Name :Riya Varyani **Div : D15A** **Roll No: 64**

1. Introduction :

Case Study Overview: This case study explores a DevOps-driven approach to software development and deployment. We focus on integrating Jenkins and SonarQube to implement a Continuous Integration (CI) pipeline with static code analysis. The primary objective is to ensure that code remains reliable and maintainable through continuous quality checks during integration. This study specifically applies to my third-year project, EventEase.

Key Features:

- **Jenkins:** Automates CI by triggering builds and static code analysis whenever code changes are made.
- **SonarQube:** Provides continuous feedback on code quality, reliability, and security.
- **AWS Cloud9:** Serves as the development environment for seamless integration.

2. Third-Year Project Integration: EventEase

EventEase is a comprehensive event management system that helps users plan and manage events, handle attendees, and schedule activities. The platform includes real-time notifications, ticketing, and a history of past events. By integrating Jenkins and SonarQube into EventEase, the following benefits are achieved:

- **Automated Testing:** Jenkins ensures that every code commit triggers tests, keeping the platform stable and bug-free.
- **Code Quality Monitoring:** SonarQube monitors code quality, enforcing clean, maintainable code.
- **Security Scanning:** SonarQube catches vulnerabilities before deployment, protecting sensitive event data.

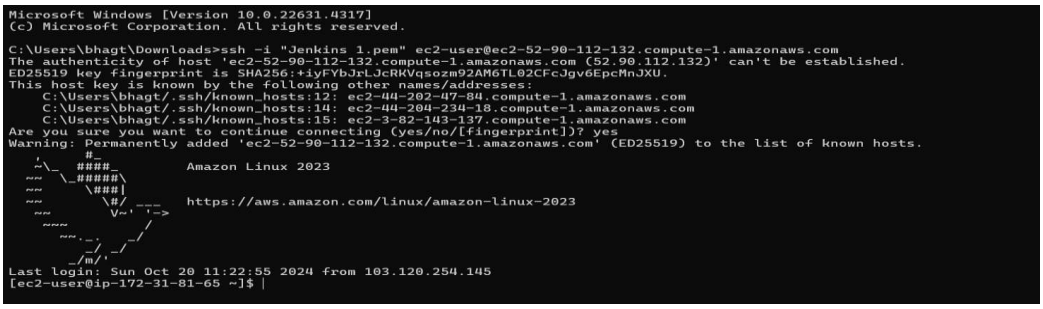
3. Demonstration :

Problem Statement: Jenkins on an EC2 instance

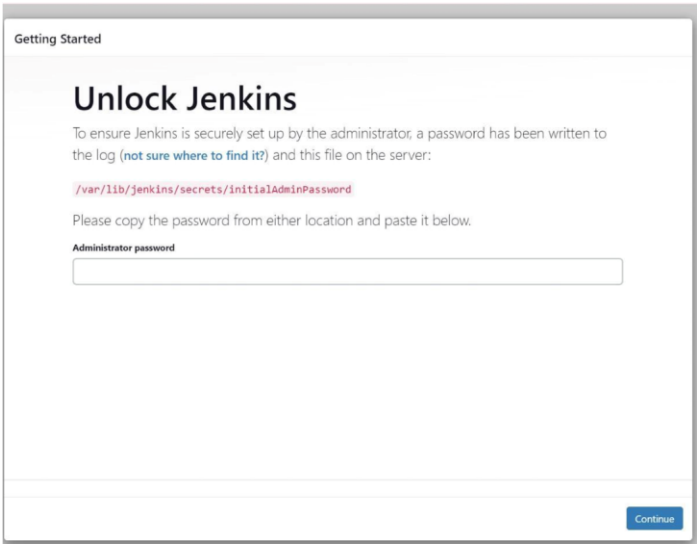
- Set up an AWS EC2 instance with Linux (t2.micro) and configure Jenkins to automate the CI pipeline.
- Execute commands to install and configure Jenkins.

