

Lab Exercise 19

Setting up Snyk for SAST in Jenkins

Name:- Vansh Bhatt

Sap ID:- 500125395

Batch:- DevOps B1

R. NO.:- R2142231689

To:- Hitesh Kumar Sharma Sir

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

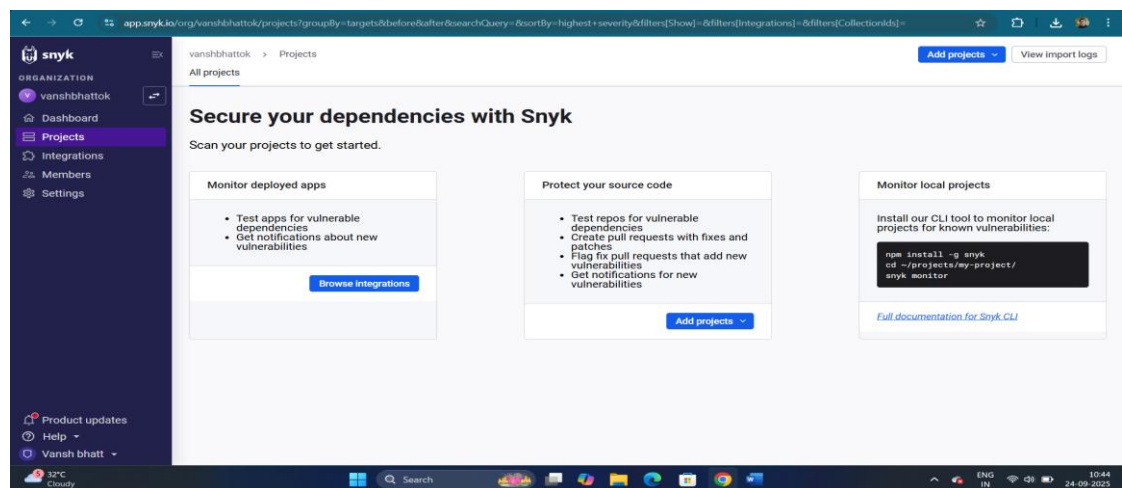
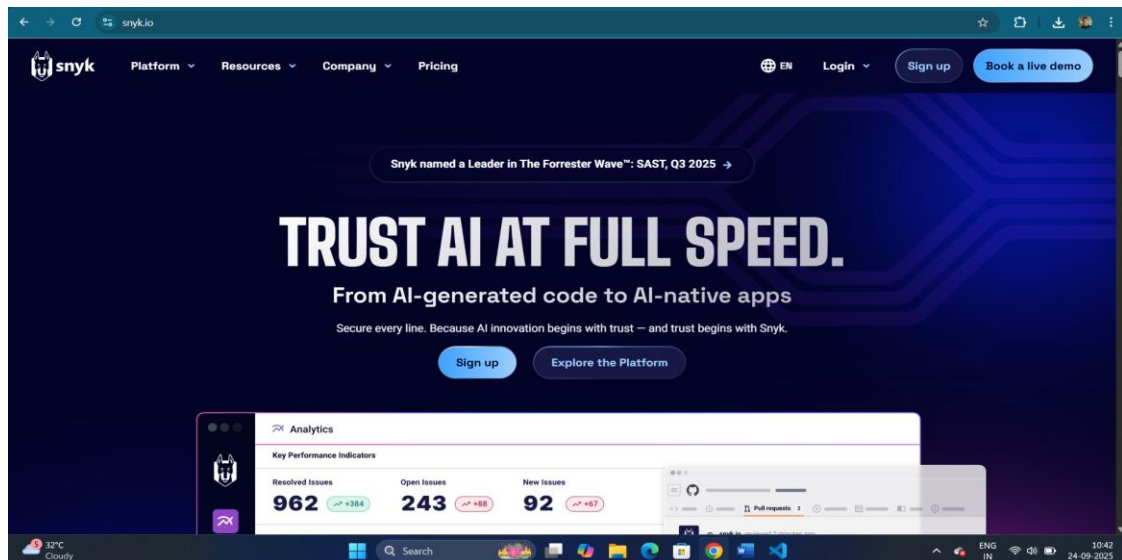
Prerequisites: None

Steps to be followed:

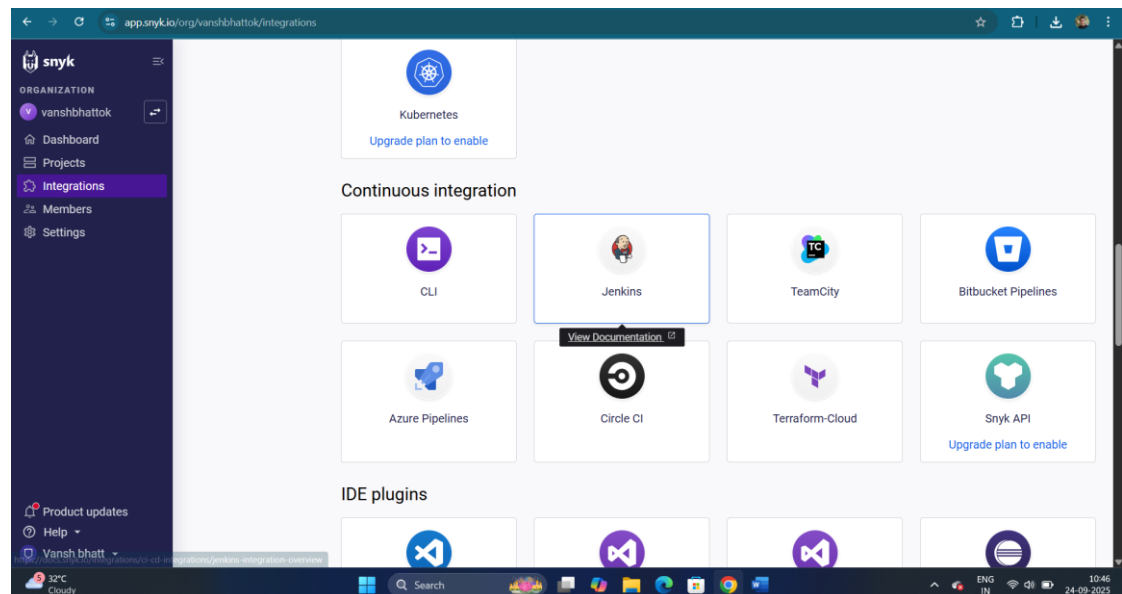
1. Configure Snyk as a SAST scan tool
2. Create and configure a Jenkins job for Snyk integration
3. Manage Snyk API and Jenkins credentials
4. Configure the Jenkins job for scanning

