

Lab Exercise 19

Setting up Snyc for SAST in Jenkins

Pratik Agrawal

500123601

Devops B2

Objective: To demonstrate the setup of the Snyc 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: Snyc

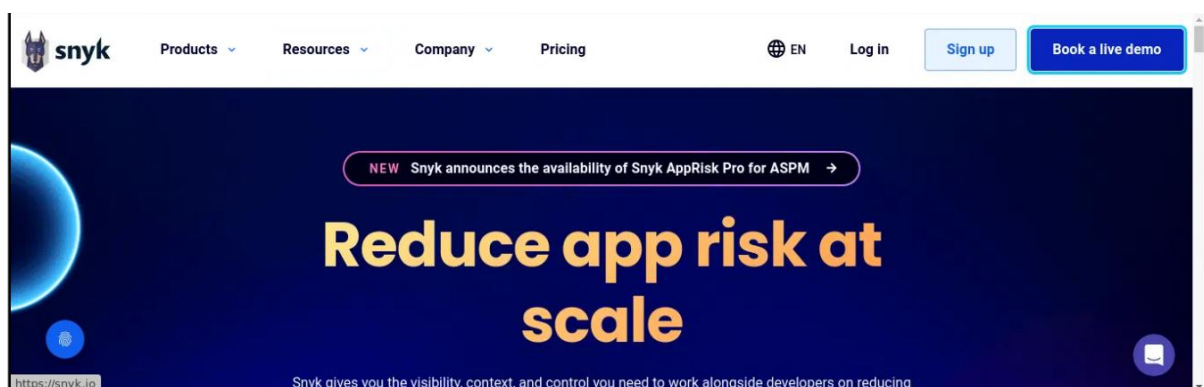
Prerequisites: None

Steps to be followed:

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

Step 1: Configure Snyc as a SAST scan tool

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



OUTPUT :-

All projects

Secure your dependencies with Snyk

Scan your projects to get started.

Monitor deployed apps

- Test apps for vulnerable dependencies
- Get notifications about new vulnerabilities

[Browse integrations](#)

Protect your source code

- Test repos for vulnerable dependencies
- Create pull requests with fixes and patches
- Flag fix pull requests that add new vulnerabilities
- Get notifications for new vulnerabilities

[Add projects](#)


Monitor local projects

Install our CLI tool to monitor local projects for known vulnerabilities:

```
npm install -g snyk
cd ~/projects/my-project/
snyk monitor
```

[Full documentation for Snyk CLI](#)


2. Navigate to **Integrations** and select **Jenkins**




Kubernetes

[Upgrade plan to enable](#)

Continuous integration




CLI




Jenkins


[View Documentation](#)




TeamCity




Bitbucket Pipelines




Azure Pipelines



Circle CI




Terraform-Cloud





Snyk API


[Upgrade plan to enable](#)

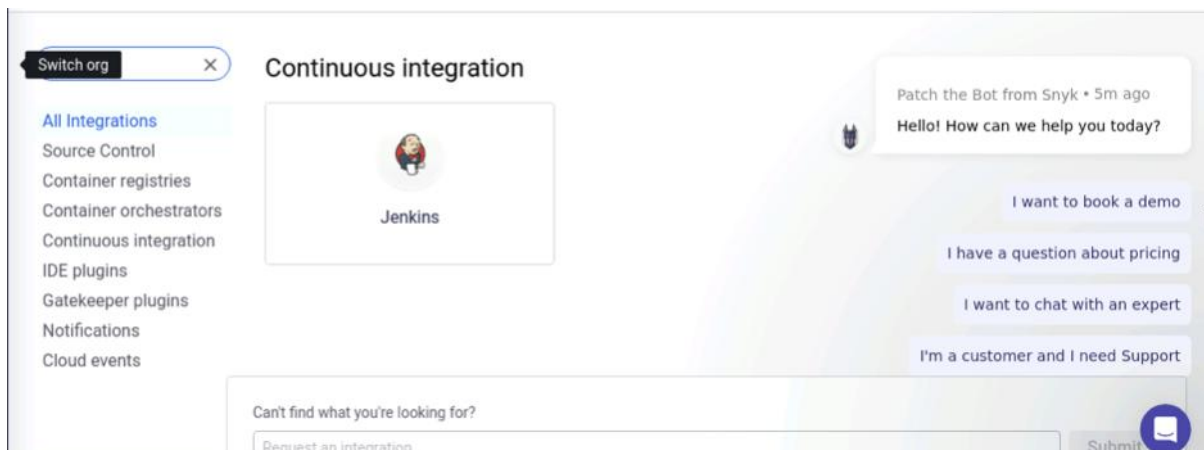
IDE plugins



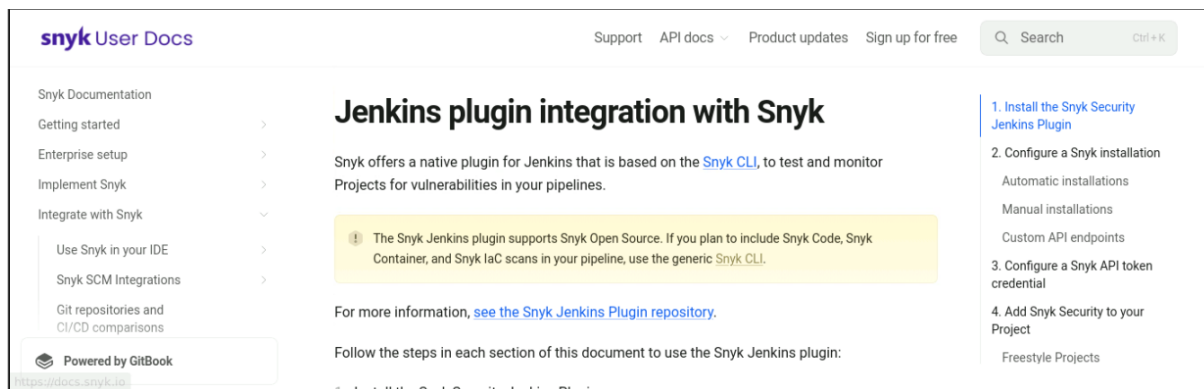








This will direct you to the documentation for integrating Snky with Jenkins.

