# **Case Study Report**

### **4. Continuous Integration with Static Code Analysis**

* **Concepts Used**: Jenkins, SonarQube, and AWS Cloud9.
* **Problem Statement**: "Set up a Jenkins pipeline using AWS Cloud9 IDE to perform a static analysis of a Java/Python application. Integrate SonarQube for code quality checks."
* **Tasks**:
  + Install Jenkins and set up a basic pipeline.
  + Configure SonarQube as part of the pipeline for static code analysis.
  + Run the pipeline and generate a report for code quality issues.

**2022.sushant.tulasi@ves.ac.in**

**Sushant Tulasi**

**Advops Case Study Assignment**

**17-10-24**

## **Introduction**

In modern software development, Continuous Integration (CI) and Continuous Deployment (CD) practices are essential for delivering high-quality software quickly and efficiently. This report outlines a case study on setting up a CI pipeline using Jenkins and SonarQube for static code analysis of a Java/Python application on AWS Cloud9. By integrating these tools, we can automate code quality checks, enhancing overall software quality and maintainability.

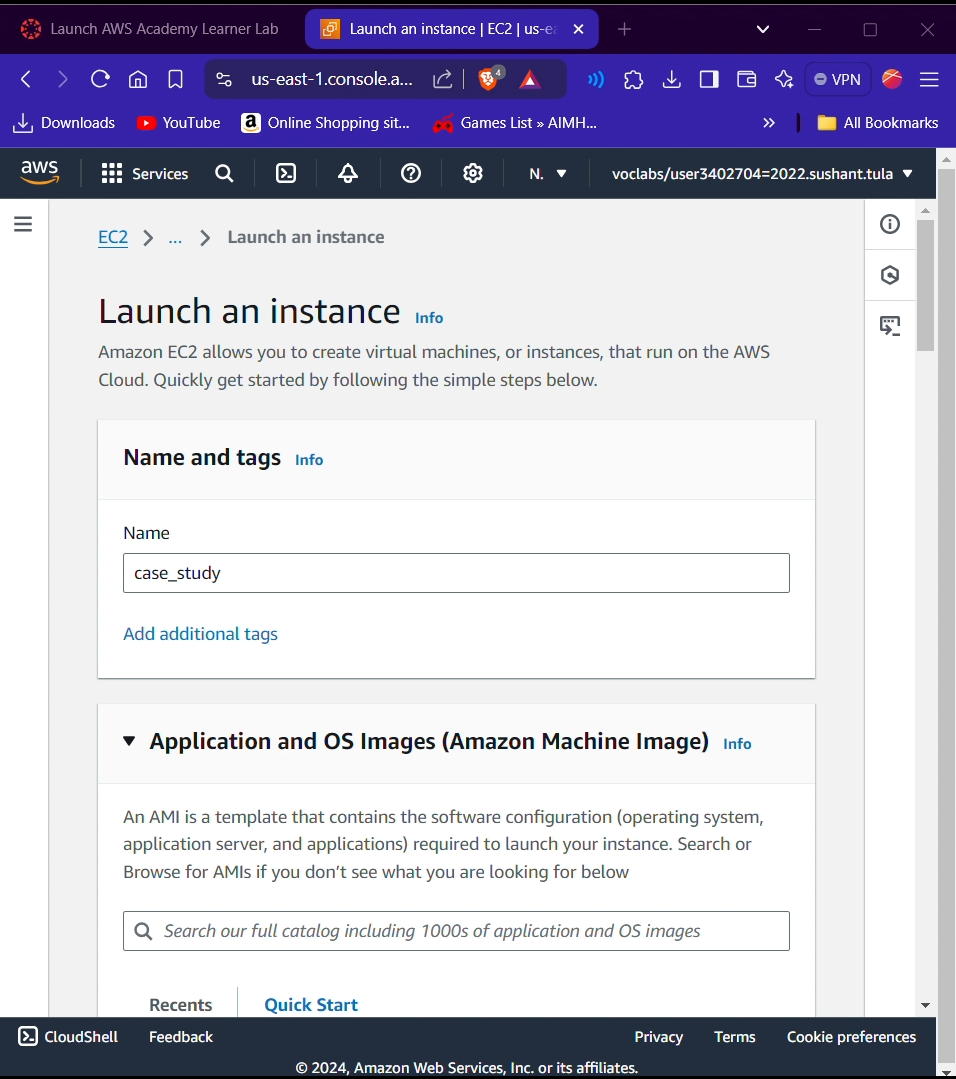
## **Components Used**

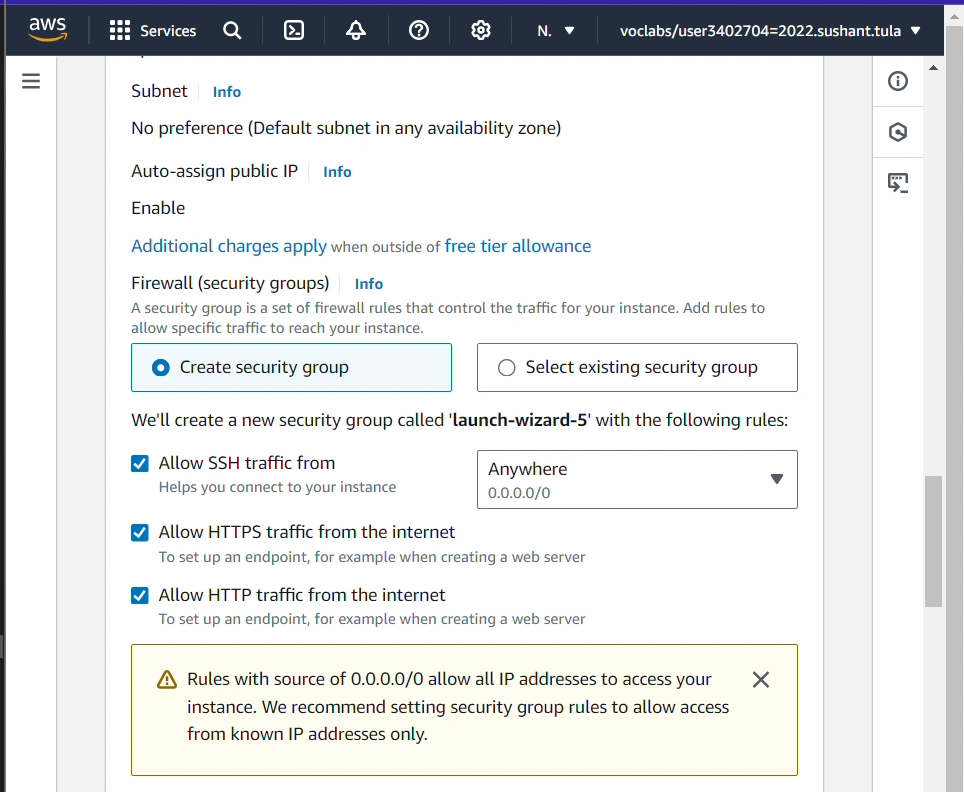
* **AWS EC2**: A scalable cloud computing service from Amazon.
* **Docker**: A platform for developing, shipping, and running applications in containers.
* **SonarQube**: An open-source platform for continuous inspection of code quality, providing static analysis for various programming languages.
* **Jenkins**: An open-source automation server for building, testing, and deploying applications.

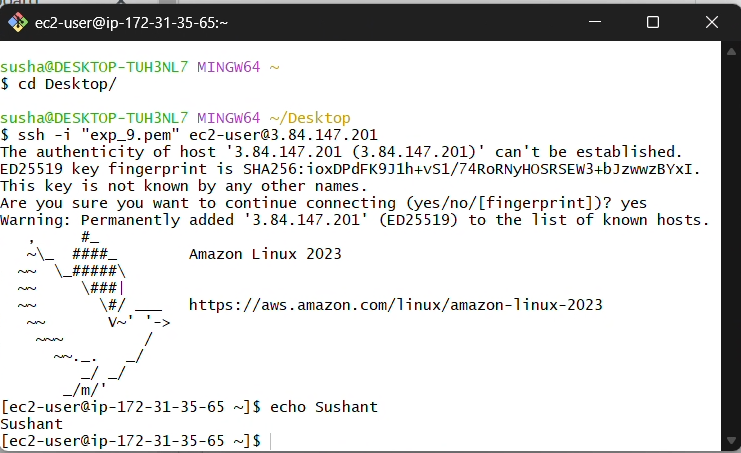
## **Step-by-Step Implementation**

### **1. Creating an EC2 Instance in AWS**

1. Log in to your AWS Management Console.
2. Navigate to the **EC2 Dashboard**.
3. Click on **Launch Instance**.
4. Choose the **Amazon Linux 2 AMI**.
5. Select the **t2.micro** instance type (free tier eligible).
6. Configure instance details as needed and click **Next** until you reach the **Review and Launch** section.
7. Click **Launch** and select or create a new key pair. Download the key pair for SSH access.