Step 1: Configure Snyk as a SAST scan tool

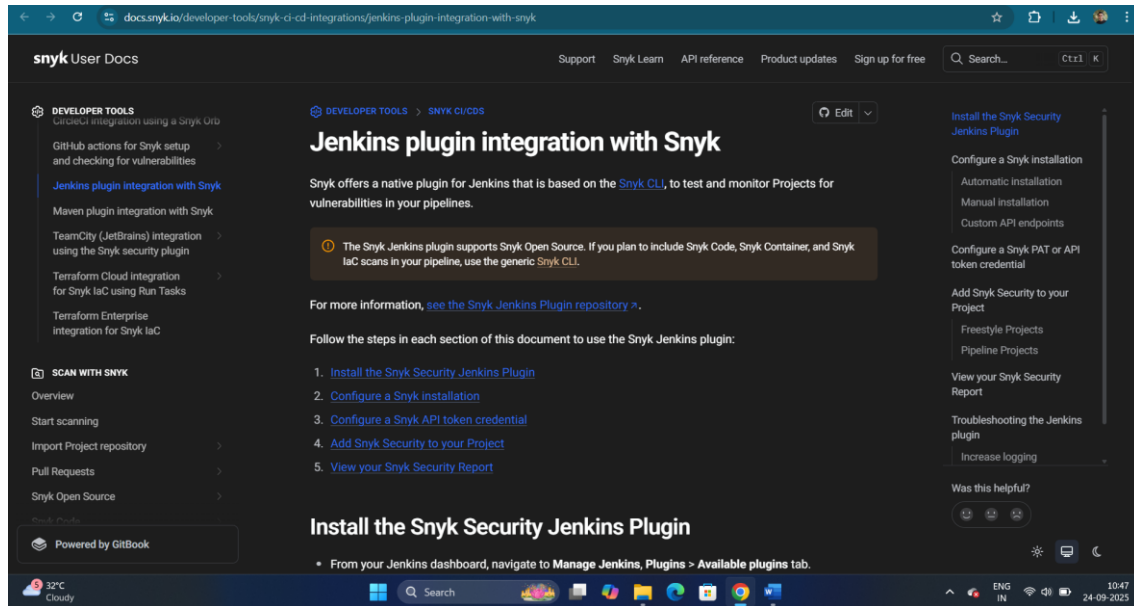
- 1.1 Visit <https://snyk.io/>, sign up for a new Snyk account, and log in



1.2 Navigate to Integrations and select Jenkins

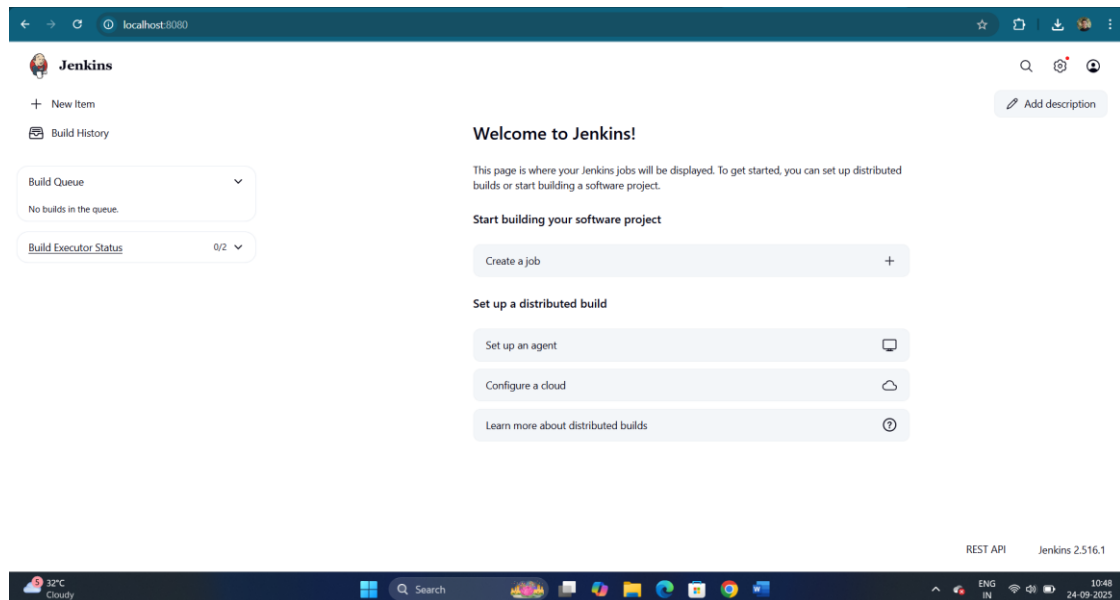


This will direct you to the documentation for integrating Snky with Jenkins.



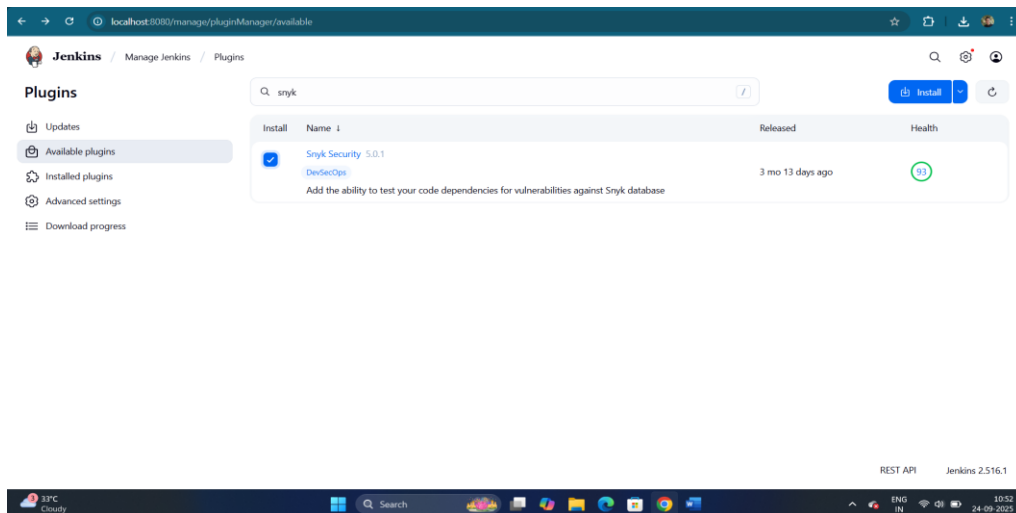
Step 2: Create and configure a Jenkins job for Snky integration

