|  |  |
| --- | --- |
| Name: | Deekshith KR |
| Lab User ID: | 23SEK3324\_U08 |
| Date: | 10/01/2024 |
| Application Name: | [**Vulnerable Java Web Application**](https://github.com/D33ksh1th/VulnerableJavaWebApplication) – **o1** |

**Follow the below guidelines:**





System Architecture:

(Understand the system and document the physical and logical architecture of the system, use the shapes and icons to capture the system architecture)

AWS

Ubuntu instance

Docker

Web Server

Container

Docker image

Define system’s normal behavior:

(Define the steady state of the system is defined, thereby defining some measurable outputs which can indicate the system’s normal behavior)

This web application is deployed successful, and it resembles advertisement page.

It consists of sign-up page to create account and to create an advertisement of one’s choice

When the user creates his account and inputs the details of the advertisement that output returns positive

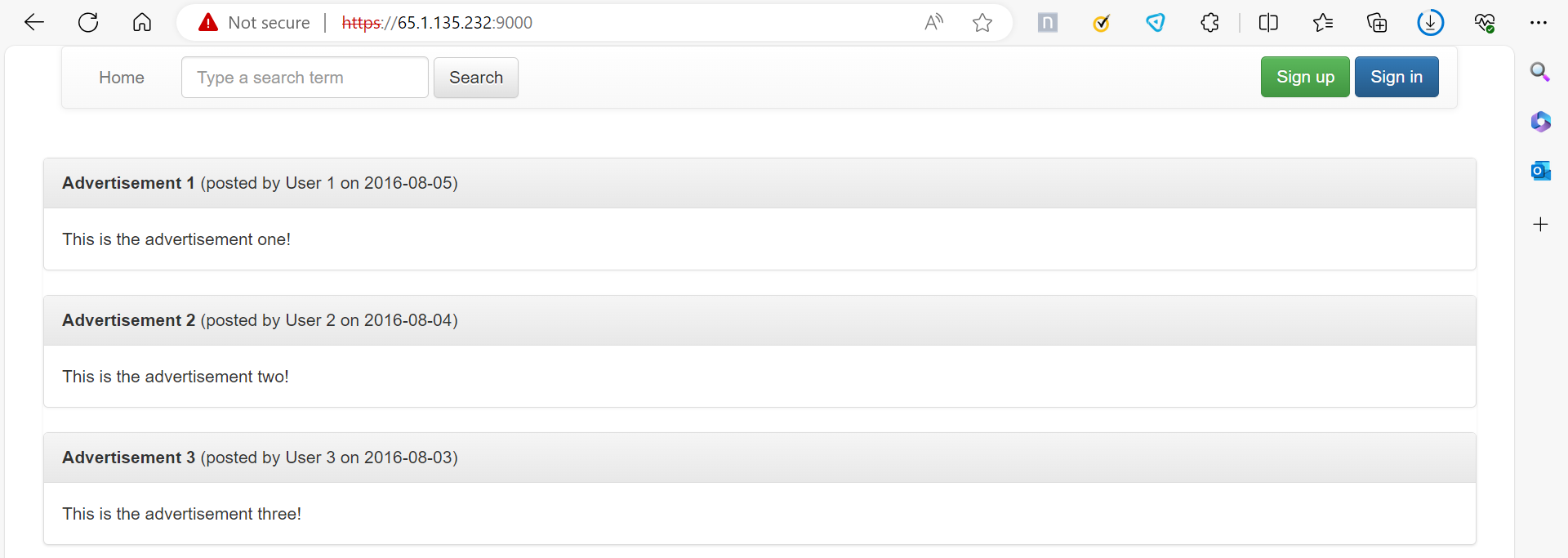
Hypothesis:

