# **Devsecops-essentials-beginner-to-advanced-hands-on-demos**

Course Link: [**https://www.udemy.com/course/devsecops-essentials-beginner-to-advanced-hands-on-demos/learn/lecture/35474688?start=1#overview**](https://www.udemy.com/course/devsecops-essentials-beginner-to-advanced-hands-on-demos/learn/lecture/35474688?start=1#overview)

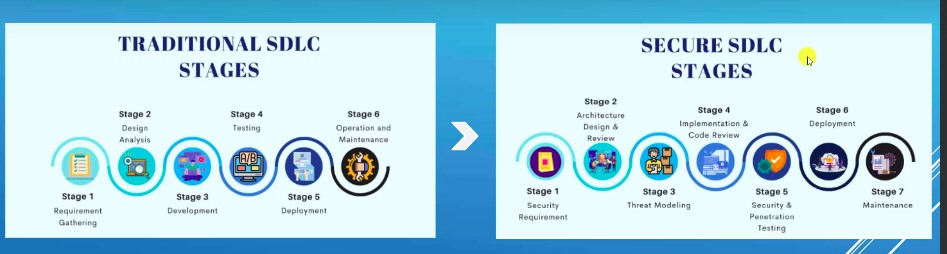
**DevSecOps**

is an approach in software development that focuses on integration of security practices in the devops lifecycle.

While DevOps focuses on the speed and efficiency of development and deployment, DevSecOps focuses on the security. It involves incorporating security practices and tools into the development process from the beginning.

This may include performing security testing, scanning code for vulnerabilities or implementing security controls as part of continuous integration and delivery process.

The goal of DevSecOps is to ensure that security is considered at every stage of the development lifecycle rather than being addressed as a separate concern.



As a DevSecOps engineer we will be involving in stage 4, where we will be doing SAST, DAST, SCA, etc.

### **Roles and Responsibilities:**

1. To perform Static Application Security Testing (SAST) using IDE Plugins e.g. **SonarLint**
2. To implement credential scanning tools like **Git Guardian** in CI/CD pipeline or at commit level e.g. **GIT Secrets** (This does not allow commits which contains hardcoded passwords)
3. Know to integrate **SAST tools like SonarQube, SonarCloud, Fortify Source Code Analysis, Veracode, Checkmarx** tools in CI/CDpipelines.
4. Know to integrate **SCA** (Software Composition Analysis Scan, used to scan for vulnerabilities in third party apps) **tools like OWASP Dependency Check, Snyk, BlackDuck** in CI/CD Pipeline.
5. Know to integrate **DAST** (Used to scan vulnerabilities in WebApps, APIs) **tools like OWASP ZAP, WebInspect, Veracode DAST, Burp** in CI/CD Pipeline.
6. Know to integrate **container scanning tools like Trivy, AQUA** in CI/CD Pipeline
7. Know to integrate **IAC scanning tools like Snyk, Bridgecrew** in CI/CD pipeline
8. Know to perform SAST, DAST and SCA scanning and perform False Positive Analysis on security vulnerabilities.
9. Report security vulnerabilities in tool like Jira and assign to dev team
10. Write YAML or any script for integrating tools in CI/CD pipelines. This is called Shift Left Approach.
11. To improve the DevSecOps maturity levels of overall Development Life Cycle.
12. To implement Cloud security controls and recommendations for compliance.

### **SAST**

Is a type of security testing which involves analyzing the source code of the software application to identify security vulnerabilities. It is typically performed during the development phase of an application, before it is deployed.

SAST tools are designed to analyse source code for known vulnerabilities and security weakness. They can also identify areas of the code that may be prone to security issues.

SAST tools typically operate by scanning the source code for patterns or specific vulnerabilities. The results of a SAST scan can be used to identify and fix vulnerabilities before an application is deployed, this helps to improve the overall security of the application.

Commercial SAST tools are checkmarx, Microfocus Fortify Source Code Analysis, Microfocus Fortify On Demand, Veracode, SonarQube, SonarCloud. All these tools can be integrated to CI/CD Platforms

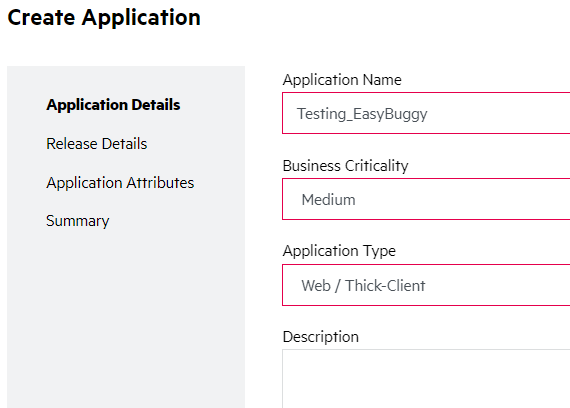
SAST Commercial IDE plugin are provided by Checkmarx, Veracode and Microfocus

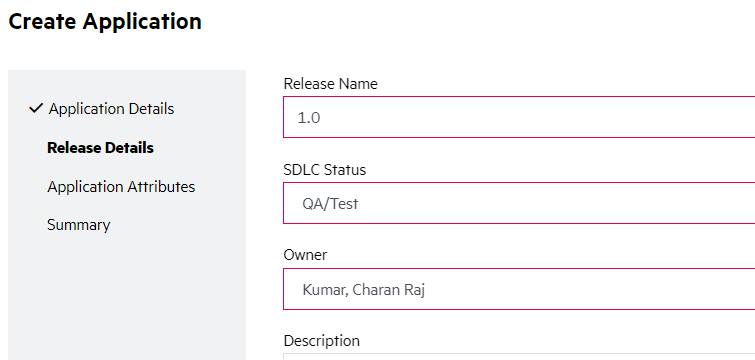
Open-Source SAST tools are SonarQube, SonarCloud and Snyk

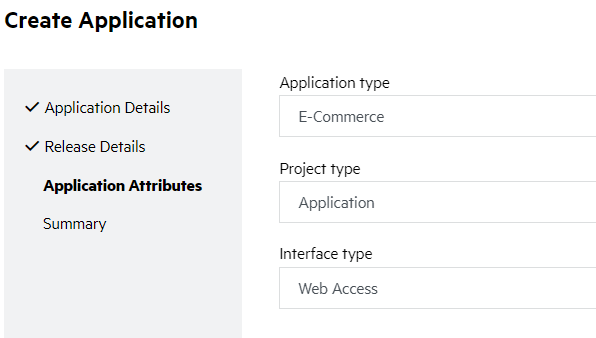
Free SAST IDE plugins are SonarLint, Snyk

### **SAST Demo with Fortify On-Demand**

In this use case we will be manually uploading the code in zip format to FOD and then scanning the code for vulnerabilities.