2.1 Open Jenkins and log in to the Jenkins account:

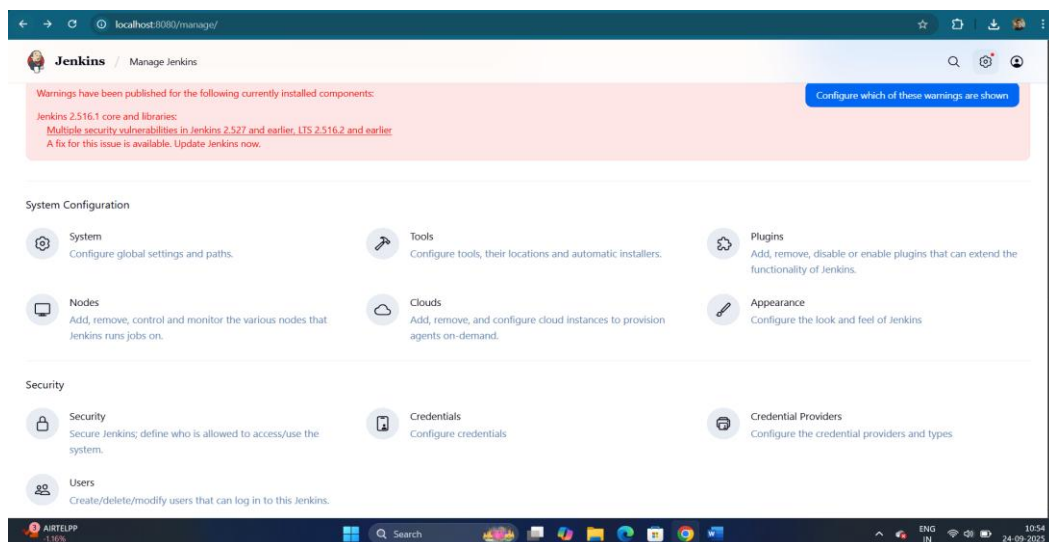


2.2 To install the Snky plugin, navigate to **Manage Jenkins** and click **Available Plugins**, search for **Snky Security** plugin, and then click **Install**

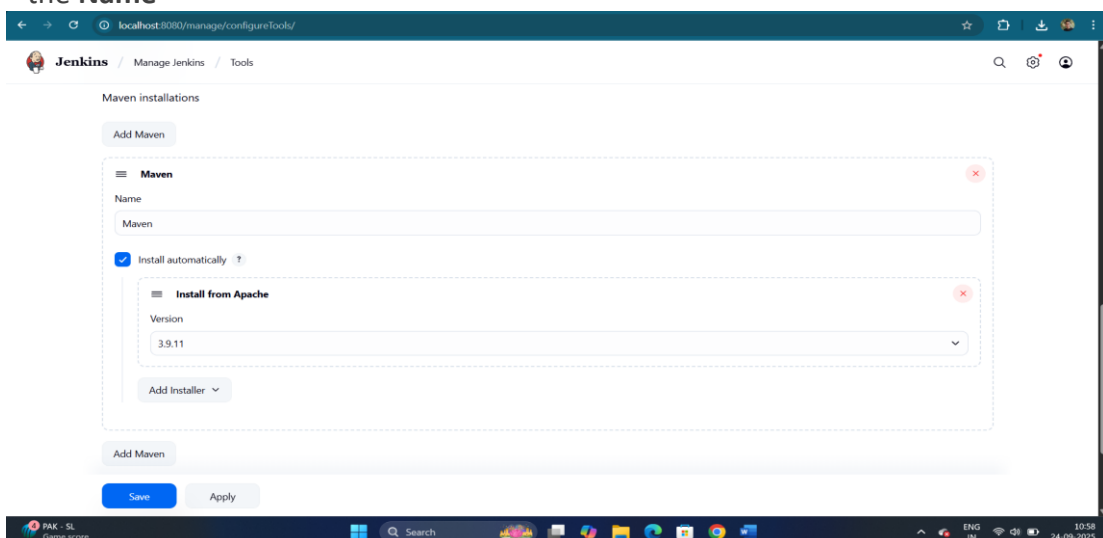
Note: The credentials for accessing Jenkins in the lab are Username: **admin** and Password: **admin**.



