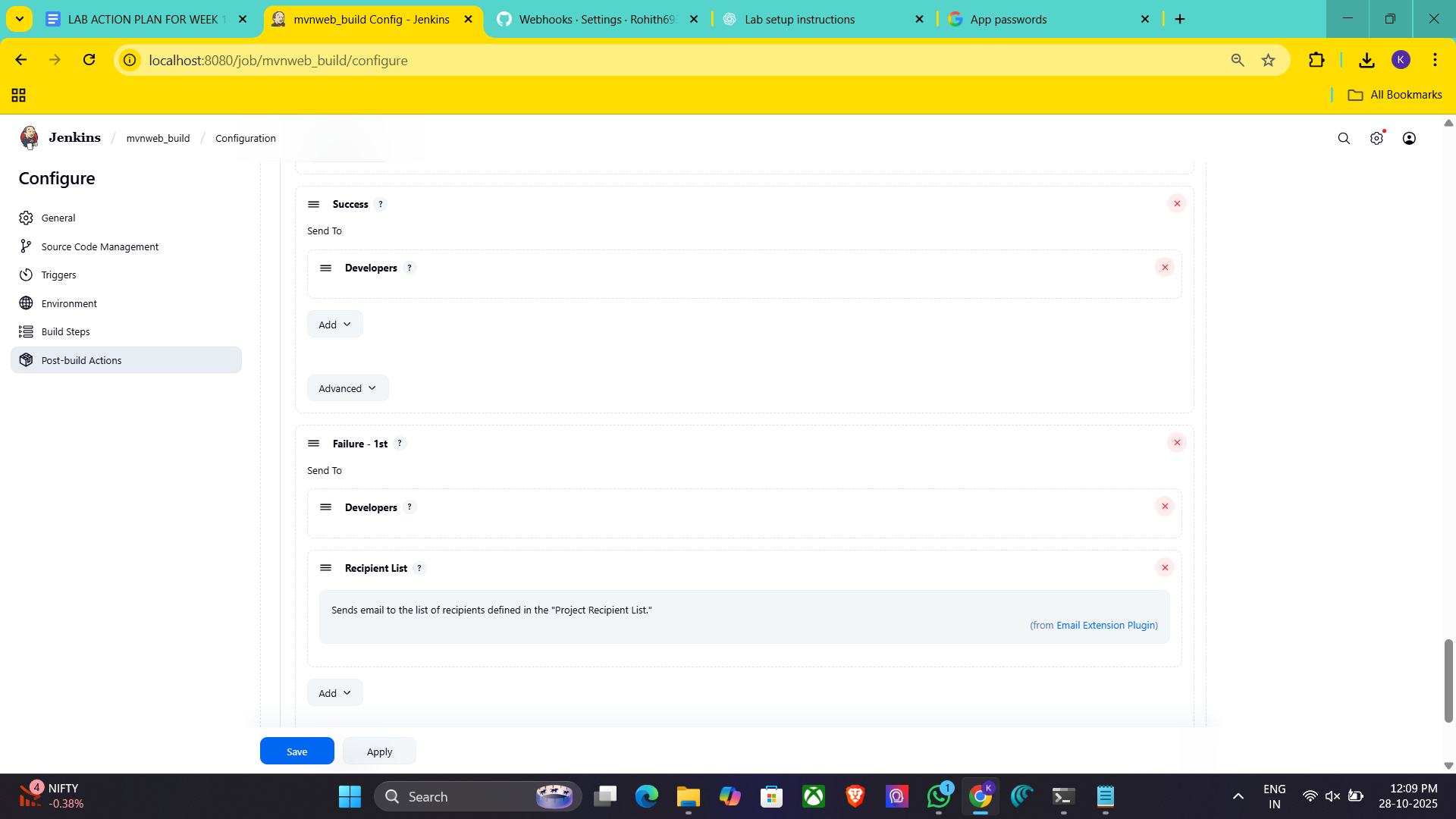