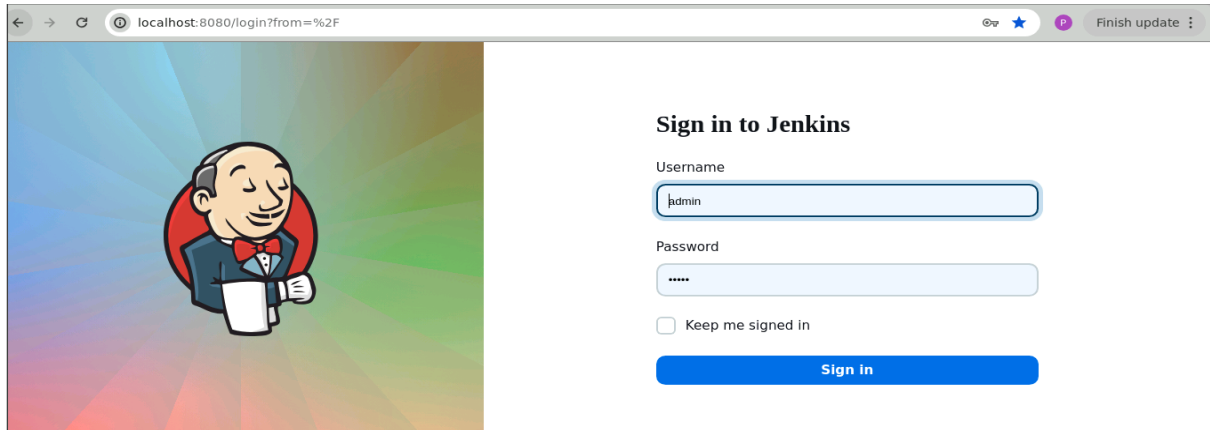


OUTPUT :-



Step 2: Create and configure a Jenkins job for Snyc integration

1. Open Jenkins and log in to the Jenkins account:



localhost:8080/login?from=%2F

Sign in to Jenkins

Username
admin

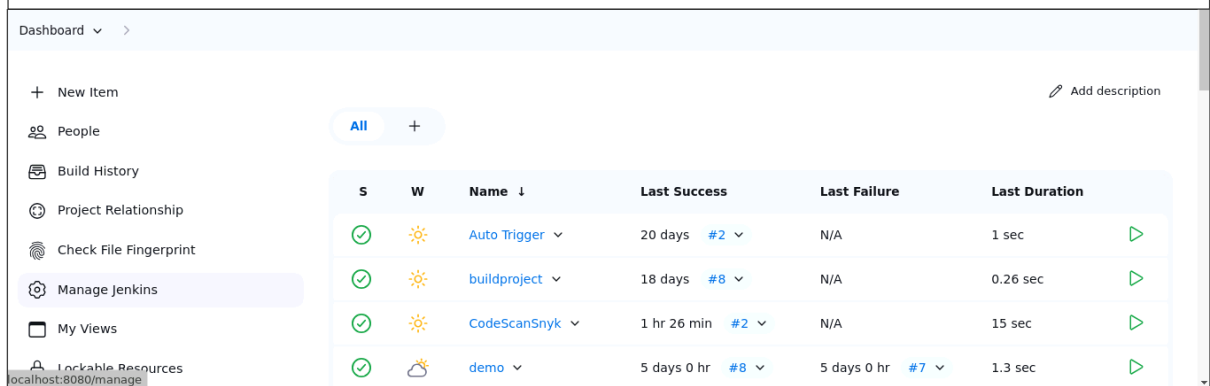
Password
.....

☐ Keep me signed in

Sign in


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

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



S	W	Name ↓	Last Success	Last Failure	Last Duration
✓	☀	Auto Trigger ▾	20 days #2 ▾	N/A	1 sec ▶
✓	☀	buildproject ▾	18 days #8 ▾	N/A	0.26 sec ▶
✓	☀	CodeScanSnyk ▾	1 hr 26 min #2 ▾	N/A	15 sec ▶
✓	☀	demo ▾	5 days 0 hr #8 ▾	5 days 0 hr #7 ▾	1.3 sec ▶

OUTPUT :-

 Jenkins

Manage Jenkins / Plugins

Search

🔍

🔔

👤

Plugins

📄 Updates 62

📦 Available plugins

⚙️ Installed plugins


⚙️ Advanced settings

📶 Download progress

🔍 sny

Name ↓	Health	Enabled
<div>Snyk Security Plugin 5.0.1</div> <div>Add the ability to test your code dependencies for vulnerabilities against Snyk database</div> <div>Report an issue with this plugin</div>	93	<input checked="" type="checkbox"/>

REST API Jenkins 2.516.1

 Jenkins

Search (CTRL+K)