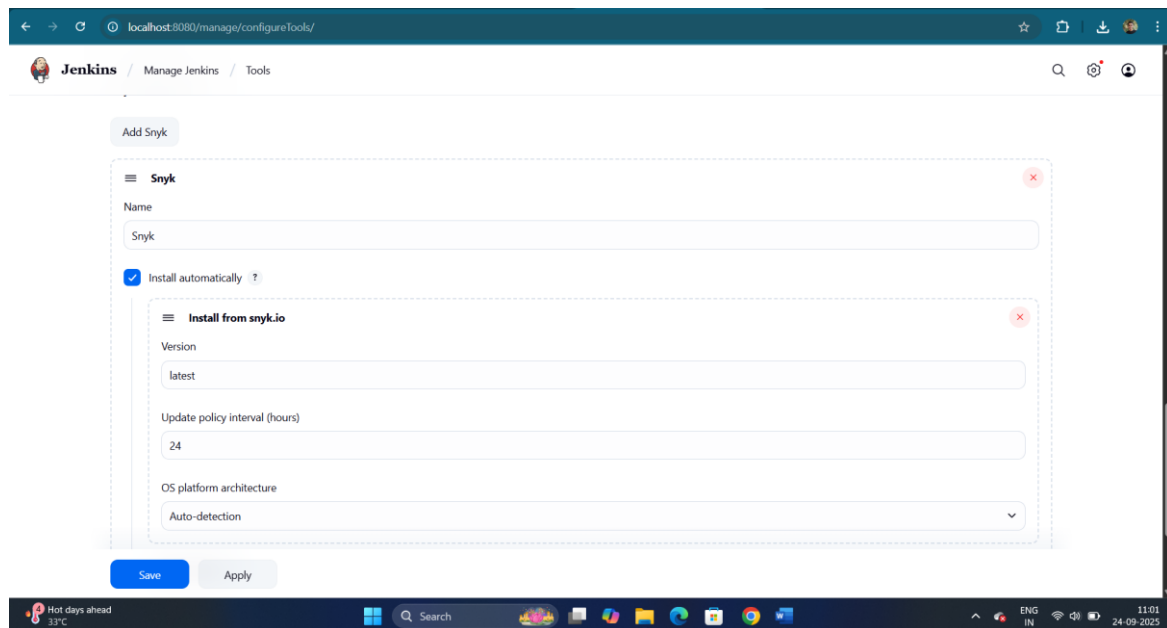
2.3 To configure Maven and Snyk in the **Global Tool Configuration**, click on **Tools** inside **Manage Jenkins**



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

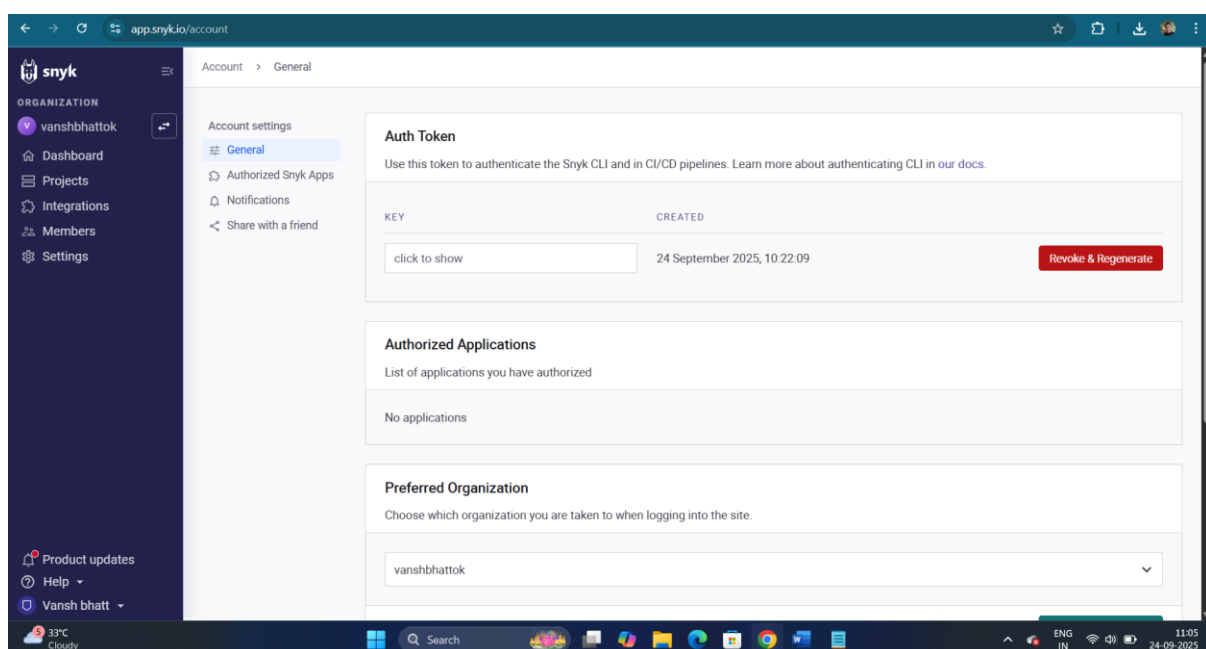


2.5 To add Snky, click on **Add Snky** under **Snky Installations**, add **Name** as **Snky**, and click on the **Save** button

