

# Lab Exercise 19

## Setting up Snyk for SAST in Jenkins

**Objective:** To demonstrate the setup of the Snyk plugin in Jenkins for Static Application Security Testing (SAST), to automatically detect vulnerabilities in their codebase during development, thereby enhancing application security before deployment

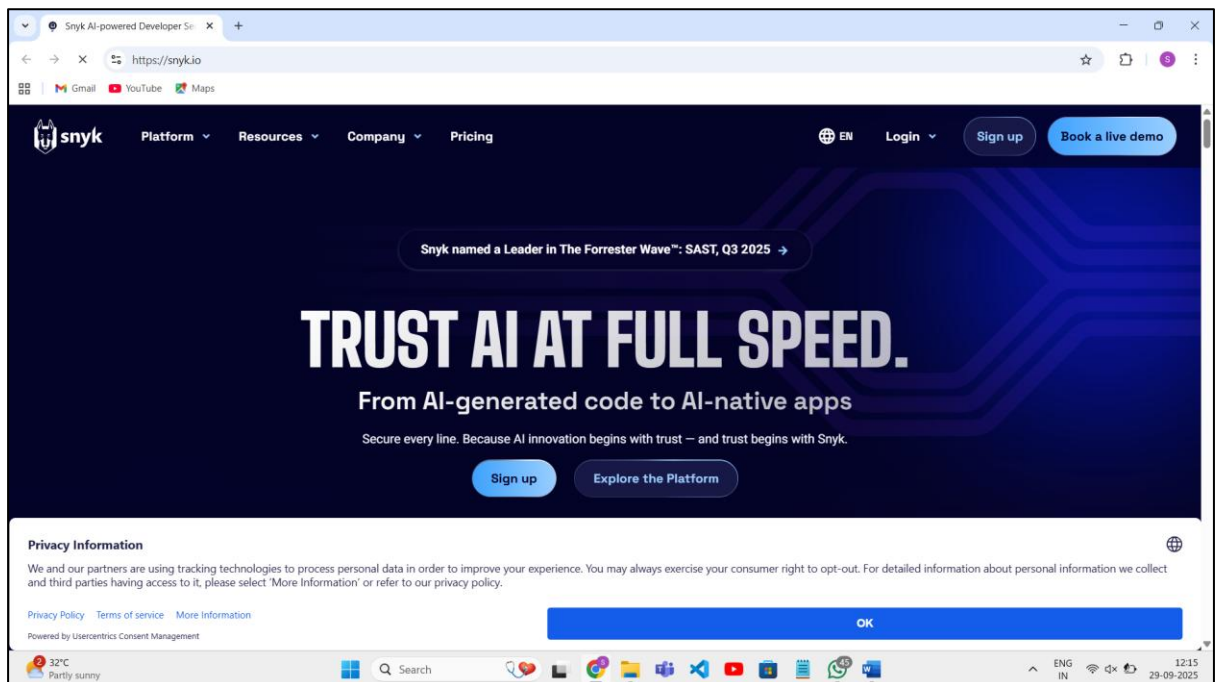
**Tools required:** Snyk

Steps to be followed:

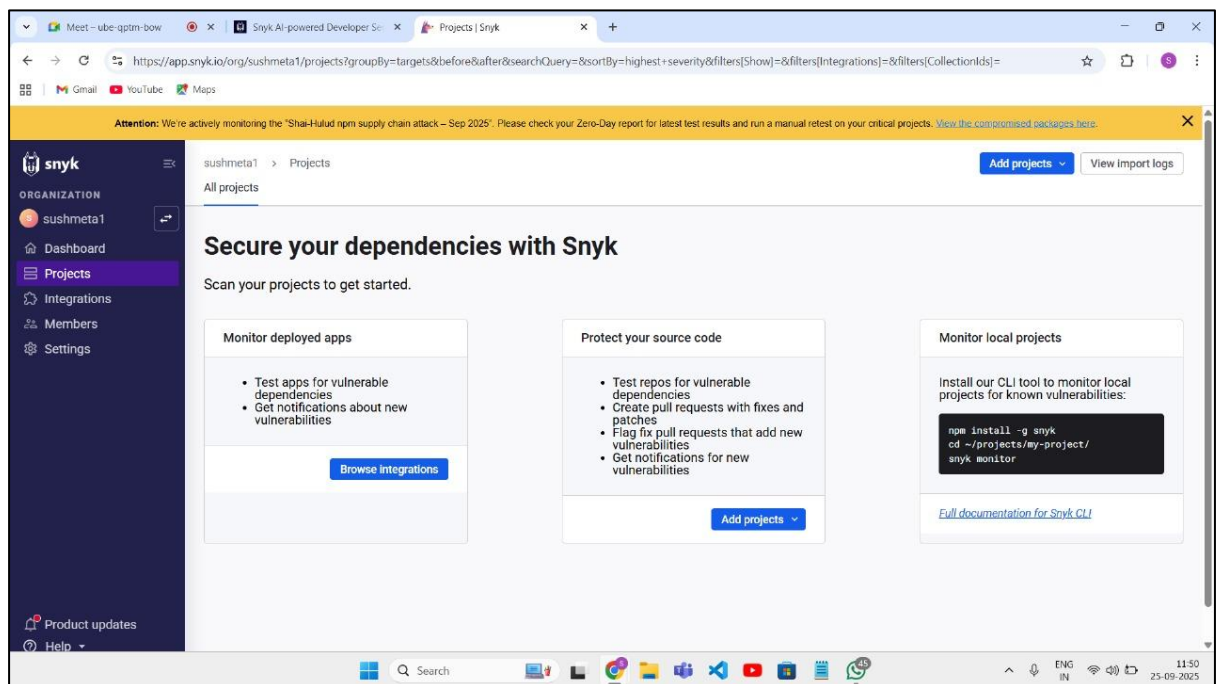
1. Configure Snyk as a SAST scan tool
2. Create and configure a Jenkins job for Snyk integration
3. Manage Snyk API and Jenkins credentials
4. Configure the Jenkins job for scanning

