

Experiment No 08

AIM

To perform Static Application Security Testing (SAST) using Snyk integrated with Jenkins and generate comprehensive vulnerability reports for a sample application.

TOOLS & TECHNOLOGIES

Tool/Technology	Purpose	Version
Jenkins	Continuous Integration/Delivery automation	Latest LTS
Snyk	Vulnerability scanning and dependency analysis	Latest
Maven	Build automation and project management	3.x
GitHub	Source code management and version control	Latest

THEORY

DevSecOps & Security Integration

DevSecOps is a development paradigm that integrates security practices across the software development lifecycle (SDLC). It follows a **Shift-Left** methodology, detecting vulnerabilities early in the process—making fixes faster and less expensive.

Key Principles:

- **Automation-First Approach:** Automated security testing within CI/CD
- **Continuous Monitoring:** Real-time security oversight
- **Collaboration:** Dev, Sec, and Ops teams work together, not in silos
- **Risk-Based Decision Making:** Security priorities determined by risk assessments

Static Application Security Testing (SAST)

SAST is a **white-box** testing methodology analyzing source code, bytecode, or binaries **without execution**, providing:

- Early vulnerability detection
- Code-wide coverage
- Compliance and secure coding practices

SAST Process:

- 1. Code Parsing: Transforms code to abstract syntax tree (AST)
- 2. Data Flow Analysis: Tracks data movement
- 3. Control Flow Analysis: Examines execution paths
- 4. Pattern Matching: Locates known vulnerabilities
- 5. Rule Engine: Applies rules to detect risks

Vulnerability Types Detected:

- Code logic flaws: XSS, SQLi, buffer overflow
- Unsafe and outdated dependencies
- Insecure configurations
- Input validation errors
- Auth & access control issues

Snyk: Developer-first SAST Tool

- Database of 700,000+ vulnerabilities
- Real-time updates via multiple sources
- Integrates with GitHub, Jenkins, Maven, npm, etc.
- Produces risk-prioritized reports and actionable fixes

Severity Classification:

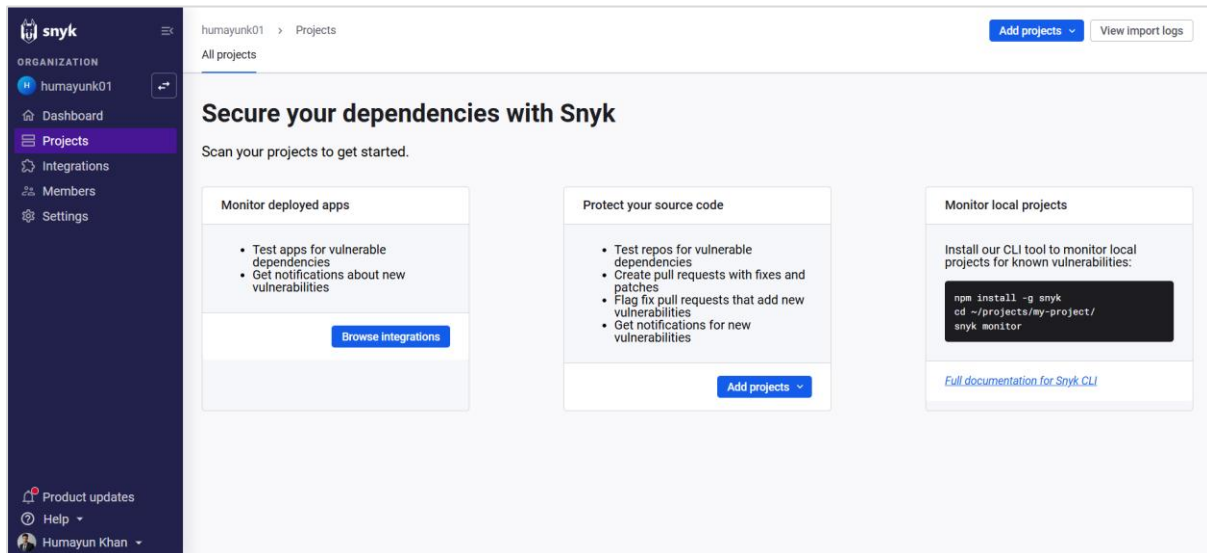
Severity Level	CVSS Score	Risk Level	Action Required
Critical	9.0–10.0	Extreme	Immediate fix
High	7.0–8.9	High	Fix in 30 days
Medium	4.0–6.9	Moderate	Fix in 90 days
Low	0.1–3.9	Low	Fix as convenient