**

**

**

### **2. Setting Up Docker on the Instance**

SSH into your EC2 instance:  
ssh -i "your-key.pem" ec2-user@your-ec2-public-ip

Update the package manager:  
For an Ubuntu instance, here are the equivalent commands:

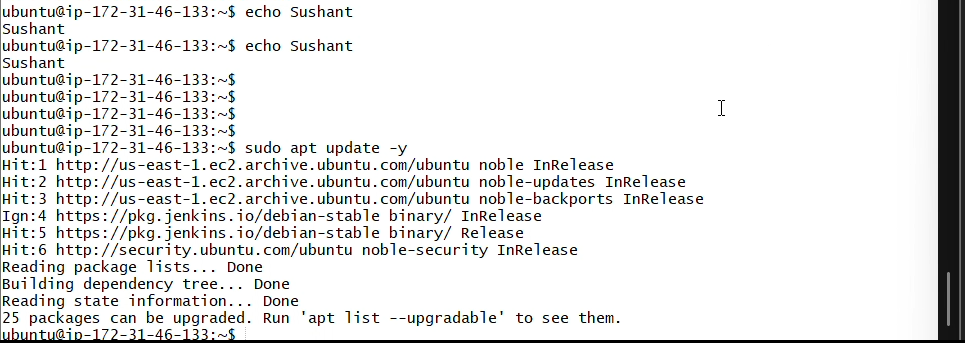
**Update the package list**  
1. sudo apt update -y

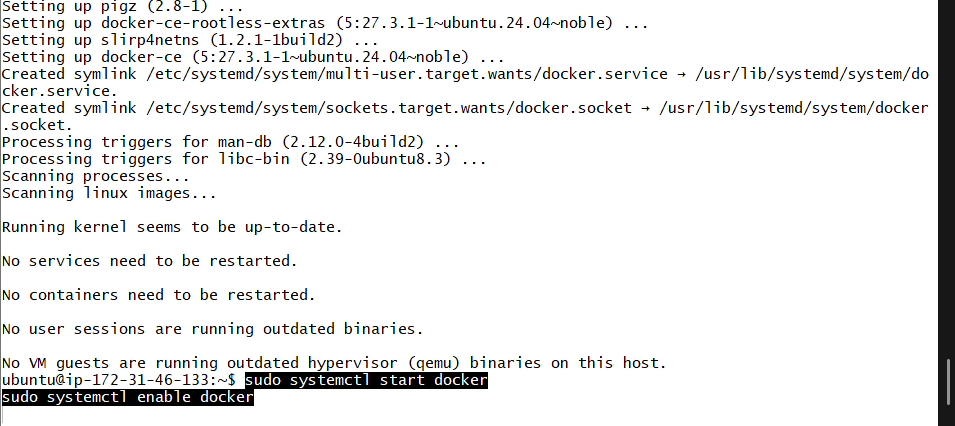
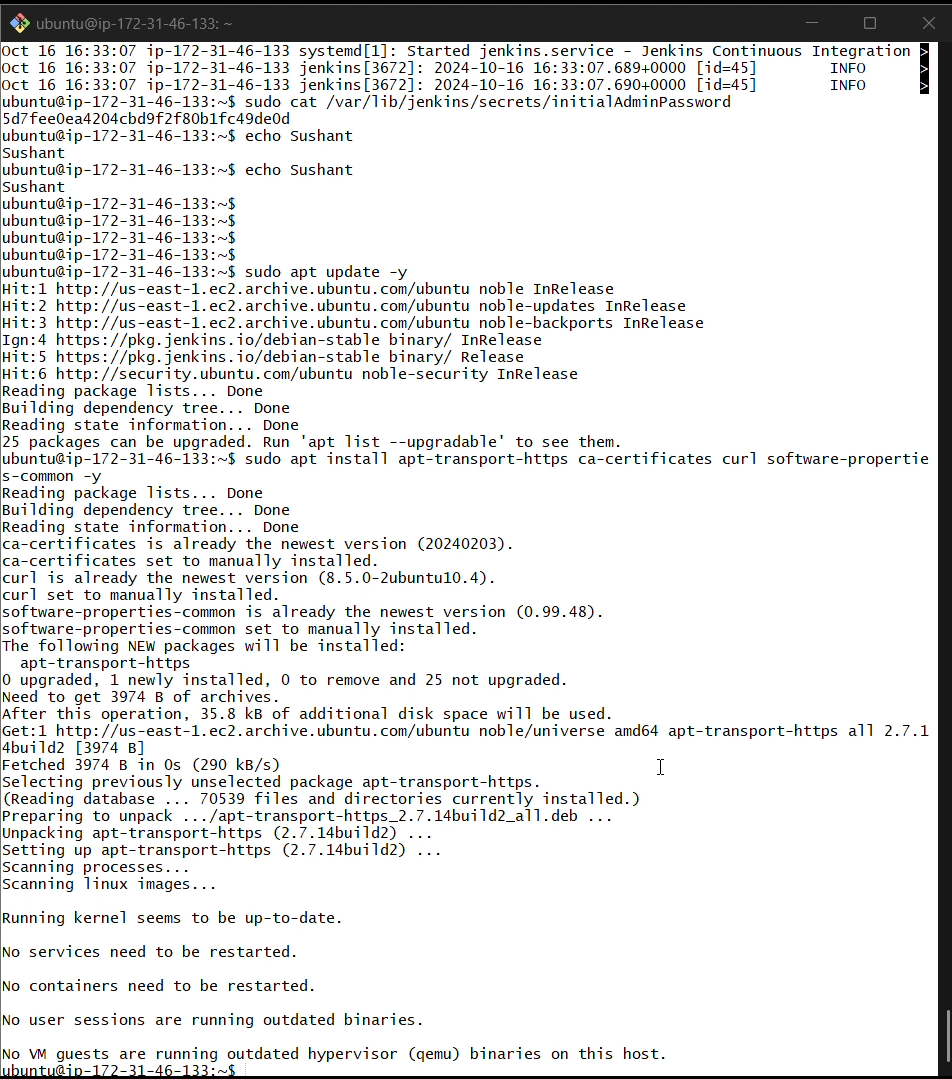
**Install Docker:**  
2. sudo apt install docker.io -y