🔔 3

🛡️ 2

👤 admin

🚪 log out

Dashboard > Manage Jenkins > Plugins

Plugins

📄 Updates 19

📦 Available plugins

⚙️ Installed plugins

⚙️ Advanced settings

🔍 snyk

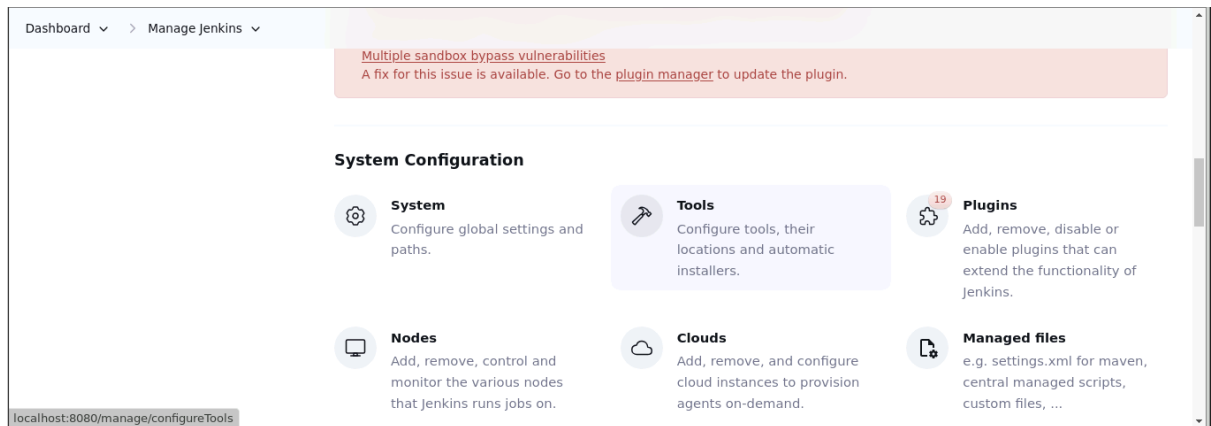
Install

🔄

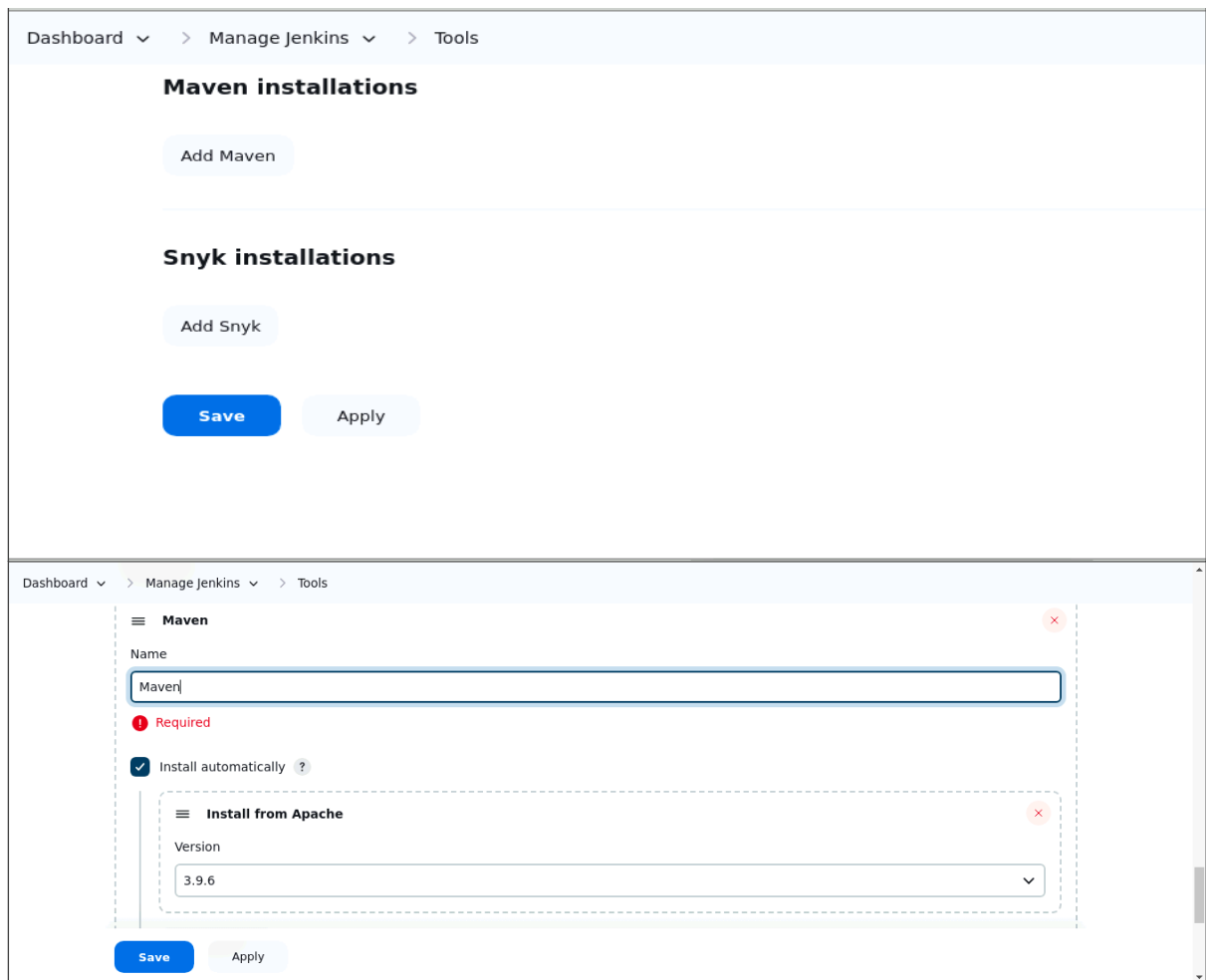
Install	Name ↓	Released
<input type="checkbox"/>	<div>Snyk Security 4.0.2</div> <div>DevSecOps</div> <div>Add the ability to test your code dependencies for vulnerabilities against Snyk database</div>	7 mo 29 days ago