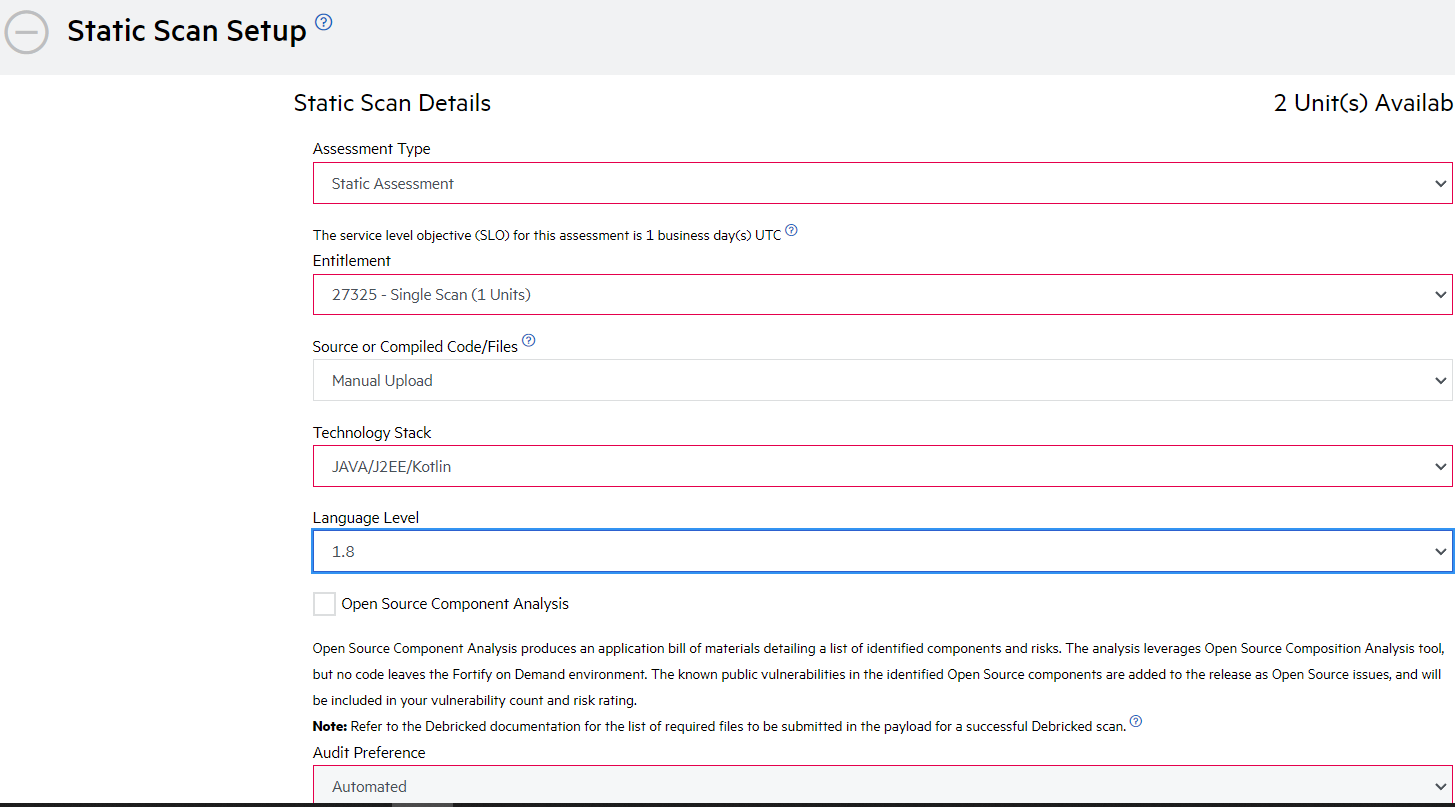




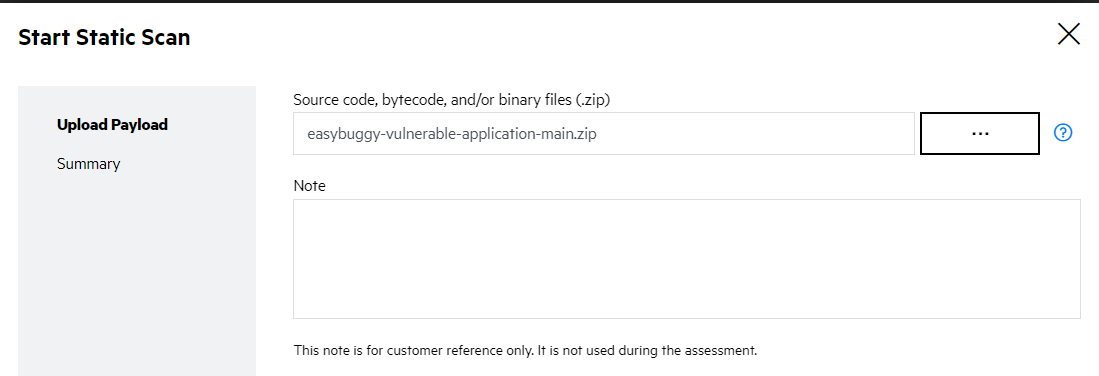


Start Manual Scan

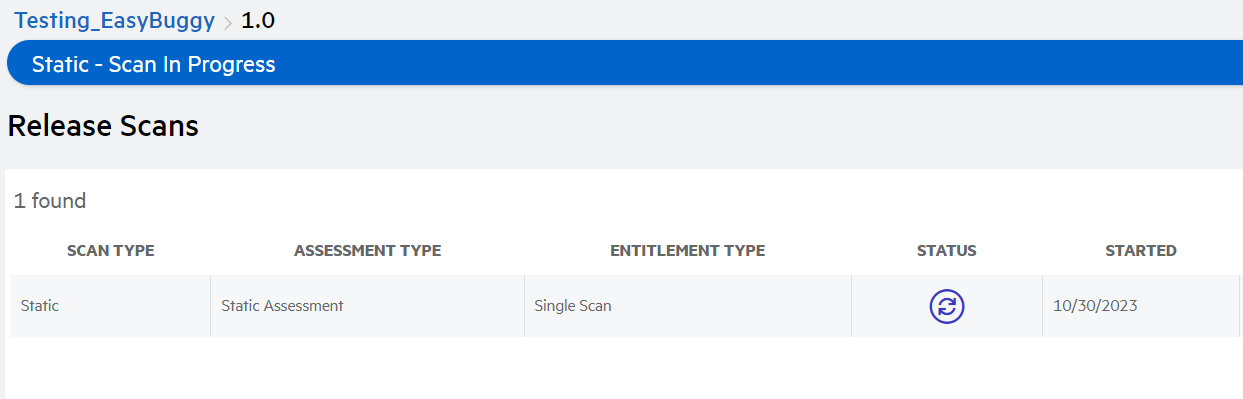
Choose the options based on the type of your requirement, click start scan.



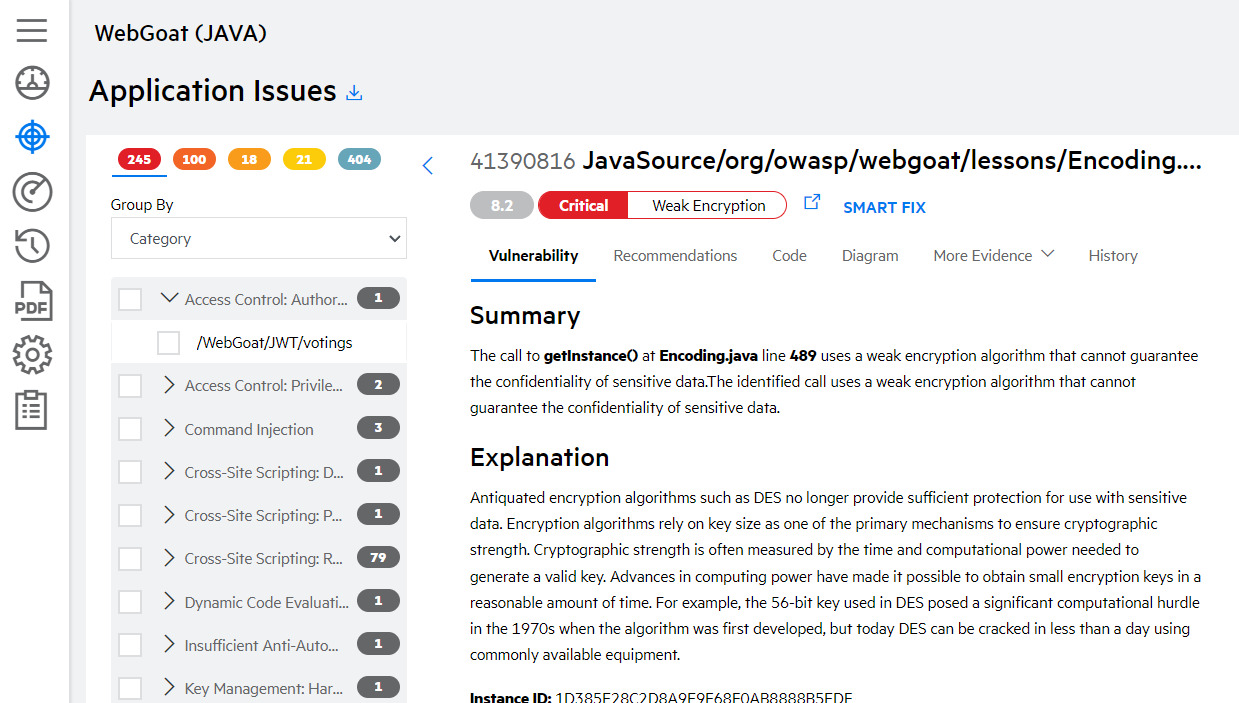
Upload your application and start the scan



Scan is in Progress



Once Scan is complete, you can browse through the issues and possible fixes as suggested



### **SBOM: Software Bill of Materials**

* It includes a list of all the software components and their versions that are included in a particular application or system.
* SBOM includes both the first-party components that was developed by the application vendor and also third-party components that was used to build the application.
* The purpose of SBOM is to provide a complete and accurate inventory of all the components that make up the application.
* SBOM is an important part of SCA and are increasingly being used in the SDLC to ensure the security and compliance of applications.
* SCA scan will use the list prepared by SBOM to identify and analyze the third-party libraries, frameworks and other components that are used in the application. The goal of SCA is to identify potential security vulnerabilities, licensing issues, and other risks associated with the use of them in the applications.

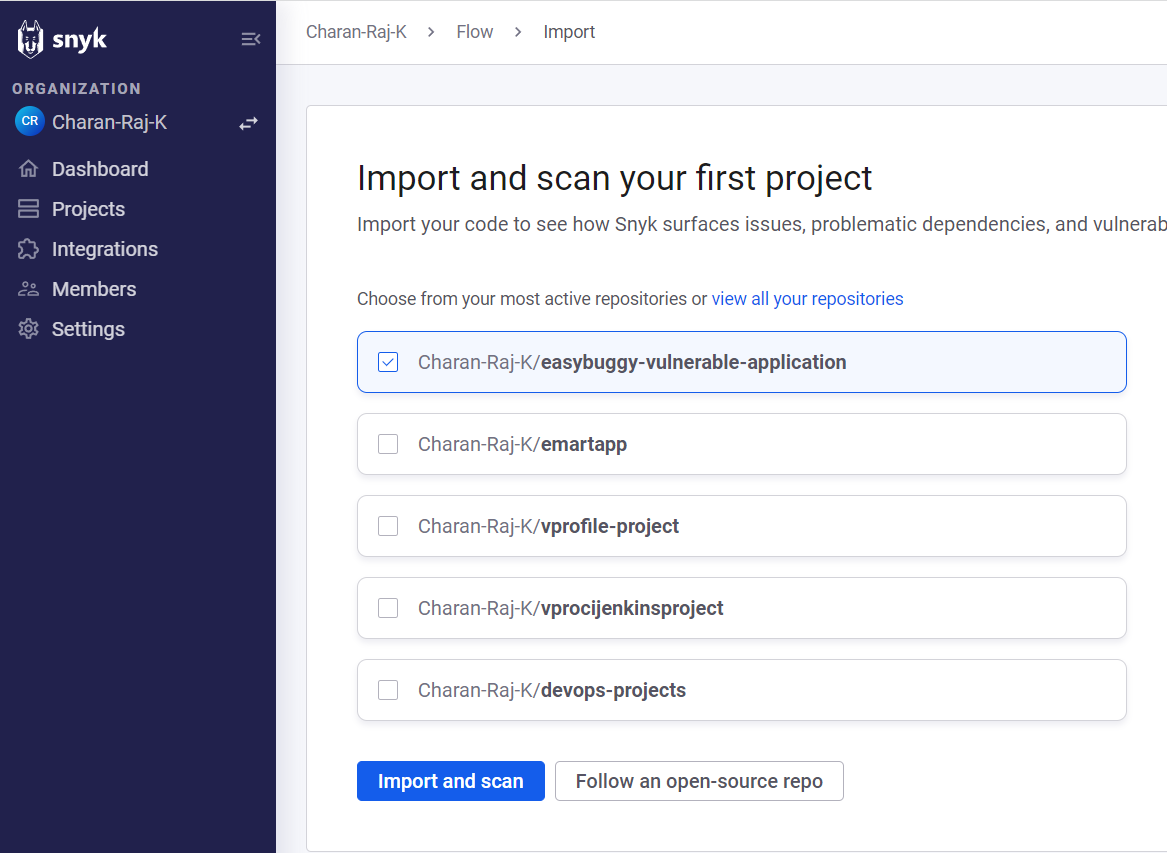
#### **SCA Tools:**

Commercial SCA tools are Snyk, Veracode SCA, BlackDuck. All these tools can be integrated with popular CI/CD platforms like Jenkins, GitLab, etc.

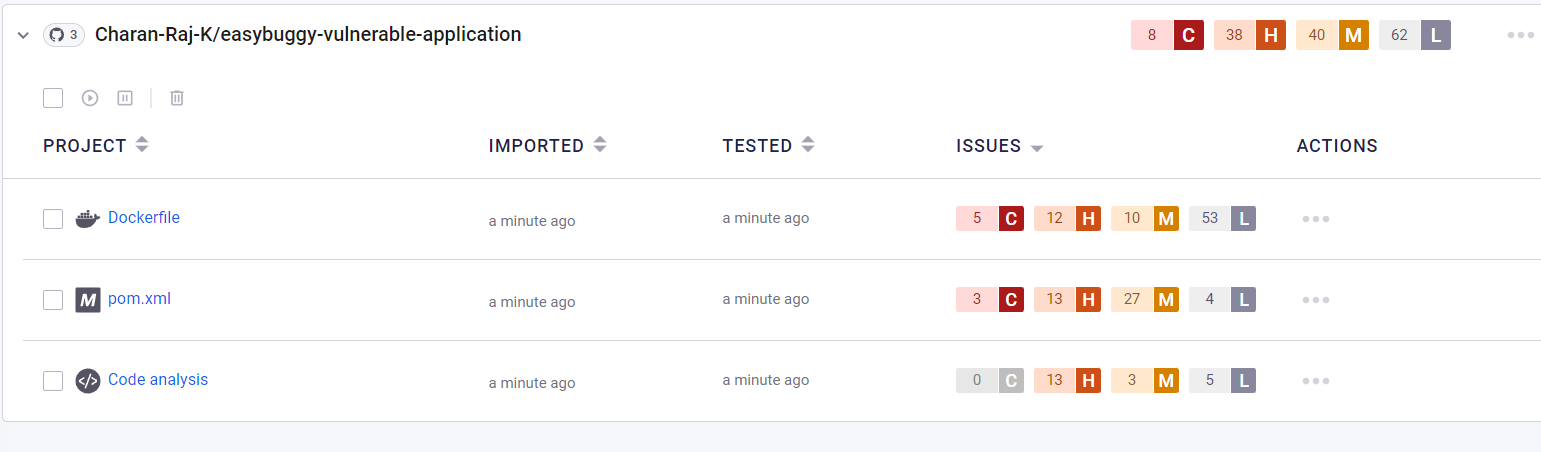
Open Source SCA tools are OWASP Dependency Check.