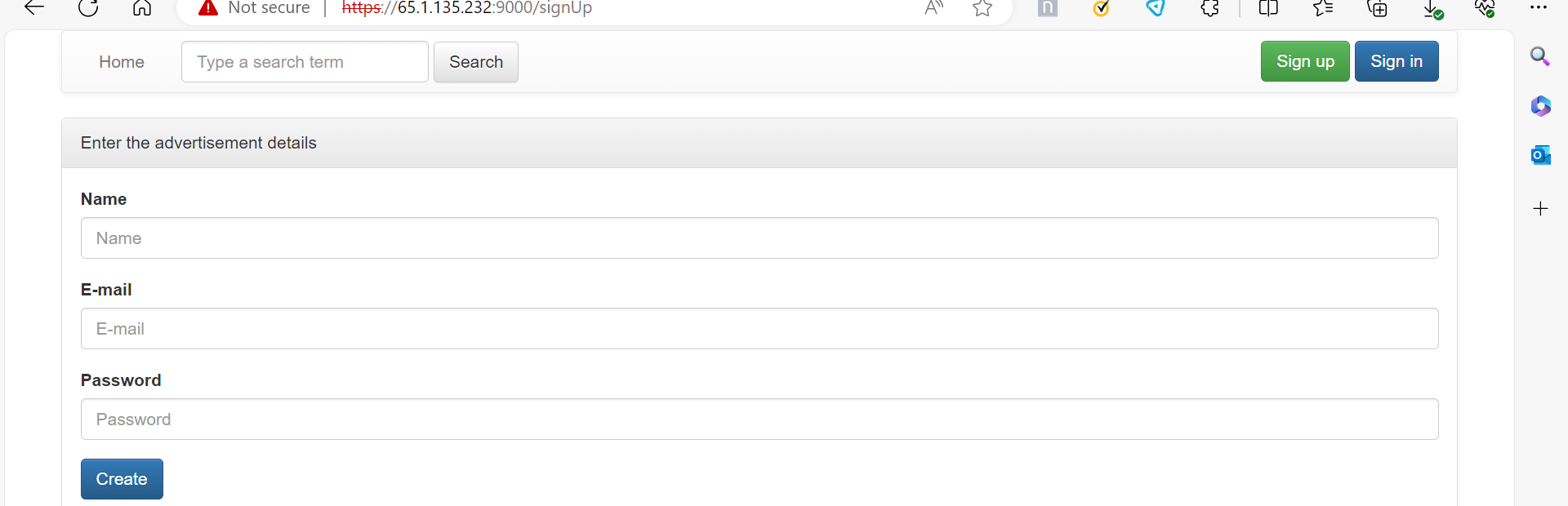
(During an experiment, we need a hypothesis for comparing to a stable control group, and the same applies here too. If there is a reasonable expectation for a particular action according to which we will change the steady state of a system, then the first thing to do is to fix the system so that we accommodate for the action that will potentially have that effect on the system. For eg: "If one of our database servers fails, our service will automatically switch to a backup server, and users will not experience any downtime or data loss.")