REST API Jenkins 2.426.3

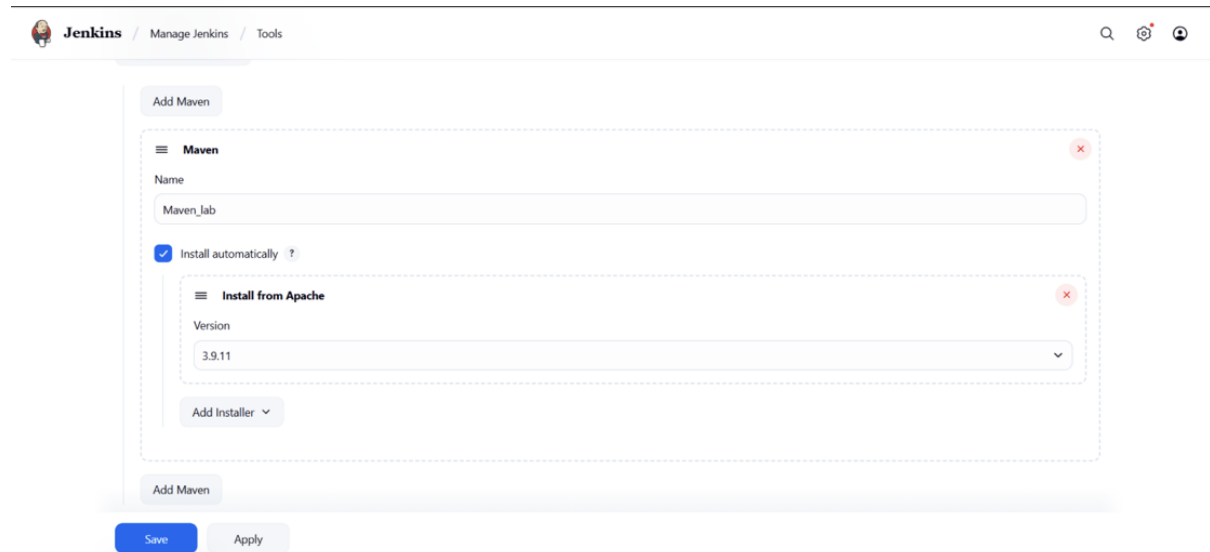
- To configure Maven and Snyk in the **Global Tool Configuration**, click on **Tools** inside **Manage Jenkins**



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

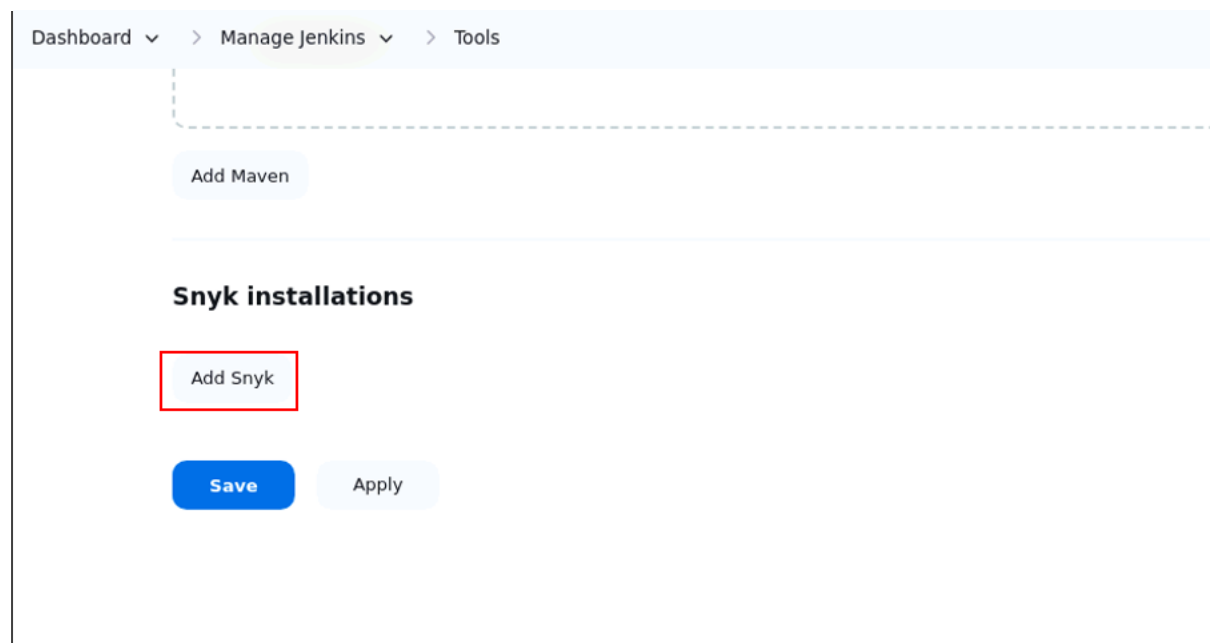


OUTPUT :-



The screenshot shows the Jenkins 'Tools' configuration page. The breadcrumb navigation at the top reads 'Jenkins / Manage Jenkins / Tools'. The main content area is titled 'Add Maven' and contains a dashed box for configuring a new tool. Inside this box, the tool name is 'Maven'. The 'Name' field is set to 'Maven_lab'. The 'Install automatically' checkbox is checked. Below this, there is a sub-section titled 'Install from Apache' with a 'Version' dropdown menu set to '3.9.11'. An 'Add Installer' button is visible at the bottom of the dashed box. Below the dashed box, there is another 'Add Maven' button. At the bottom of the page, there are 'Save' and 'Apply' buttons.

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



The screenshot shows the Jenkins 'Tools' configuration page with the breadcrumb navigation 'Dashboard > Manage Jenkins > Tools'. The 'Add Maven' button is visible. Below it, the 'Snyk installations' section is shown. In this section, the 'Add Snyk' button is highlighted with a red rectangle. At the bottom of the page, there are 'Save' and 'Apply' buttons.

Jenkins / Manage Jenkins / Tools

Snyk

Name
Snyk

☒ Install automatically ?

Install from snyk.io

Version
latest

Update policy interval (hours)
24

OS platform architecture
Auto-detection

Add Installer

Save Apply

Dashboard > Manage Jenkins > Tools

Snyk

Name
Snyk

Required

☒ Install automatically ?