### Step 1: Configure Snyk as a SAST scan tool

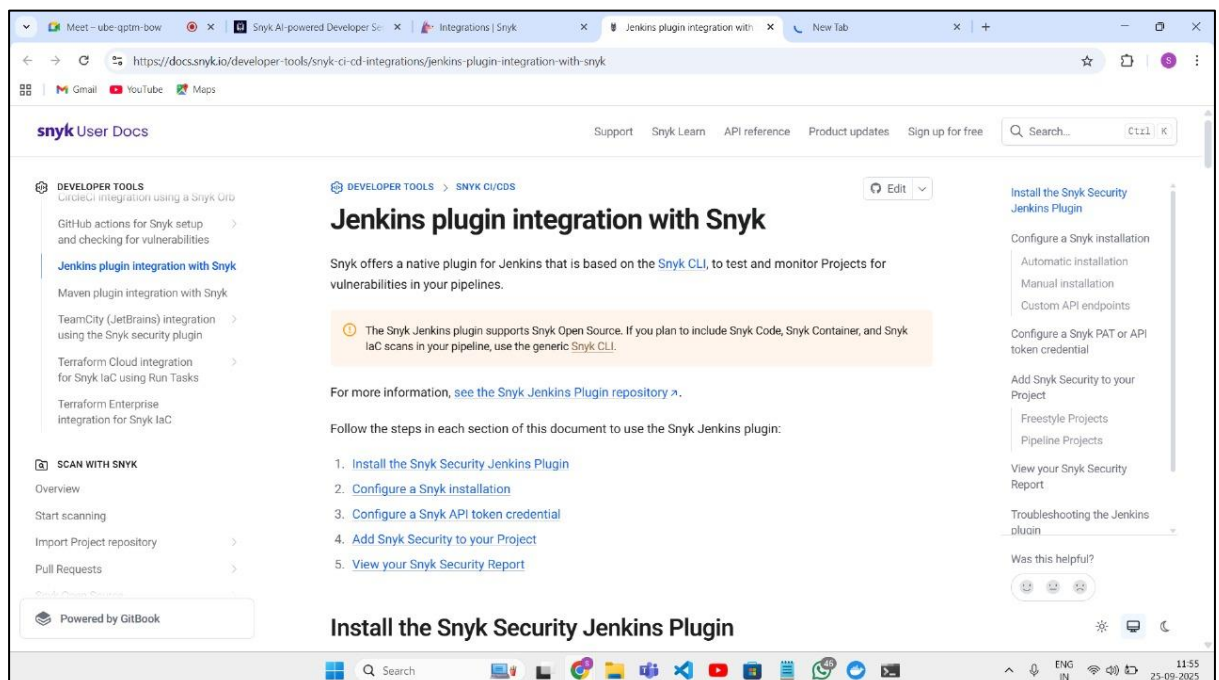
1.1 Visit <https://snyk.io/>, sign up for a new Snyk account, and log in



## 1.2 Navigate to **Integrations** and select **Jenkins**

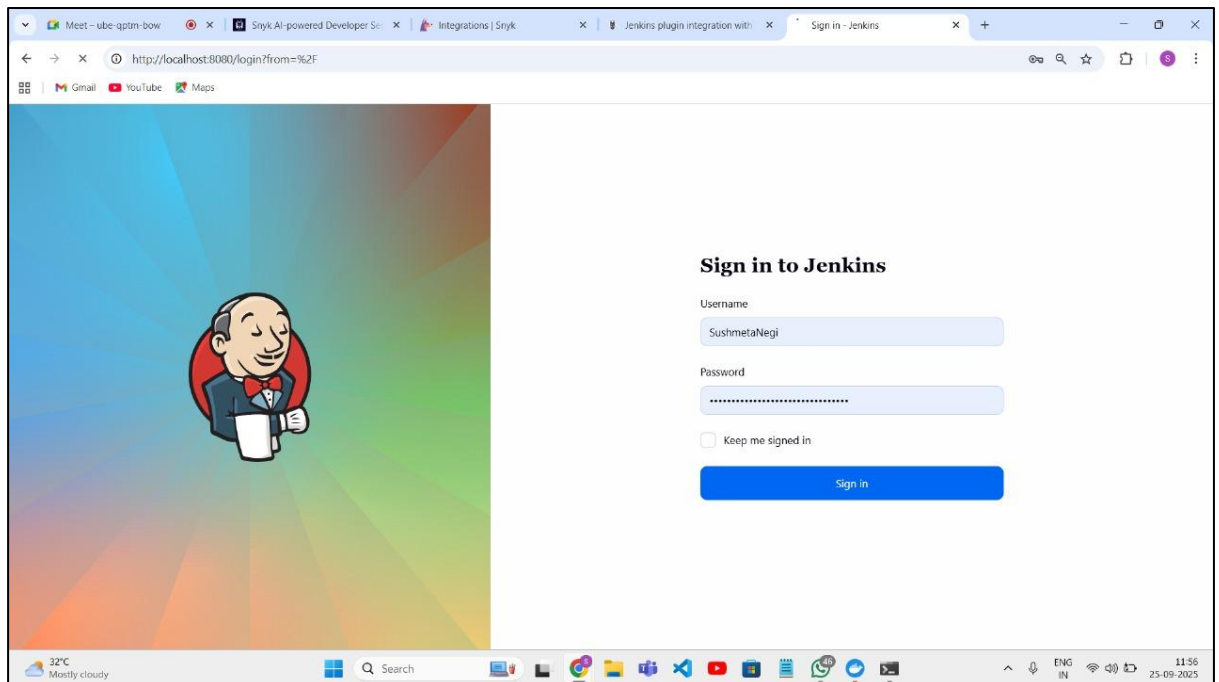


This will direct you to the documentation for integrating Snyk with Jenkins.



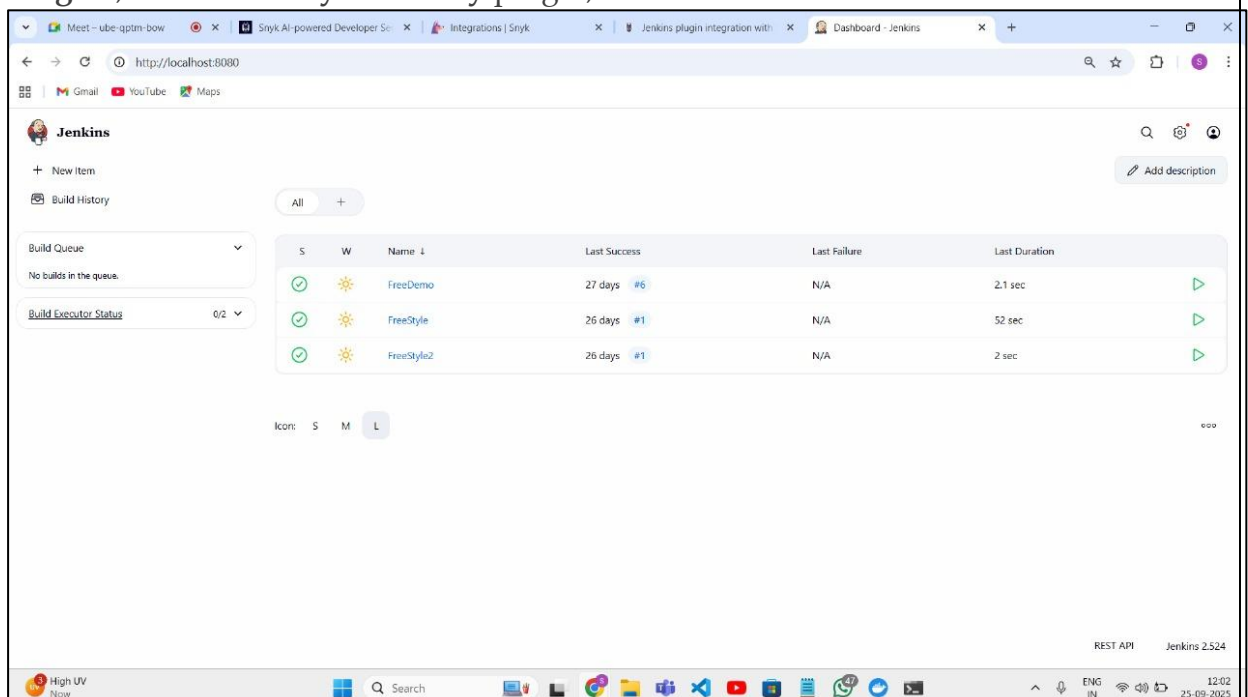
## Step 2: Create and configure a Jenkins job for Snyk integration