**Start the Docker service:**3. sudo systemctl start docker

**Enable Docker to start on boot**  
4. sudo systemctl enable docker

**Add the ubuntu user to the Docker group:**5. sudo usermod -aG docker $USER

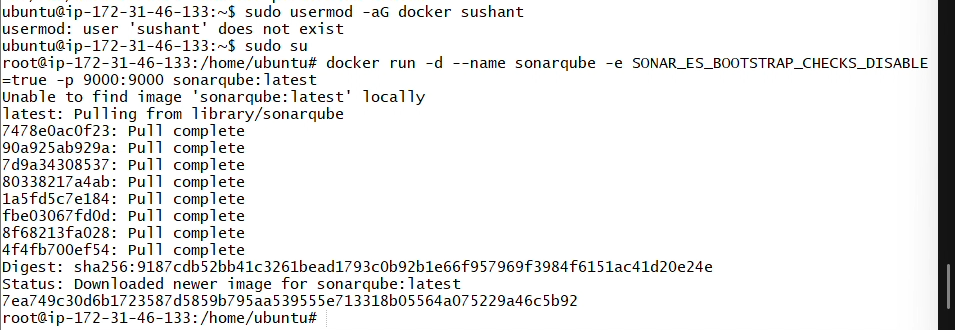
**

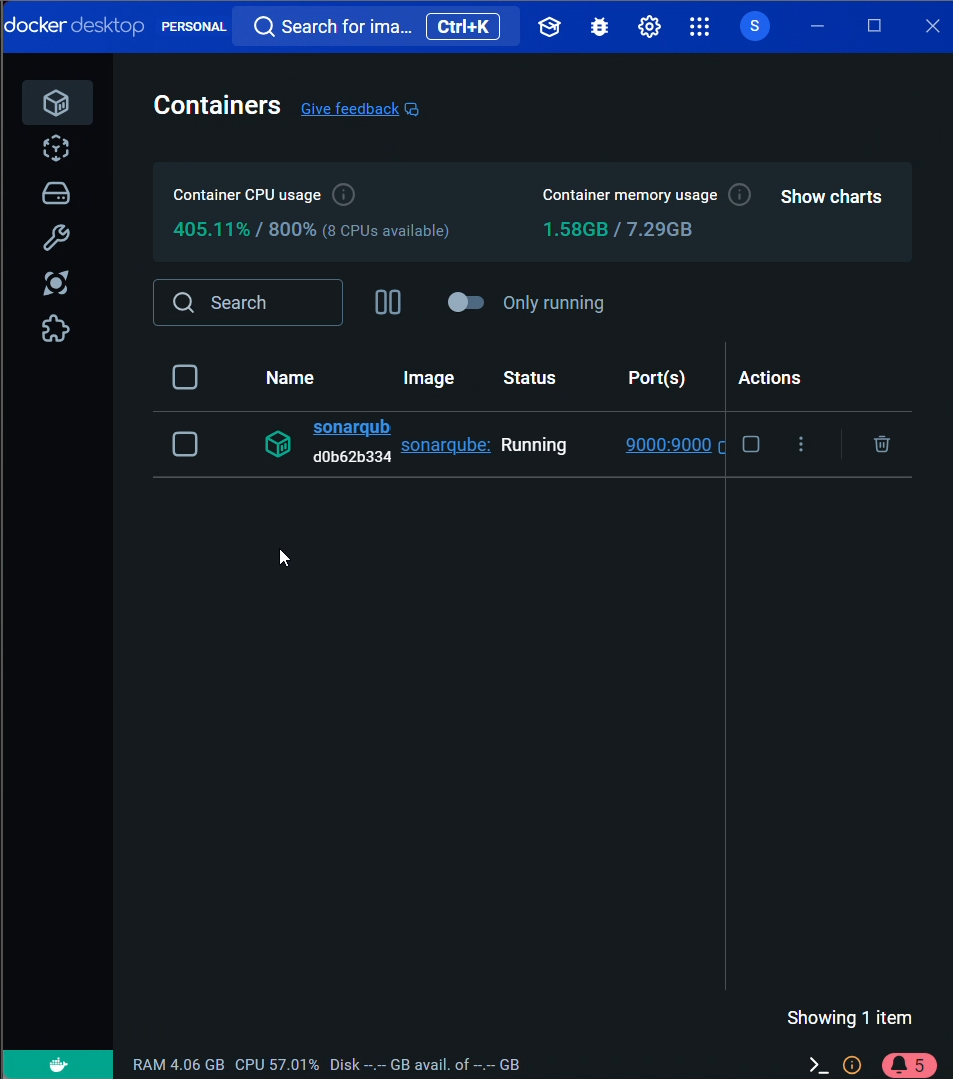
**

### **3. Turning on SonarQube in Docker**

Pull the SonarQube Docker image  
1. docker pull sonarqube

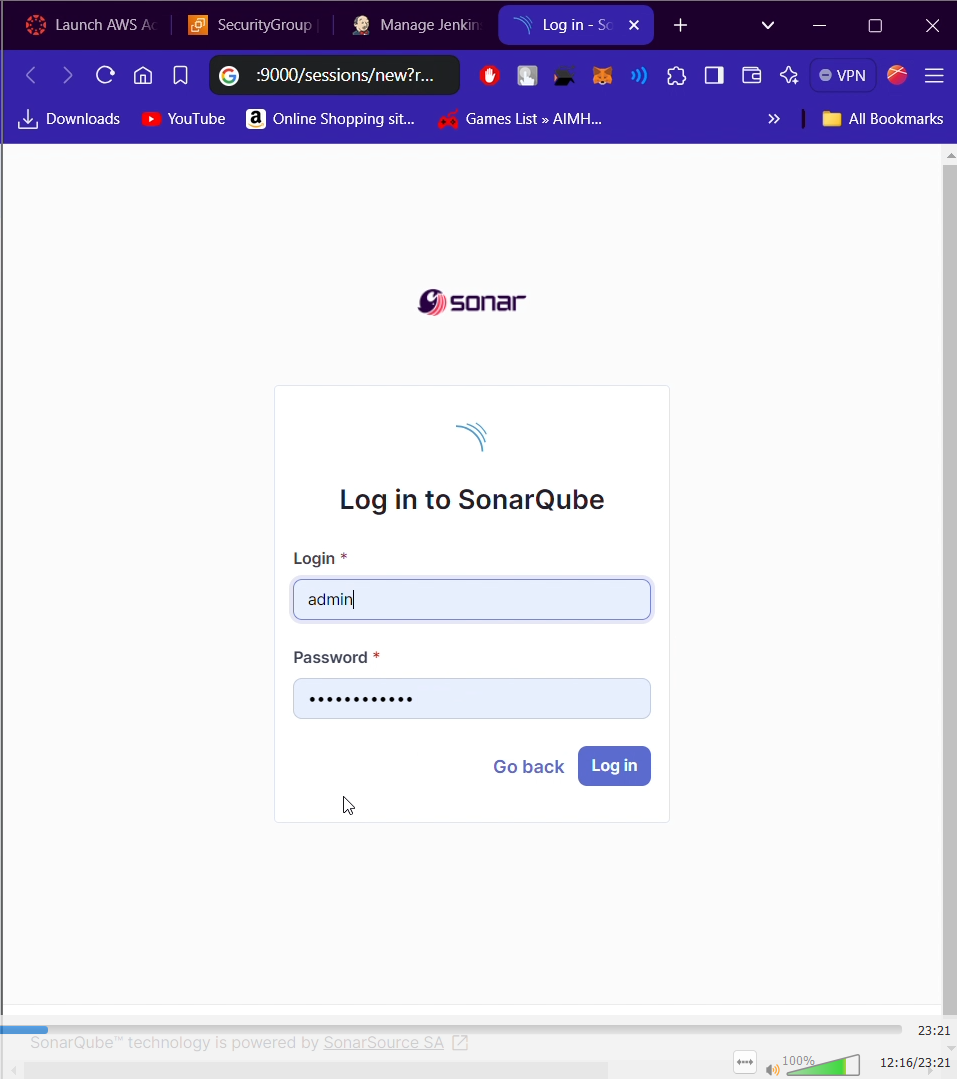
Run the SonarQube container:  
  
2. docker run -d --name sonarqube -p 9000:9000 sonarqube

**

**

### **4. Logging in to SonarQube**

1. Open your browser and navigate to http://your-ec2-public-ip:9000.
2. Log in with the default credentials (admin/admin).

**

### **5. Setting Up Jenkins for SonarQube Scanner**

Install Jenkins

1. Ubuntu ec2 instance
2. Allow security rule for port 8080
3. Sudo apt-get update
4. Sudo apt-get install openjdk-11-jdk -y
5. sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \

https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key

echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc]" \

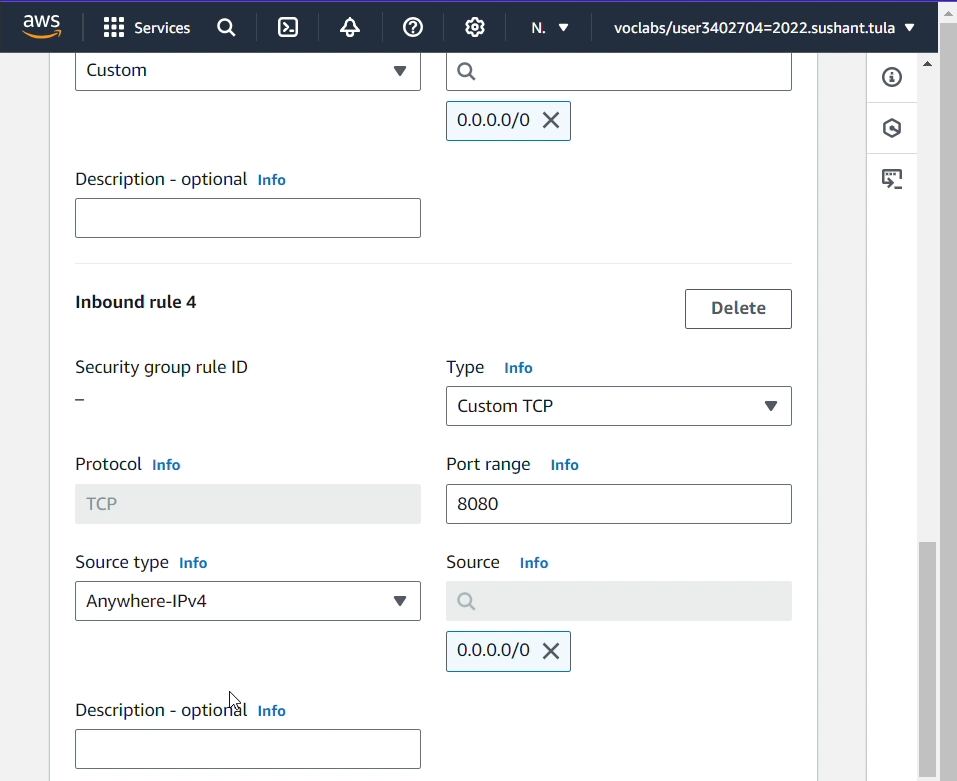
https://pkg.jenkins.io/debian-stable binary/ | sudo tee \