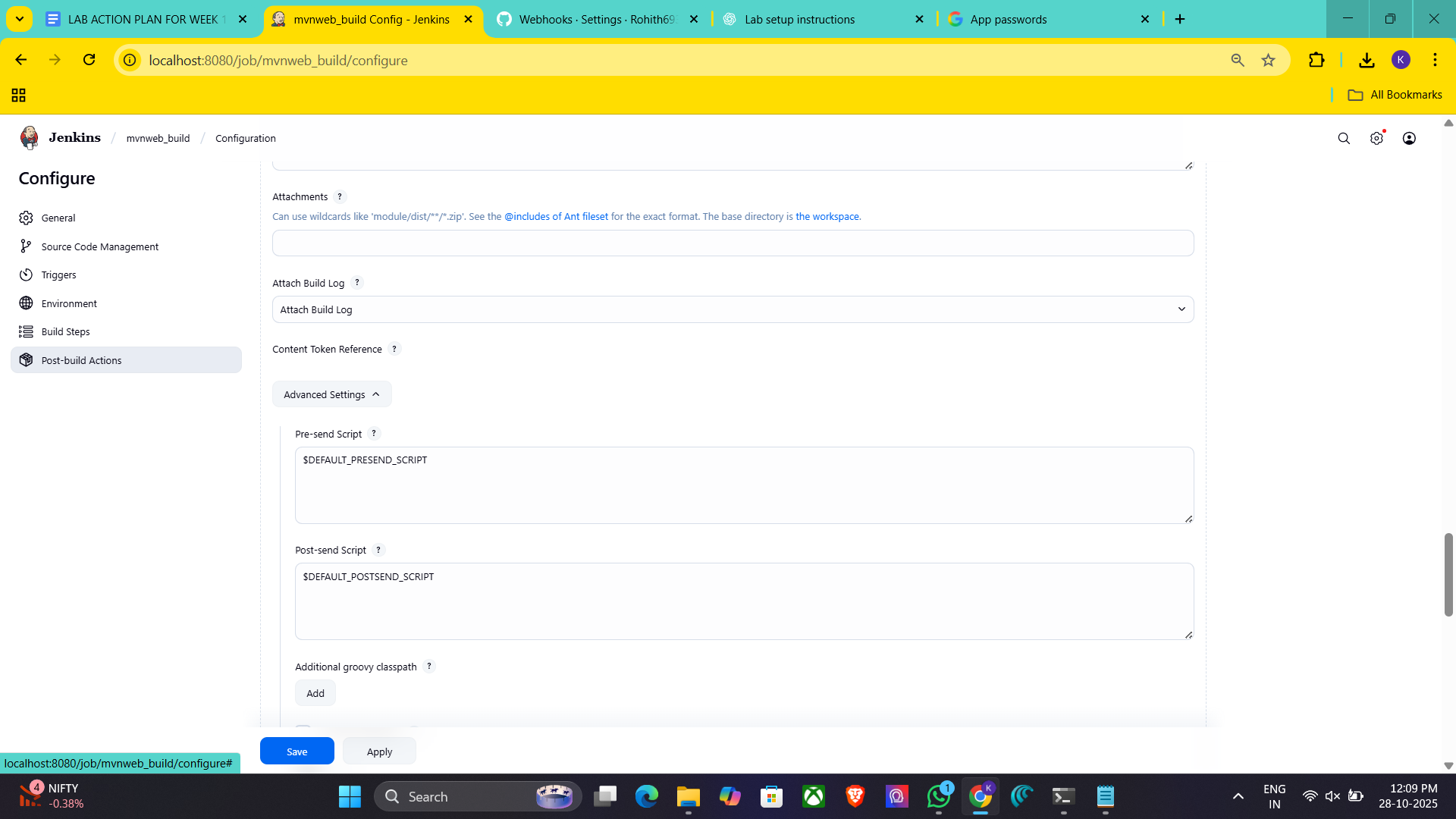