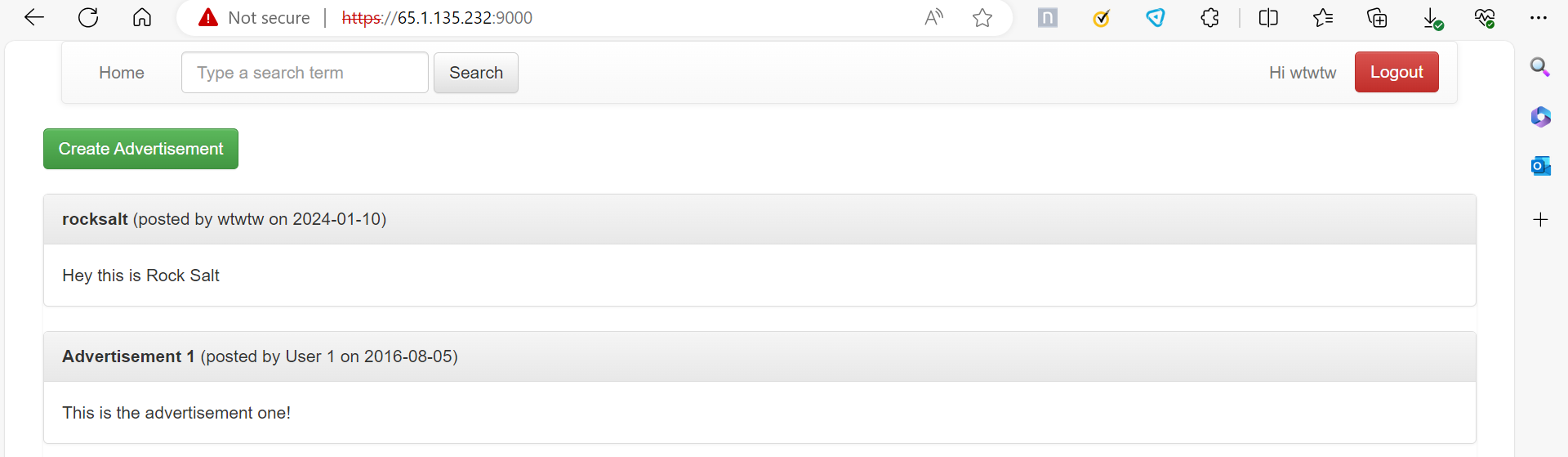


This web application is deployed using AWS ubuntu instance using docker , a docker image is built and is run

exposing port 9000. Once the container is running once can use their Ip in the web i.e. localhost:9000.







Experiment:

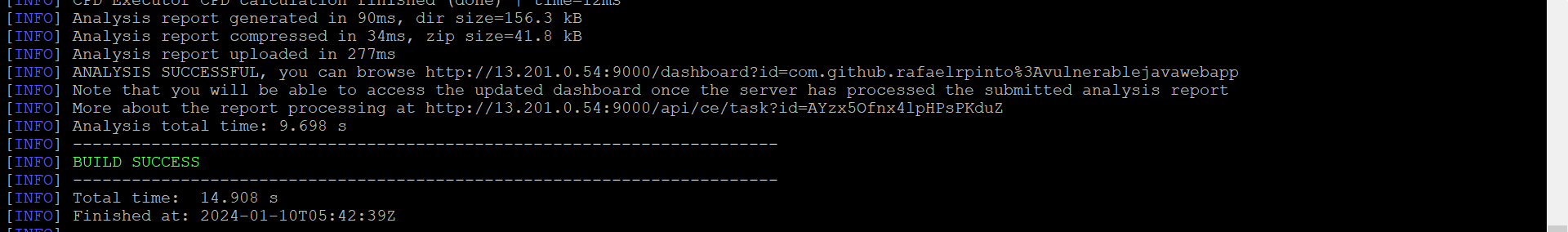
(Document your Preparation, Implementation, Observation and Analysis )

1. Testing the code using SonarQube
2. Trivy
3. SNYK

Test using Maven and SonarQube

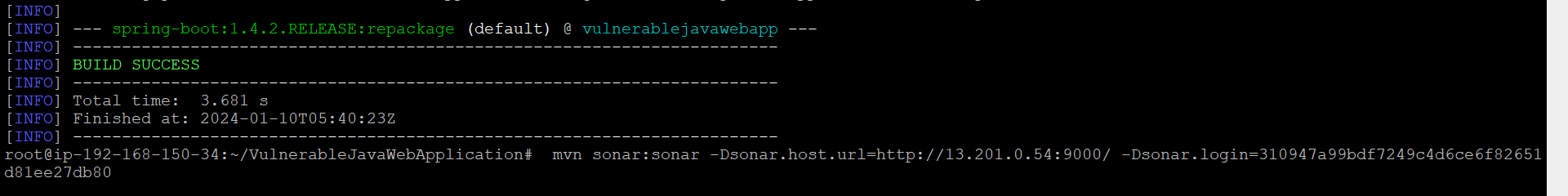
For this test we need maven and sonarqube which should be installed and configured in the same system.

We are going to clean the pom.xml file which lies in the repository using mvn clean.