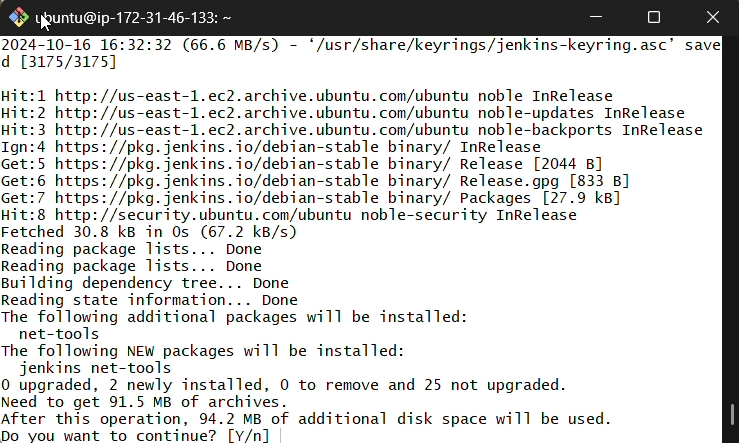
/etc/apt/sources.list.d/jenkins.list > /dev/null

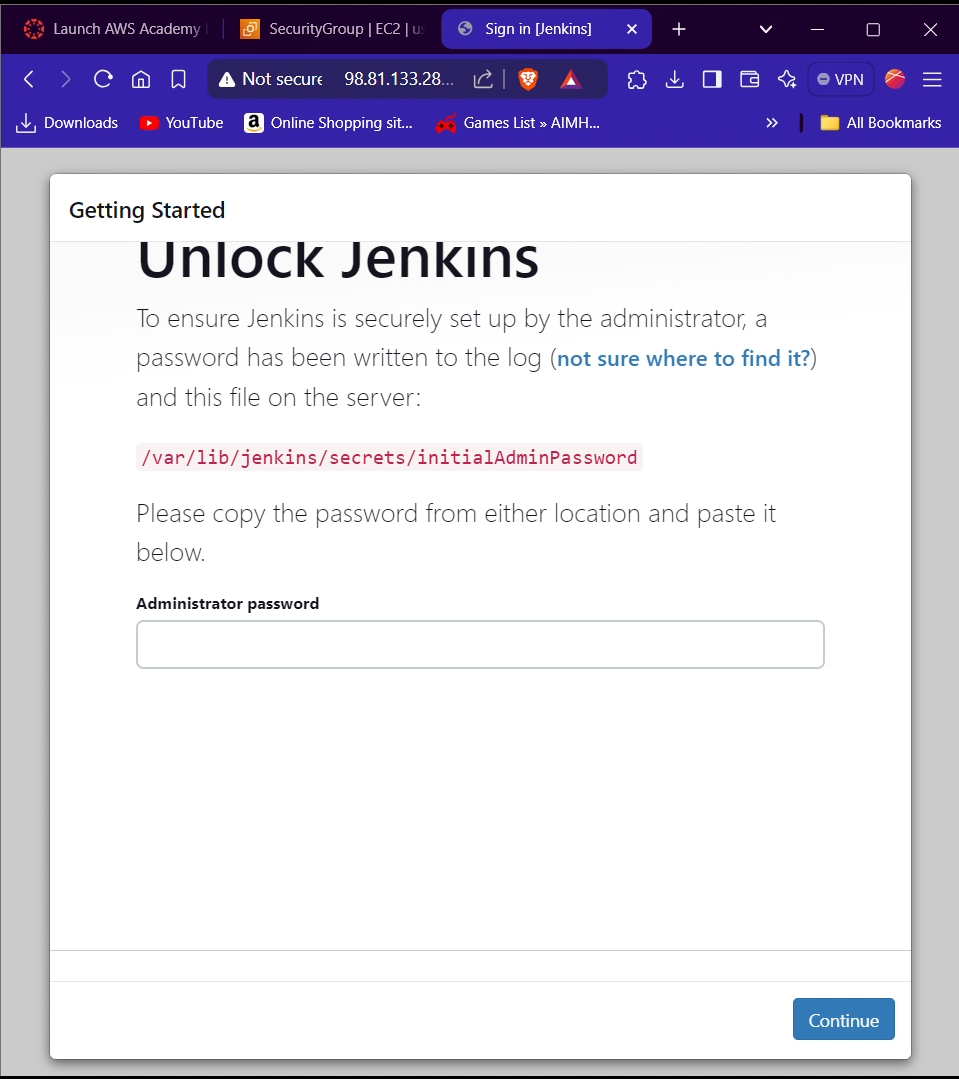
sudo apt-get update

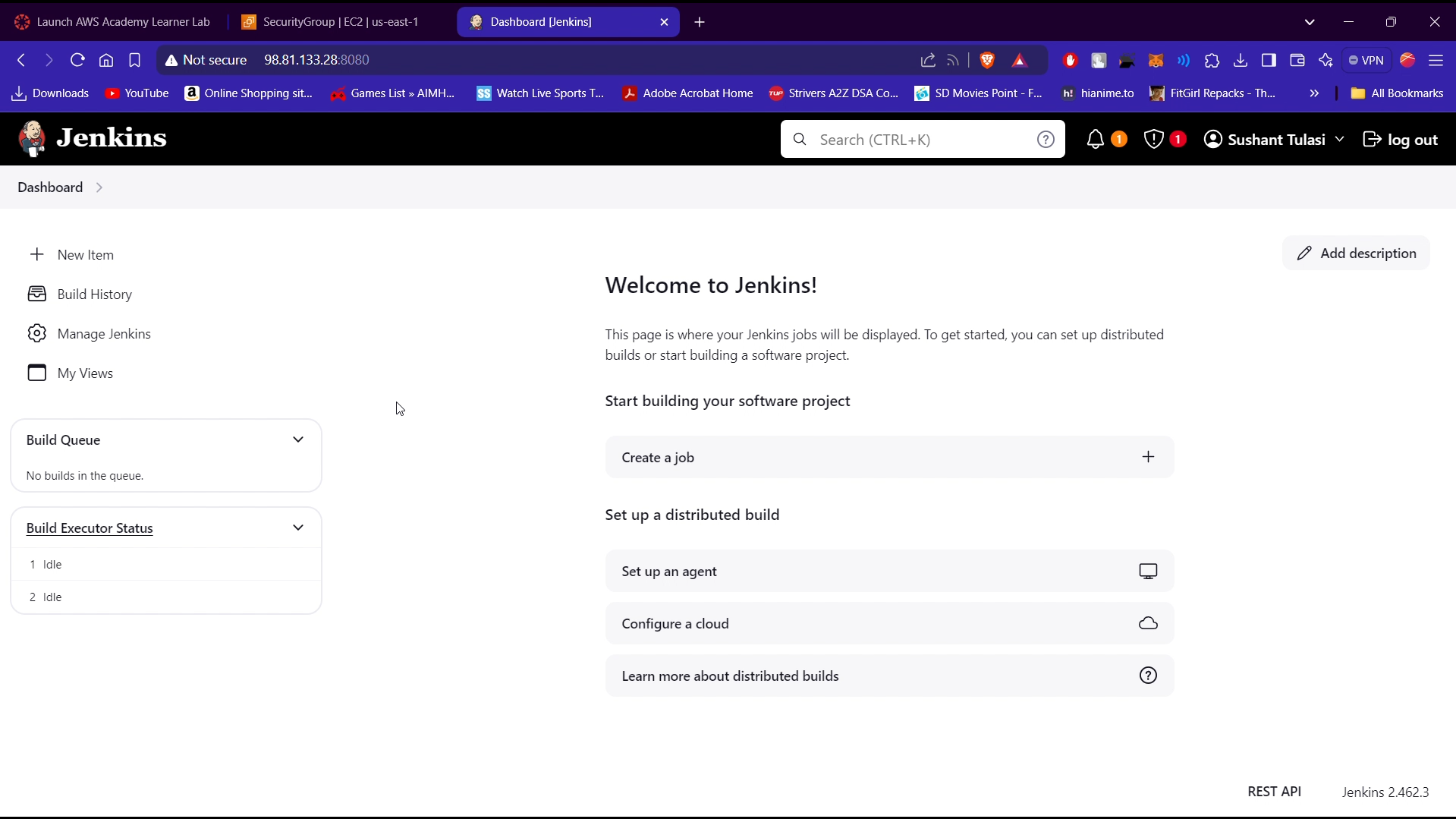
sudo apt-get install jenkins

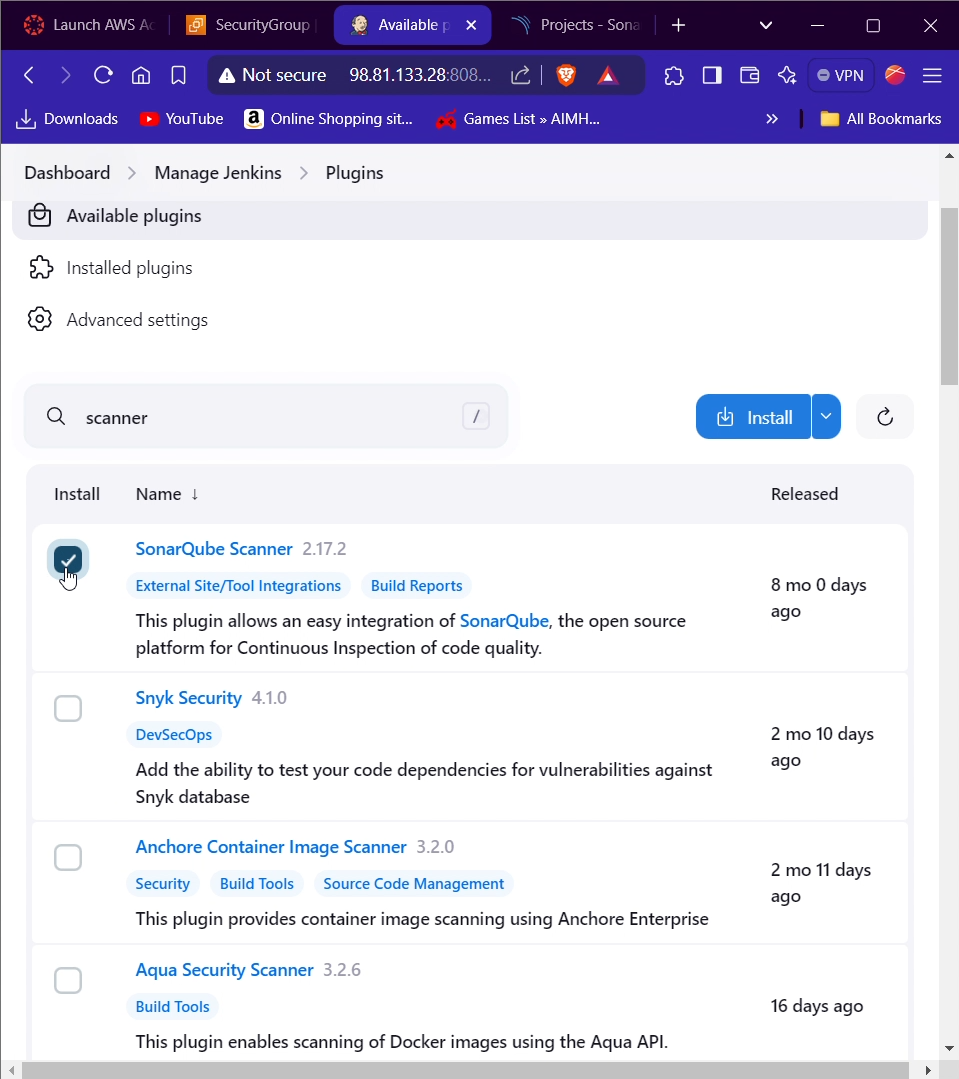
1. sudo systemctl enable jenkins
2. sudo systemctl start jenkins
3. sudo systemctl status jenkins
4. docker run -d --name jenkins -p 8080:8080 jenkins/jenkins:lts
5. Access Jenkins at http://your-ec2-public-ip:8080.
6. Follow the instructions to unlock Jenkins using the initial admin password (found in the container logs).