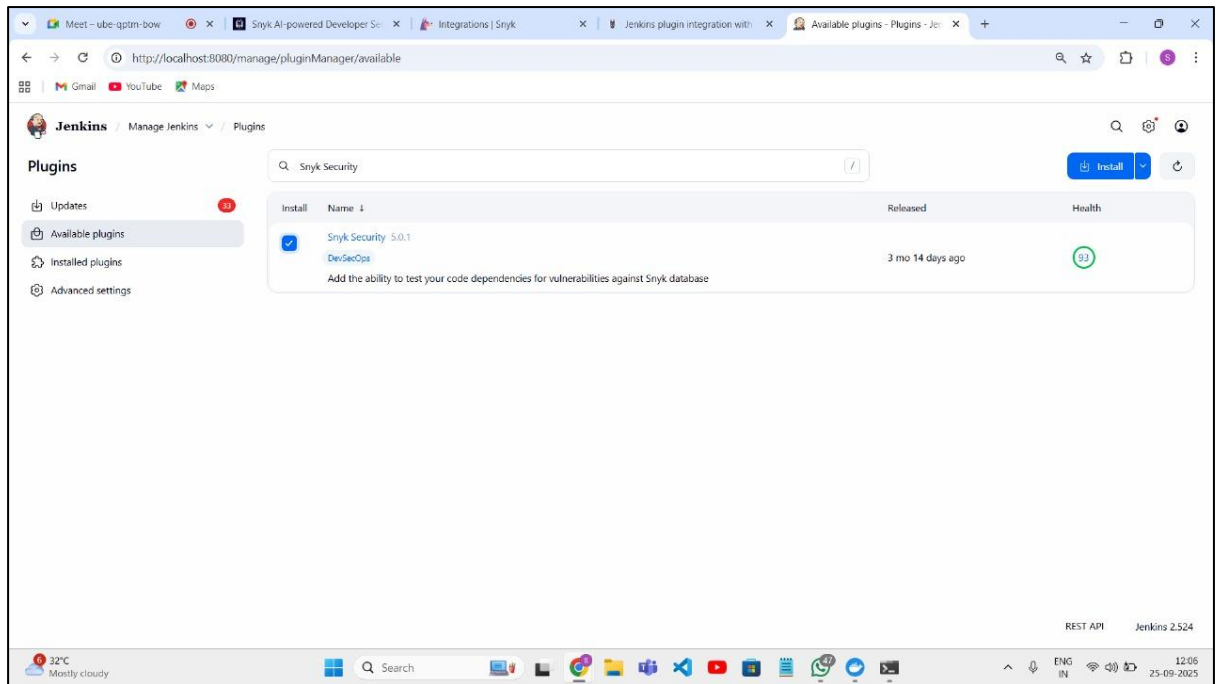
### 2.1 Open Jenkins and log in to the Jenkins account:



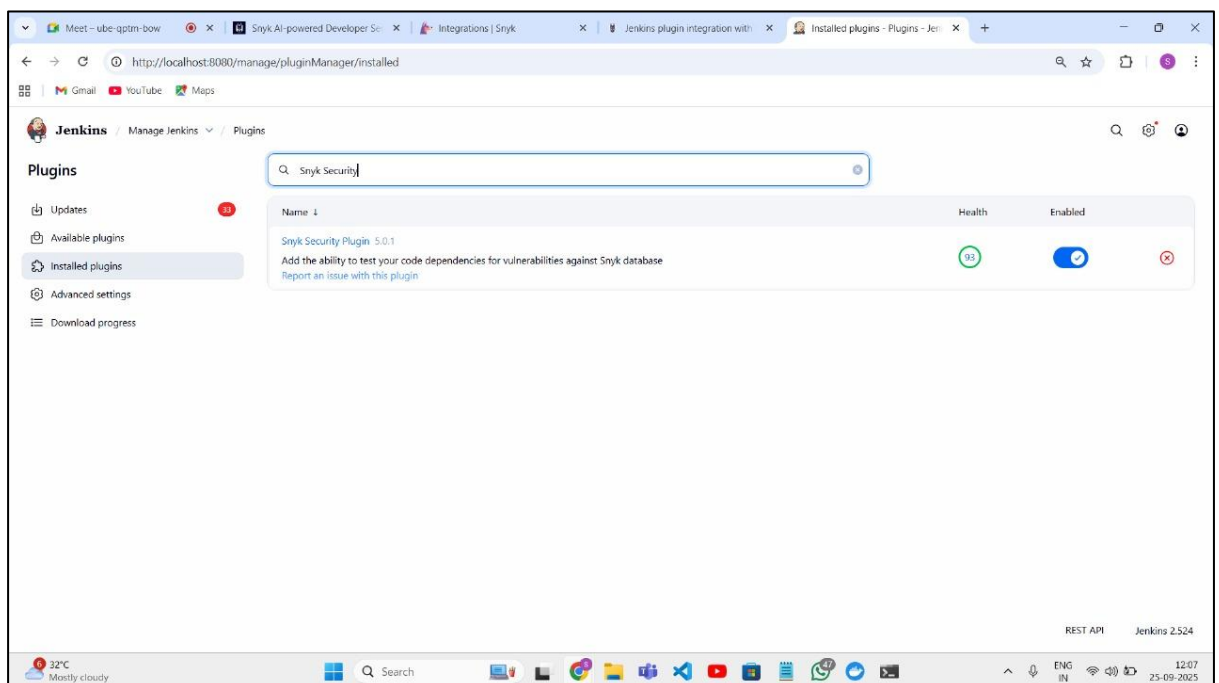
**Note:** The credentials for accessing Jenkins in the lab are Username: **admin** and Password: **admin**.