PROCEDURE

METHOD 1: Snyc Dashboard Integration

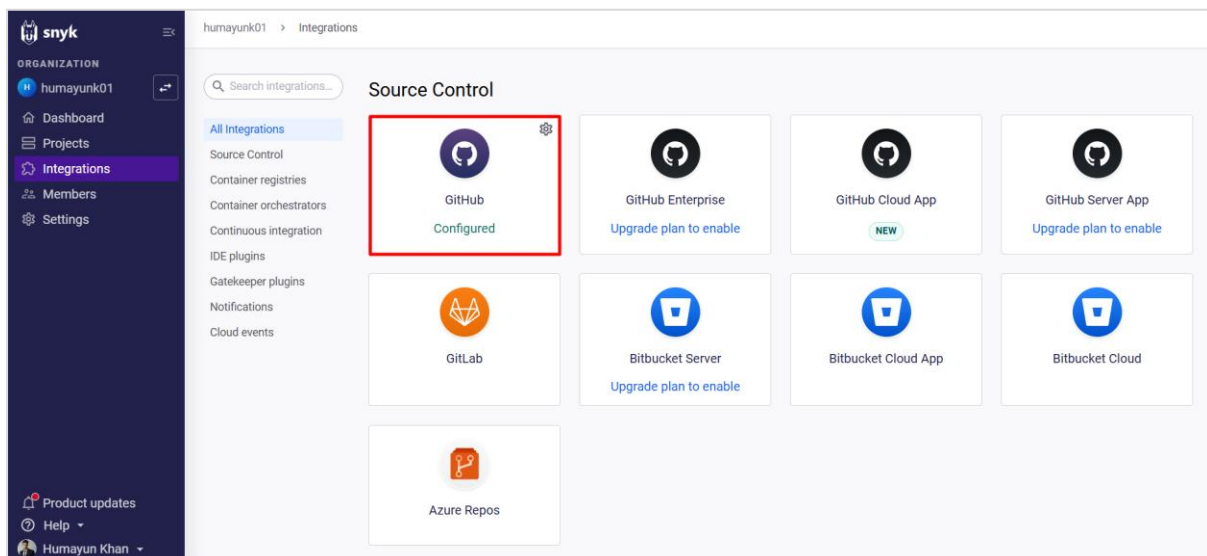
Step 1: Account Setup

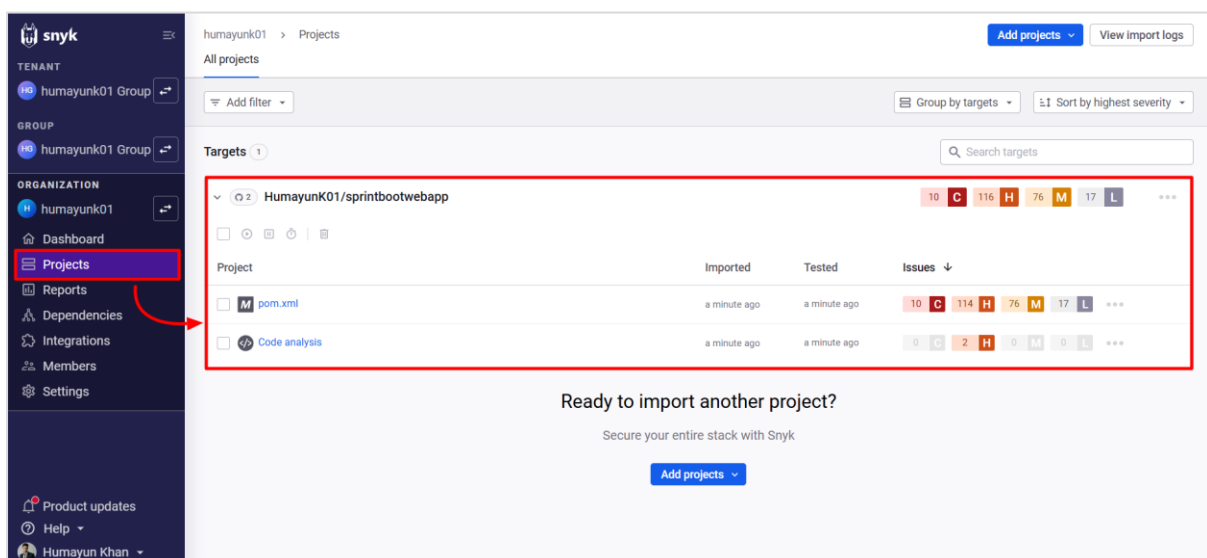