Install from snyk.io

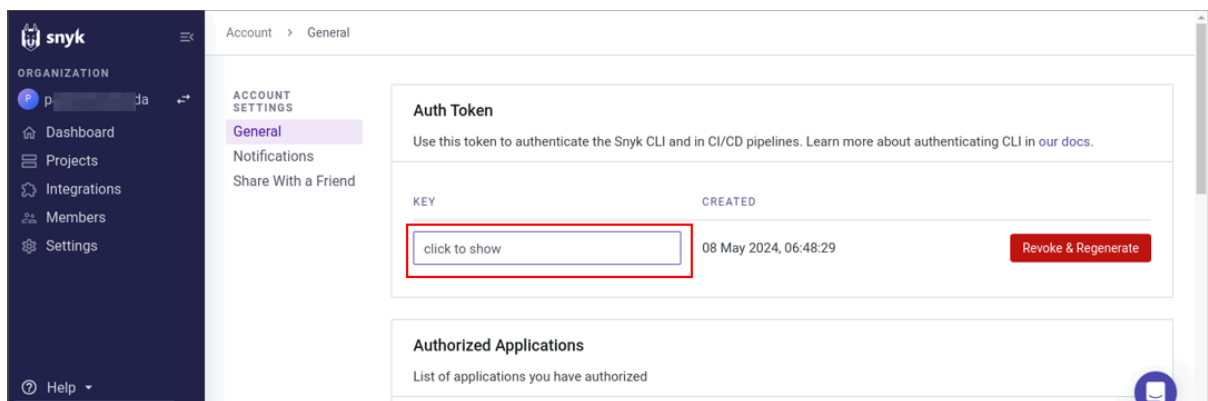
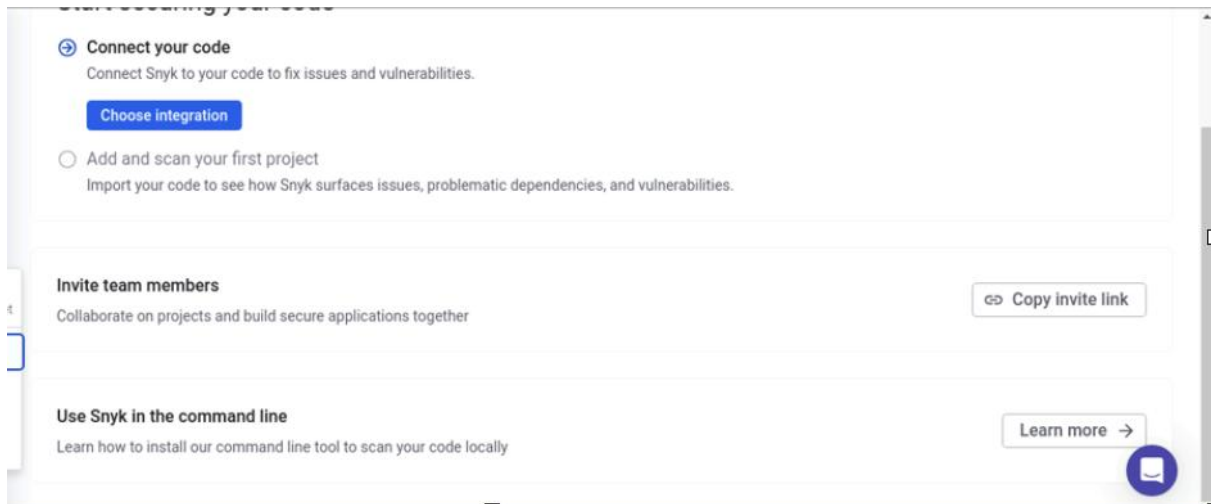
Version
latest