### 2.2 To install the Snyk plugin, navigate to **Manage Jenkins** and click **Available Plugins**, search for **Snyk Security** plugin, and then click **Install**

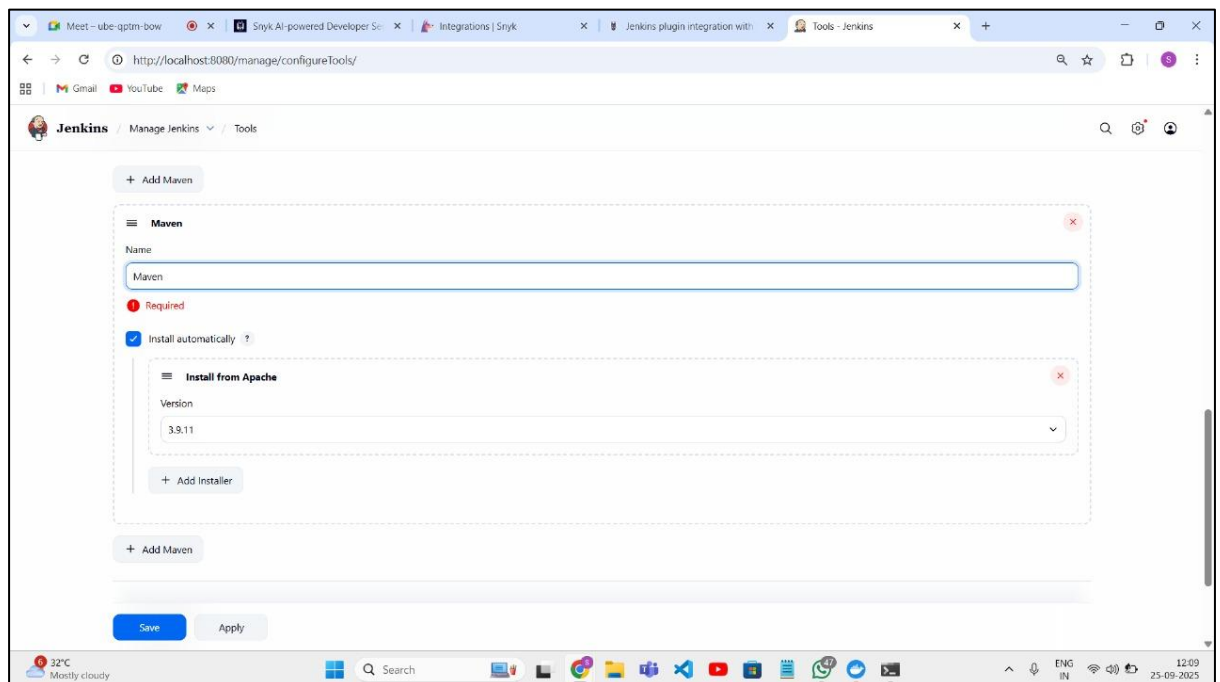




2.3 To configure Maven and Snyk in the **Global Tool Configuration**, click on **Tools** inside **Manage Jenkins**

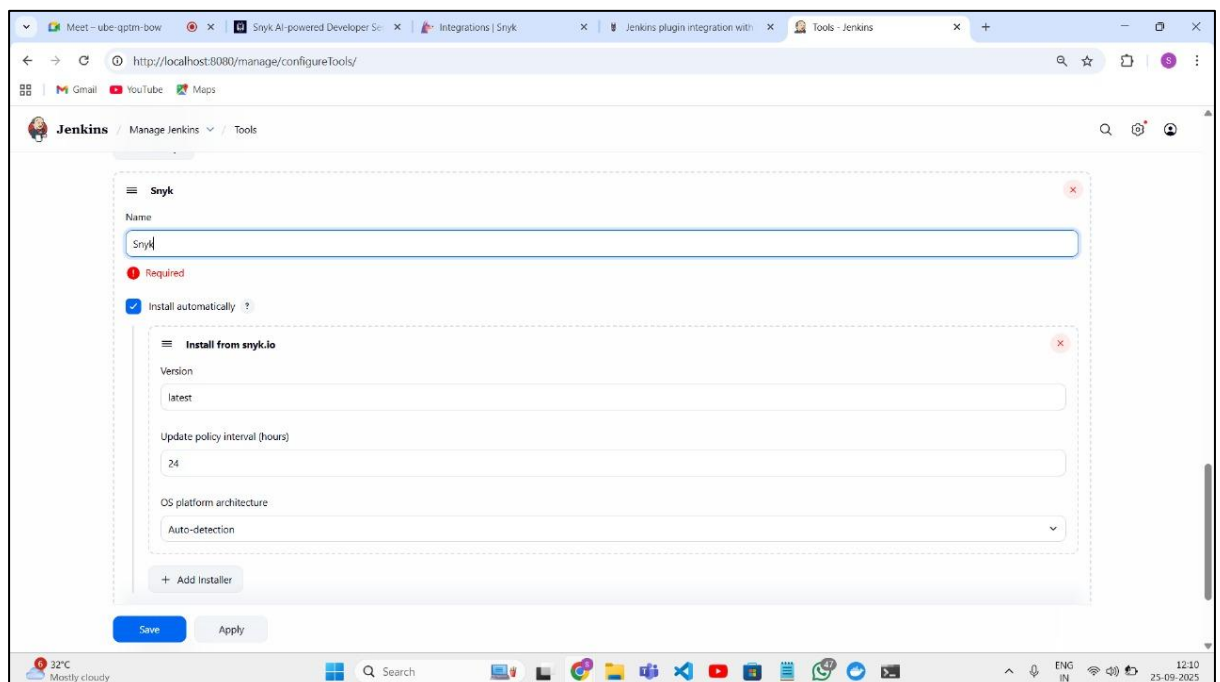


2.4 To add Maven, click on **Add Maven** under **Maven installations** and enter **Maven** as the **Name**



The screenshot shows the Jenkins 'Tools' configuration page for Maven. The browser address bar indicates the URL is `http://localhost:8080/manage/configureTools/`. The page title is 'Jenkins / Manage Jenkins / Tools'. A '+ Add Maven' button is at the top left. Below it, a dashed box contains the configuration for a tool named 'Maven'. The 'Name' field is filled with 'Maven'. A red 'Required' error message is shown below the name field. The 'Install automatically' checkbox is checked. Below this, a section titled 'Install from Apache' contains a 'Version' dropdown menu set to '3.9.11'. An '+ Add Installer' button is at the bottom of this section. At the bottom of the dashed box, there is another '+ Add Maven' button. Below the dashed box, there are 'Save' and 'Apply' buttons. The bottom of the page shows a Windows taskbar with various icons and a system tray showing '32°C Mostly cloudy' and the date '25-09-2025'.