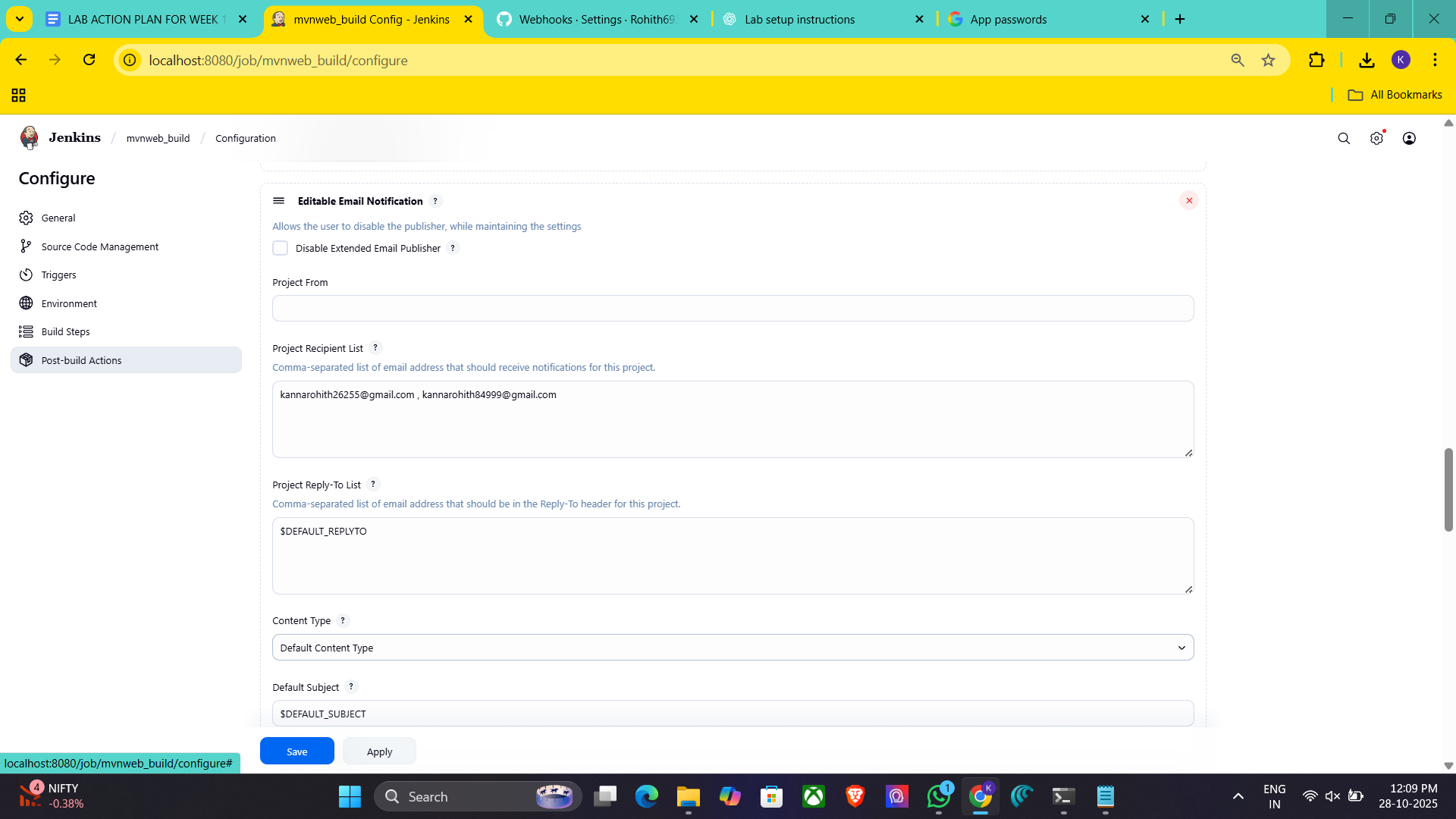