4. Jenkins Configuration:

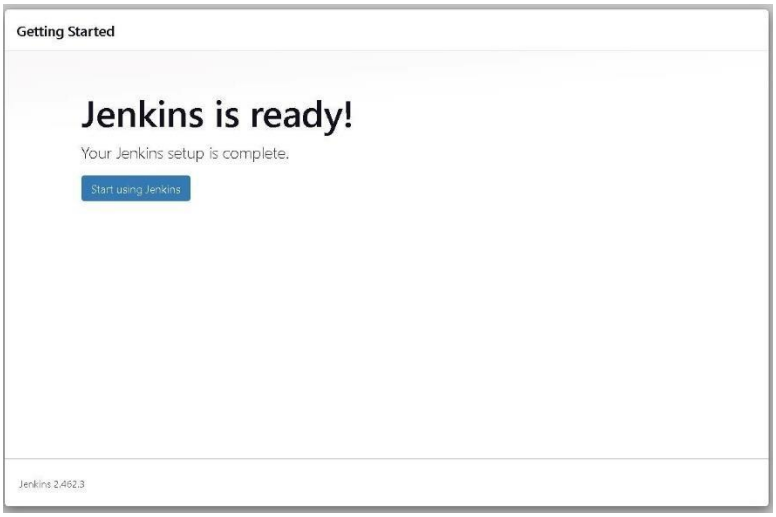
Name :Riya Varyani **Div : D15A** **Roll No: 64**



- **sudo yum update -y**
- **sudo wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhatstable/jenkins.repo**
- **sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key**
- **sudo yum install jenkins -y**
- **sudo systemctl enable jenkins**
- **sudo systemctl start jenkins**
- **sudo systemctl status jenkins**



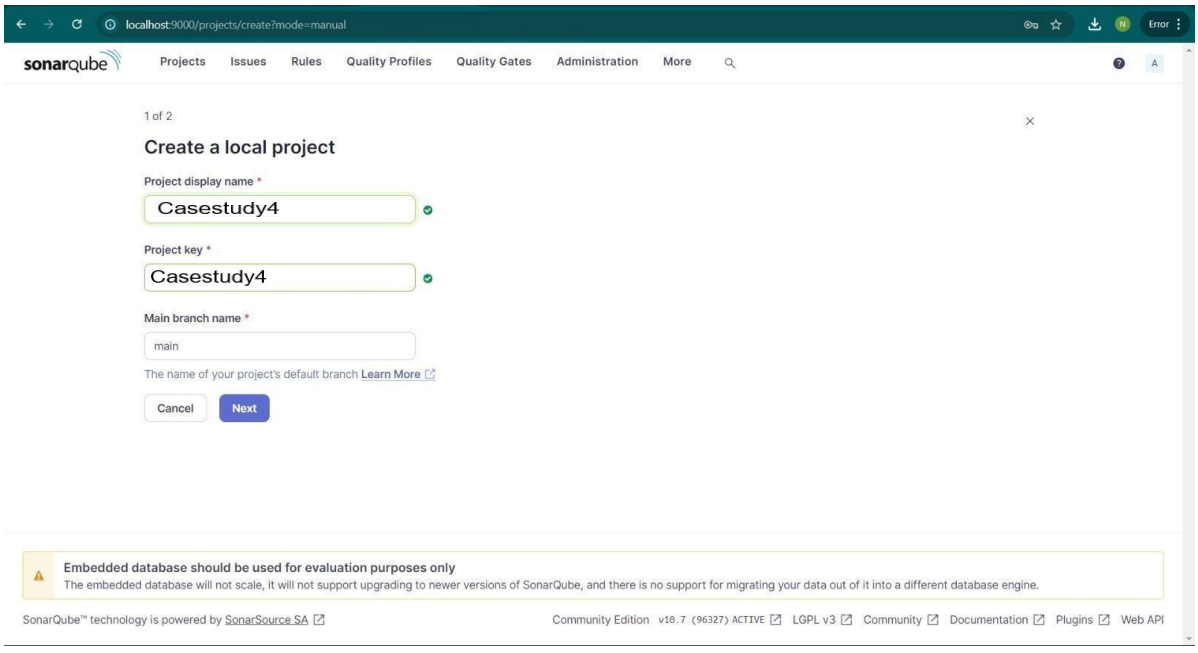
Name :Riya Varyani **Div : D15A** **Roll No: 64**