#### **SNYK Demo:**

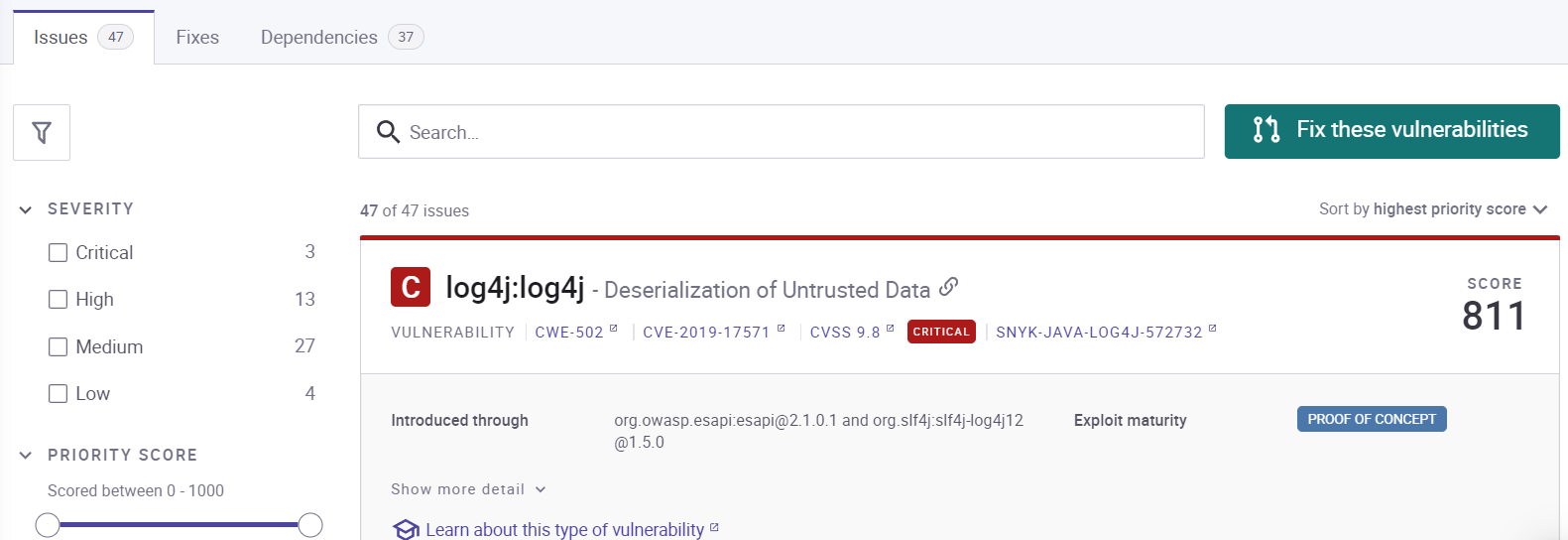
Import the project from GitHub Repo to be scanned and SCA test to be run



After the scan is complete, it presents the results with the issues. Click on the files to get more details on the issues.



In the below section, we can find the list of issues based on the type of severity and possible solutions to fix them.



### **DAST: Dynamic Application Security Testing**

* Is a method of testing in which we interact with the application, either via GUI or API via Swagger/Postman to find the vulnerabilities.
* It is typically performed after the application is developed and deployed as it helps in identifying the vulnerabilities that may have not been detected during development phase.
* It involves sending various inputs to the application and observing the response looking for the indications of vulnerabilities such as unhandled exceptions, error messages or any other unusual behaviors like cross site tokens, missing CSRF tokens or missing exceptions.
* It is important to note that DAST is not a substitute to SAST or SCA or Penetration Testing. It is often used in combination with these tools.
* Commercial DAST tools are Veracode DAST, WebInspect, these can be integrated with CI/CD platforms.
* Open Source tools are OWASP ZAP, Burp Community edition

Note: Read about basics of vulnerabilities like Cross site scripting, Owasp top 10, SQL injection

#### **DAST Demo: QWASP ZAP with hosted version**

<https://hostedscan.com/owasp-vulnerability-scan> go to this URL and enter <http://example.com> for handson

### **Container Security**

* Containers are a way of packaging and deploying application in a lightweight and portable format.
* They allow developers to package the application and dependencies together in a single package, making it easy to deploy and run the application on any machine that is compatible with container runtime.
* Commercial container Security tools are AQUA, Prisma Cloud, Snyk. These tools can be integrated to CI/CD pipeline.
* Open Source tools are SNYK community edition, Trivy.

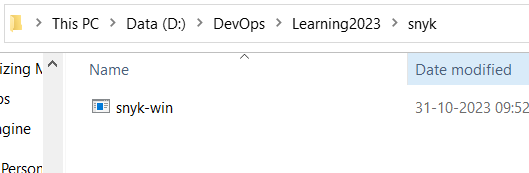
#### **SNYK Demo**

URL: <https://app.snyk.io/org/charan-raj-k>

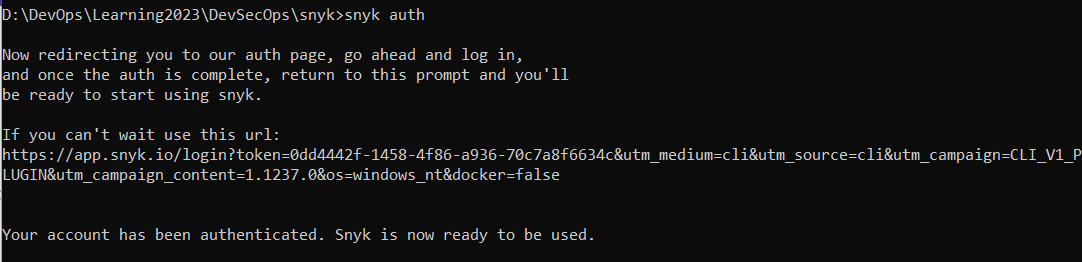
Download SNYK CLI https://github.com/snyk/cli/releases

SNYK will require docker runtime, hence ensure docker is installed in your environment.

Running SNKY in windows via cmd

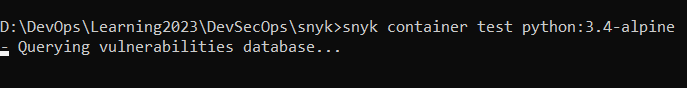


Rename the executable to snyk.exe and run snyk auth



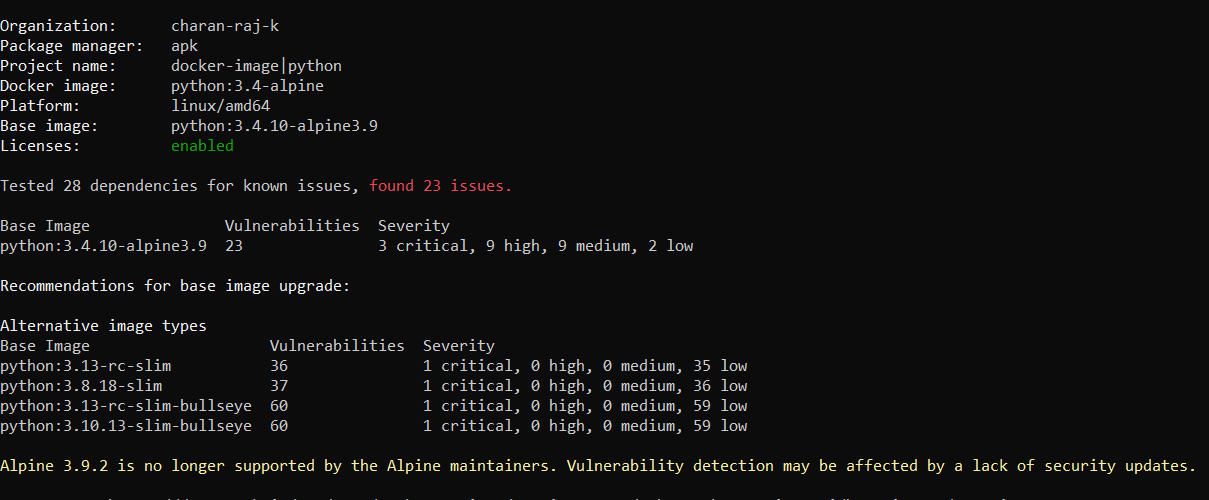
#### **SNYK Container Test**

Now run the test on any image



Scan Results:

SNKY will give list of low, medium, high and critical issues and recommended fix in the report.

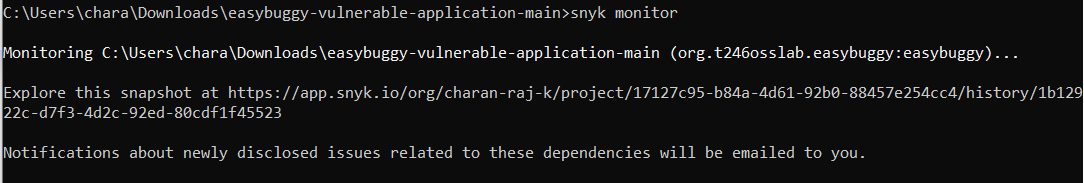


#### **SNYK Monitor**

The snyk monitor command creates a project in your Snyk account to be continuously monitored for open source vulnerabilities and license issues. After running this command, log in to the Snyk website and view your projects to see the monitor.

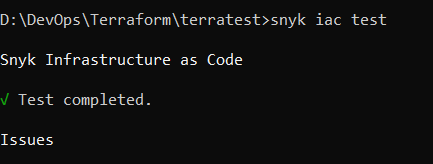
If you make changes to the project, you must run the monitor command again.

For SNYK monitor to work, you need to run it inside the project directory as scan runs on the pom or json or graddle files.



#### SNYK IAC

Run snyk iac test in the directory having tf configurations to scan them and get the results.



#### Exporting reports to a readable format.

https://docs.snyk.io/snyk-cli/scan-and-maintain-projects-using-the-cli/cli-tools/snyk-to-html

Install the snyk-to-html plugin

$npm install snyk-to-html -g

Export the report at runtime to html format

$ snyk container test charanrajkumar9/vprofiledb:latest --json | snyk-to-html -o results-container.html

Reports will be exported to the location where snyk executable is installed.

A screenshot of a computer

Description automatically generated



### **IAC Security and its Tools**

It is a process of analyzing IAC code and configurations to identify potential security vulnerabilities and misconfigurations.

Commercial IAC scan tools are Checkov from BridgeCrew, Snyk and Cloudspoilt. These tools can be integrated with CI/CD platform

Open-Source IAC tools is SNYK Community Edition.

#### **Checkov Tool**

Is a CLI based tool and run to identify the vulnerabilities.