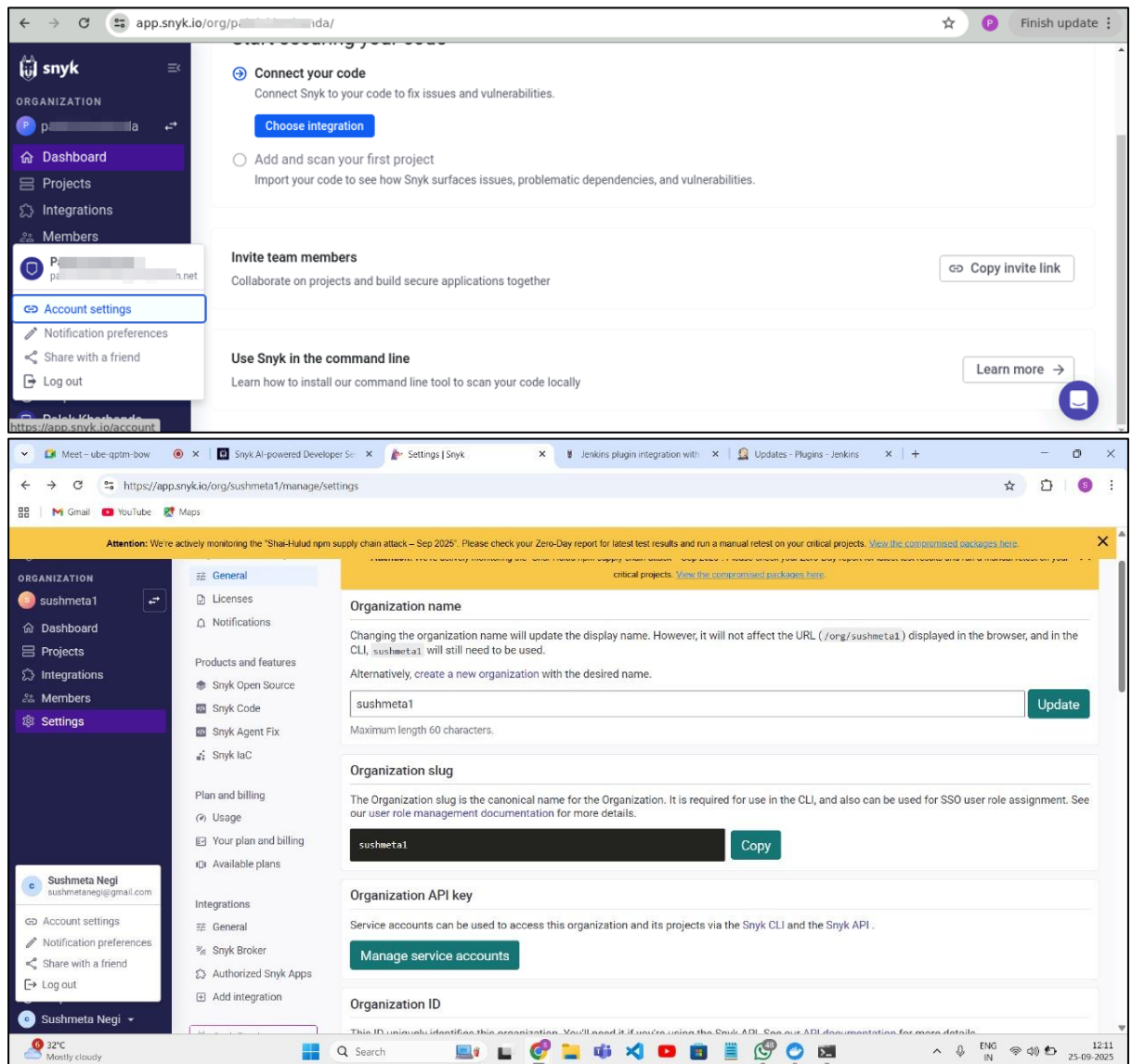
2.5 To add Snky, click on **Add Snky** under **Snyk Installations**, add **Name** as **Snyk**, and click on the **Save** button



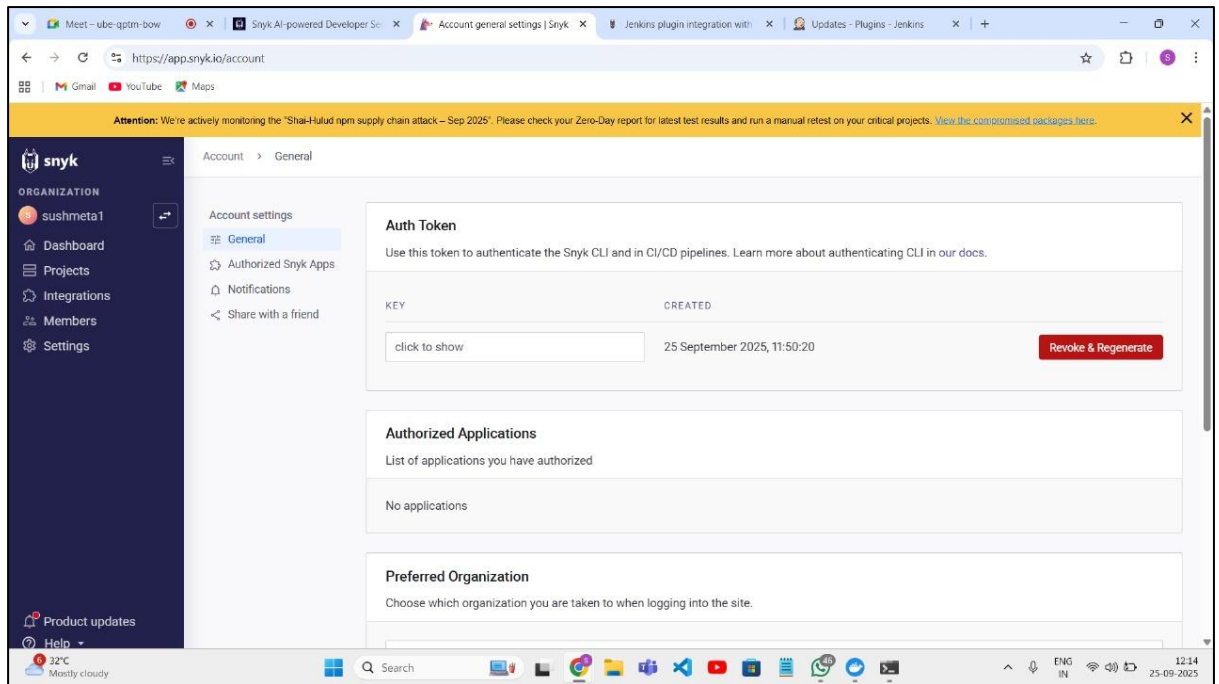
The screenshot shows the Jenkins 'Tools' configuration page for Snyk. The browser address bar indicates the URL is `http://localhost:8080/manage/configureTools/`. The page title is 'Jenkins / Manage Jenkins / Tools'. A '+ Add Snyk' button is at the top left. Below it, a dashed box contains the configuration for a tool named 'Snyk'. The 'Name' field is filled with 'Snyk'. A red 'Required' error message is shown below the name field. The 'Install automatically' checkbox is checked. Below this, a section titled 'Install from snyk.io' contains several fields: 'Version' (set to 'latest'), 'Update policy interval (hours)' (set to '24'), and 'OS platform architecture' (set to 'Auto-detection'). An '+ Add Installer' button is at the bottom of this section. At the bottom of the dashed box, there is another '+ Add Snyk' button. Below the dashed box, there are 'Save' and 'Apply' buttons. The bottom of the page shows a Windows taskbar with various icons and a system tray showing '32°C Mostly cloudy' and the date '25-09-2025'.

