Name: Manav Jawrani

Roll No.: 19

Subject: Advanced DevOps

Experiment No.: 8

Experiment 8.

Aim: Create a Jenkins CICD pipeline with Sonar Qube / Gitlab integration to perform a static analysis of code to detect bugs, code smells and security vulnerabilities on a sample web / Java / python application.

Theory:

- What is pipeline?

A pipeline is a concept that introduces a series of events or tasks that are connected in a sequence to make quick software releases. For example, there is a task, that task has got five different stages and each stage has got some steps. All the steps in phase one have to be completed, to mark the latter stage to be completed.

- What is a CI/CD pipeline?

CI/CD pipeline refers to the Continuous Integration/ Continuous Delivery pipeline. The CI/CD pipeline acts as the backbone of DevOps approach. This pipeline is responsible for building codes, running tests and deploying software version. The pipeline executes the job in a defined manner by first coding it and then structuring it inside several blocks that may include several steps or tasks.

SAST with CI/CD Pipeline?

SAST tools analyze application source code to find security weaknesses or vulnerabilities that malicious content can exploit. SAST tools use a white-box approach to testing which analyzes the application from inside. Integrating SAST tools into a process is critical to building sustainable projects. SAST must be automated and must be integrated into the CI/CD toolset to improve efficiency, consistency and easy detection.

SAST can be applied to all stages of software development cycle and can catch both unintentional bugs and malicious tampering. Here is how SAST can contribute at each stage of development:

1. Initial build
2. Staging and acceptance
3. Production Deployment

FOR EDUCATIONAL USE

Sundaram

Implementation:

Prerequisites:

1. Docker Installed

Download from here: https://www.docker.com

2. Jenkins

Download from here: https://www.jenkins.io/download/

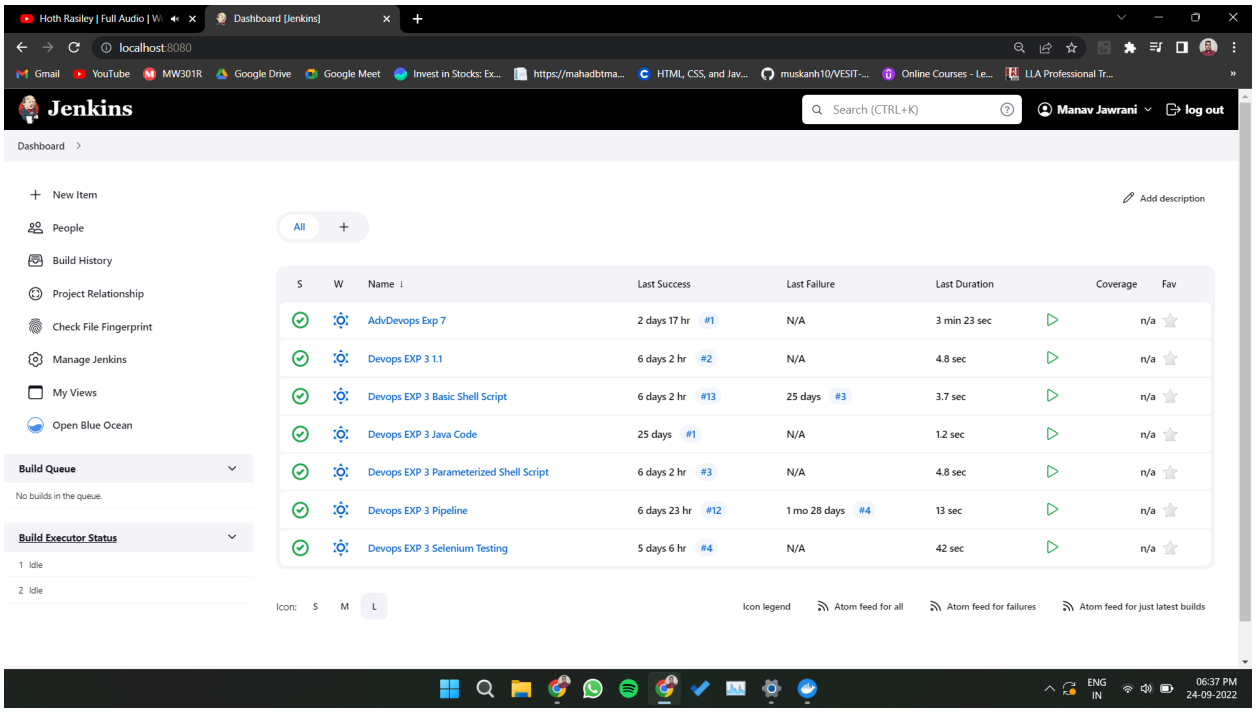
3. SonarQube Docker image.