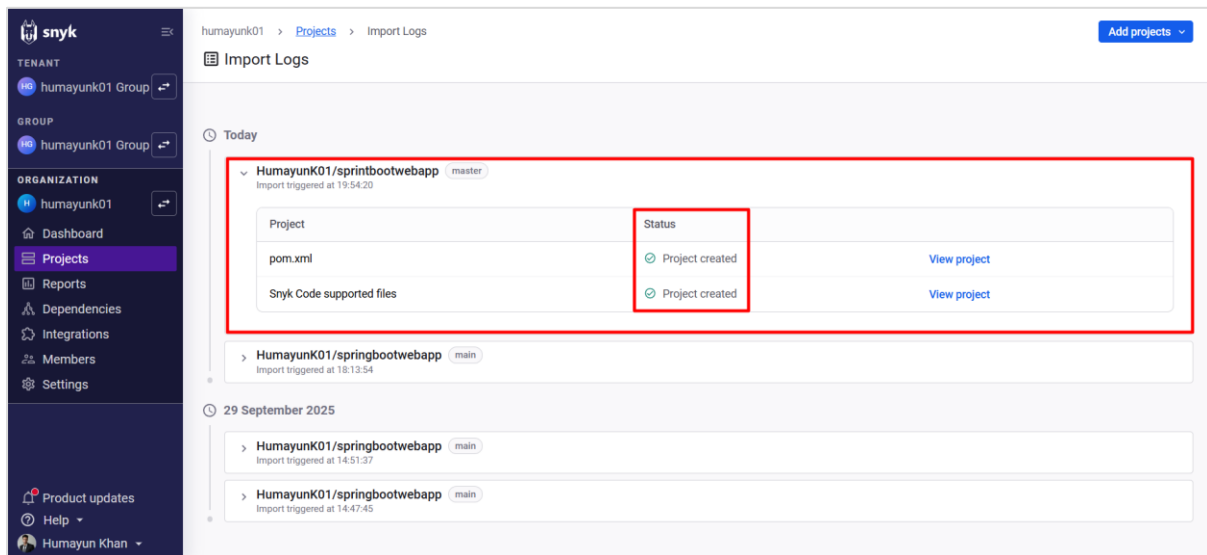
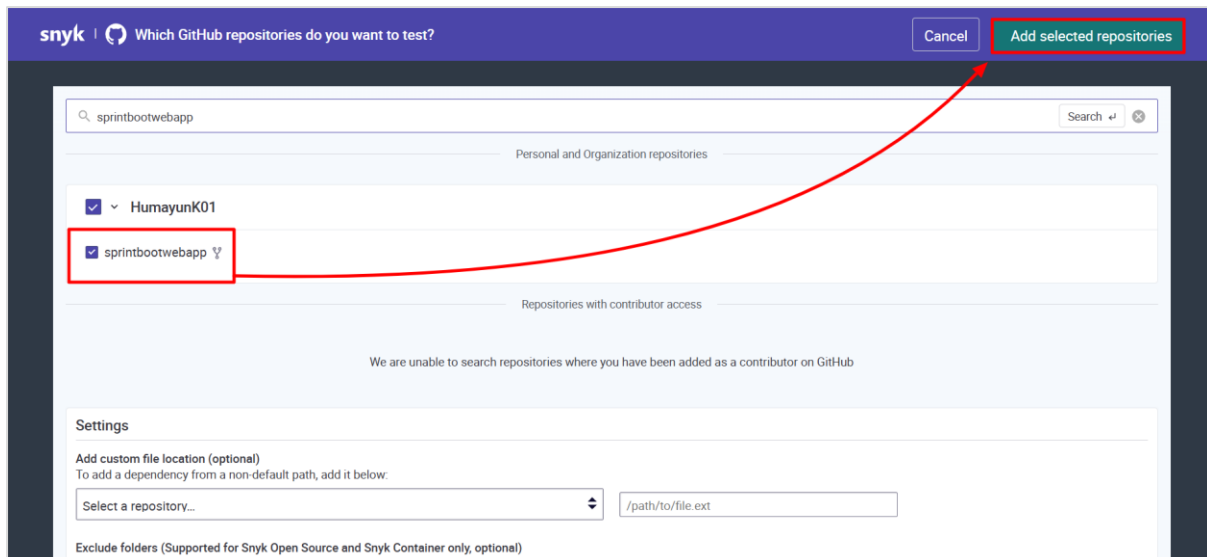
- Go to snyk.io, register, authenticate via GitHub/Google/email