5. Task: SonarQube analysis of a Java/Python Project on Jenkins Pipeline:-

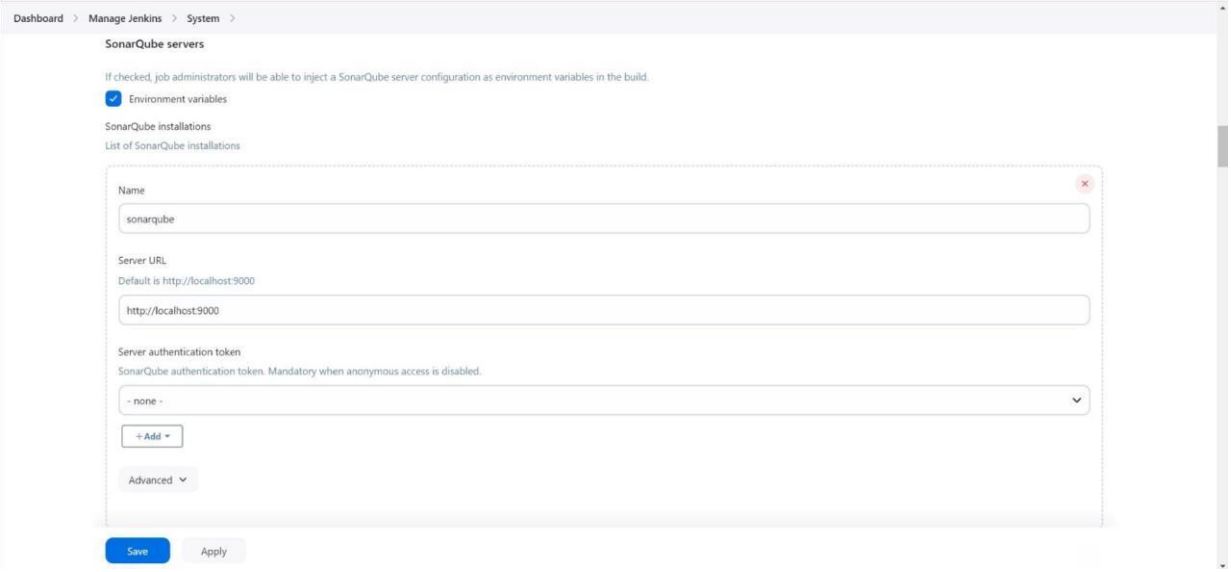
A] Sonarqube project:-
Python Project : https://github.com/piomin/sample-java-sonar

1) Create a sonarqube project named casestudy_24.