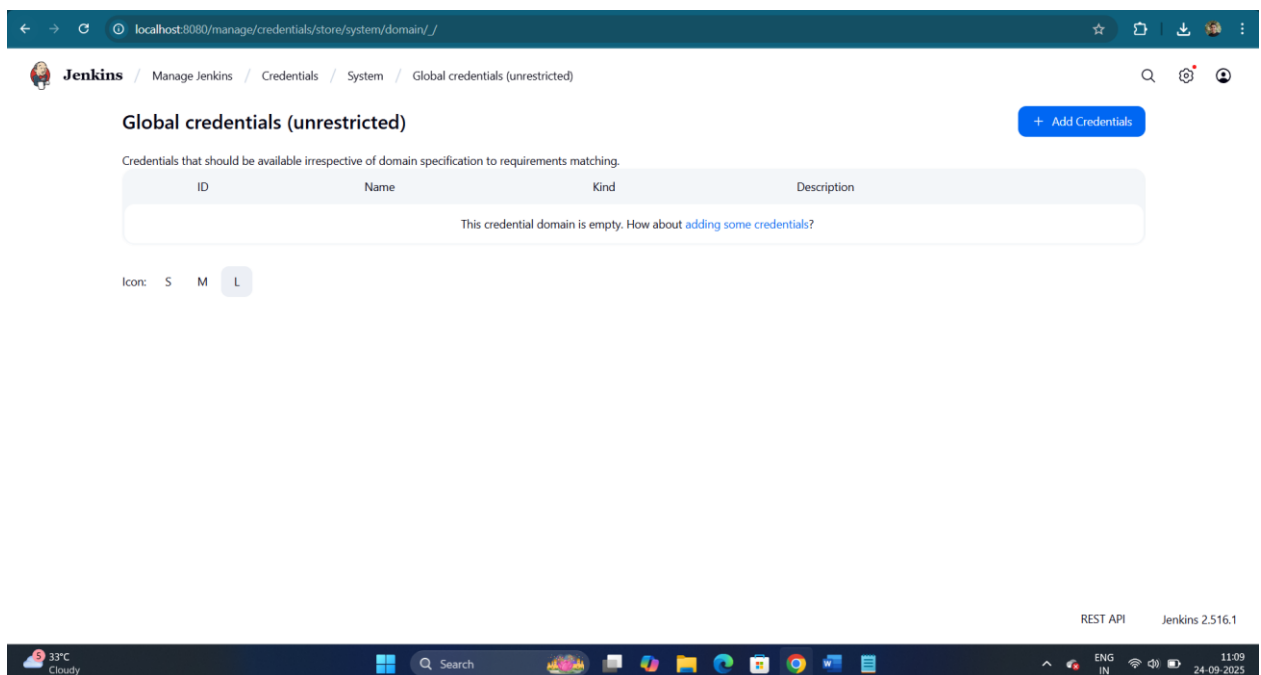
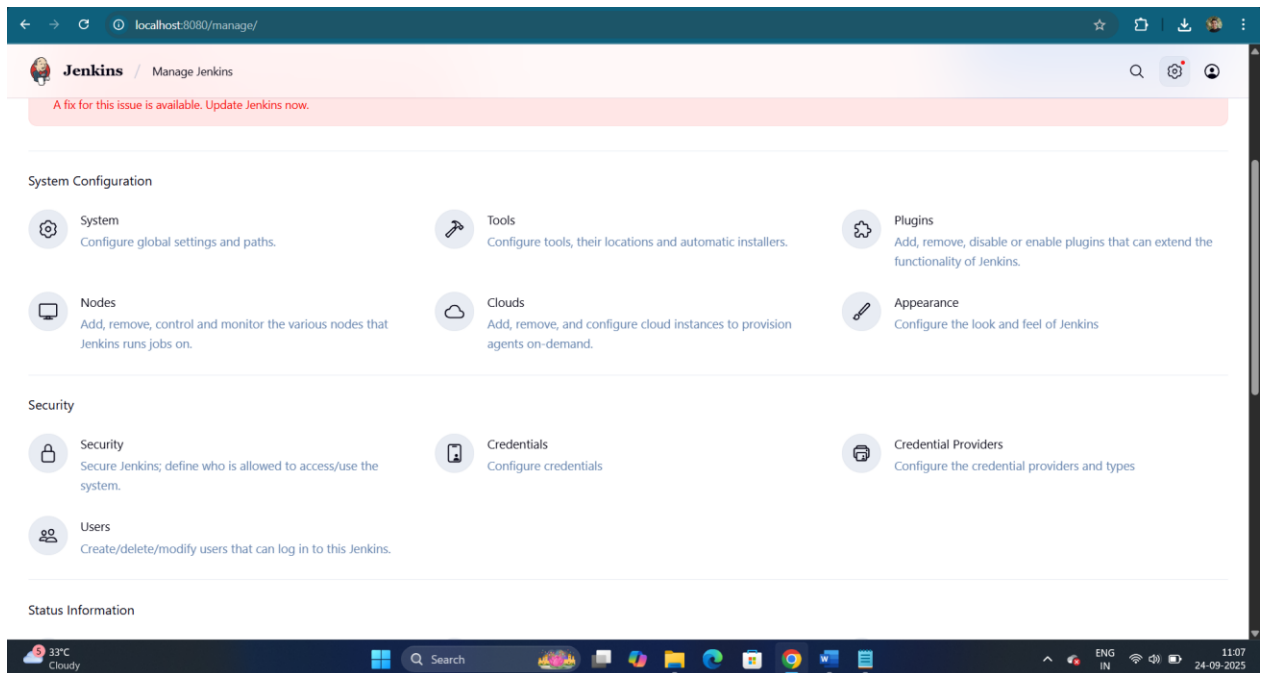


Step 3: Manage Snky API and Jenkins credentials

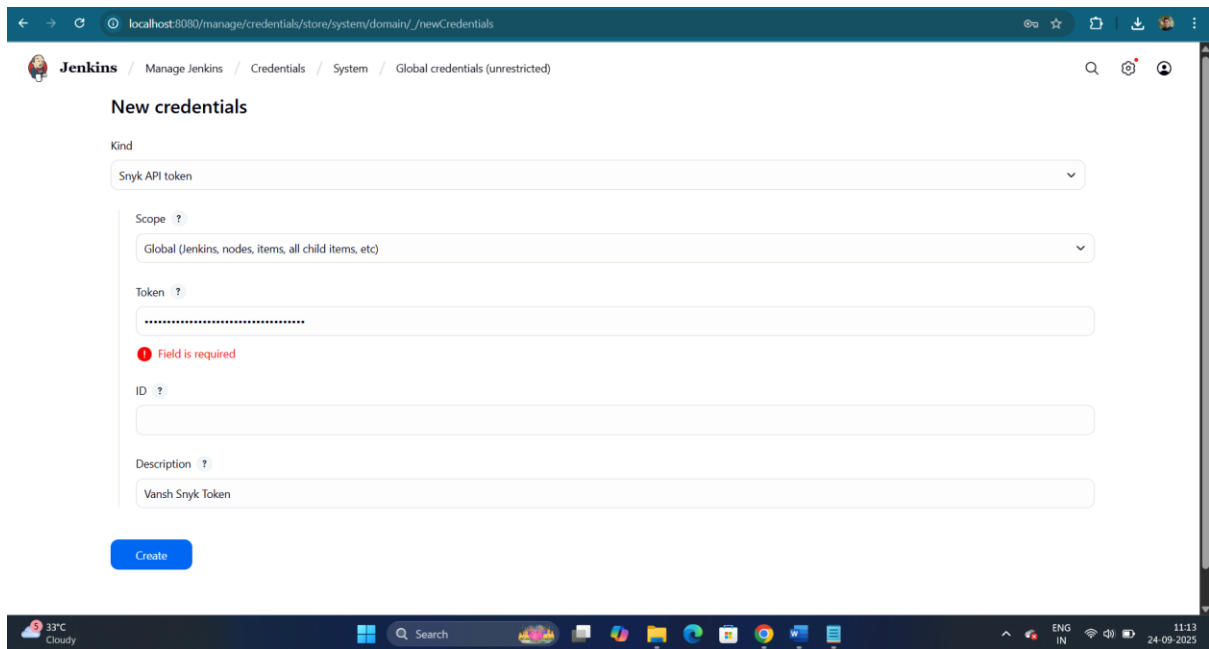
3.1 To retrieve your Snky API token, go to **Account Settings** in your Snky account, click on **Click to show** under the Auth Token key field, and copy the token for further reference