##### [**Requirements**](https://github.com/bridgecrewio/checkov#requirements)

Python >= 3.7 (Data classes are available for Python 3.7+)

Terraform >= 0.12

##### Refer for installation

<https://github.com/bridgecrewio/checkov>

##### Execution

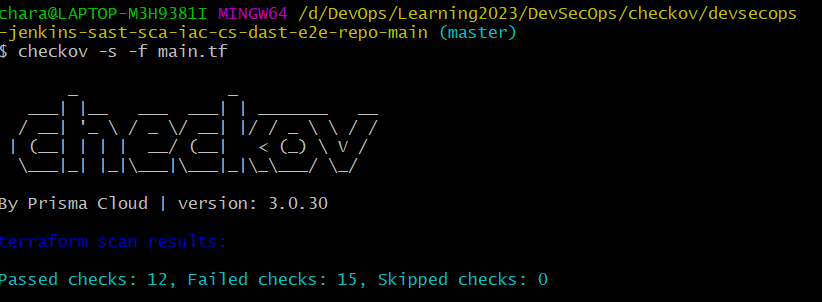
Run checkov from the directory where code is present using below command:

$checkov -s -f main.tf

(-s stand for soft fail and -f is used to pass filename to checkov)

-s flag is used to avoid any build failures in CICD pipelines if security misconfigurations are identified.

Note: Checkov will throw exit code 1 if there is misconfiguration in tf file and will fail the build, if -s used it will throw error code 0 and build will not fail.



Results



#### **SNYK Scan**

Scan the terraform files or the terraform plan files using snyk.

D:\DevOps\Learning2023\DevSecOps\snyk>snyk iac test resources.tf --report

A screenshot of a computer program

Description automatically generated

D:\DevOps\Learning2023\DevSecOps\snyk>snyk iac test tf-plan.json | snyk-to-html -d -o results-tf-plan.html

#### **Bridgecrew Tool**

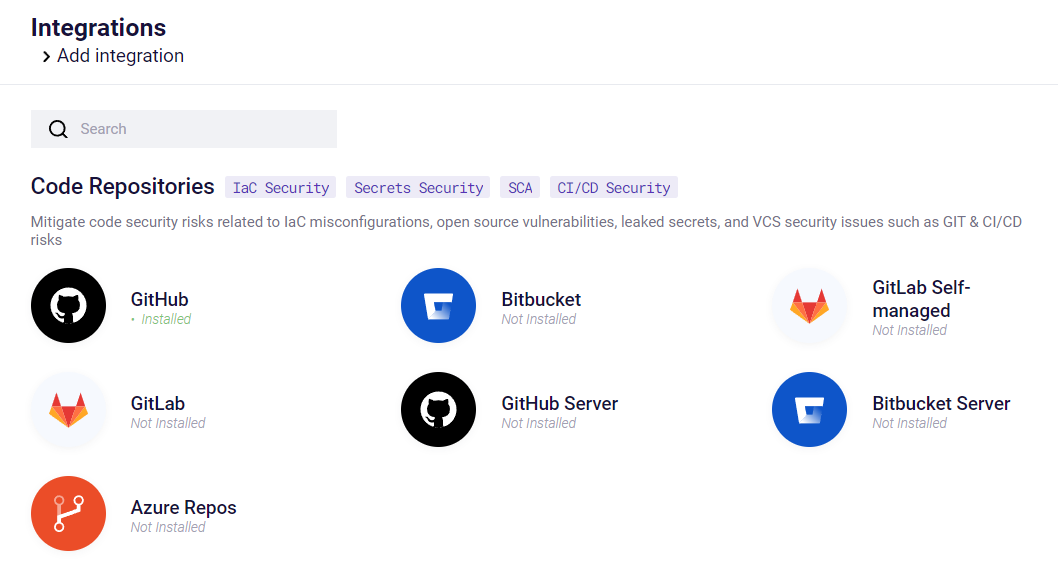
Is a SaaS tool that scans iac code for vulnerabilities, it’s a product from Prisma Cloud.

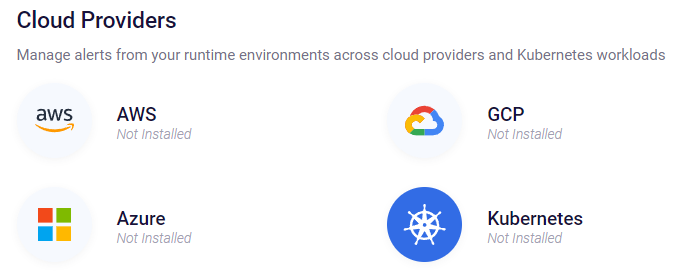
UI:

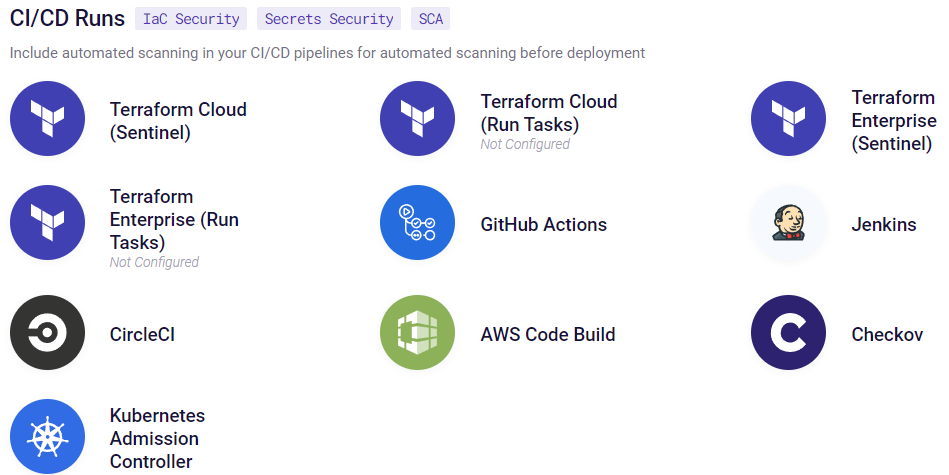


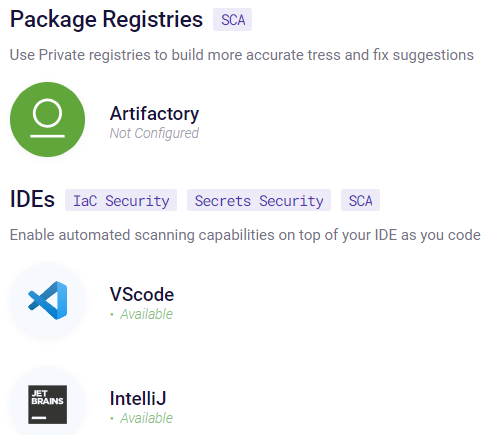
Integrations:

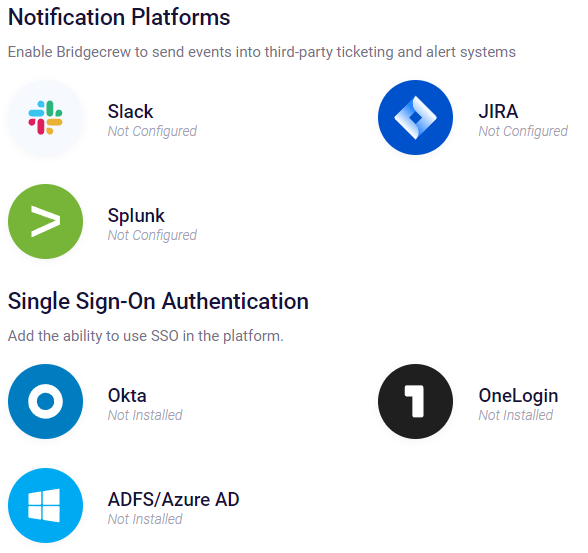
Code Repos







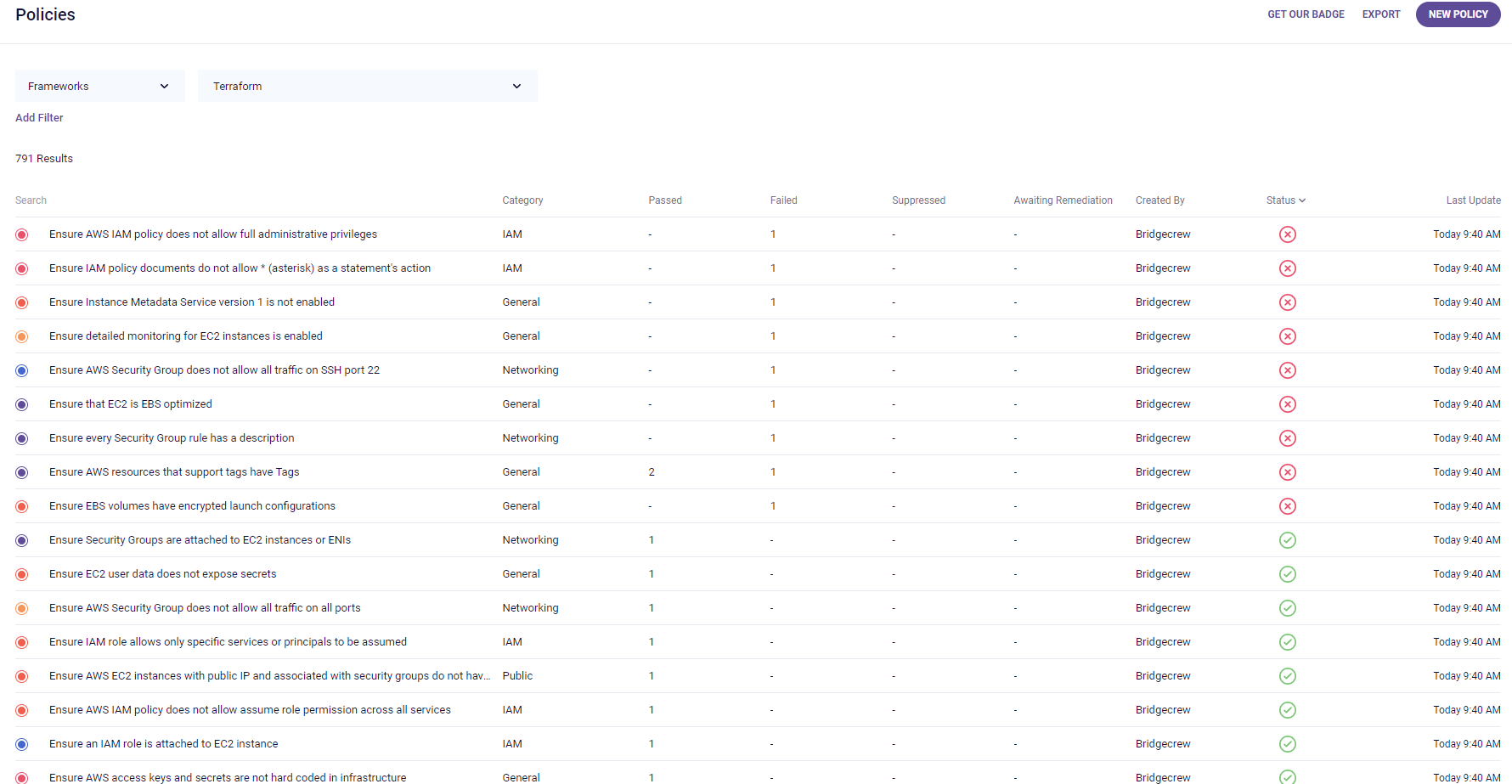




Scan Dashboard



This dashboard displays the list of benchmark policies used to scan and the number of pass and failed cases.