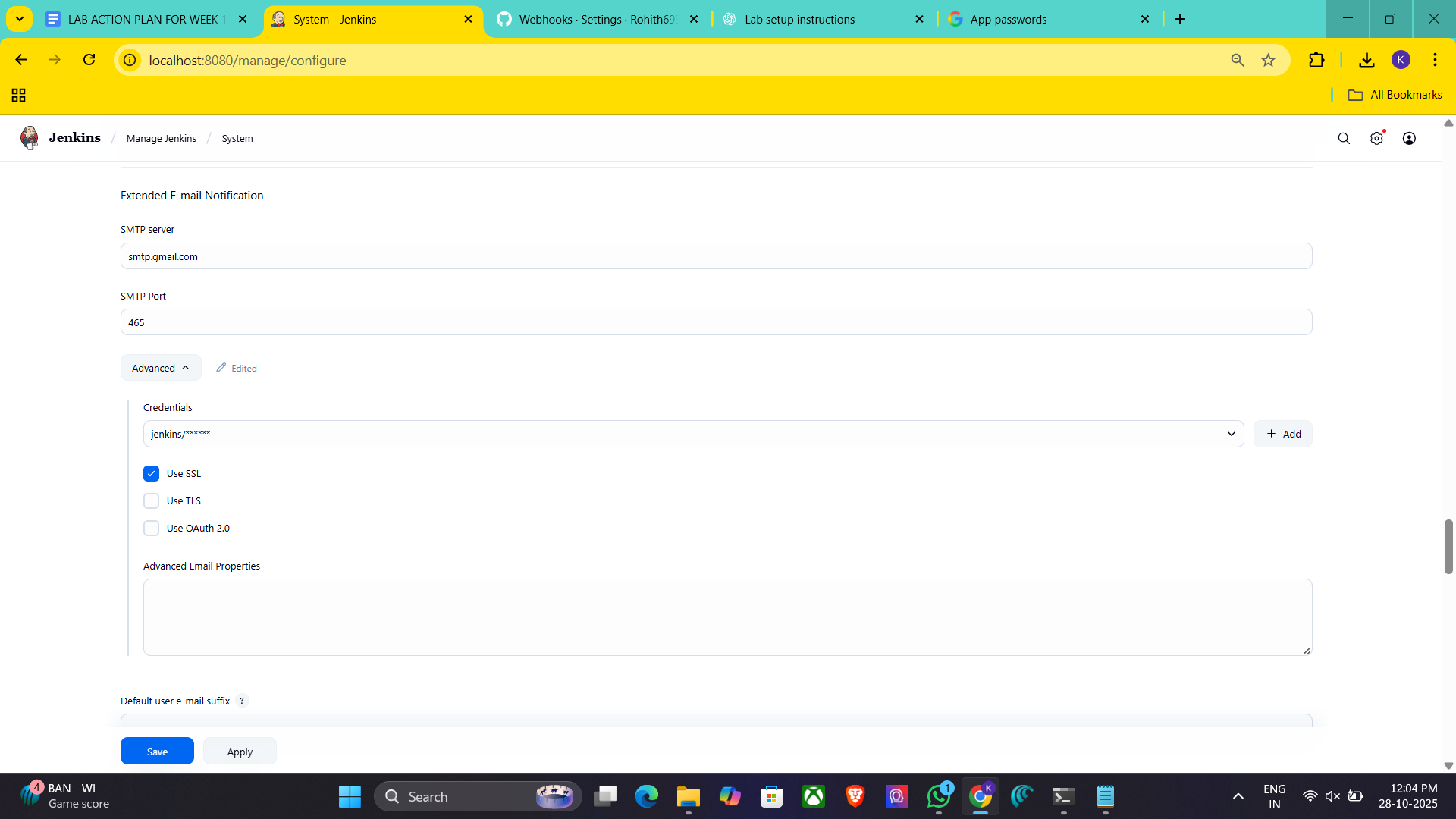