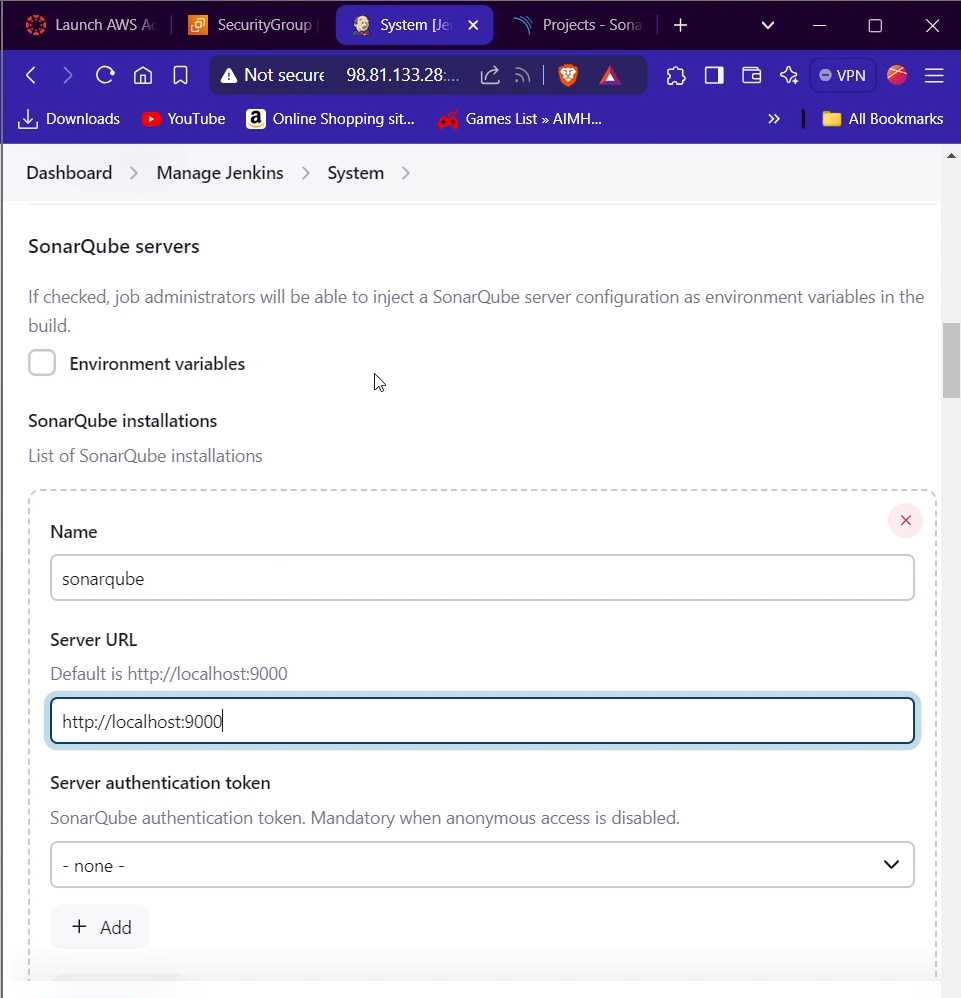
**

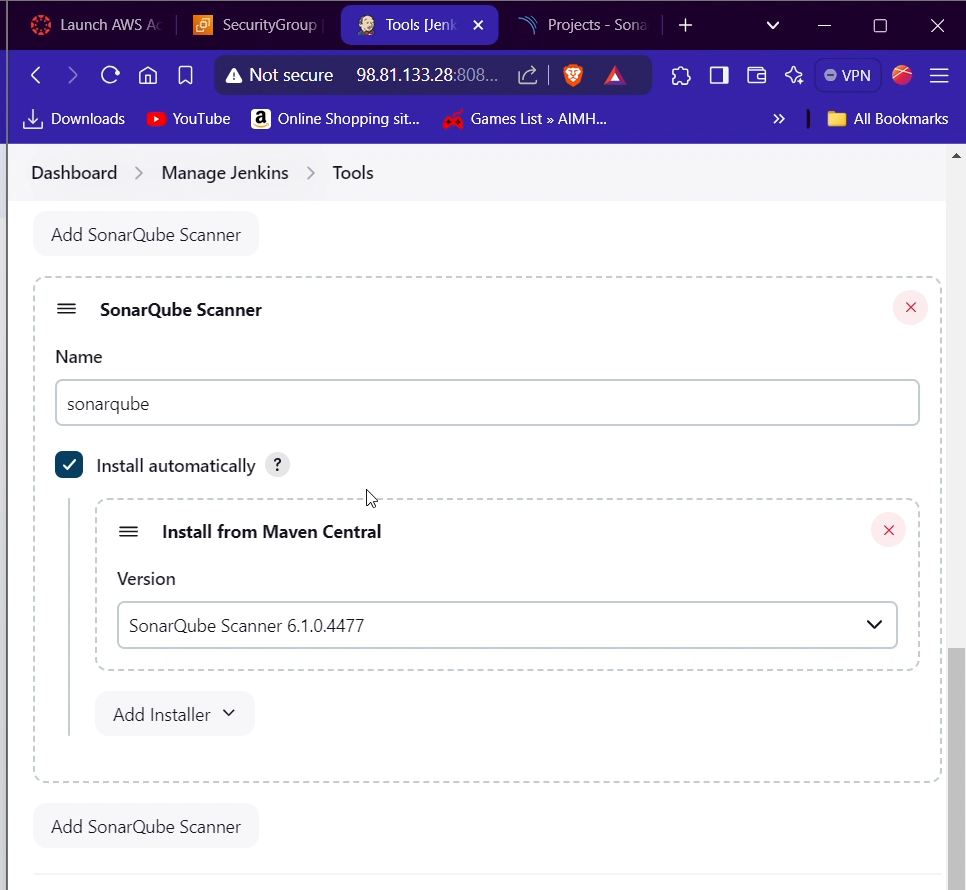
**

**

**

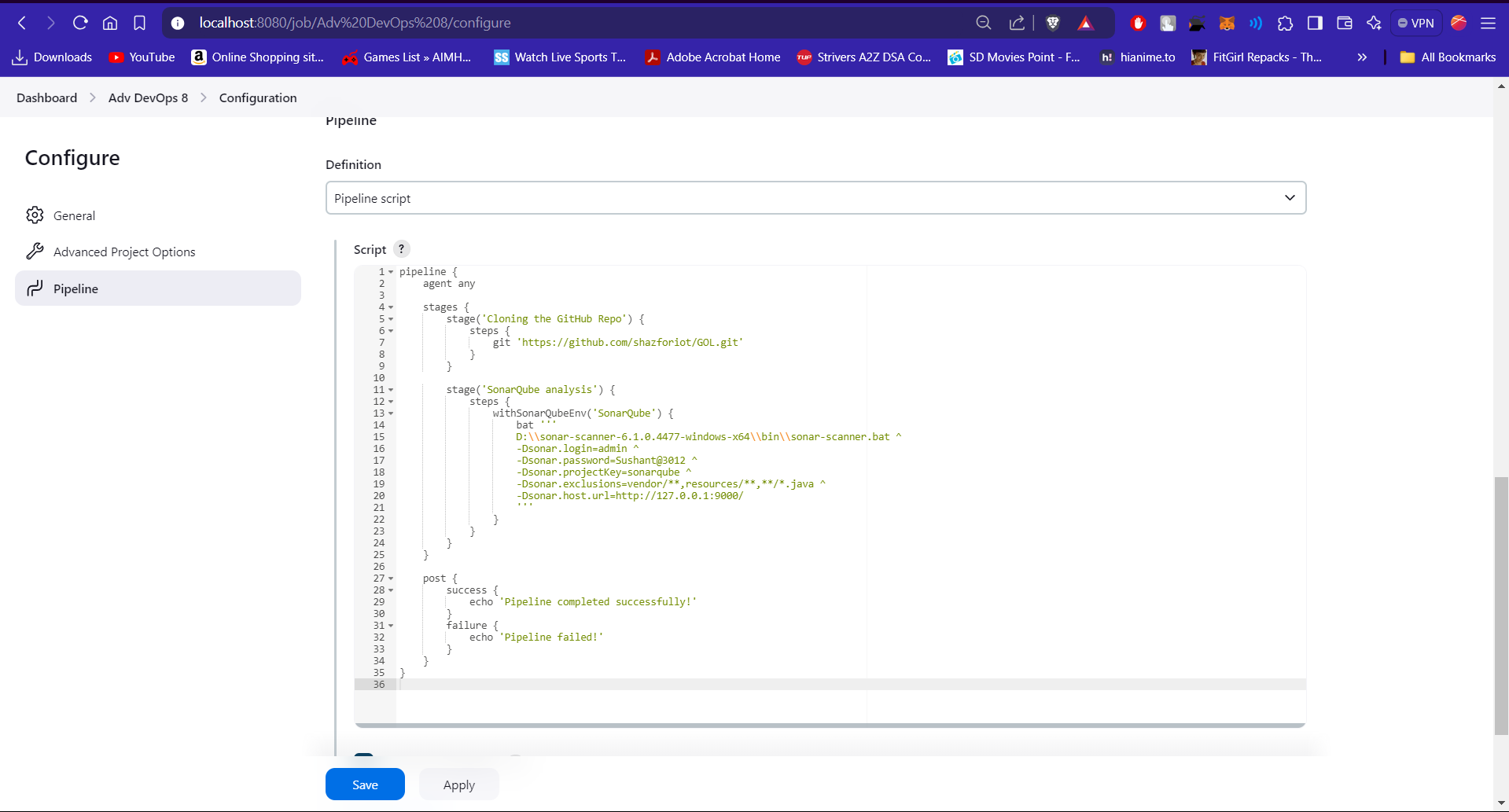
**

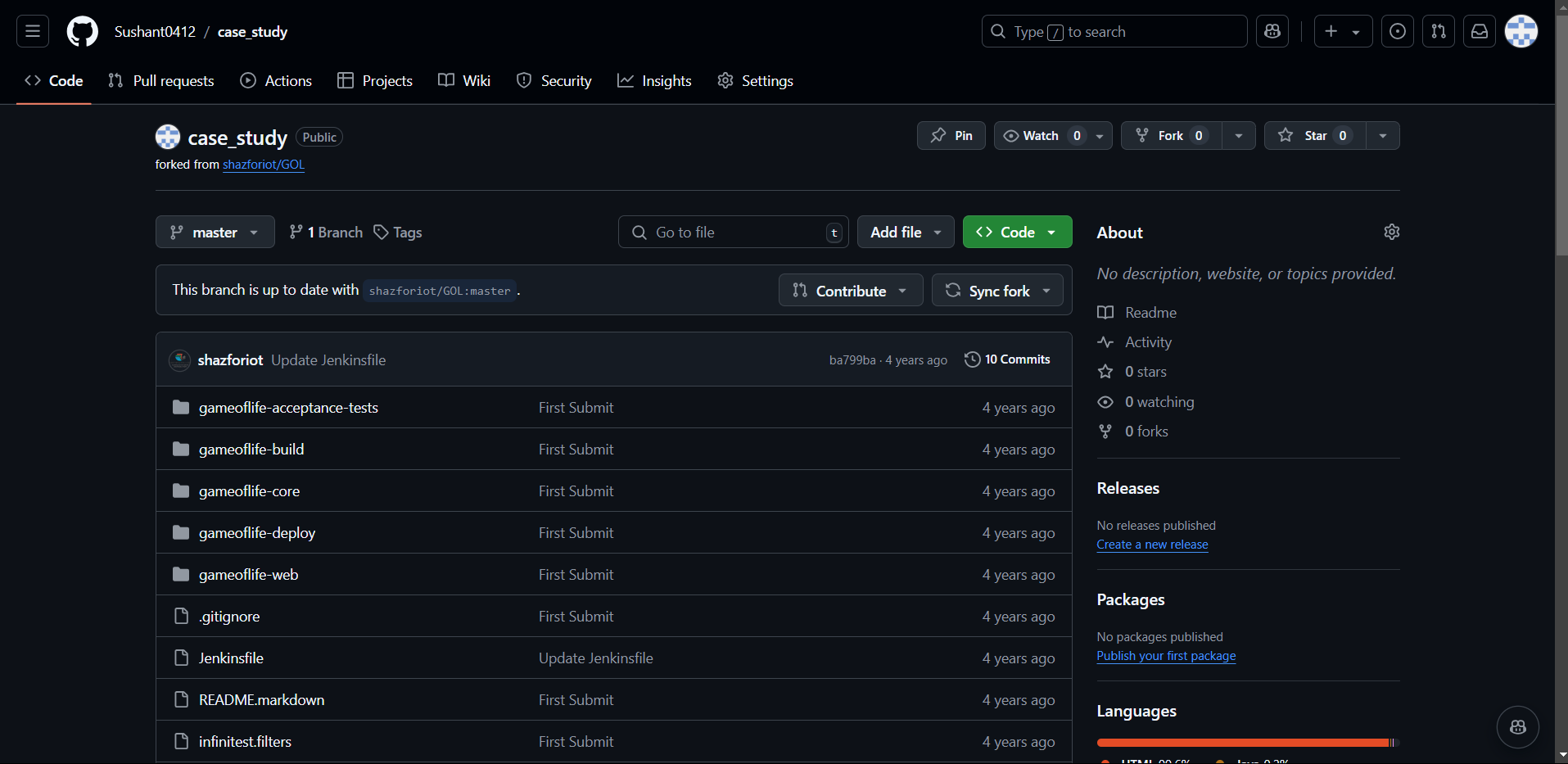
**

**

### **6. Adding a Build for Scanner Analysis**

1. In Jenkins, create a new job (freestyle project).
2. Under **Build Environment**, check "Provide Configuration File Provider Context".
3. Under **Build**, select "Execute shell" and add the commands for running the SonarQube scanner.