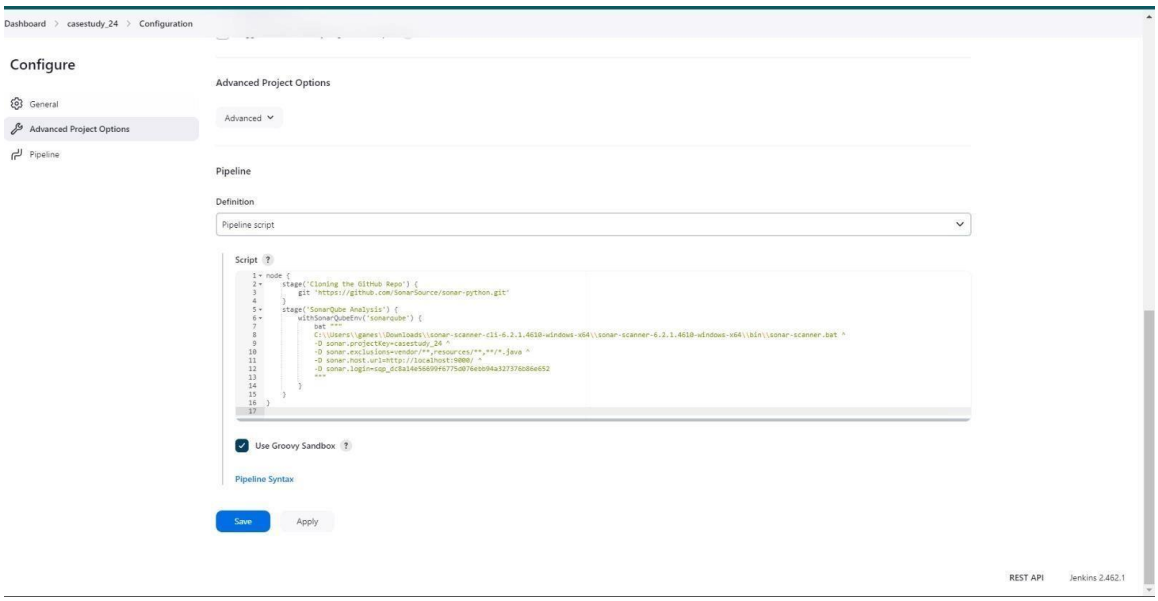


2) Configuration of SonarQube in Jenkins.

Name :Riya Varyani **Div : D15A** **Roll No: 64**



3) Deploy Code on Jenkins Pipeline :



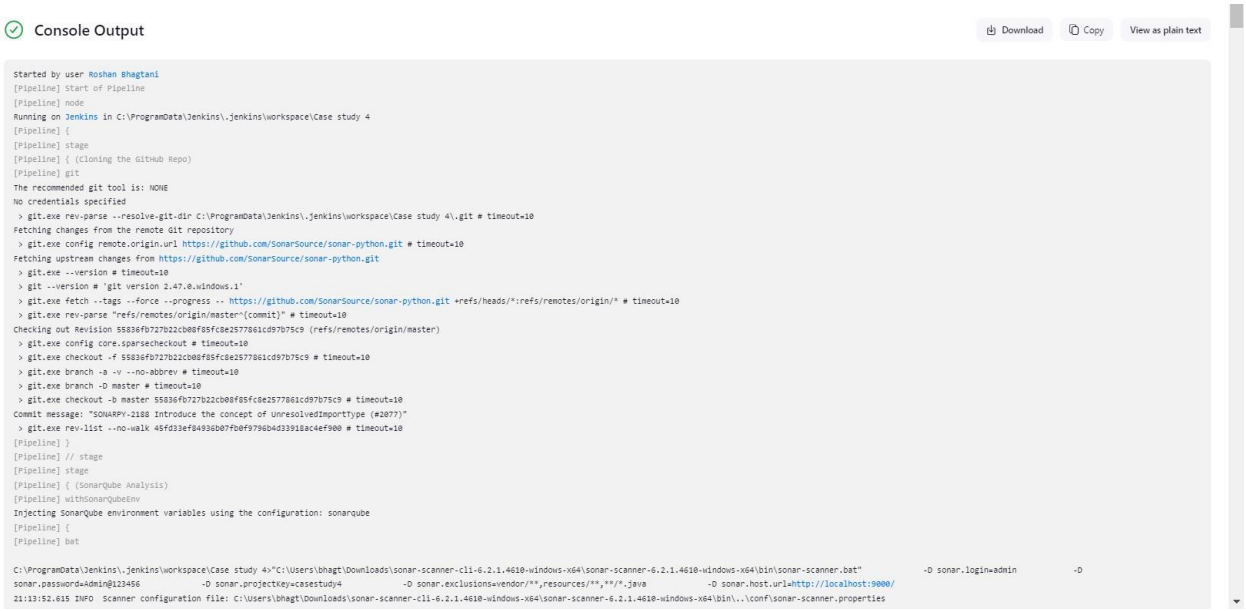
4)Pipeline Script :