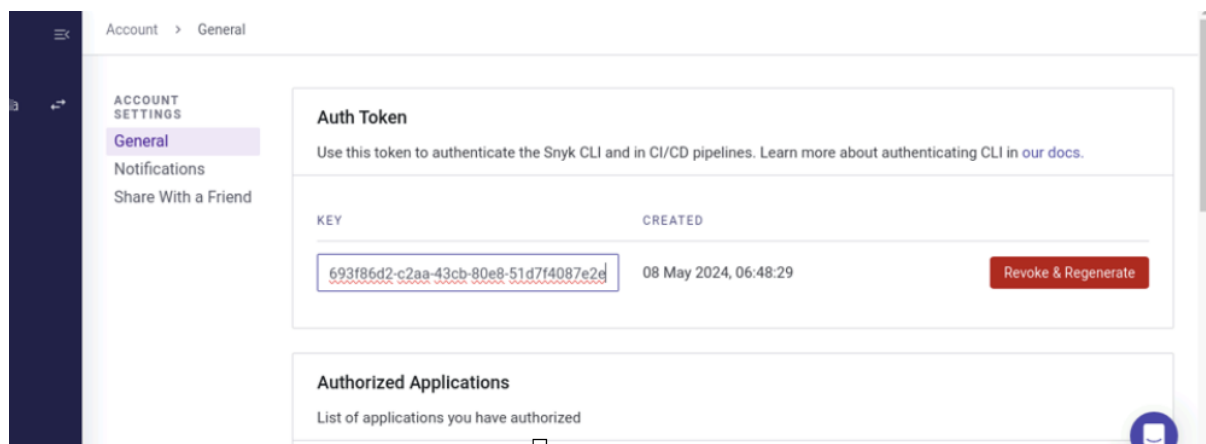
Save Apply

OUTPUT :-

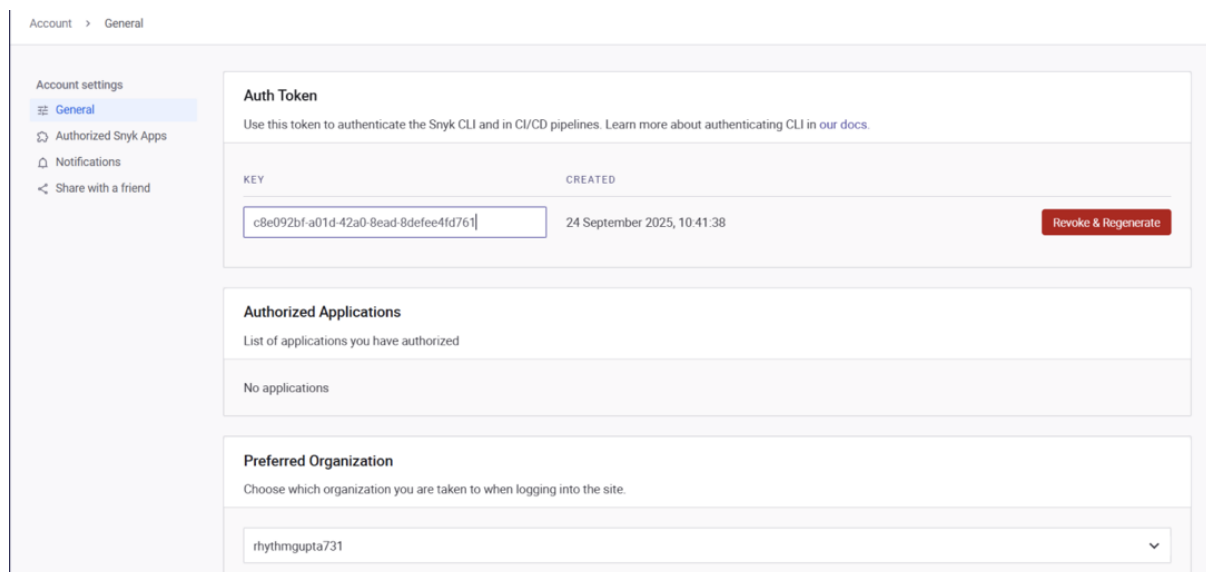
Step 3: Manage Snyc API and Jenkins credentials

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





OUTPUT :-



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

Dashboard >

+ New Item

People

Build History

Project Relationship

Check File Fingerprint

Manage Jenkins

My Views

Lockable Resources

All

+

S	W	Name ↓	Last Success	Last Failure	Last Duration
✓	☀	Auto Trigger	20 days #2	N/A	1 sec
✓	☀	buildproject	18 days #8	N/A	0.26 sec
✓	☀	CodeScanSnyk	1 hr 26 min #2	N/A	15 sec
✓	☀	demo	5 days 0 hr #8	5 days 0 hr #7	1.3 sec

localhost:8080/manage

Add description

Dashboard > Manage Jenkins

Add, remove, control and monitor the various nodes that Jenkins runs jobs on.

Add, remove, and configure cloud instances to provision agents on-demand.

e.g. settings.xml for maven, central managed scripts, custom files, ...

Security

Security

Secure Jenkins; define who is allowed to access/use the system.

Credentials

Configure credentials

Credential Providers

Configure the credential providers and types

Users

Create/delete/modify users that can log in to this Jenkins.

localhost:8080

Status Information

Dashboard > Manage Jenkins > Credentials

System (global) f00a1fd3-209e-43ea-9131-71fb4358dd39

Stores scoped to Jenkins

P	Store ↓	Domains
System	System	(global)

Icon: S M L

REST API Jenkins 2.426.3

- 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
geeks	admin/***** (geeks)	Username with password	geeks
geeks01	admin/***** (geeks01)	Username with password	geeks01
Maven	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 ?

.....

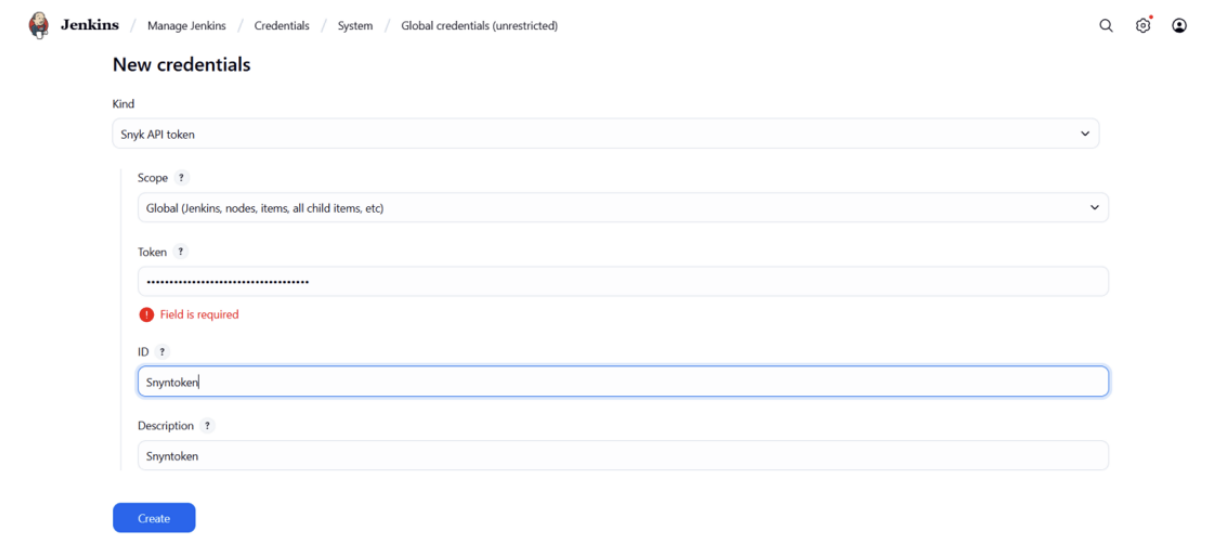
ID ?

Description ?