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



localhost:8080/manage/credentials/store/system/domain/_/newCredentials

Jenkins / Manage Jenkins / Credentials / System / Global credentials (unrestricted)

New credentials

Kind: Snyk API token

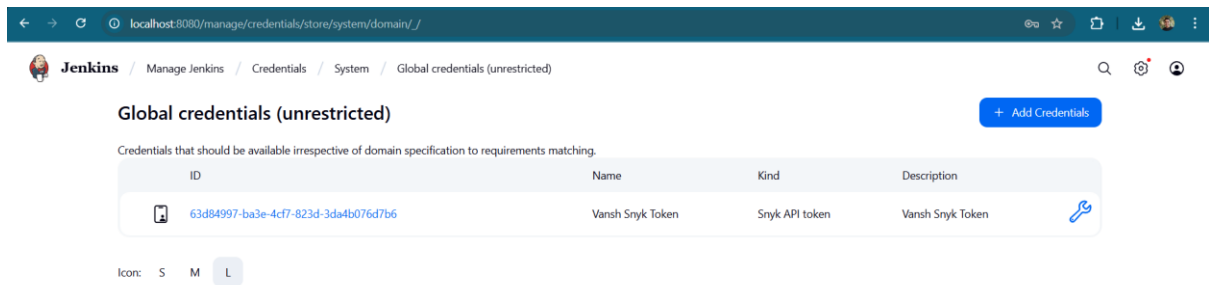
Scope: Global (Jenkins, nodes, items, all child items, etc)

Token:
 Field is required

ID:
 Description: Vansh Snyk Token

Create

33°C Cloudy 11:13 24-09-2025



localhost:8080/manage/credentials/store/system/domain/_/

Jenkins / Manage Jenkins / Credentials / System / Global credentials (unrestricted)

Global credentials (unrestricted) [+ Add Credentials](#)

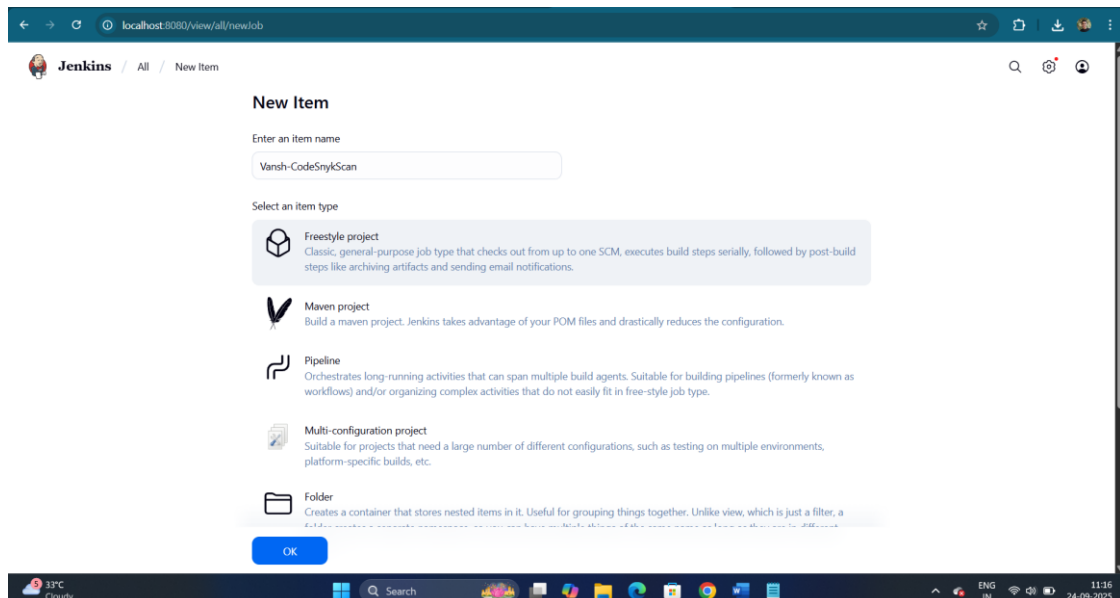
Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
63d84997-ba3e-4c77-823d-3da4b076d7b6	Vansh Snyk Token	Snyk API token	Vansh Snyk Token

Icon: S M L

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**



localhost:8080/view/all/newJob

Jenkins / All / New Item

New Item

Enter an item name: Vansh-CodeSnykScan

Select an item type:

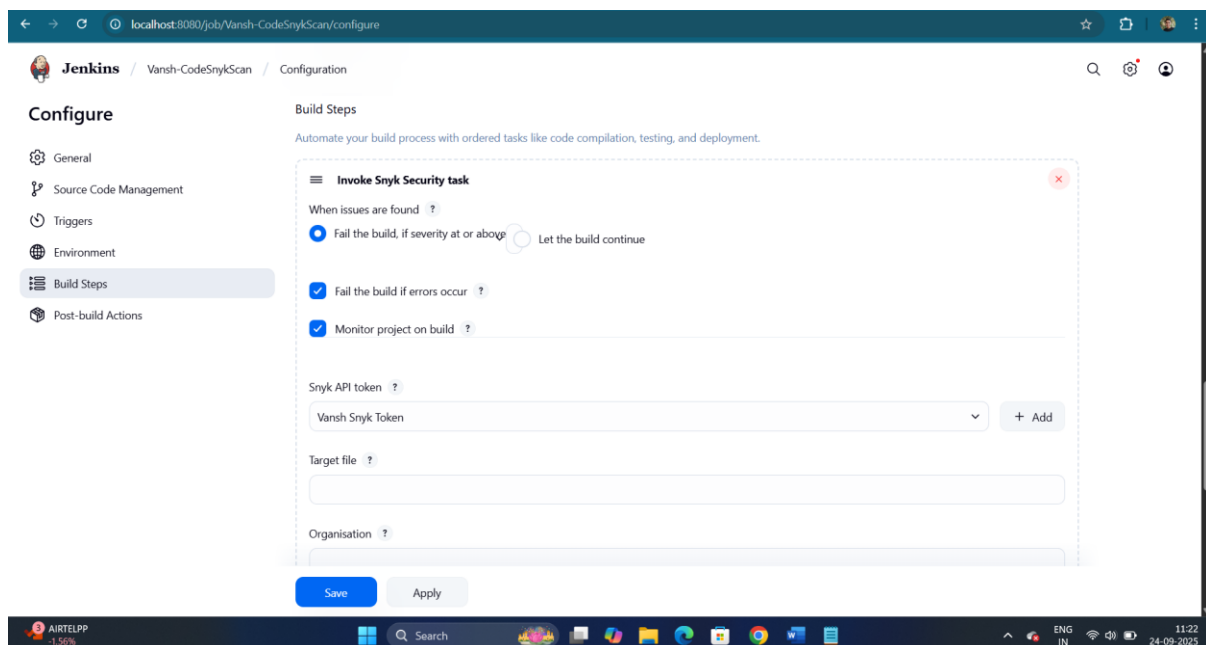
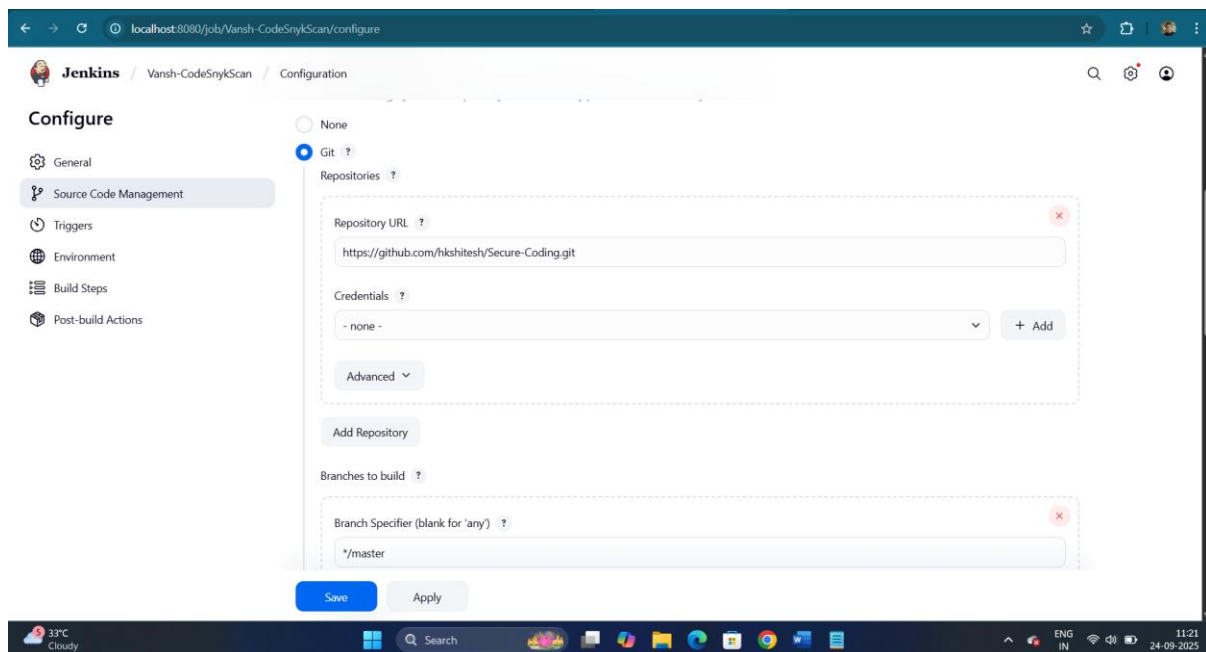
- Freestyle project**
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.
- Maven project
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.
- Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
- Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
- Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate workspace.

OK

33°C Cloudy 11:16 24-09-2025

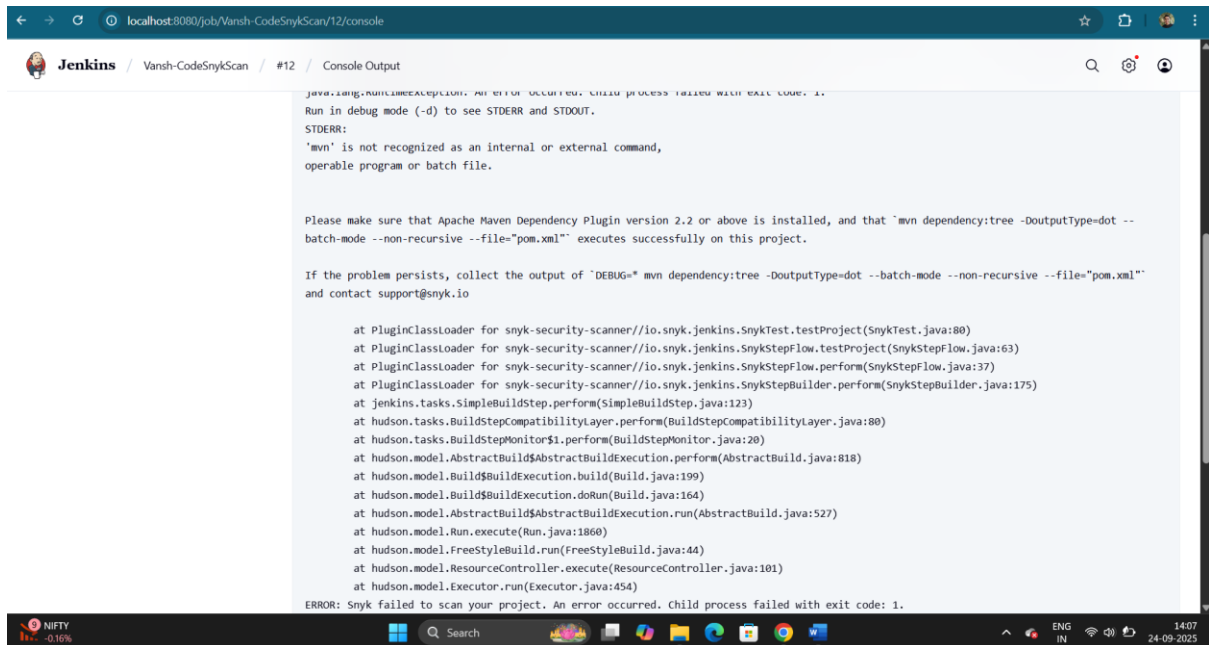
4.2 After creating a job, go to **Source Code Management** and enter the GitHub repository URL. Then, under **Build Steps**, add the build step **Invoke Snyk Security task** with the name **SnykToken**. Finally, click the **Save** button to create the build.