## Step 3: Manage Snky API and Jenkins credentials

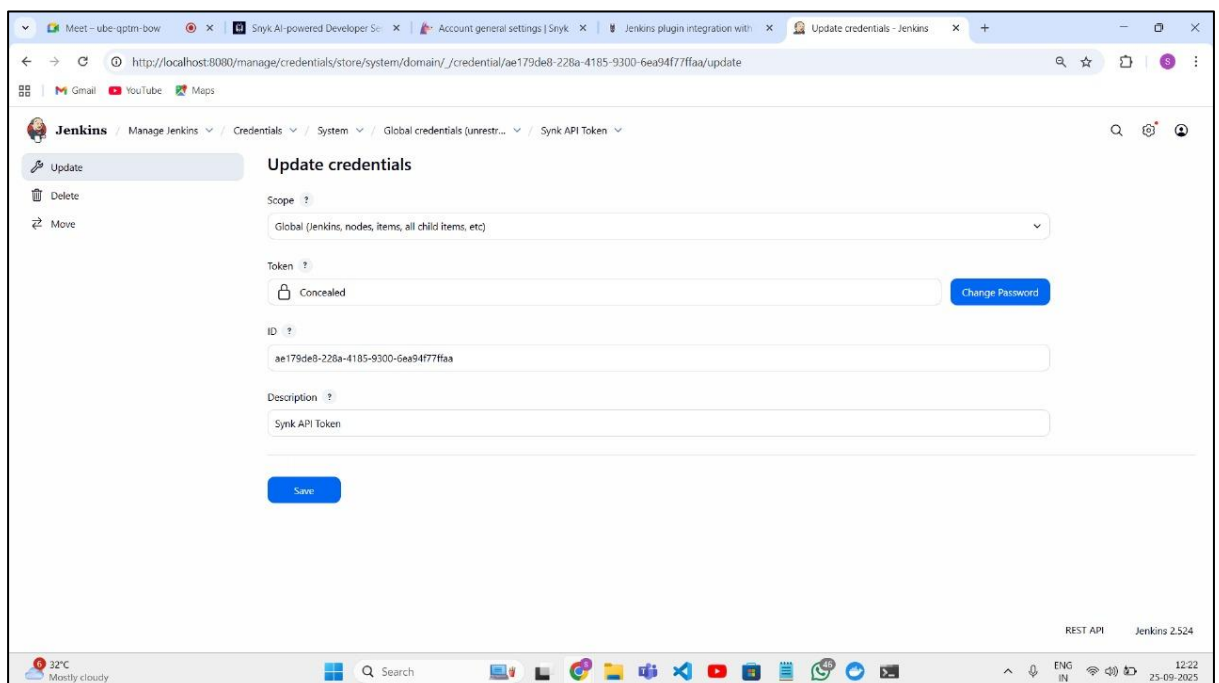
- 3.1 To retrieve your Snky API token, go to **Account Settings** in your Snky account, click on **Click to show** under the Auth Token key field, and copy the token for further reference







3.2 In the Jenkins interface, go to **Manage Jenkins**, select **Security**, then choose **Credentials** and select **global** to add global credentials



3.3 Click on **Add Credentials**, select the **Snyk API token** from the **Kind** field, paste the copied token from step 3.1 into the **Token** field, and then click the **Create** button

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted)

Global credentials (unrestricted)

+ Add Credentials

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
<a href="#">geeks</a>	admin/***** (geeks)	Username with password	geeks
<a href="#">geeks01</a>	admin/***** (geeks01)	Username with password	geeks01
<a href="#">Maven</a>	GithubResources1/***** (Mavengit)	Username with password	Mavengit
localhost:8080/manage/credentials/store/system/domain/_/newCredentials		GithubResources1/*****	Username with

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted)

Kind

Snyk API token

Username with password

GitHub App

SSH Username with private key

Secret file

Secret text

Snyk API token

Certificate

Token

Field is required

ID

Create

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted)

Scope

Global (jenkins, nodes, items, all child items, etc)

Token

.....

ID

Description

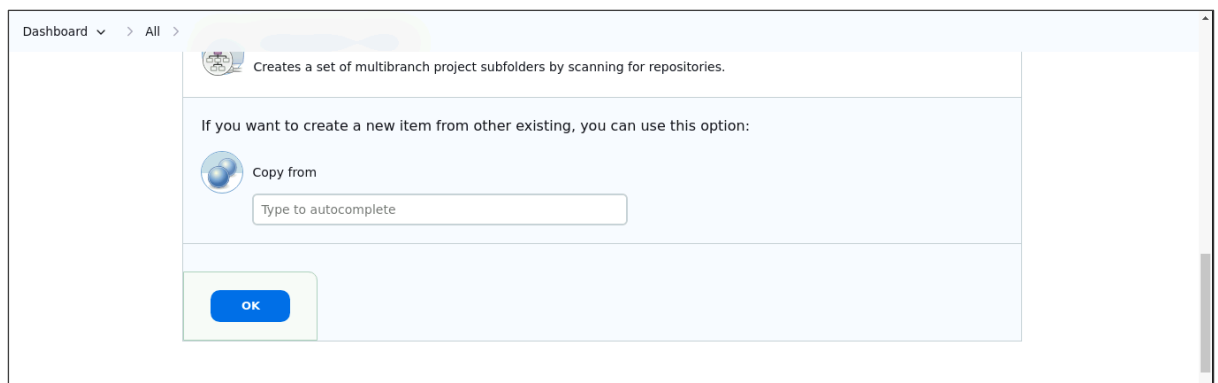
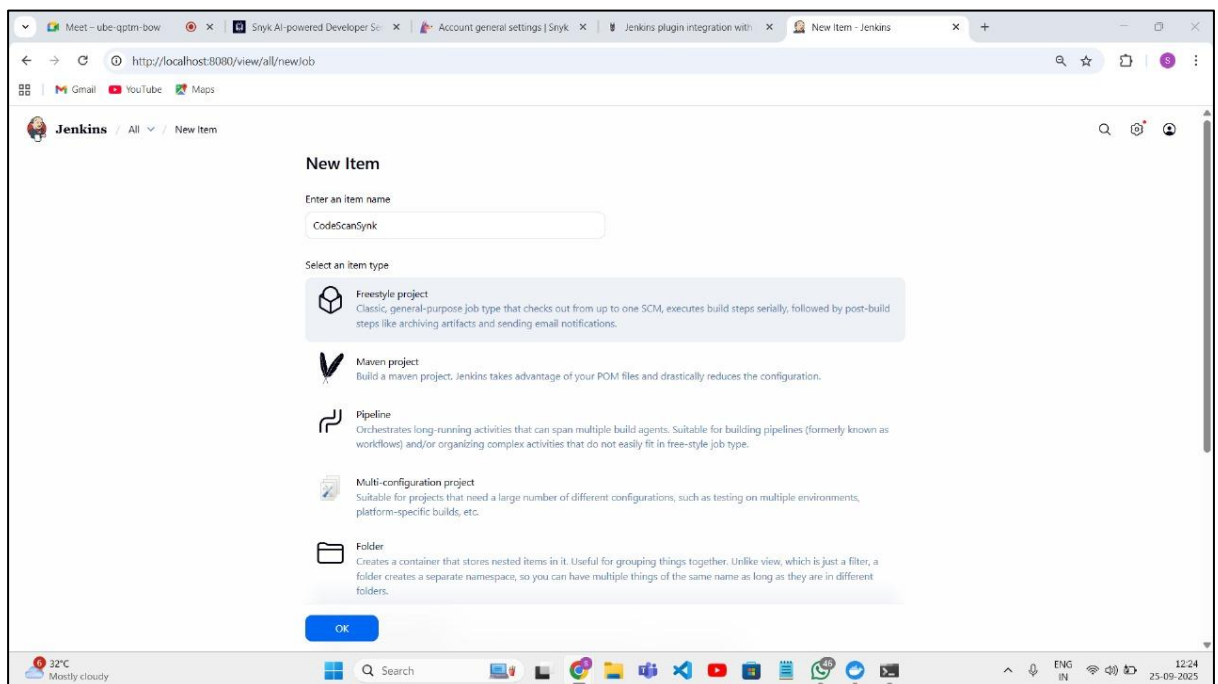
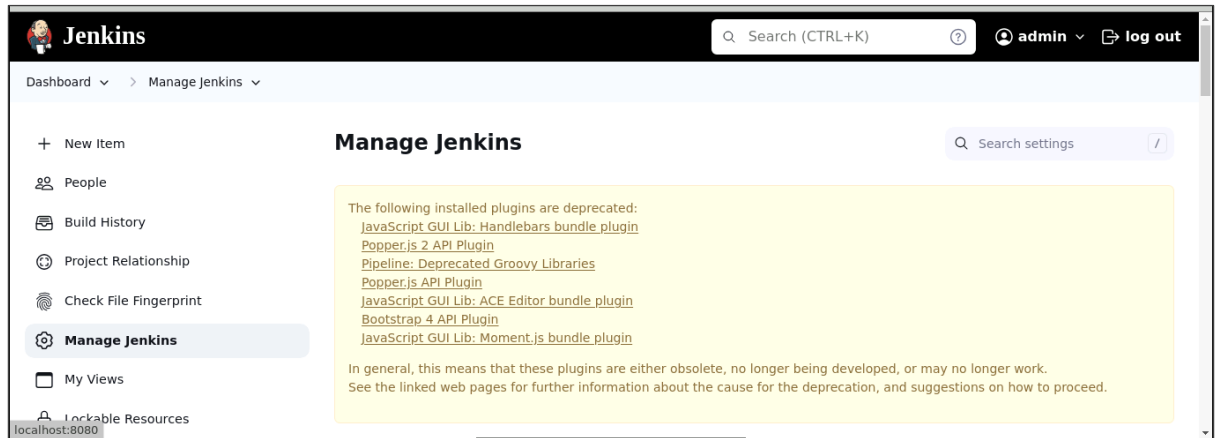
SynkToken