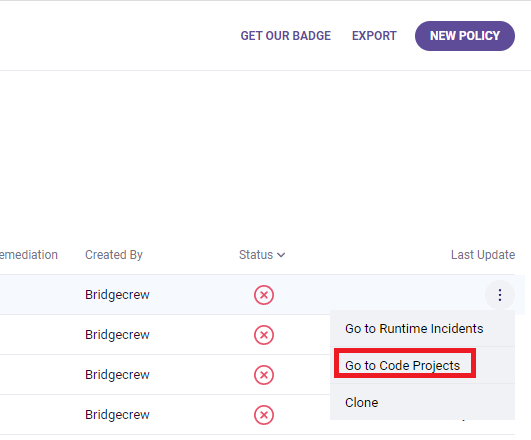


Click on any vulnerability shows detailed information as below,

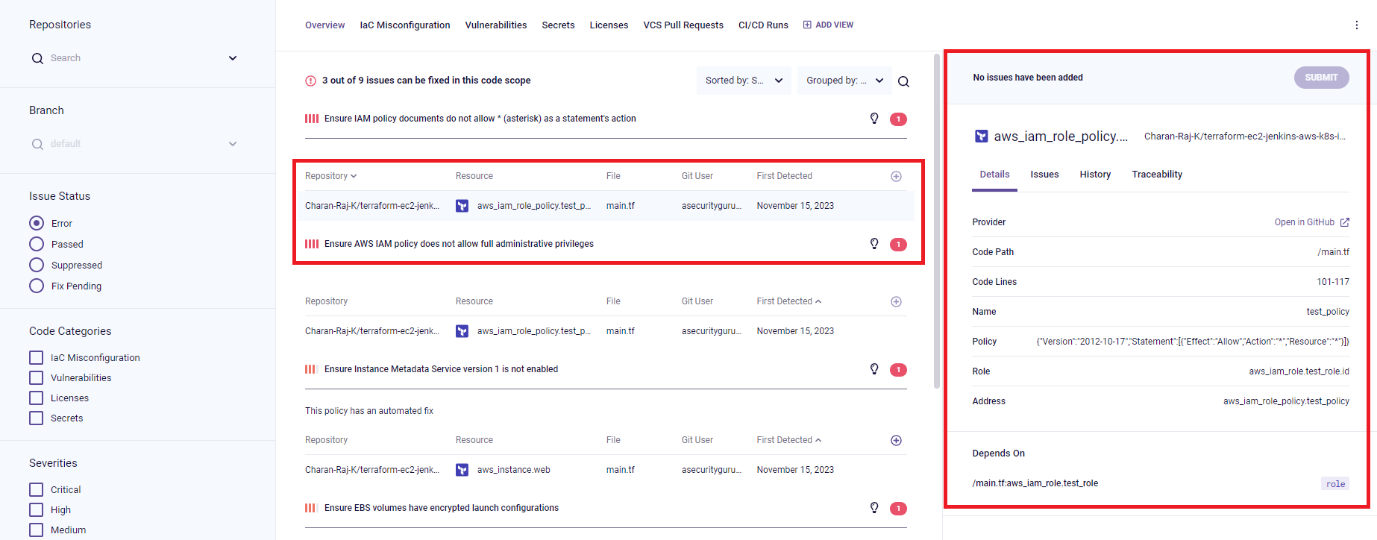


The guideline link will have possible suggestions to fix the vulnerability.

Click on the kebab menu and click on Go to Code Projects to see the code which has issues reported.



From the below UI we can see where exactly the issue is reported in the code and also what fix we can apply is also suggested by the tool.



### **CWE, CVE and CWS**

**CWE (Common Weakness Enumeration)** is a list of common software weaknesses that could lead to security vulnerabilities.

It is a standard way of identifying and describing these weaknesses so that they can be easily understood and addressed.

CWE list is maintained by MITRE corporation.

**CVE (Common Vulnerabilities and Exposures)** is a database of publicly disclosed cybersecurity vulnerabilities maintained by MITRE.

Each CVE entry includes a unique identifier, a description of the vulnerability and information about the affected software or hardware.

**CVSS (Common Vulnerability Scoring System)** is a standardized method for evaluating the severity of vulnerabilities.

It is designed to provide a consistent way of measuring the risk posed by a particular vulnerability, regardless of the specific software or hardware affected.

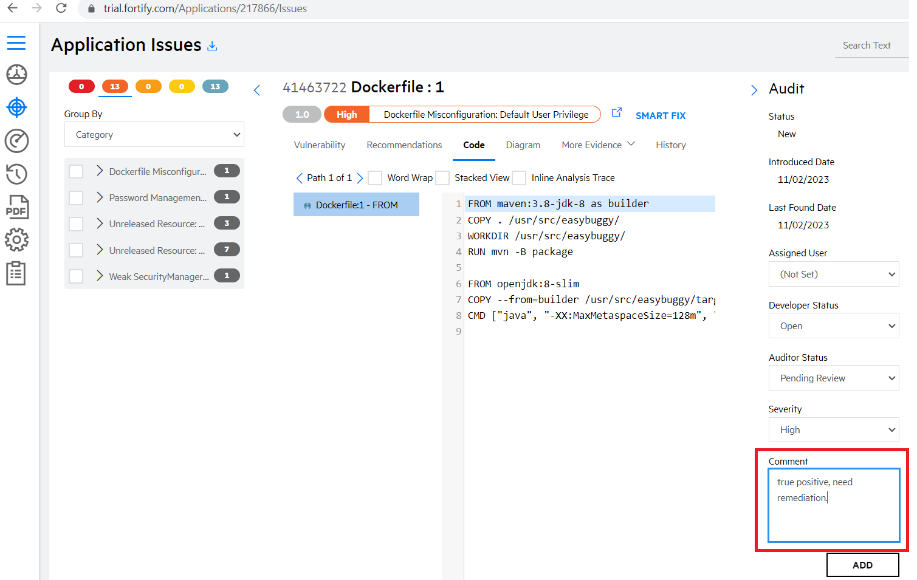
### **False Positive Analysis**

Is the process of identifying and analyzing false positives in a system or process.

A false positive, also known as false alarm is an error that occurs when a system or process wrongly identifies something as a threat or problem when it is not.

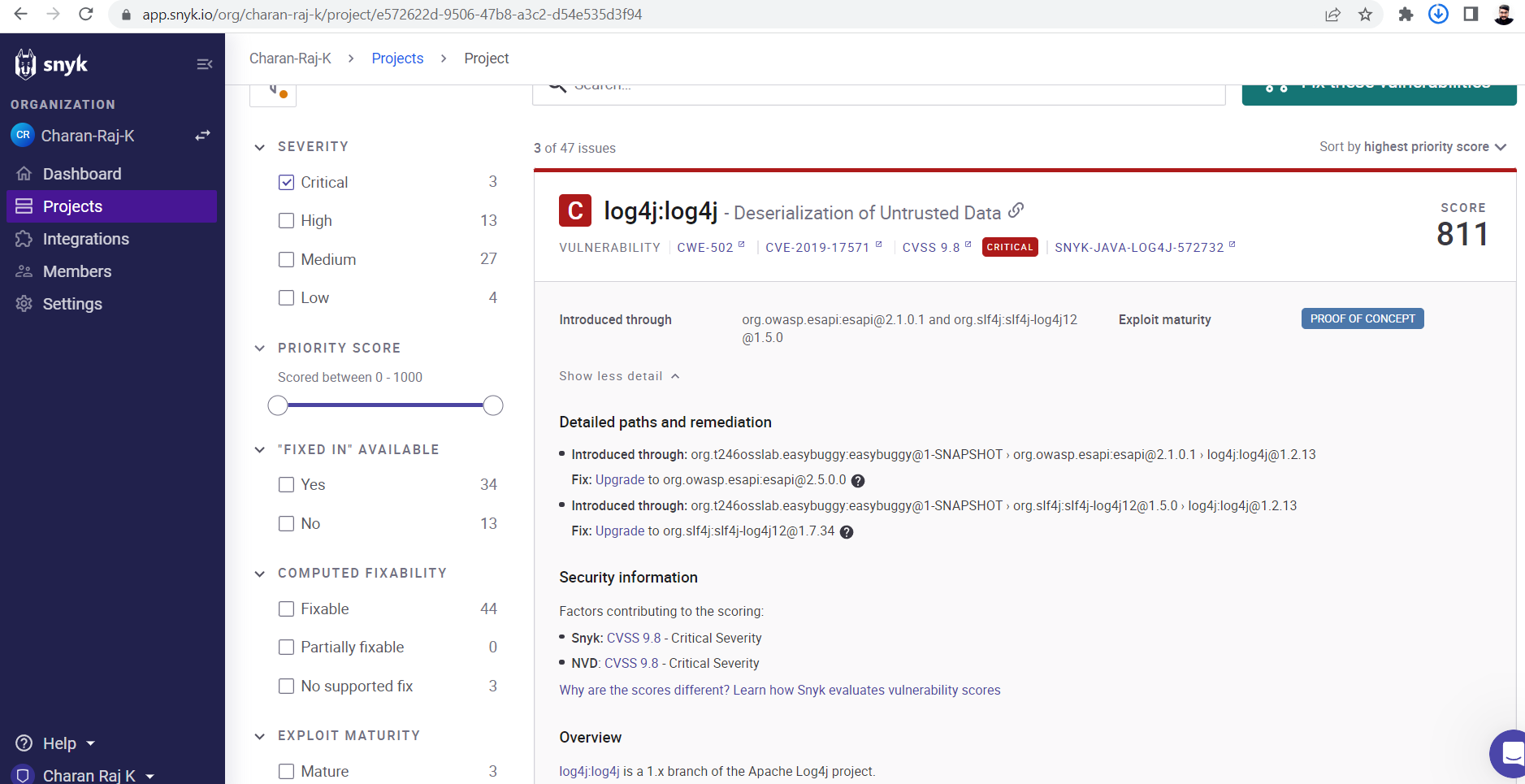
In context to cybersecurity, a false positive refers to instances where a security system or process wrongly identifies a code being vulnerable. For instance, Sonarqube says that CSRF token is not implemented in the system but the toke is implemented in another name would lead to False Positive event generated by it.

#### **Demo for SAST, SCA, DAST, IAC and Container Security**

****

FPA for SAST is done by manually going through the reported vulnerability and checking if they are true positive.