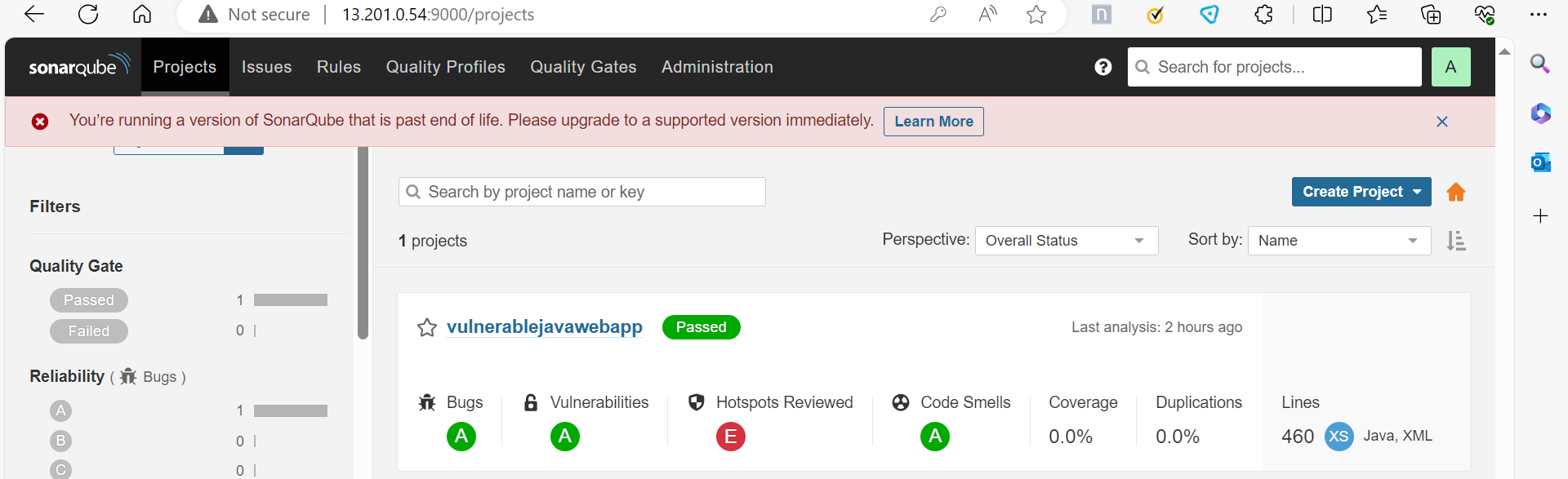


Also same with mvn package which was a build success

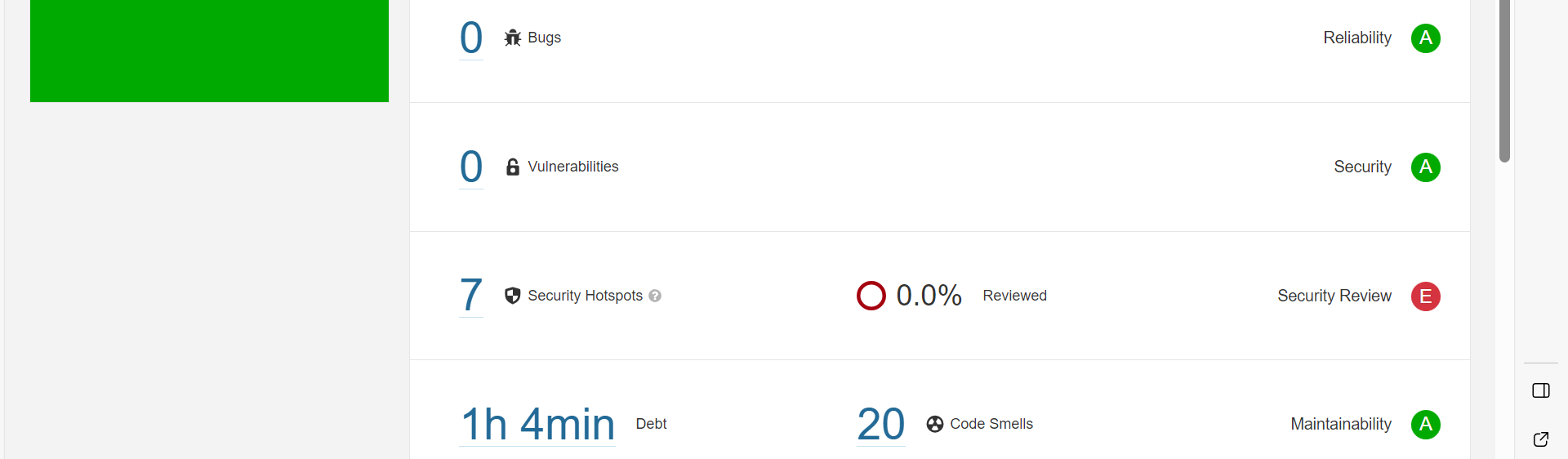
Next run the sonarqube:



Now the sonarqube application goes live and also test the code to check for vulnerabilities :



When we go in depth we can find that this code has 0 bugs and 0 vulnerabilities , 7 security hotspots and has passed the test



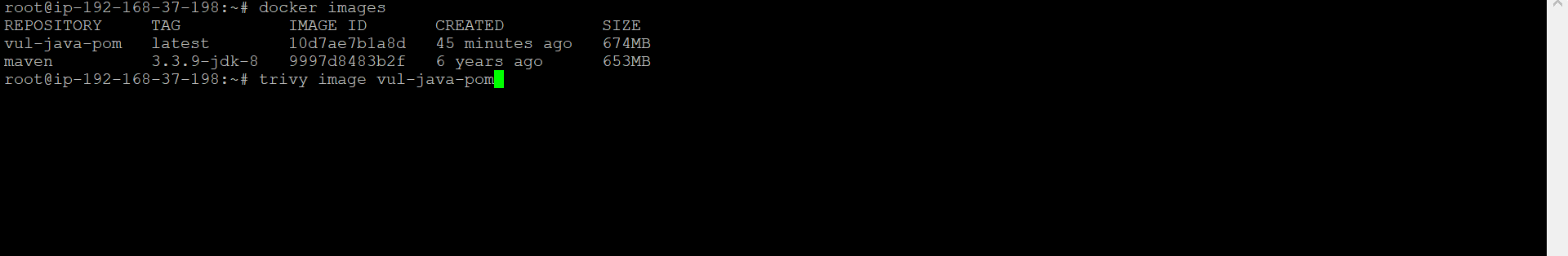
This was the result of this test.

Next test: Trivy tool

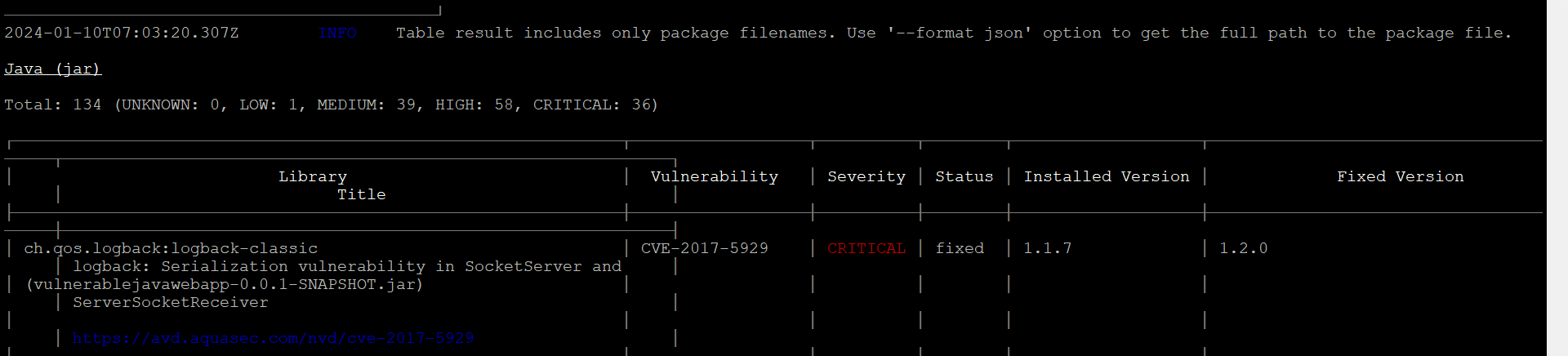
We are going to trivy to scan the image to find the vulnerability and to check the severity of the threat

We shall use this command to run the test:

Trivy image <image name>



The output of this test is :



There threat severity are :

LOW=1 , MEDIUM=39, HIGH= 58 , CRTICAL = 36

Lets see 2 of the critical threat :

1. CVE-2021-26291 - rate 9.1

Apache Maven will follow repositories that are defined in a dependency’s Project Object Model (pom) which may be surprising to some users, resulting in potential risk if a malicious actor takes over that repository or is able to insert themselves into a position to pretend to be that repository

2)CVE-2016-1000027 – rate 9.8:

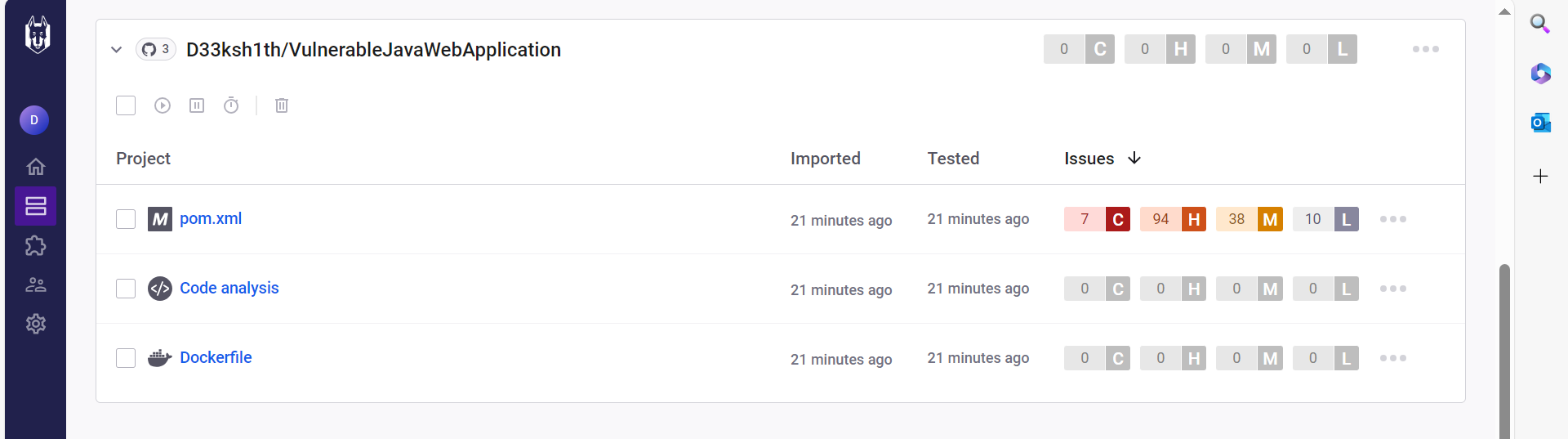
Pivotal Spring Framework through 5.3.16 suffers from a potential remote code execution (RCE) issue if used for Java deserialization of untrusted data. Depending on how the library is implemented within a product, this issue may or not occur, and authentication may be required. NOTE: the vendors position is that untrusted data is not an intended use case. The products behavior will not be changed because some users rely on deserialization of trusted data

We can also use SNYK tool to check out the vulnerabilities.

We need to sign up in snyk portal and use the git repository that you want test.

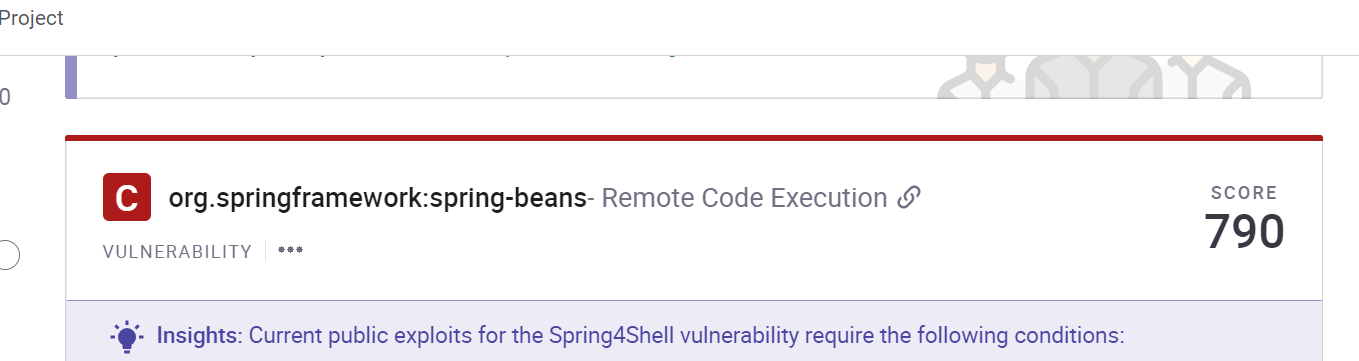
Here I shall use the same java application repository to scan for tests ;