Create



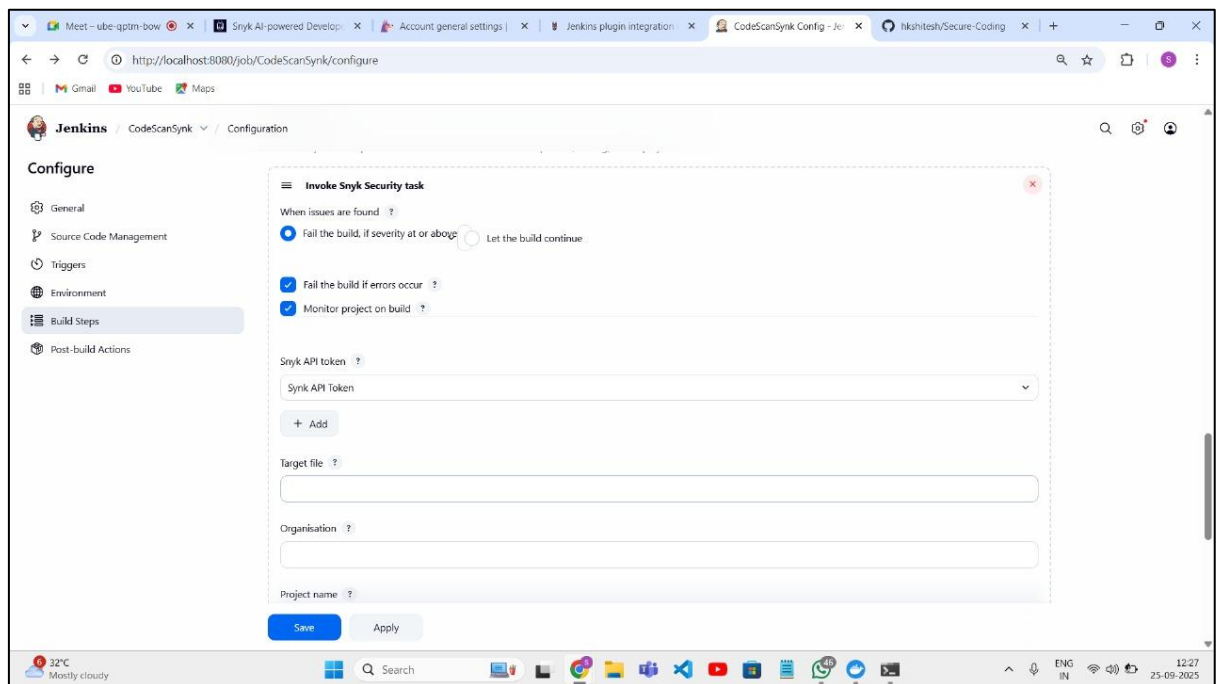
## Step 4: Configure the Jenkins job for scanning

4.1 To create a new Jenkins job, click on **New Item**, enter the item name as **CodeScanSnyk**, select **Freestyle project**, and then click **OK**

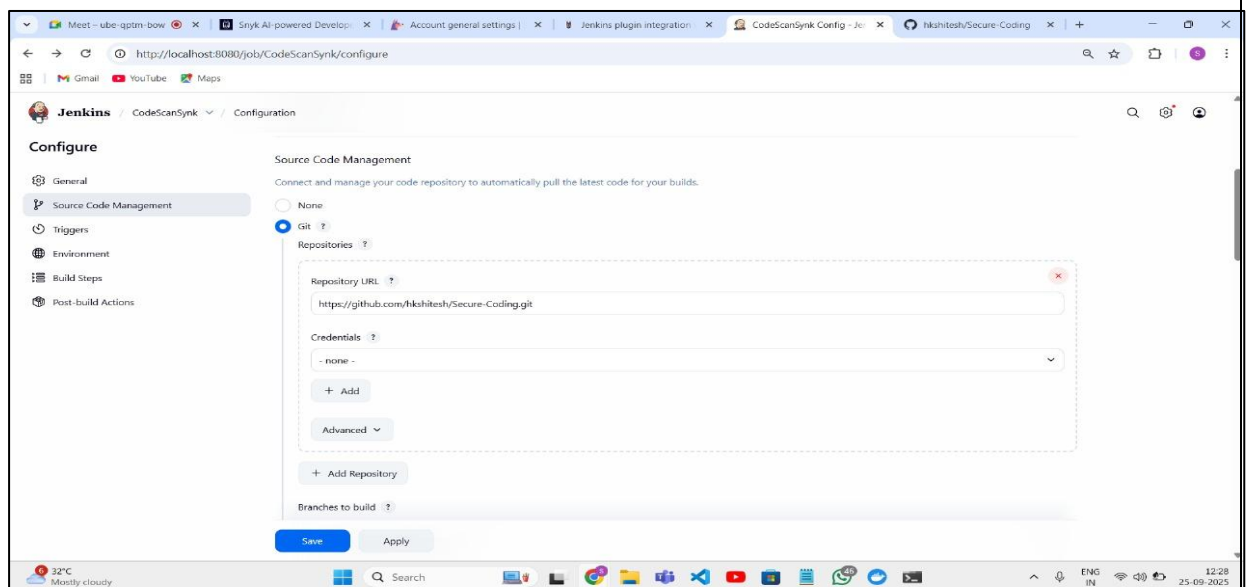


4.2 After creating a job, go to **Source Code Management** and enter the GitHub repository URL. Then, under **Build Steps**, add the build step **Invoke Snynk Security task** with the name **SnykToken**. Finally, click the **Save** button to create the build.