Script:- node

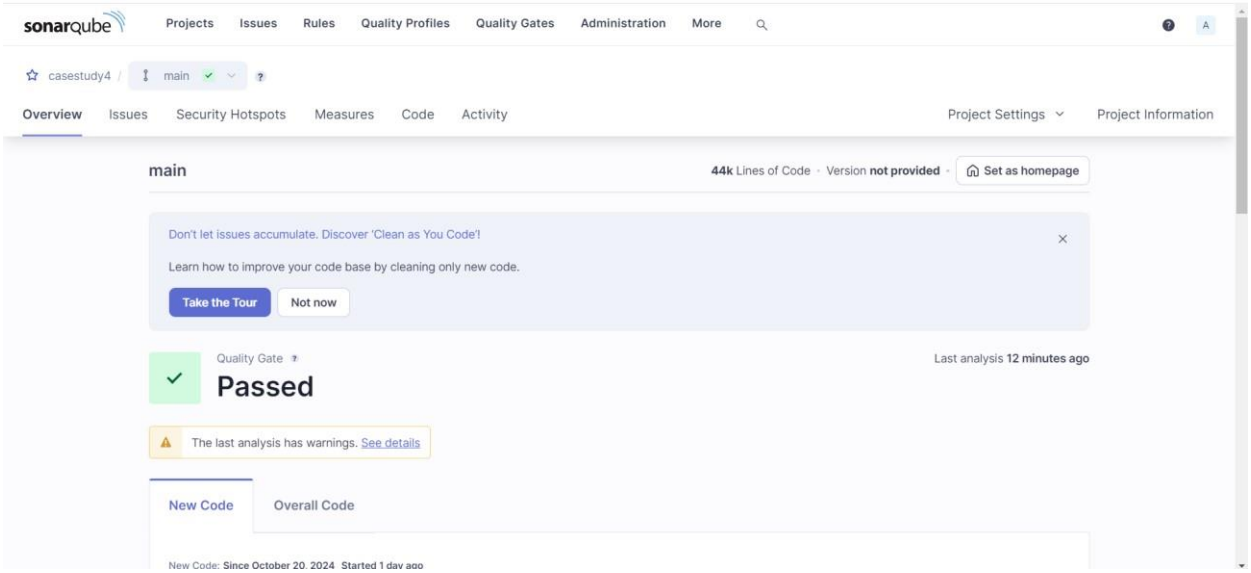
```
{
   stage('Cloning the GitHub Repo') { git
      'https://github.com/SonarSource/sonar-python.git'
   }
   stage('SonarQube Analysis') {
      withSonarQubeEnv('sonarqube') { bat
"C:\\bhagt\\nish\\Downloads\\sonar-scanner-cli-6.1.0.4477-windows-x64\\sonar-scanner-6.1.0
.4 477-windows-x64\\bin\\sonar-scanner.bat " +
            "-D sonar.login=admin " +
            "-D sonar.password=Admin@123456 " +
```

```
            "-D sonar.projectKey=casestudy_24 " +
            "-D sonar.exclusions=vendor/**,resources/**,**/*.java
            " + "-D sonar.host.url=http://localhost:9000/"
        }
    }
}
```

5)Open Console Output on Jenkins to check whether the Output is Success or not.



6)SonarQube assessment(Analysis):



Thus, the Python project was successfully analyzed with SonarQube.

6. Conclusion :

This case study highlights the successful integration of Jenkins and SonarQube to automate continuous integration and static code analysis for Python and Java projects. This CI pipeline enhances code quality and ensures compliance with security standards before deployment, fostering continuous improvement and accountability within the development team. Ultimately, this integration reduces vulnerabilities and streamlines the software delivery lifecycle.

enhances code quality and ensures compliance with security standards before deployment, fostering continuous improvement and accountability within the development team. Ultimately, this integration reduces vulnerabilities and streamlines the software delivery lifecycle.