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



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

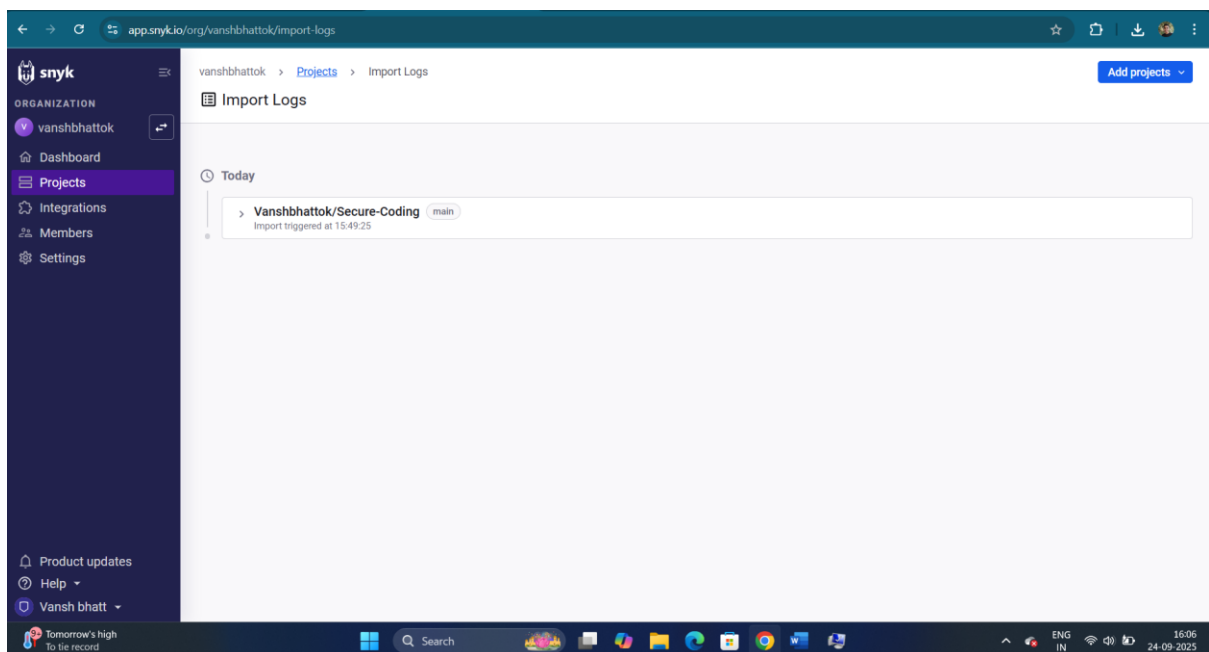
4.3 To check the build status, click on the build link under **Permalinks**. After that, click on **Console Output**



The screenshot shows the Jenkins web interface for a build named 'Vansh-CodeSnykScan' with ID '#12'. The 'Console Output' tab is selected, displaying the following text:

```
java.lang.RuntimeException: An error occurred. Child process failed with exit code: 1.  
Run in debug mode (-d) to see STDERR and STDOUT.  
STDERR:  
'mvn' is not recognized as an internal or external command,  
operable program or batch file.  
  
Please make sure that Apache Maven Dependency Plugin version 2.2 or above is installed, and that 'mvn dependency:tree -DoutputType=dot --  
batch-mode --non-recursive --file="pom.xml"' executes successfully on this project.  
  
If the problem persists, collect the output of 'DEBUG=* mvn dependency:tree -DoutputType=dot --batch-mode --non-recursive --file="pom.xml"'  
and contact support@snyk.io  
  
at PluginClassLoader for snyk-security-scanner//io.snyk.jenkins.SnykTest.testProject(SnykTest.java:80)  
at PluginClassLoader for snyk-security-scanner//io.snyk.jenkins.SnykStepFlow.testProject(SnykStepFlow.java:63)  
at PluginClassLoader for snyk-security-scanner//io.snyk.jenkins.SnykStepFlow.perform(SnykStepFlow.java:37)  
at PluginClassLoader for snyk-security-scanner//io.snyk.jenkins.SnykStepBuilder.perform(SnykStepBuilder.java:175)  
at jenkins.tasks.SimpleBuildStep.perform(SimpleBuildStep.java:123)  
at hudson.tasks.BuildStepCompatibilityLayer.perform(BuildStepCompatibilityLayer.java:80)  
at hudson.tasks.BuildStepMonitor$1.perform(BuildStepMonitor.java:20)  
at hudson.model.AbstractBuild$AbstractBuildExecution.perform(AbstractBuild.java:818)  
at hudson.model.Build$BuildExecution.build(Build.java:199)  
at hudson.model.Build$BuildExecution.doRun(Build.java:164)  
at hudson.model.AbstractBuild$AbstractBuildExecution.run(AbstractBuild.java:527)  
at hudson.model.Run.execute(Run.java:1860)  
at hudson.model.FreeStyleBuild.run(FreeStyleBuild.java:44)  
at hudson.model.ResourceController.execute(ResourceController.java:101)  
at hudson.model.Executor.run(Executor.java:454)  
ERROR: Snyk failed to scan your project. An error occurred. Child process failed with exit code: 1.
```

4.4 To navigate to the Snyk 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