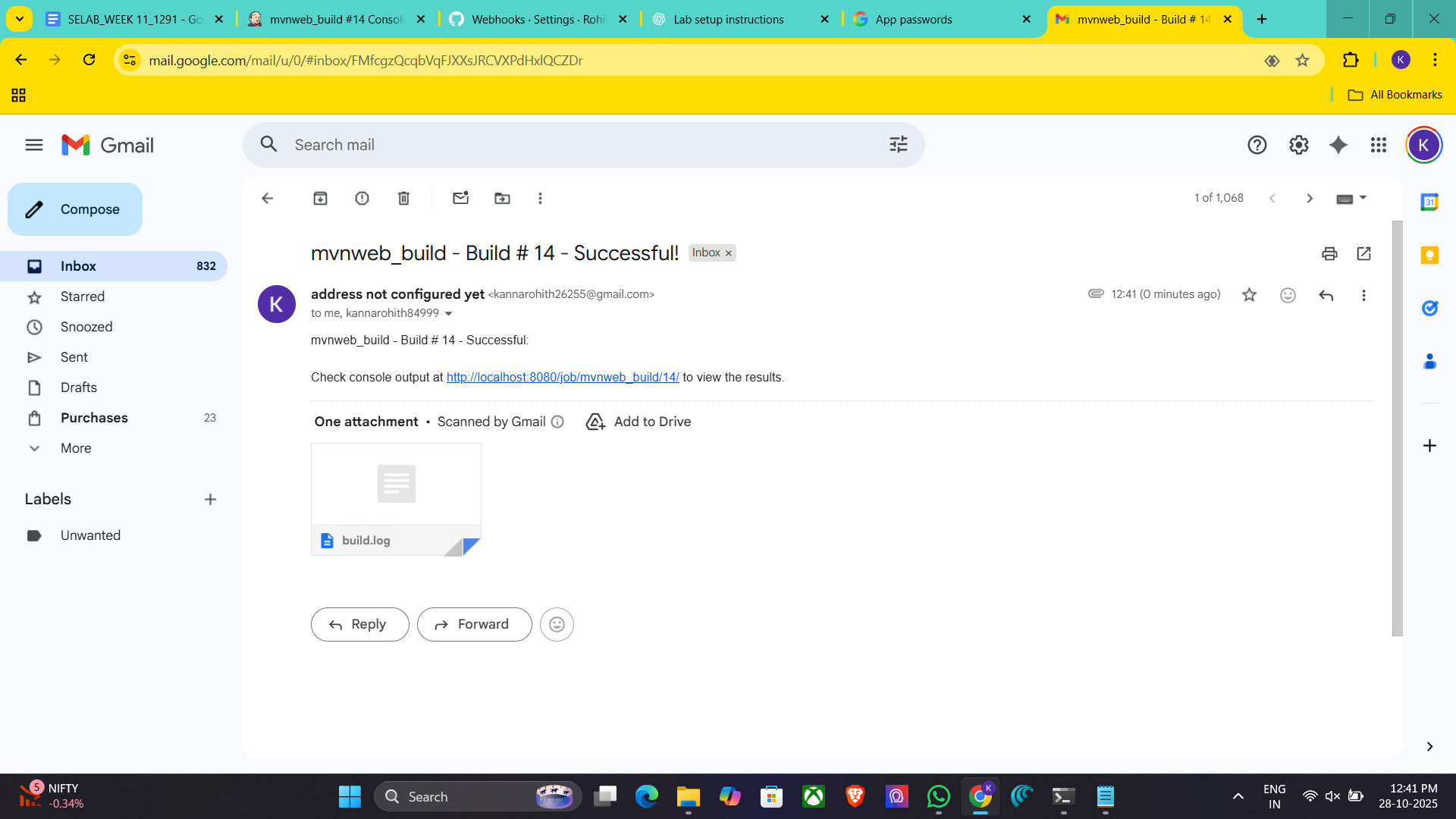












Viva Questions

1. What is Continuous Integration (CI)?

Ans:Continuous Integration (CI) is a software development practice where developers frequently integrate their code into a shared repository. Each integration is automatically verified by building and testing the code to detect issues early, improve code quality, and reduce integration problems.

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

Ans:**Continuous Delivery:** Ensures code is always in a deployable state but requires manual approval for production.

**Continuous Deployment:** Automatically deploys every change to production without manual intervention.

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

Ans: Jenkins automates building, testing, and deploying code, enabling continuous integration and delivery.

1. What is a webhook in GitHub?

Ans: A webhook automatically sends notifications to external tools like Jenkins whenever an event (like a push) happens in a repo.

1. Why are webhooks used in Jenkins integration?

Ans: Webhooks are used so that Jenkins can be instantly notified whenever a new code change is pushed to GitHub. This eliminates the need for Jenkins to constantly check (poll) for changes and enables immediate, event-based build triggering — making the CI/CD process faster and more efficient.

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

Ans: 1.Build periodically

2. Poll SCM  
 3.GitHub hook trigger  
 4. Trigger after another project  
 5.Trigger remotely via URL

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

Ans:Polling checks for changes at intervals; webhooks trigger builds instantly when changes occur.

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

Ans:Ngrok is a tunneling tool that exposes a local server (like Jenkins running on localhost) to the internet via a secure public URL. It is used when Jenkins is running locally and needs to be accessible by GitHub for webhook communication.

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

Ans:It creates a secure tunnel from the internet to your local Jenkins, letting GitHub send webhook data successfully.

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

Ans:To automatically notify developers about build success or failure, helping monitor and fix issues quickly.