Step 2: Repository Integration

- Add a GitHub project, grant access, configure settings (scan frequency, notifications)





Step 3: Scan Execution

- Snyk detects dependencies and scans automatically
- Generates vulnerability report, ranks by severity

The screenshot shows the Snyk Targets page. At the top, there's a search bar and a 'Targets' tab with a count of 1. Below, a table lists targets. The first target is 'Humayunk01/sprintbootwebapp' with a summary bar showing 10 Critical (C), 116 High (H), 76 Medium (M), and 17 Low (L) issues. A red box highlights the 'View report' button in the actions column. Below the table, there's a list of projects: 'pom.xml' and 'Code analysis', both imported and tested 3 minutes ago. A dropdown menu is open for the first target, showing options: 'Add projects', 'Retry last import', and 'Delete'.

Step 4: Report Analysis

- Review vulnerability details (CVE, CVSS, affected package, fix info)

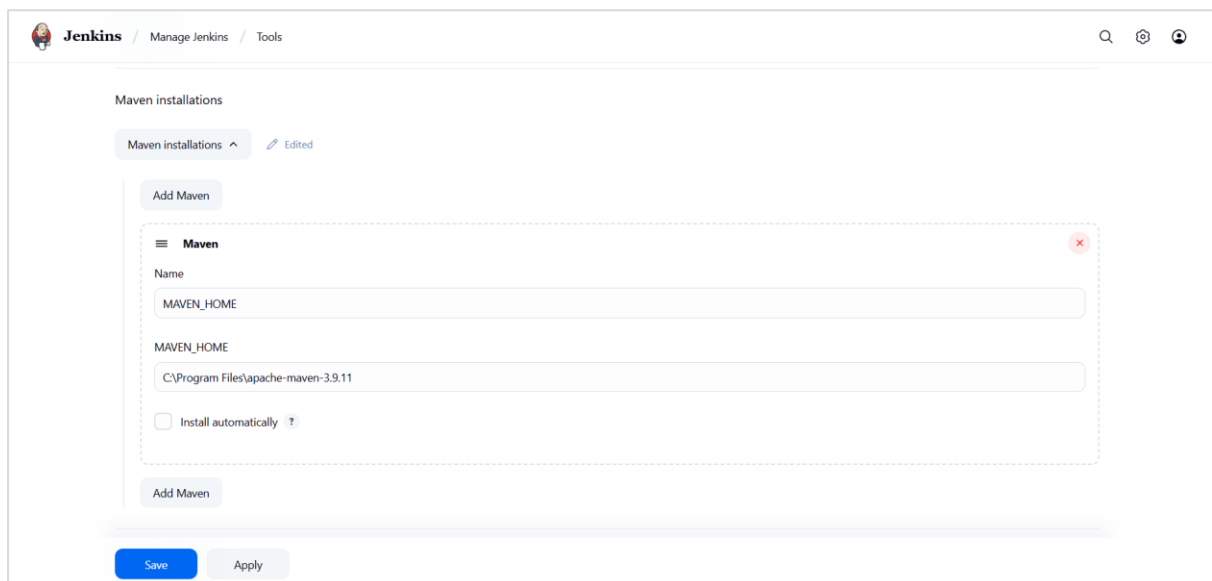
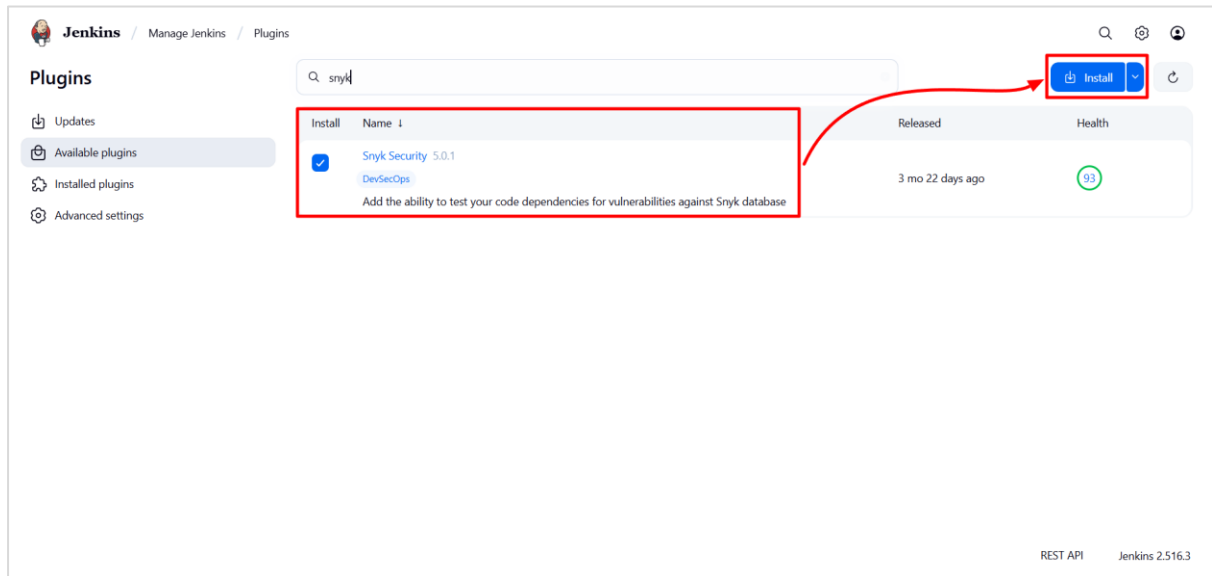
The screenshot shows the Snyk Issues Detail page. The left sidebar contains navigation links: Dashboard, Projects, Reports (selected), Dependencies, Integrations, Members, and Settings. The main content area has a 'Vulnerabilities Detail' tab selected. A red box highlights the summary section, which includes 'Total Issues' (0) and 'Unique Vulns' (0). Below this, there's a breakdown by severity: Critical (0), High (0), Medium (0), and Low (0). The 'Issue details' section shows 'No Data'. At the top, there's an 'Export to PDF' button and filters for 'Issue Status: Open' and 'Project Target: Humayunk01/sprintbootwebapp'.