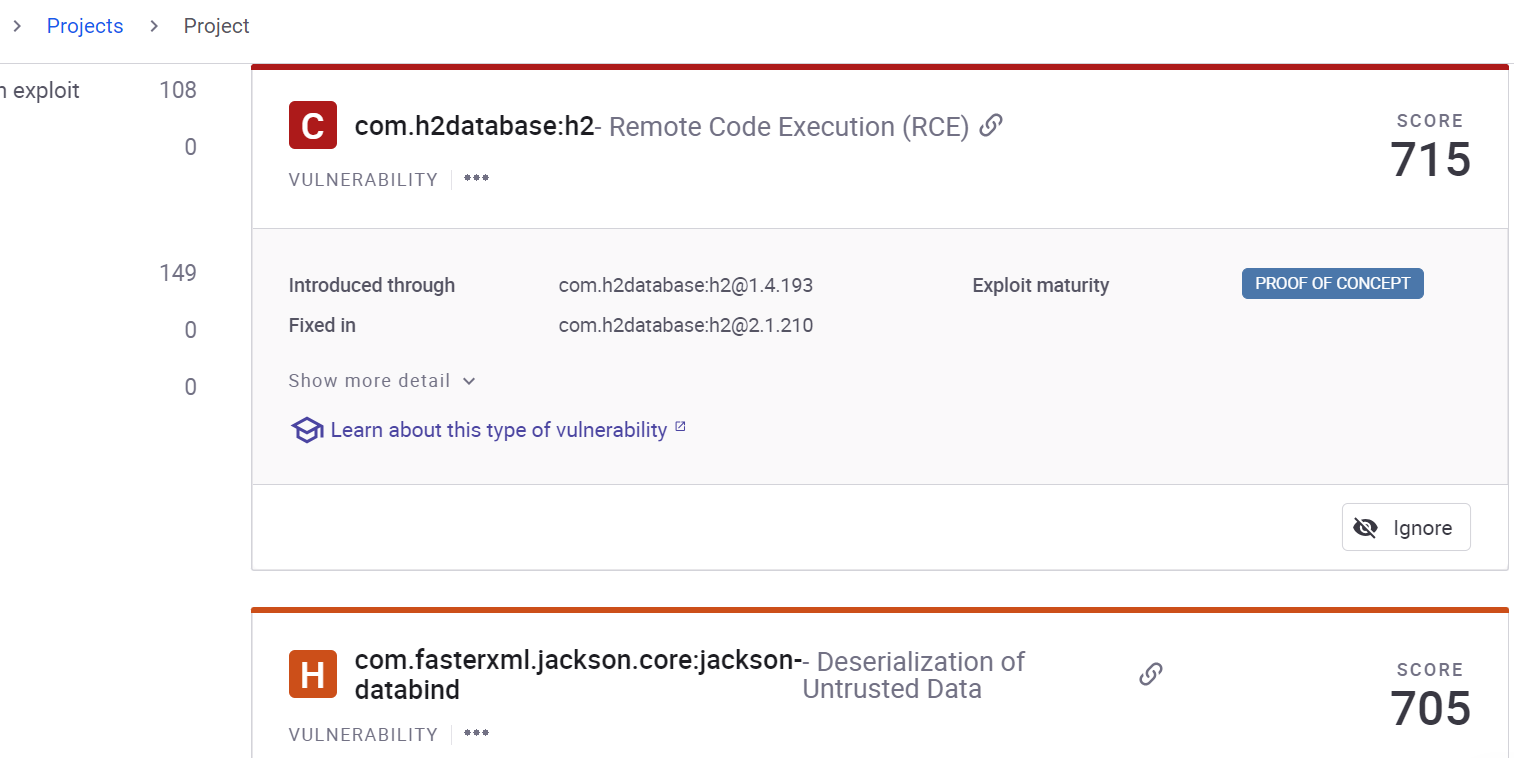
This is the result after the successful test:



Here we can see that the pom.xml file contains the vulnerability it has 7 critical 94 high and 38 medium threats

These are the few details of the test :





Experiment:

(Document your Preparation, Implementation, Observation and Analysis )