**

**

*Script:*pipeline {

agent any

stages {

stage('Cloning the GitHub Repo') {

steps {

git 'https://github.com/shazforiot/GOL.git'

}

}

stage('SonarQube analysis') {

steps {

withSonarQubeEnv('SonarQube') {

sh '''

/opt/sonar-scanner/bin/sonar-scanner \

-Dsonar.login=admin \

-Dsonar.password=your\_password \

-Dsonar.projectKey=sonarqube \

-Dsonar.exclusions=vendor/\*\*,resources/\*\*,\*\*/\*.java \

-Dsonar.host.url=http://public\_ip\_address:9000/

'''

}

}

}

}

post {

success {

echo 'Pipeline completed successfully!'

}

failure {

echo 'Pipeline failed!'

}

}

}

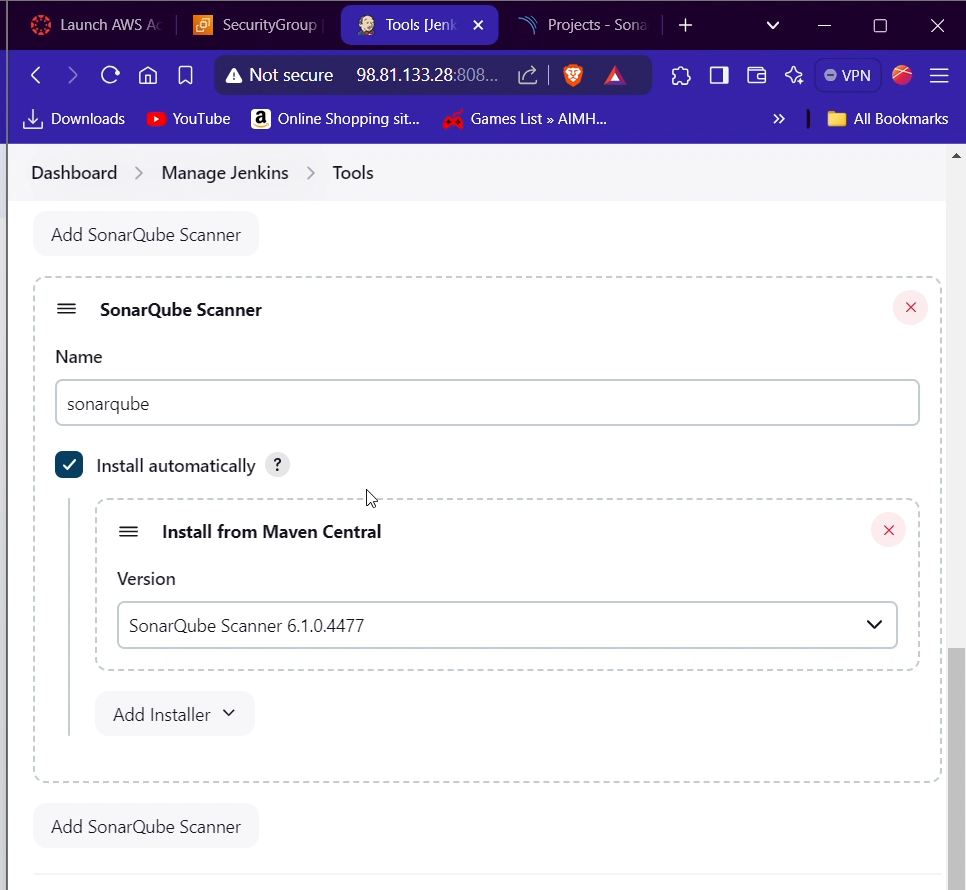
### **7. Adding SonarQube Configuration**

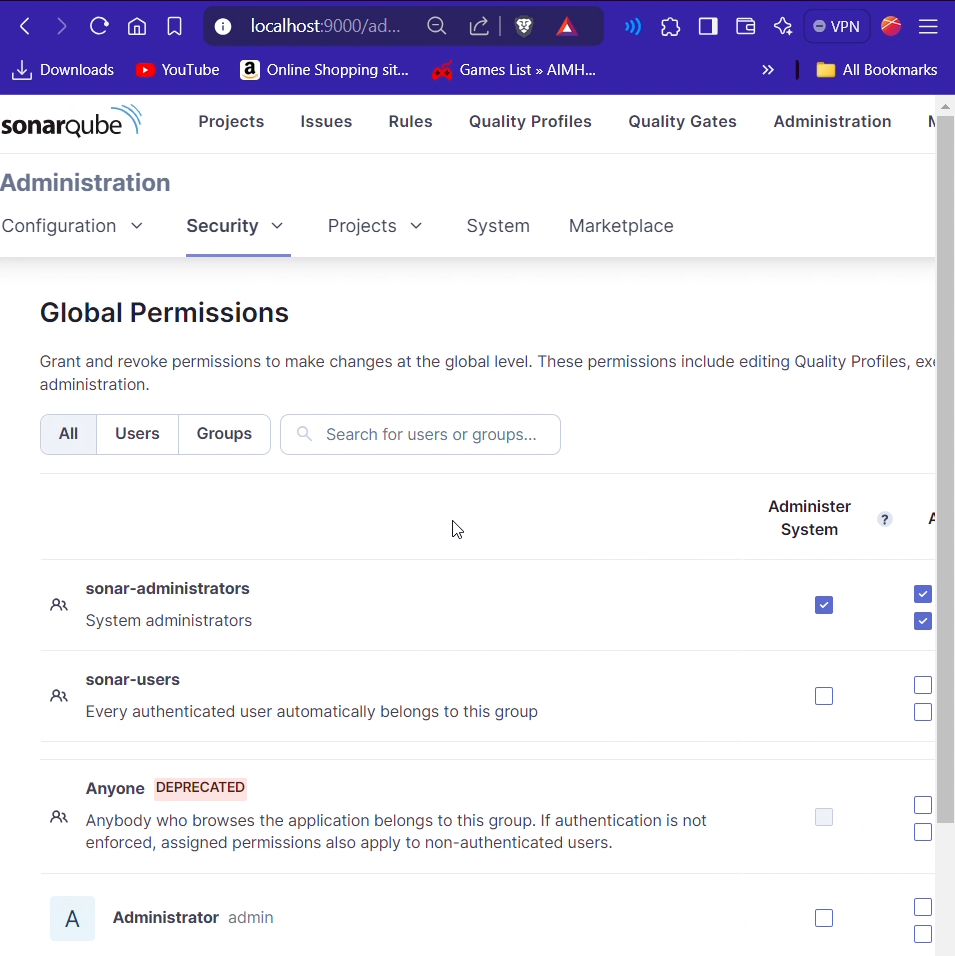
1. In your Jenkins job configuration, add the SonarQube server details under **Post-build Actions**.
2. Enter the SonarQube URL and authentication token.