SynkToken

Create

OUTPUT :-



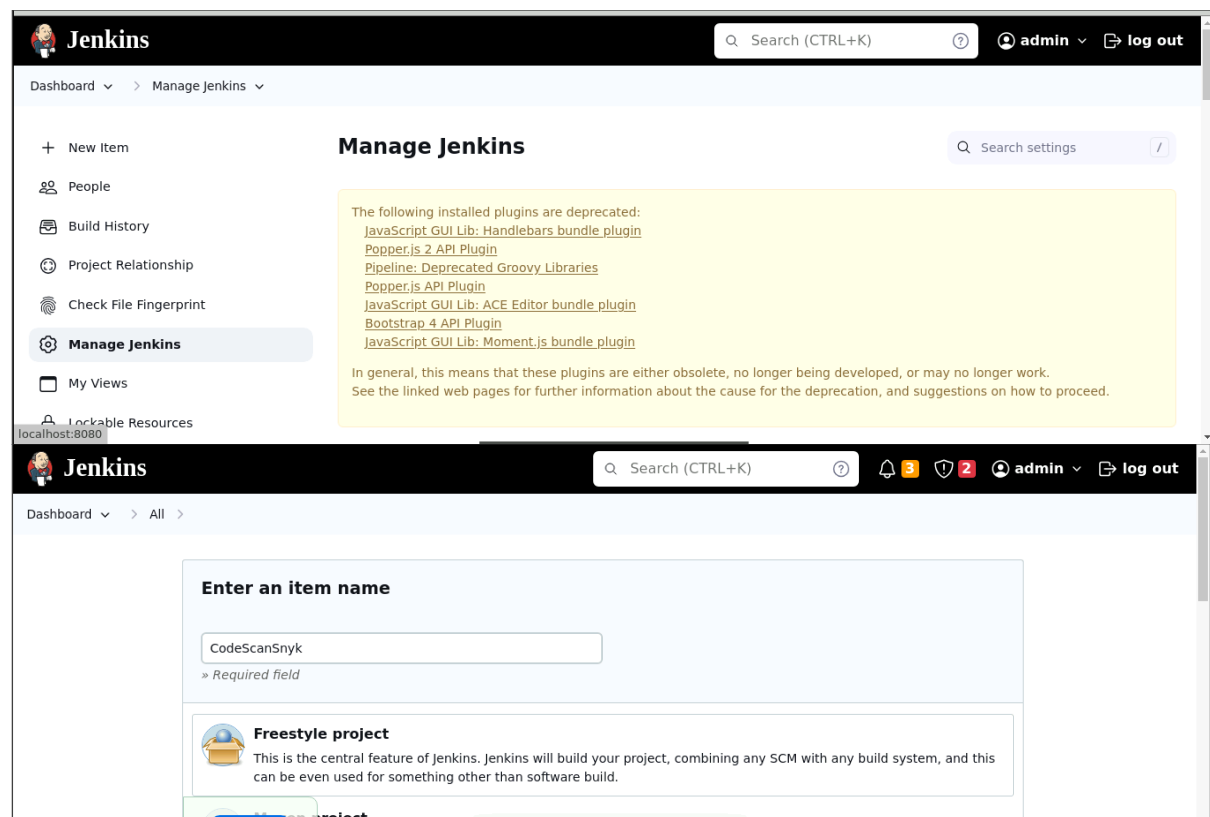
The screenshot shows the 'New credentials' page in Jenkins. The breadcrumb navigation at the top is 'Jenkins / Manage Jenkins / Credentials / System / Global credentials (unrestricted)'. The form has the following fields:

- Kind:** A dropdown menu with 'Snyk API token' selected.
- Scope:** A dropdown menu with 'Global (Jenkins, nodes, items, all child items, etc)' selected.
- Token:** A text input field with a red error message 'Field is required' below it.
- ID:** A text input field containing 'Snyntoken'.
- Description:** A text input field containing 'Snyntoken'.

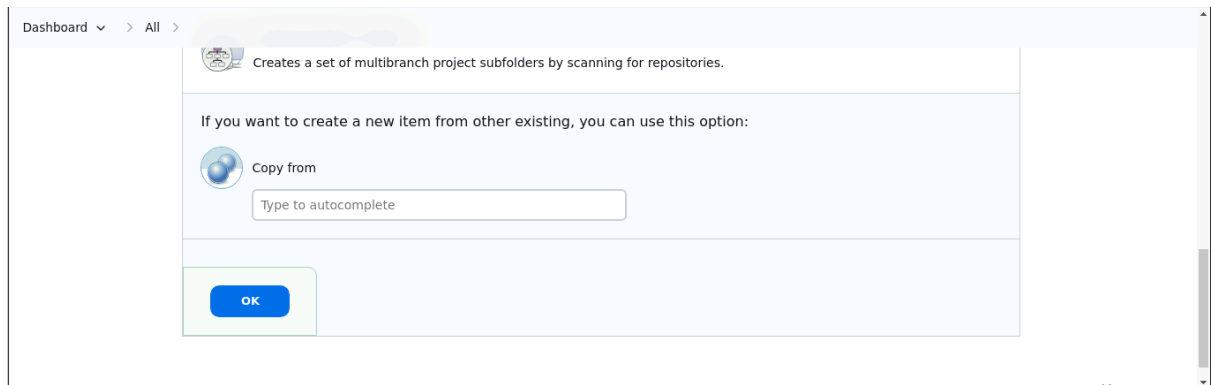
A blue 'Create' button is located at the bottom left of the form.