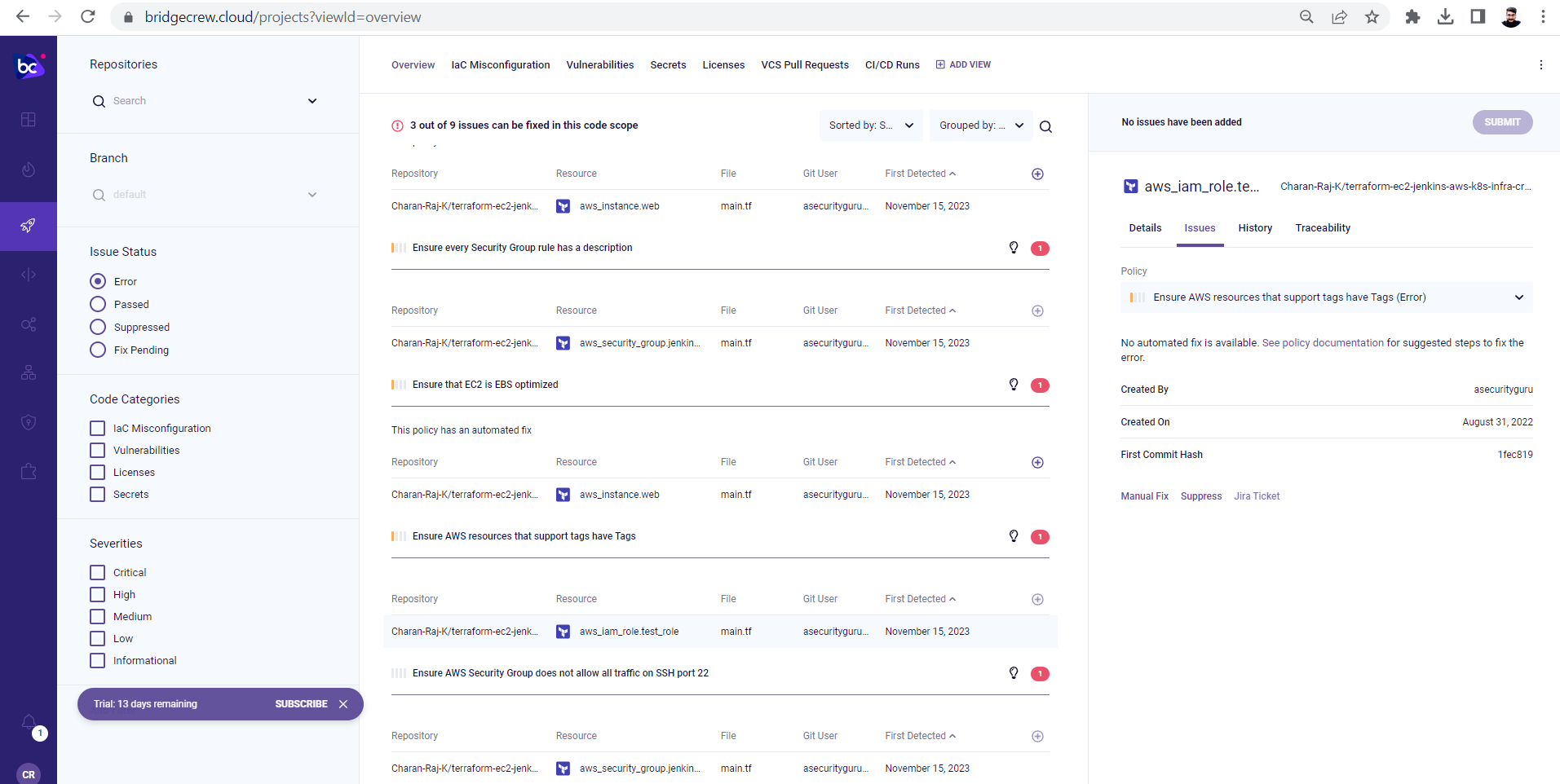
FPA for SCA



In case of SCA we can provide the fix recommendation to the developer as suggested in the tool and get the vulnerability fixed.

FPA for IAC

It is done on bridgecrew,



## DevSecOps Maturity Model (DMM)

* Is used to define and measure the maturity of a secure sdlc process, a level is assigned to your sdlc process.
* There are 5 stages in maturity model which is assigned based on the maturity of the processes.
* The DMM. Is a framework that helps organizations to understand and improve their DevSecOps practices.
* It defined a set of best practices and capabilities that organizations can strive to achieve in order to improve the security and quality of their software development and delivery process.
* There are several versions of DevSecOps Maturoty Model which can be categorized into stages as below:
  + **Ad-Hoc**: in this initial stage, security is an afterthought and is not integrated into the software development process.
  + **Repeatable:** In this stage, security practices are established but they are still separate from the software development process. (when the dev team and security team are not working together)
  + **Defined:** In this stage, security practices are integrated into the sdlc process also the standard procedures and policies related to security are defined. (When the security practices are implemented along with coding phase)
  + **Managed:** In this stage, security practices are continuously monitored and optimized to improve the efficiency and effectiveness. (all the security reports are reviewed and actions are taken)
  + **Optimizing:** In this Final stage, security practices are continuously improved and adapted to meet the evolving needs of an organization. (Suppose if the tool used for scanning is not yielding expected results, we have to look for another optimized tool to fulfill our requirements, hence optimizing the process)
* OWASP DevSecOps Maturity Model (DSOMM), a standard and model defined by OWASP.
  + <https://owasp.org/www-project-devsecops-maturity-model/>
  + <https://dsomm.owasp.org/> (Different stages of maturity model at different stages of SDLC)

## SAST Demo with SonarQube Docker Instance

Vagrant Execution Folder: D:\DevOps\Learning2023\DevSecOps\sonar-sast-demo

Repo: <https://github.com/Charan-Raj-K/devsecops-jenkins-k8s-tf-sast-sca-sonarcloud-snyk-repo>

SonarDashboard: <http://192.168.56.31:9000/dashboard>

1) Download or clone Vulnerable code to local system (https://github.com/asecurityguru/devsecops-jenkins-k8s-tf-sast-sca-sonarcloud-snyk-repo)

2) Make sure Java11 is installed (https://www.oracle.com/in/java/technologies/javase/jdk11-archive-downloads.html)

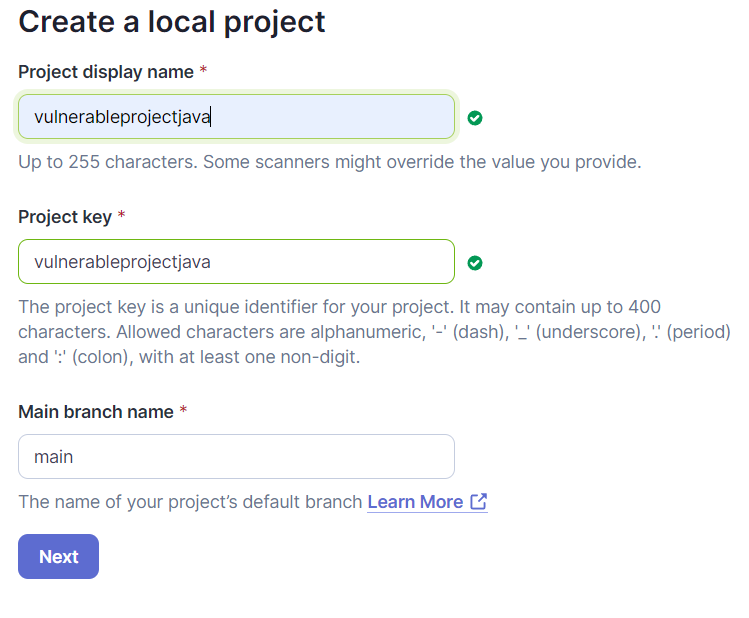
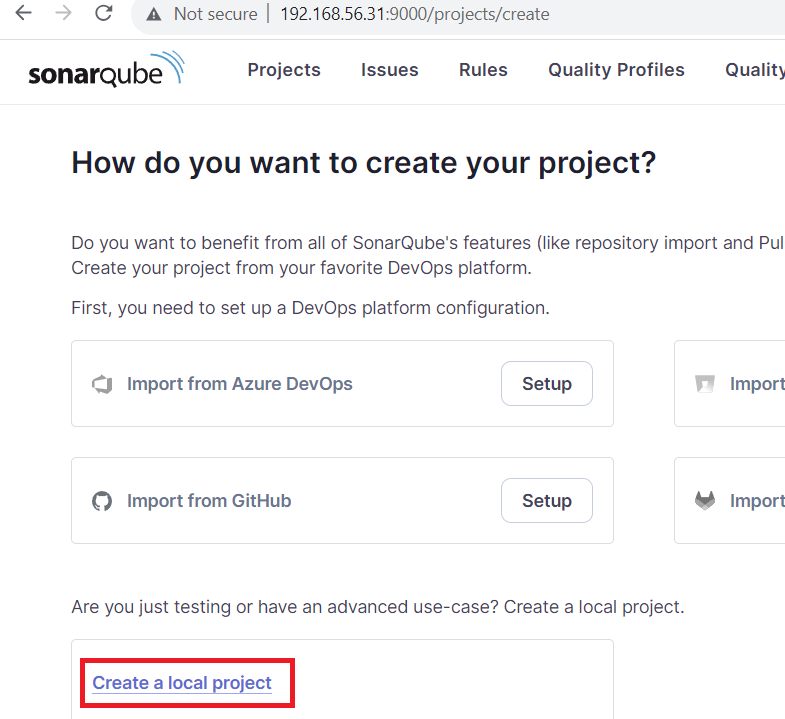
3) Make sure maven is installed and environment variable is set (https://maven.apache.org/download.cgi)

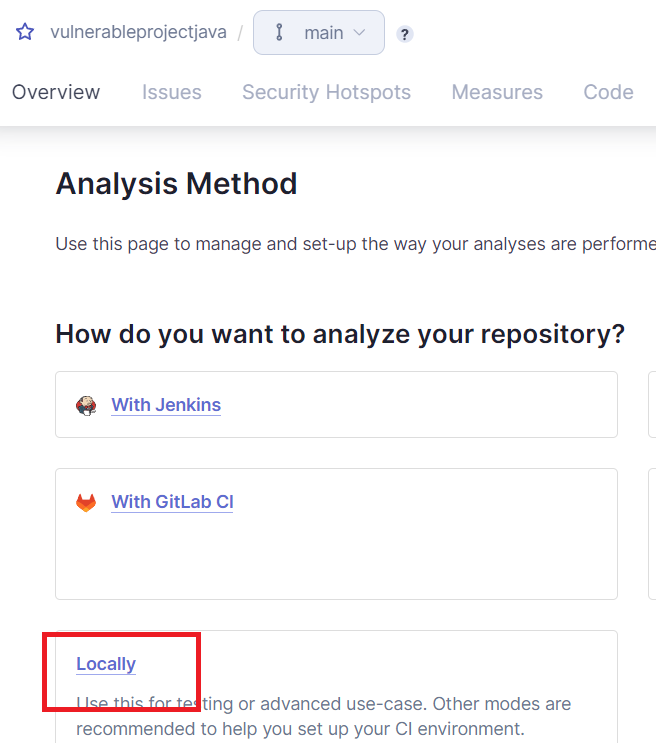
4) Install Docker

5) Download and run SonarQube image in Docker using commands:

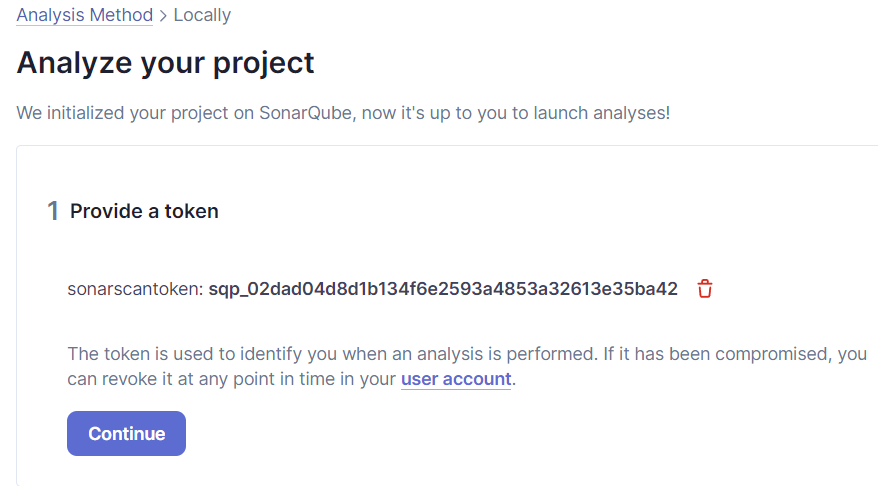
docker run -d --name sonarqube -p 9000:9000 -p 9092:9092 sonarqube

6) Create a project in SonarQube:

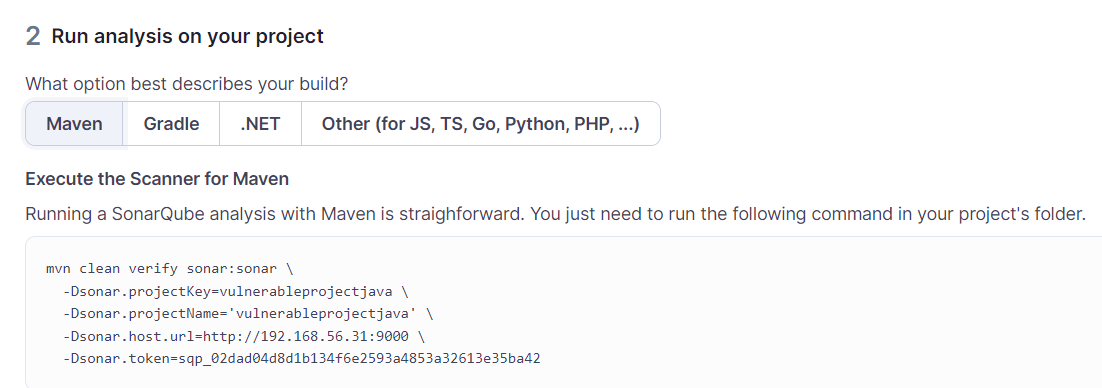




Generate Token

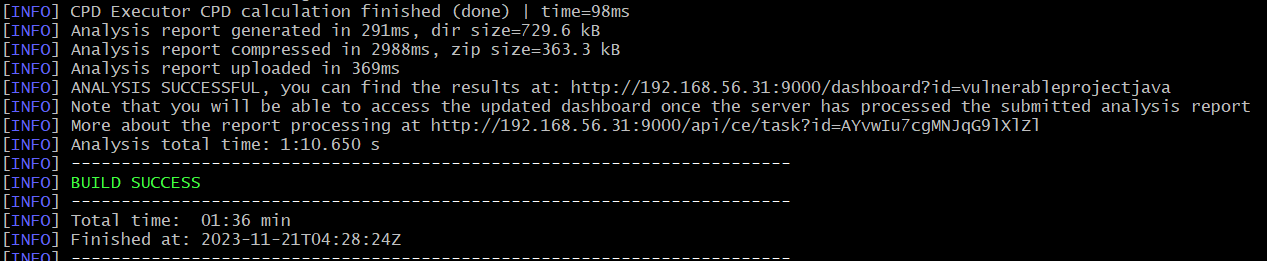


Copy the Maven command and run in your project folder in local to scan

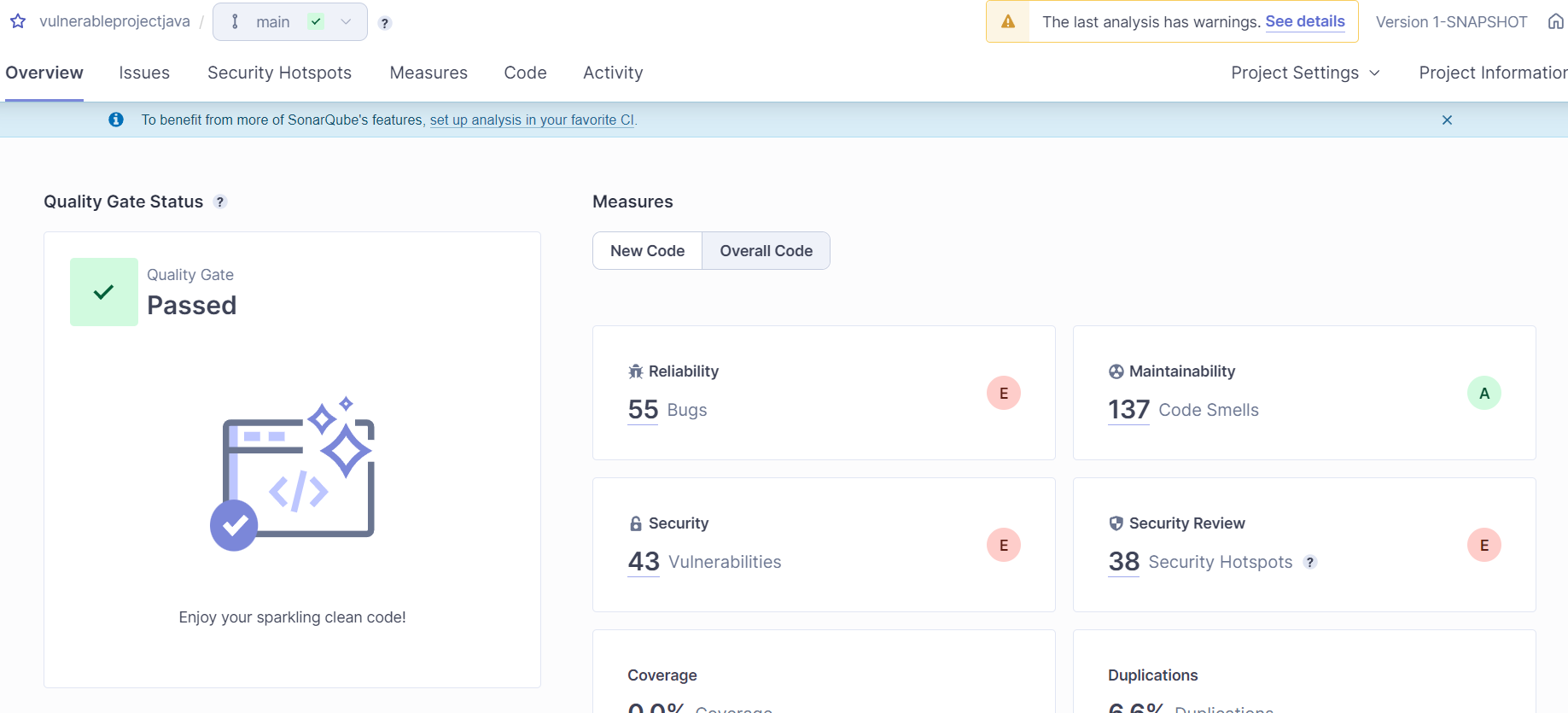




Build output



Scan results dashboard



## DevSecOps Pipeline

This is an end-to-end devsecops pipeline where one tool will be intergrated for each type of scan, one tool for SAST(SonarQube), one tool for SCA(snyk), one tool for IAC(checkov), one for container(snyk)



If Facing docker login issue, do below

# sudo chmod 666 /var/run/docker.sock

# sudo usermod -aG docker ${USER}

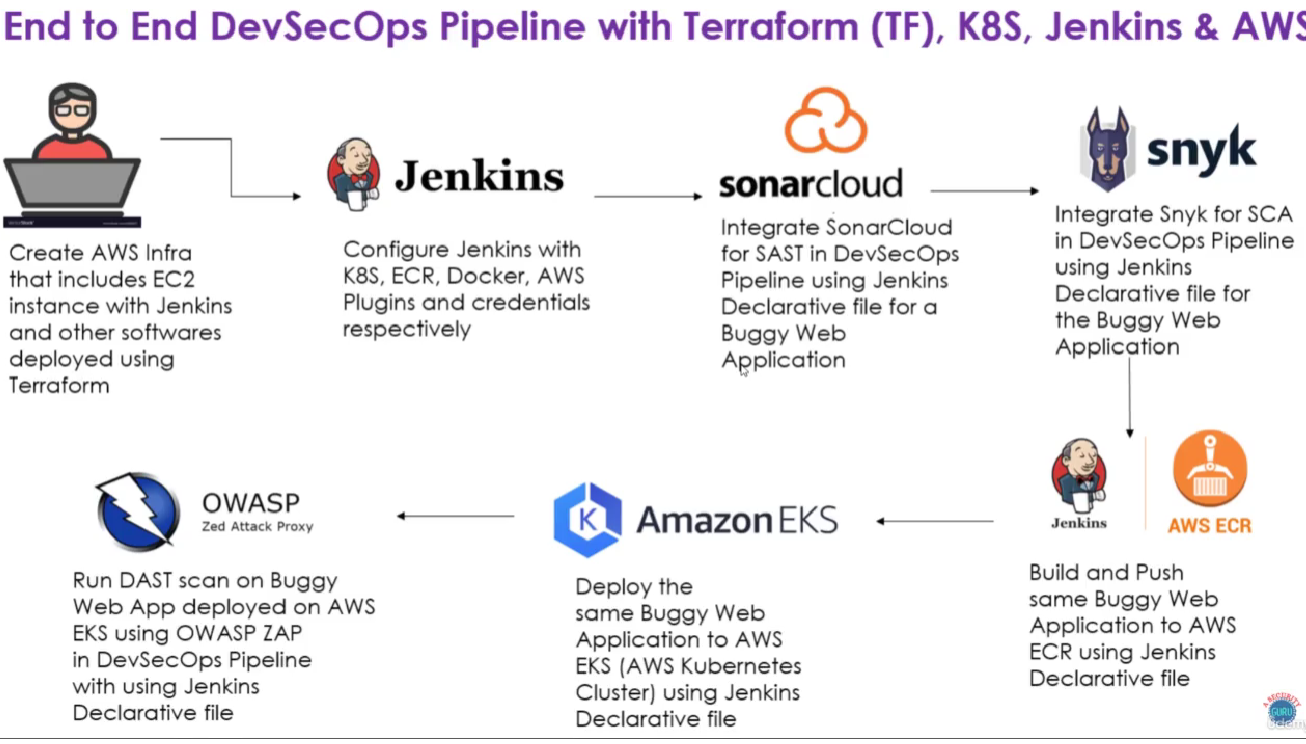
# sudo chmod 666 /var/run/docker.sock

Course link: <https://www.udemy.com/course/devsecops-essentials-beginner-to-advanced-hands-on-demos/learn/lecture/35472624#questions/20879762>