The screenshot shows the Snyk Vulnerabilities Detail page. The left sidebar is the same as the previous screenshot. The main content area has a 'Vulnerabilities Detail' tab selected. A red box highlights the 'Vulnerabilities By' section, which includes a table with columns: 'UNIQUE VULNS', 'TOTAL ISSUES', 'VULNERABILITY', 'CONFIGURATION', and 'LICENSE'. The values are all 0. Below this, the 'Vulnerability Details' section shows 'No Data'. At the top, there's an 'Export to PDF' button and filters for 'Issue Status: Open' and 'Issue Type'.

METHOD 2: Jenkins + Snyk Plugin Integration

Step 1: Jenkins Setup

- Install plugins: Snyk Security, Maven, GitHub
- Configure Maven, JDK as global tools



Step 2: Snyk Configuration

- Add API token in Jenkins credentials
- Set Snyk installation

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

Jenkins

Manage Jenkins

Credentials

System

Global credentials (unrestricted)

New credentials

Kind

Snyk API token

Scope

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

Token

API Key

ID

Snyk-API-Token

Description

Create

REST API Jenkins 2.516.3

Jenkins

Manage Jenkins

Credentials

System

Global credentials (unrestricted)

Global credentials (unrestricted)

+ Add Credentials

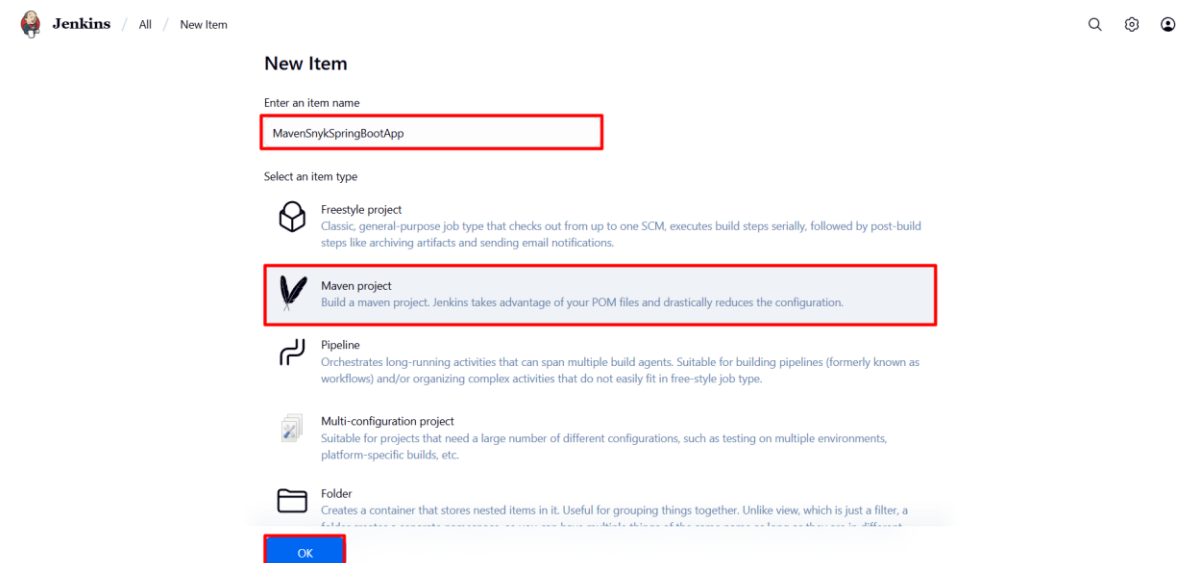
Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
tomcat-creds	humayun/***** (Tomcat Manager credentials)	Username with password	Tomcat Manager credentials
2e2019db-82a2-4176-81cf-eddb9e1e2fa8	HumayunK01/***** (GitHub Token)	Username with password	GitHub Token
snyk-api-token	snyk-api-token	Secret text	