Use GitHub Repo: <https://github.com/hkshitesh/Secure-Coding.git>



**Note:** For GitHub repository URL, use <https://github.com/hkshitesh/Secure-Coding.git>



### 4.3 To check the build status, click on the build link under **Permalinks**. After that, click on **Console Output**

The screenshot shows the Jenkins dashboard for a project named 'CodeScanSnyk'. On the left sidebar, there are links for 'Changes', 'Workspace', 'Build Now', 'Configure', 'Delete Project', 'Rename', and 'Build History'. The 'Build History' link is highlighted, and a 'trend' dropdown is visible. The main area shows 'Last Successful Artifacts' with a link to '2024-05-08T09:24-17-848173830Z\_snyk\_report.html' (13.79 KB) and a 'view' button. Below this is the 'Permalinks' section with a list of build links: 'Last build (#2), 2 min 46 sec ago', 'Last stable build (#2), 2 min 46 sec ago', 'Last successful build (#2), 2 min 46 sec ago', and 'Last completed build (#2), 2 min 46 sec ago'. At the bottom, there is a 'Filter builds...' input field.

The screenshot shows the Jenkins 'Console Output' for build #1 of the 'CodeScanSnyk' project. The output text is as follows:

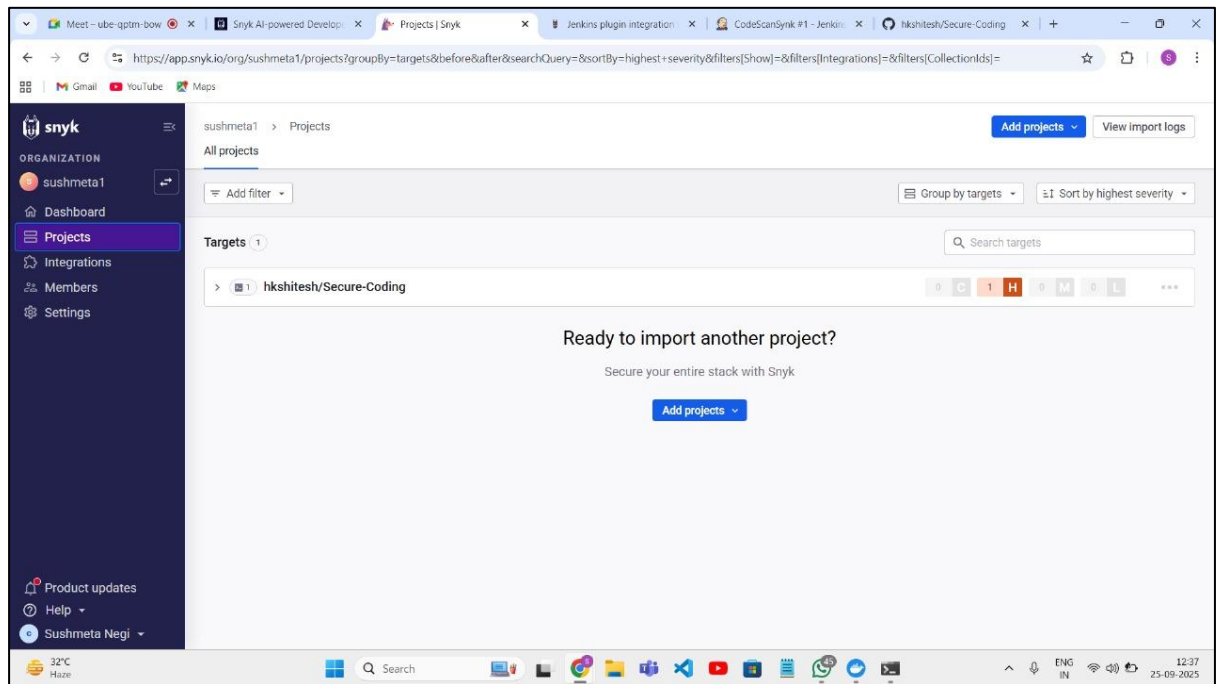
```
Started by user Sushmeta Negi
Running as SYSTEM
Building in workspace /var/jenkins_home/workspace/CodeScanSnyk
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/hkshitesh/Secure-Coding.git
> git init /var/jenkins_home/workspace/CodeScanSnyk # timeout=10
Fetching upstream changes from https://github.com/hkshitesh/Secure-Coding.git
> git --version # timeout=10
> git --version # 'git version 2.39.5'
> git fetch --tags --force --progress -- https://github.com/hkshitesh/Secure-Coding.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git config remote.origin.url https://github.com/hkshitesh/Secure-Coding.git # timeout=10
> git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision 5e3aaedae26e41b315263bf3151216fd7eb416b1 (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f 5e3aaedae26e41b315263bf3151216fd7eb416b1 # timeout=10
Commit message: "Add files via upload"
First time build. Skipping changelog.
Installing Snyk (latest)...
Testing project...
> /var/jenkins_home/tools/io.snyk.jenkins.tools.SnykInstallation/Snyk/snyk-linux test --json --severity-threshold=low
Vulnerabilities found!
Result: 1 known vulnerabilities | 6 dependencies
Generating report...
```

The screenshot shows the Jenkins build summary for build #1 of the 'CodeScanSnyk' project, dated 'Sep 25, 2025, 6:59:10 AM'. The summary includes the following information:

- Status:** Failed (indicated by a red 'X' icon).
- Build Artifacts:** A link to '2025-09-25T07-02-15-472810478Z\_snyk\_report.html' (17.50 KB) with a 'view' button.
- Started by user:** Sushmeta Negi.
- This run spent:**
  - 0.71 sec waiting
  - 3 min 42 sec build duration
  - 3 min 43 sec total from scheduled to completion
- git:**
  - Revision:** 5e3aaedae26e41b315263bf3151216fd7eb416b1
  - Repository:** https://github.com/hkshitesh/Secure-Coding.git
  - refs/remotes/origin/main
- Changes:** No changes.

At the bottom, there is a 'REST API' link and the Jenkins version '2.524'.

#### 4.4 To navigate to the Snyc tool to review code, scan reports under the **Projects** section



By following the above steps, you have successfully demonstrated the setup of the Snyk plugin in Jenkins for static application security testing (SAST), to automatically detect vulnerabilities in their codebase during development, thereby enhancing application security before deployment.