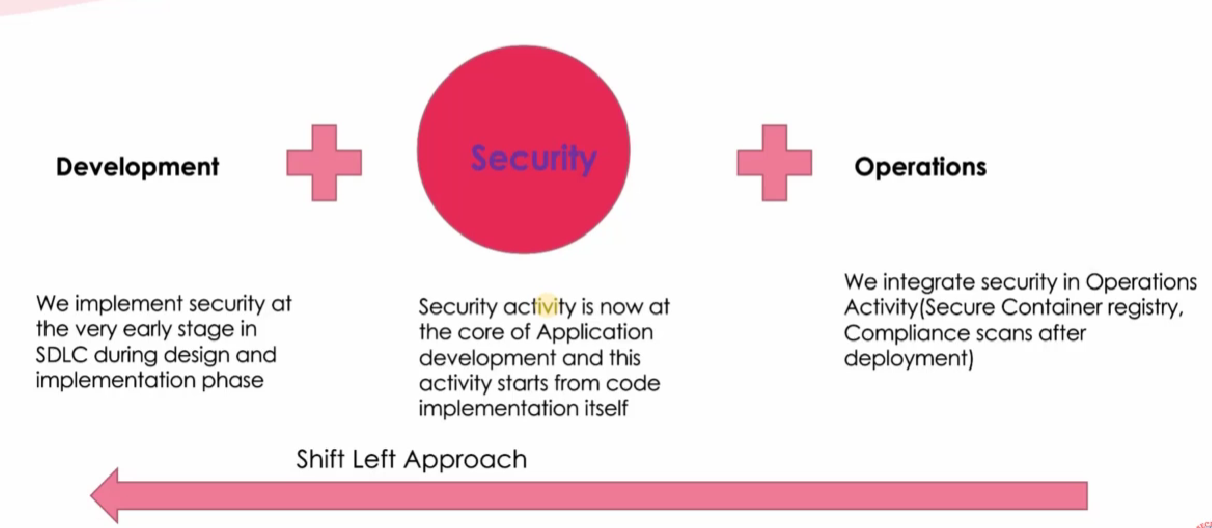
# Devsecops with Terraform, Kubernetes, Jenkins, Aws

Course Link: <https://www.udemy.com/course/devsecops-with-terraform-kubernetes-jenkins-aws/learn/lecture/34014686?start=0#overview>

## Pipeline Workflow



## What is DevSecOps



Shift-Left-Approach is a process where security scans are implemented at the early stage of development.

## Tools Used in implementing DevSecOps

### Development Phase:

1. Git Secrets: Is used to check if any security credentials or personal tokens are committed in the source code, it scans through the source code and will give warnings if any such things are present in the code which can be removed to make the code secure.
2. Security Plugins in IDE: Many companies like Fortify, Veracode, Snyk, SonarQube has release plugins which can be integrated to scan the code and mitigate the security issues at the development phase itself.

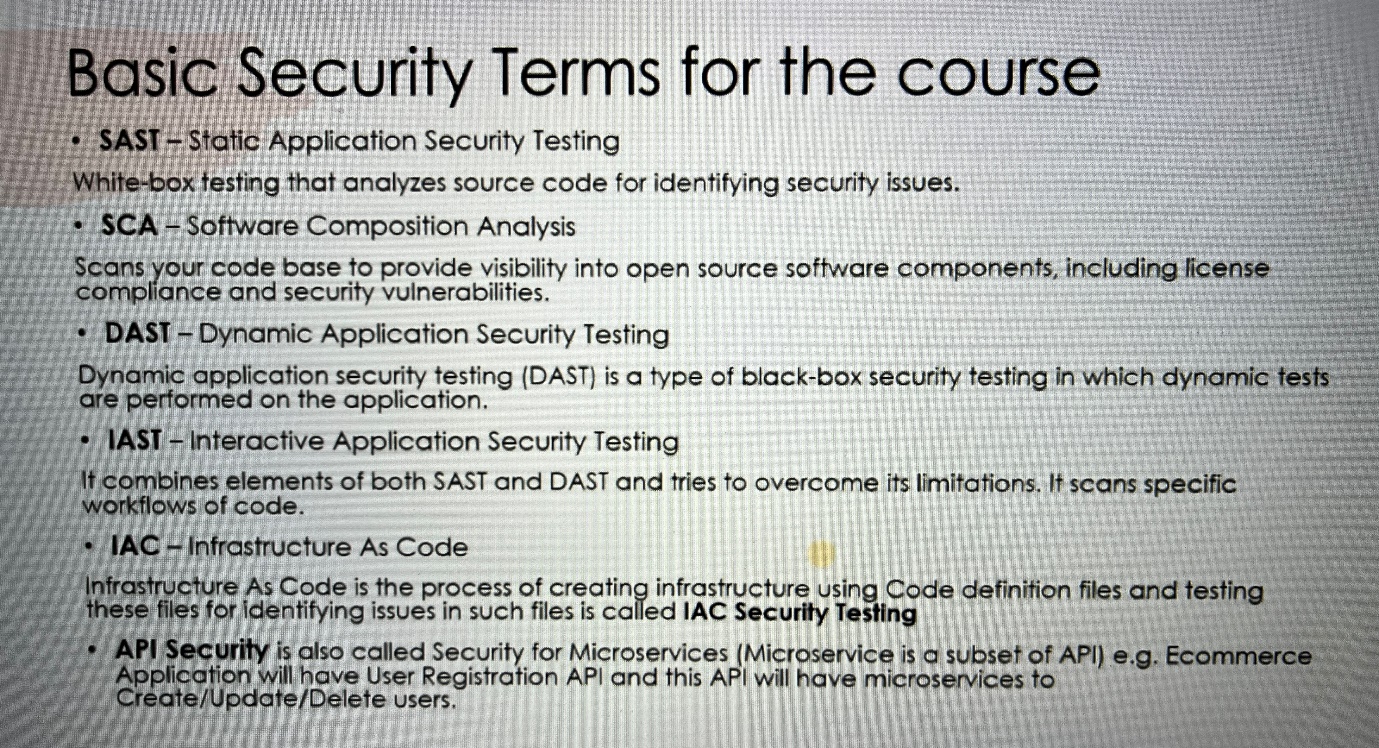
### Security Phase:

1. Code Quality Tools (SonarQube)
2. SAST Tools (Fortify, Veracode, Checkmarx)
3. SCA Tools (Snyk, Fortify, Veracode, Blackduck)
4. DAST Tool (OWASP Zap, WebInspect, Veracode Dast, Acunetix)
5. IAC Security (Snyk, BridgeCrew)
6. Container Security (Aqua, PrismaCloud, Qualys)

### Operations Phase:

1. Build Pipeline tools: Jenkins, AWS, Gitlab
2. Cloud Security Posture Management Tools: Aqua, BridgeCrew
3. Container Registry Scanning Tools: Aqua, AWS Native Registry Scanning
4. Infrastructure Scanning Tools: Chef Inspec(Compliance), Nessus
5. Cloud Security Tools: AWS Security Hub, Azure Defender

# Basic Security Terms



## Terraform-AWS-Jenkins-K8S-Pipeline

Repo: <https://github.com/Charan-Raj-K/terraform-ec2-jenkins-aws-k8s-infra-creation.git>

Course Link: <https://www.udemy.com/course/devsecops-with-terraform-kubernetes-jenkins-aws/learn/lecture/39552090#overview>

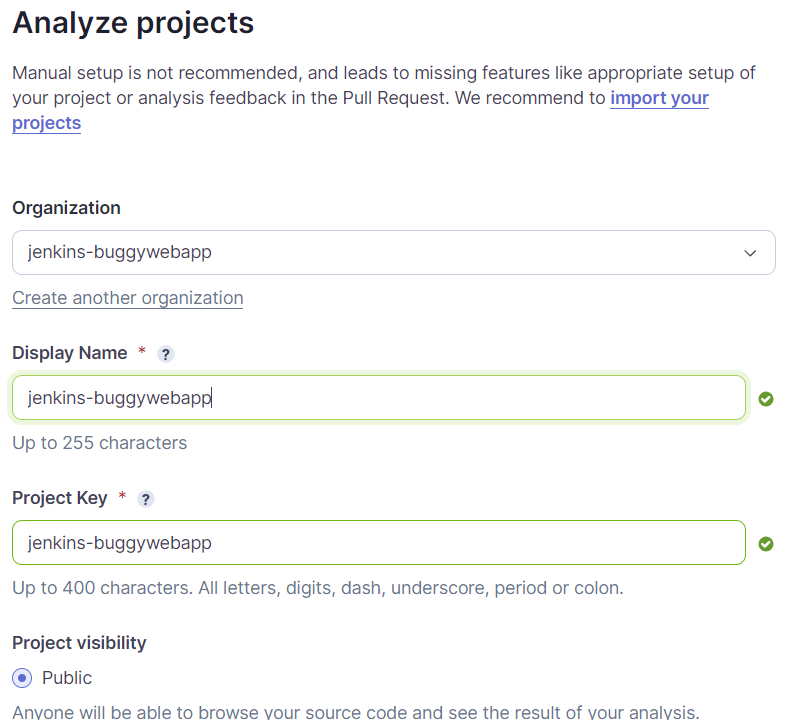
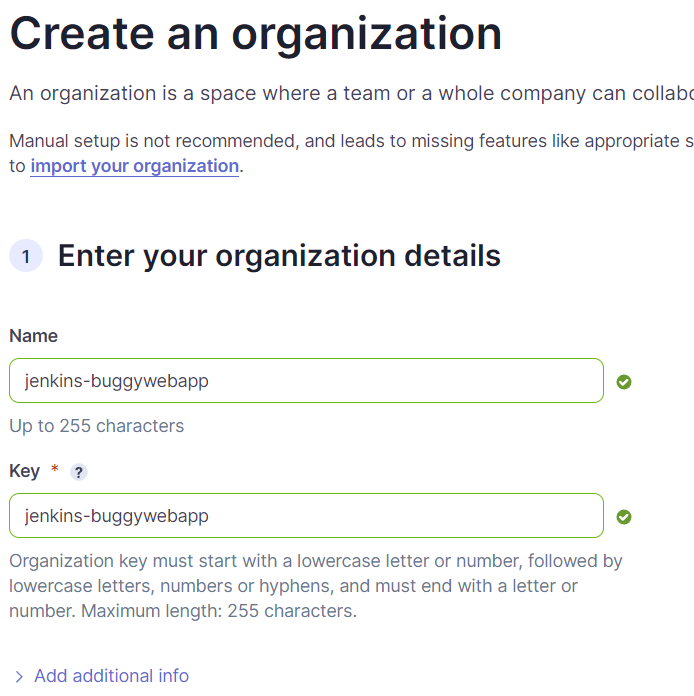
### Install Plugins in Jenkins:

1. Docker
2. Kubernetes CLI
3. Amazon ECR
4. AWS Credentials Plugin

### Integrate Sonarcloud with Jenkins

Repo Url: <https://github.com/Charan-Raj-K/devsecops-jenkins-k8s-tf-sast-sonarcloud-repo>

1. Create an organization in sonarcloud, create project to analyze



1. Add the sonar configuration in the Jenkins file 
2. Create Jenkins pipeline job and add the git url: https://github.com/Charan-Raj-K/devsecops-jenkins-k8s-tf-sast-sonarcloud-repo.git