Icon: S M L

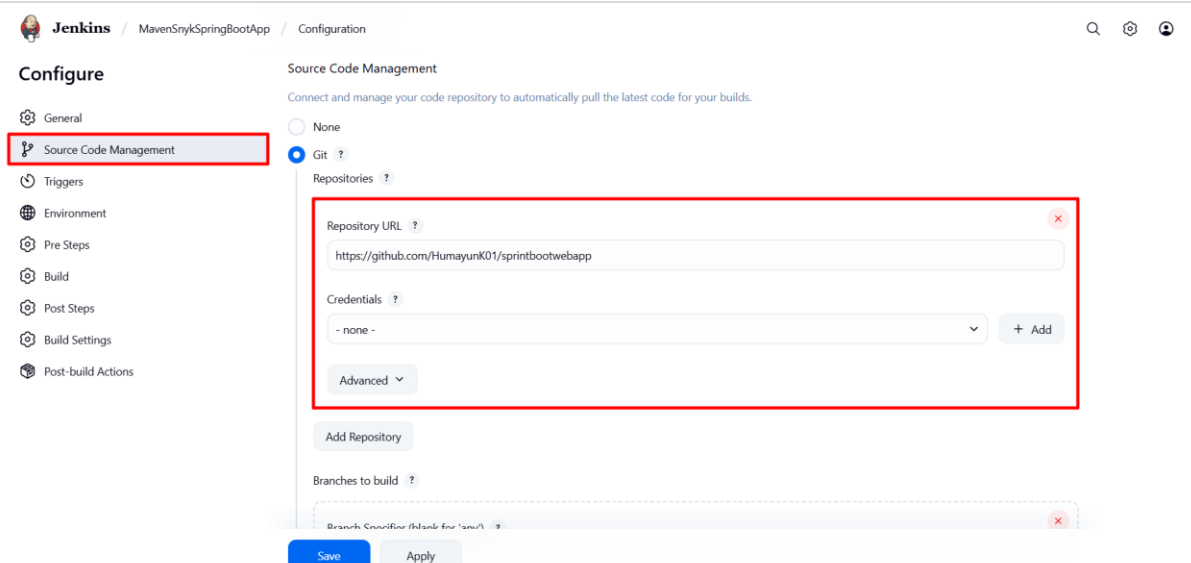
REST API Jenkins 2.516.3

Step 3: Pipeline Configuration



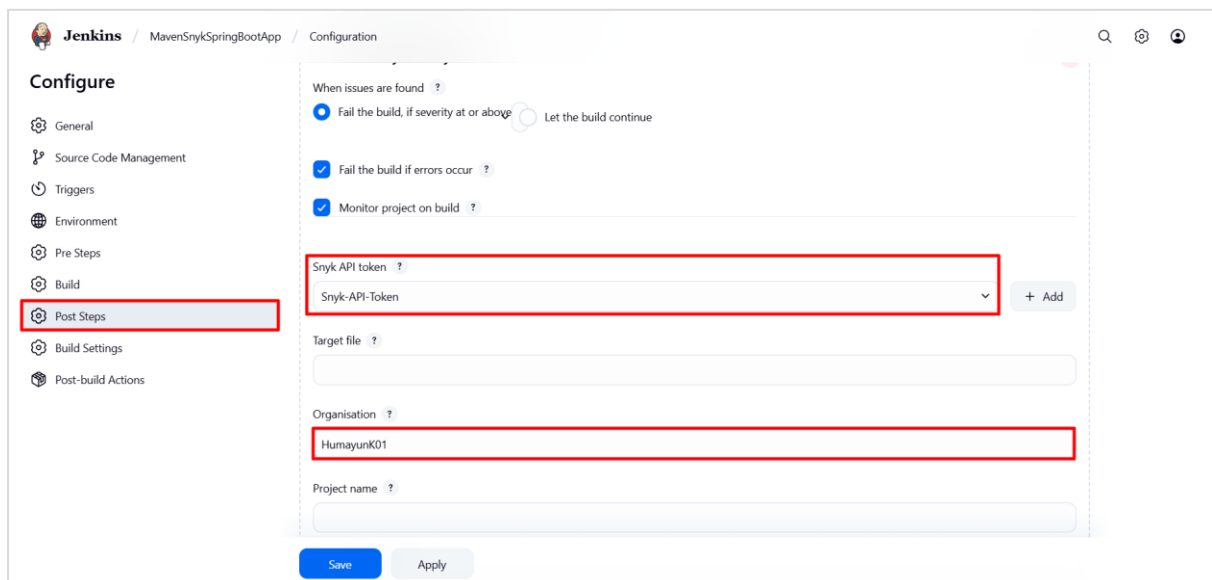
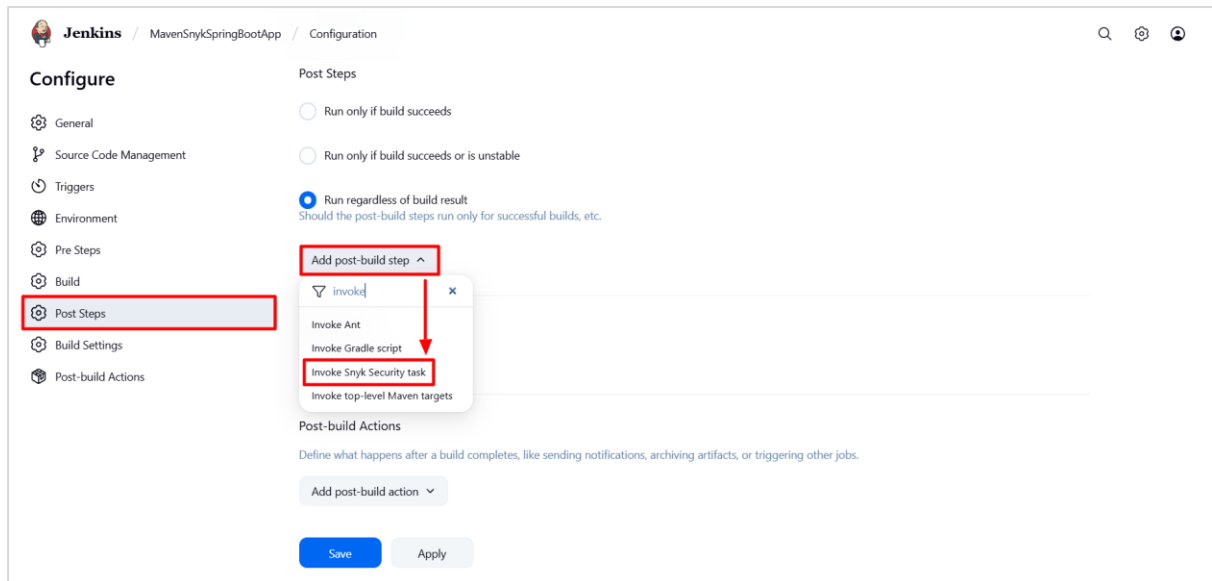
The screenshot shows the Jenkins 'New Item' configuration page. The breadcrumb navigation at the top reads 'Jenkins / All / New Item'. The main heading is 'New Item'. Below it, there is a text input field for 'Enter an item name' containing 'MavenSnykSpringBootApplication', which is highlighted with a red rectangle. Under the 'Select an item type' section, several options are listed: 'Freestyle project', 'Maven project' (highlighted with a red rectangle), 'Pipeline', 'Multi-configuration project', and 'Folder'. Each option has a brief description. At the bottom of the form, there is a blue 'OK' button, also highlighted with a red rectangle.