Step 4: Configure the Jenkins job for scanning

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

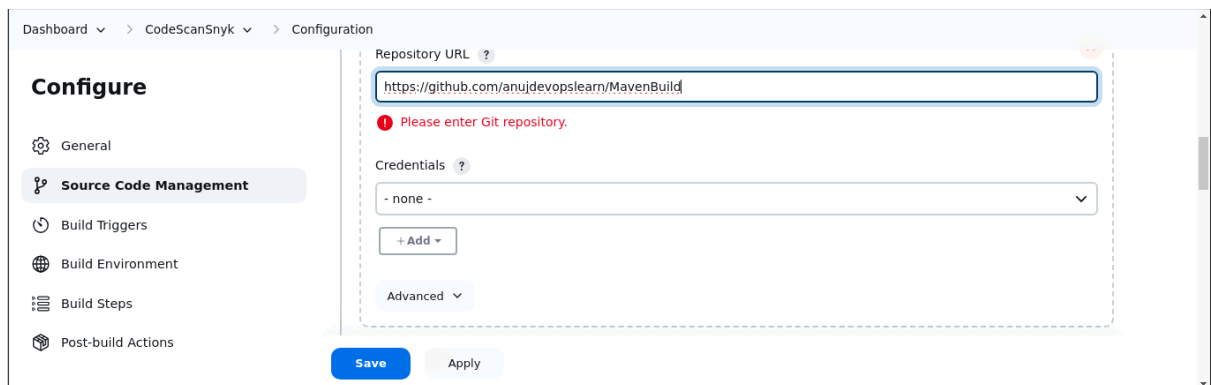


The screenshot shows the 'New Item' configuration page in Jenkins. The breadcrumb navigation is 'Dashboard > Manage Jenkins > All >'. The left sidebar shows 'New Item' as the selected option. The main area is titled 'Manage Jenkins' and contains a warning about deprecated plugins. Below this, the 'Enter an item name' section has a text input field with 'CodeScanSnyk' and a red asterisk indicating it is a required field. The 'Freestyle project' option is selected, showing a brief description: 'This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.'




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 Snky 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>



OUTPUT :-

 Jenkins

CodeScanSnyk / Configuration



Configure

General

Source Code Management

Triggers

Environment

Build Steps

Post-build Actions

Git

Repositories

Repository URL

https://github.com/rhythimgupta/Secure-Coding.git

Credentials

- none -

+ Add

Advanced

Add Repository

Branches to build

Branch Specifier (blank for 'any')

Save

Apply

Dashboard > CodeScanSnyk > Configuration

Configure

General

Source Code Management

Build Triggers

Build Environment

Build Steps

Post-build Actions

Build Steps

Add build step

Filter

Execute Windows batch command

Execute shell

Invoke Ant

Invoke Gradle script

Invoke Snyk Security task

Invoke top-level Maven targets

Provide Configuration files

Dashboard > CodeScanSnyk > Configuration

Configure

General

Source Code Management

Build Triggers

Build Environment

Build Steps

Post-build Actions

☒ Fail the build if errors occur

☒ Monitor project on build

Snyk API token

SynkToken

+ Add

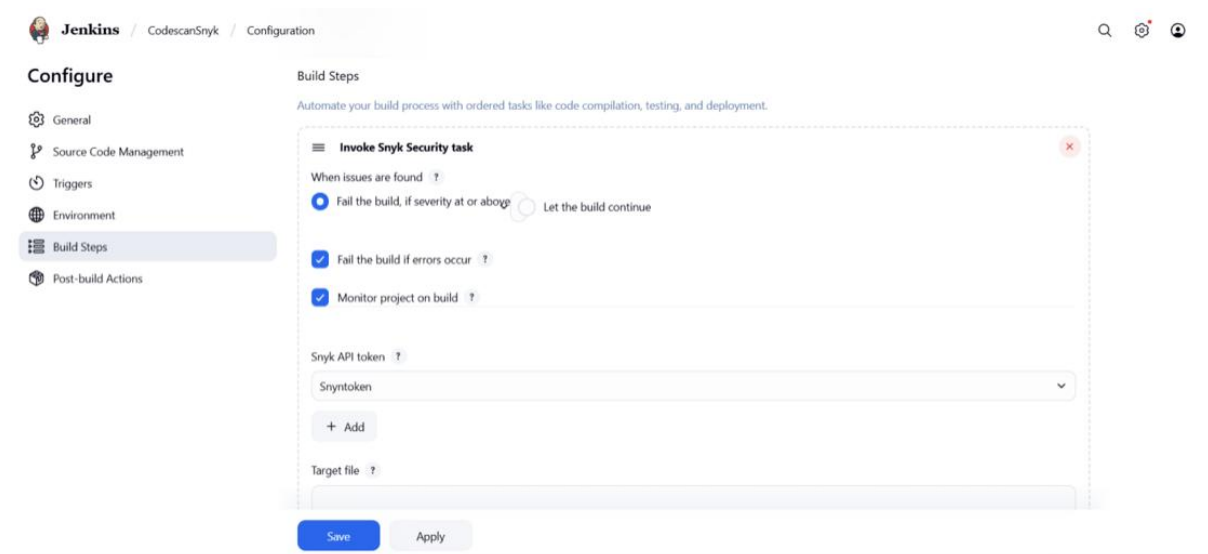
Target file

Save

Apply

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

OUTPUT :-



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

OUTPUT :-

The screenshot shows the Jenkins web interface for a project named 'CodescanSnyk'. On the left, there is a sidebar with navigation links: Status, Changes, Workspace, Build Now, Configure, Delete Project, and Rename. The main area displays the 'Permalinks' section, listing recent builds with their status and timestamps. Below this, a 'Builds' section shows a list of builds with their IDs and times. At the bottom right, there is a 'REST API' link and the Jenkins version '2.516.3'.

The screenshot shows the 'Console Output' for build #1 of the 'CodescanSnyk' project. The output text details the build process, including the user 'Rhythm gupta', the workspace path, and the execution of various commands like 'git rev-parse', 'git fetch', and 'git checkout'. It also shows the results of a Snyk security scan, indicating that 1 known vulnerability was found. The build ends with a 'FAILURE' status.

4. To navigate to the Snyk tool to review code, scan reports under the **Projects** section

The screenshot shows the Snyk web application interface. The left sidebar contains navigation links: Organization, Dashboard, Projects, Integrations, Members, and Settings. The main area displays the 'Projects' section for the organization 'palak.kharbanda'. It shows a list of projects with a search bar and filters. A specific project 'anujdevopslearn/MavenBuild' is highlighted. Below the project list, there is a section titled 'Ready to import another project?' with a button to 'Add projects'.

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.

OUTPUT :-