### 

### **Summary of Installation Steps for Sonar Scanner 6.1.0.4477**

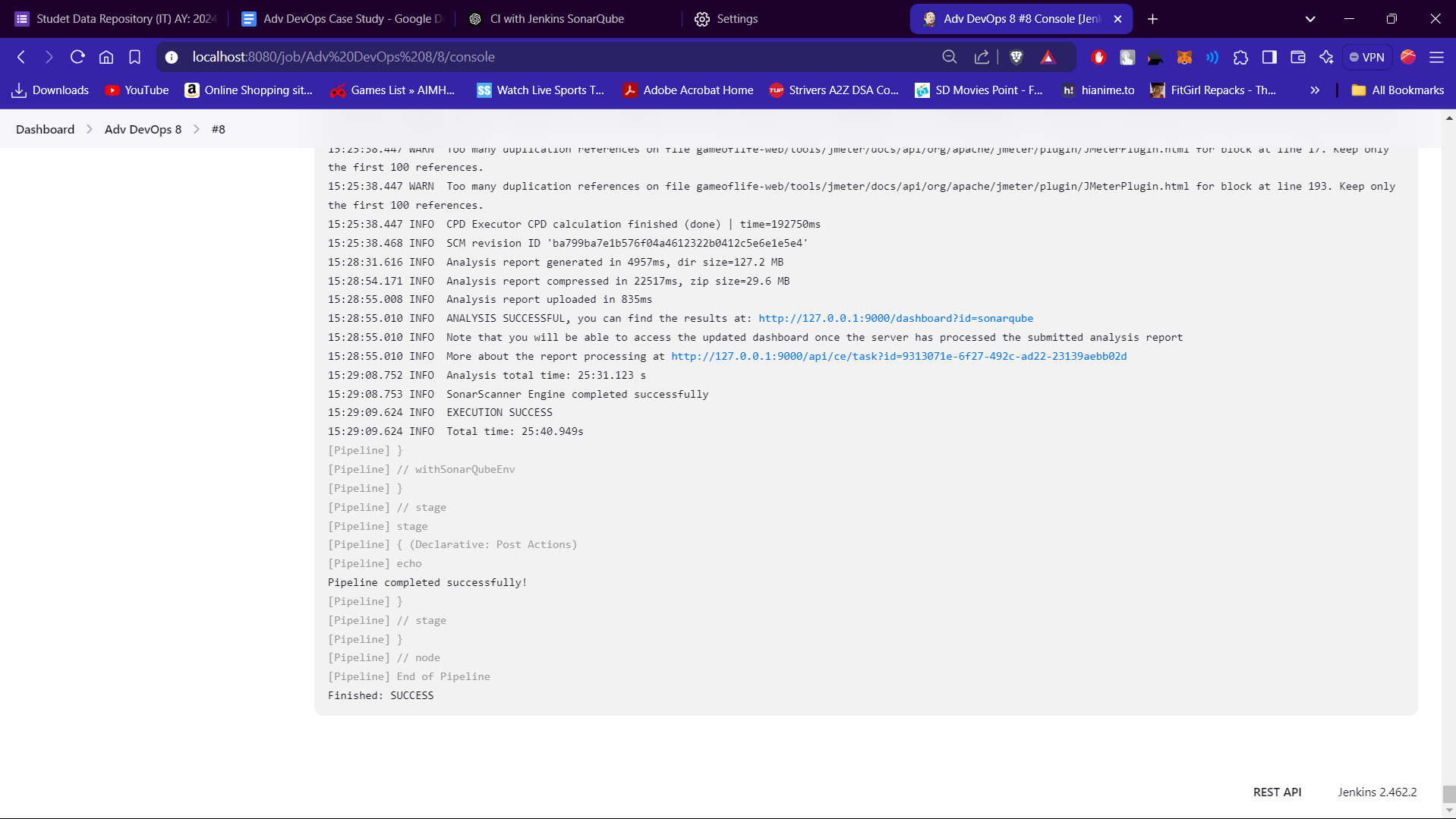
1. **Connect to the EC2 Instance**: Use SSH to connect to your Amazon EC2 instance.
2. **Download Sonar Scanner**:
3. Use wget to download the Sonar Scanner from the official source:  
     
   wget https://binaries.sonarsource.com/Distribution/sonar-scanner-cli/sonar-scanner-cli-6.1.0.4477-linux-x64.zip
4. **Install Required Packages**:
5. Ensure you have unzip installed  
   sudo apt-get update
6. sudo apt-get install unzip
7. **Unzip the Downloaded File**:
8. Unzip the downloaded .zip file:  
   unzip sonar-scanner-cli-6.1.0.4477-linux-x64.zip
9. **Move the Directory**:
10. Move the extracted directory to /opt  
    sudo mv sonar-scanner-6.1.0.4477-linux-x64 /opt/sonar-scanner
11. **Verify the Installation**:
12. Check if the installation was successful by running  
    /opt/sonar-scanner/bin/sonar-scanner --version
13. You should see output confirming the installed version (6.1.0.4477).
14. **Configure the Environment** (Optional):
15. Add the Sonar Scanner to your PATH by modifying your .bashrc or .bash\_profile  
    echo 'export PATH=$PATH:/opt/sonar-scanner/bin' >> ~/.bashrcsource ~/.bashrc

**

**

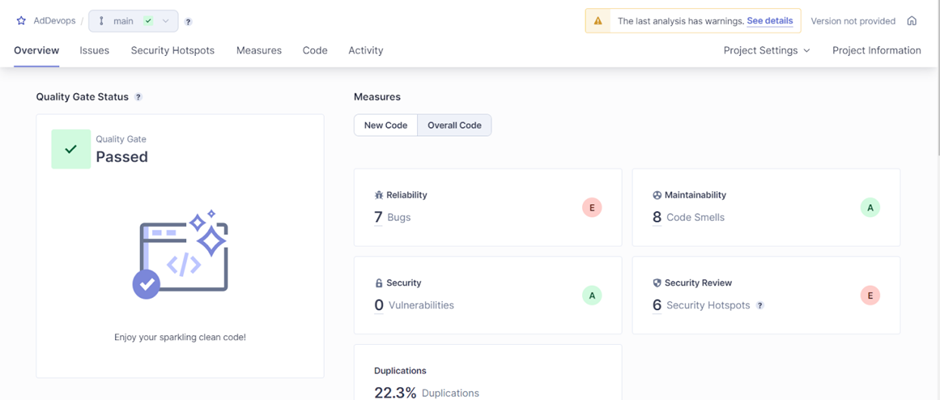
### **8. Press Build Now**

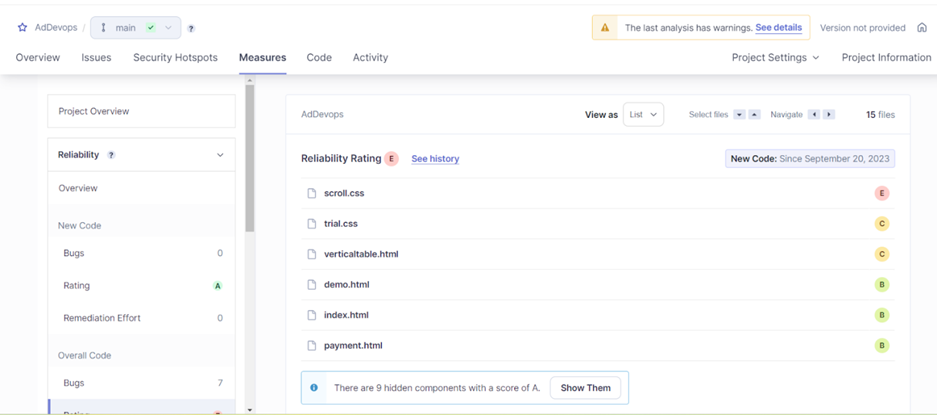
1. Save the Jenkins job configuration.
2. Click on **Build Now** to start the analysis.

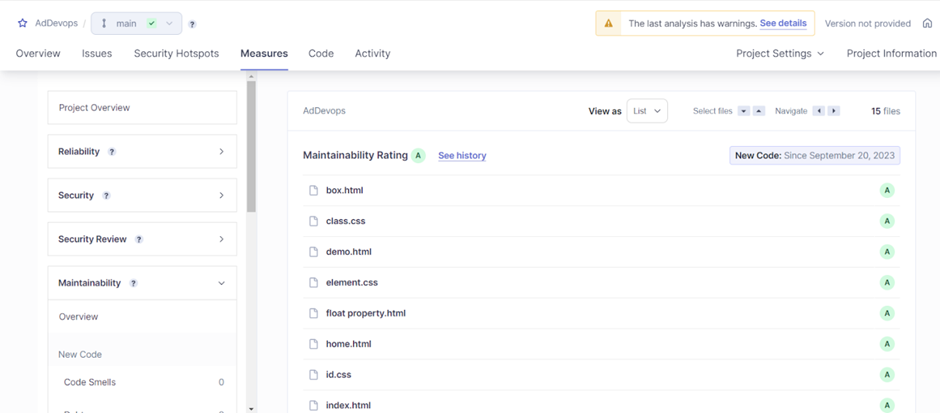
**

### **9. Look for Stats on SonarQube for Analysis**

1. Go back to SonarQube and navigate to the project dashboard.
2. Review the code quality reports and metrics generated from the analysis.

**

**

**

## **Conclusion**

In this case study, we successfully set up a Continuous Integration pipeline using Jenkins and SonarQube for static code analysis of a Java/Python application. By leveraging AWS Cloud9, we ensured a scalable and efficient environment for our development needs. The integration of these tools allows for early detection of code quality issues, leading to better software quality and reduced technical debt. This approach exemplifies the importance of CI/CD practices in modern software development.