- Create Maven job, add Git repository



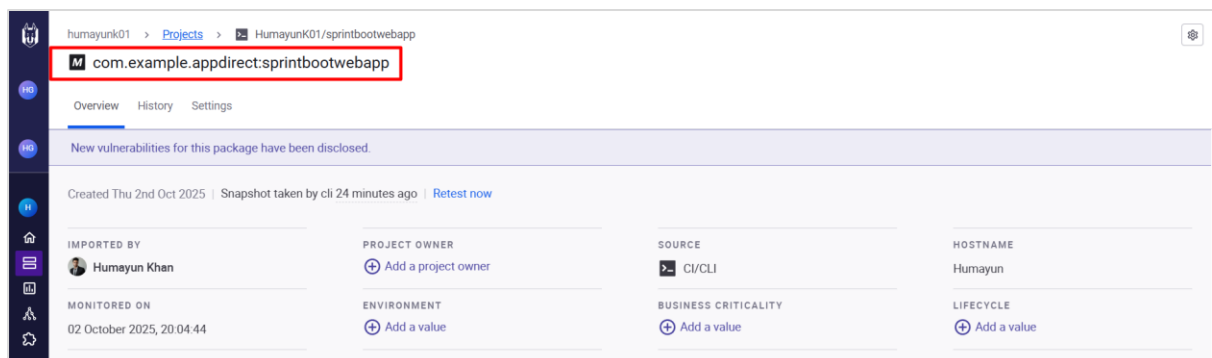
The screenshot shows the Jenkins 'Configuration' page for the 'MavenSnykSpringBootApplication' job. The breadcrumb navigation at the top reads 'Jenkins / MavenSnykSpringBootApplication / Configuration'. The main heading is 'Configure'. On the left sidebar, the 'Source Code Management' tab is selected and highlighted with a red rectangle. The main content area is titled 'Source Code Management' and includes the instruction 'Connect and manage your code repository to automatically pull the latest code for your builds.' Below this, there are radio buttons for 'None' and 'Git' (selected). Under the 'Git' section, there is a 'Repositories' section with a red rectangle highlighting the 'Repository URL' field (containing 'https://github.com/HumayunK01/springbootwebapp') and the 'Credentials' dropdown menu (set to '- none -'). There is also an 'Add Repository' button and a 'Branches to build' section at the bottom. At the very bottom, there are 'Save' and 'Apply' buttons.

- Set build goals (clean compile), add post-build 'Invoke Snyk Security Task' (target file: pom.xml, severity threshold: high, fail build on issues)



Step 4: Build & Reporting

- Manually or automatically trigger build
- Confirm Snyk scan, review console output
- Vulnerability report archived with build, synced with Snyk dashboard



humayunk01 > Projects > Humayunk01/sprintbootwebapp

com.example.appdirect:sprintbootwebapp

Overview History Settings

Reset Search...

217 of 217 issues

Group by none Sort by highest priority score

ISSUE TYPE

- ☐ Vulnerabilities 209
- ☐ License issues 8

SEVERITY

- ☒ Critical 10
- ☒ High 114
- ☒ Medium 76
- ☒ Low 17

PRIORITY SCORE

Scored between 0 - 1000

"FIXED IN" AVAILABLE

- ☐ Yes 207

org.springframework:spring-beans - Remote Code Execution (RCE) [🔗](#) **SCORE 919**

VULNERABILITY | CWE-94 | CVE-2022-22965 | CVSS 9.8 | **CRITICAL** | SNYK-JAVA-ORGSPRINGFRAMEWORK-2436751

Insights: Current public exploits for the Spring4Shell vulnerability require the following conditions:

- Built with Java Runtime Environment (JRE) version 9 or above
- Deployed on either Apache Tomcat, Payara or Glassfish
- Dependent on spring-webmvc or spring-webflux

If your application configuration applies to these conditions, we advise prioritizing remediation of this vulnerability. However, given it is technically possible for additional exploit conditions to exist we do recommend attempting upgrading to a fixed versions for all vulnerable instances.

Introduced through org.springframework.boot:spring-boot-starter-data-jpa@1.4.0.RELEASE

Fixed in org.springframework:spring-beans@5.2.20, @5.3.18

Exploit maturity **MATURE**

Show more detail

[Learn about this type of vulnerability](#)

CONCLUSION

Implementing SAST using Snyk and Jenkins within a CI/CD pipeline enabled early, automated detection of security vulnerabilities—finding 128 issues, including critical deserialization bugs.

Developers benefit from immediate actionable feedback, and security is now embedded across all stages of delivery, reducing costs and boosting the organization's security posture.

Key Outcomes:

- Early vulnerability detection reduces remediation costs & risks
- Seamless security integration for continuous delivery
- Actionable remediation, prioritization, and developer empowerment
- Continuous improvement through feedback and education

Future Enhancements:

- Integrate Dynamic Application Security Testing (DAST)
- Develop custom rules and advanced compliance dashboards
- Measure ongoing security improvement via KPIs