Create image using this command:

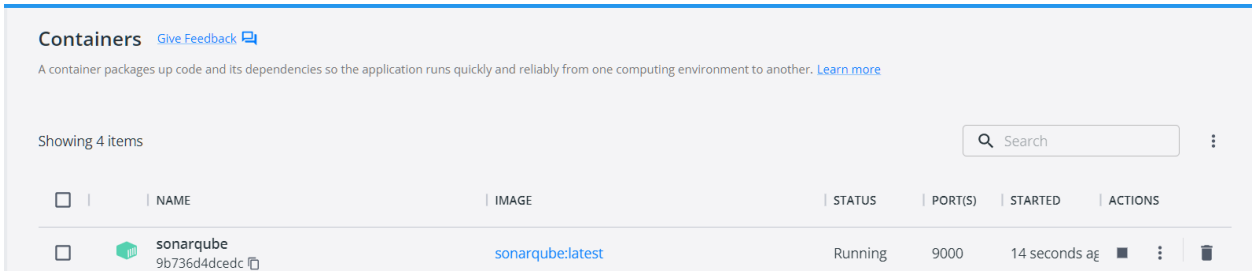
docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest

Steps

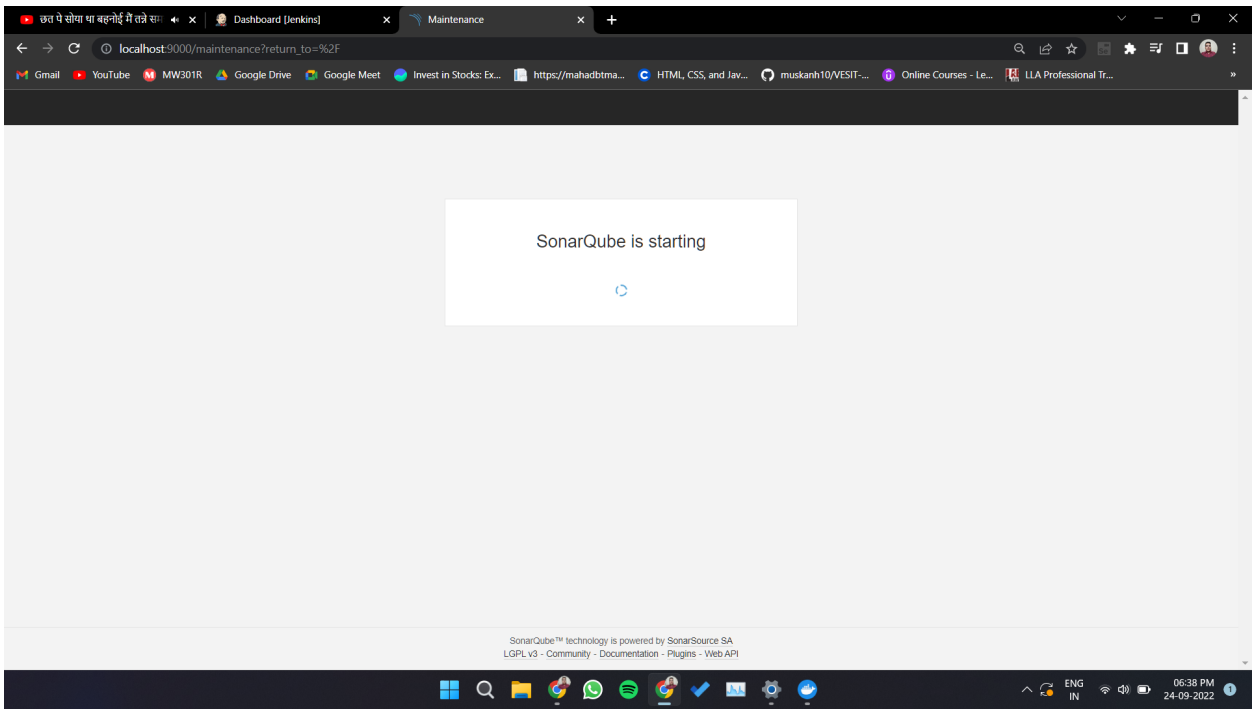
Step 1: Open up Jenkins Dashboard on localhost, port 8080 or whichever port it is at for you**.**



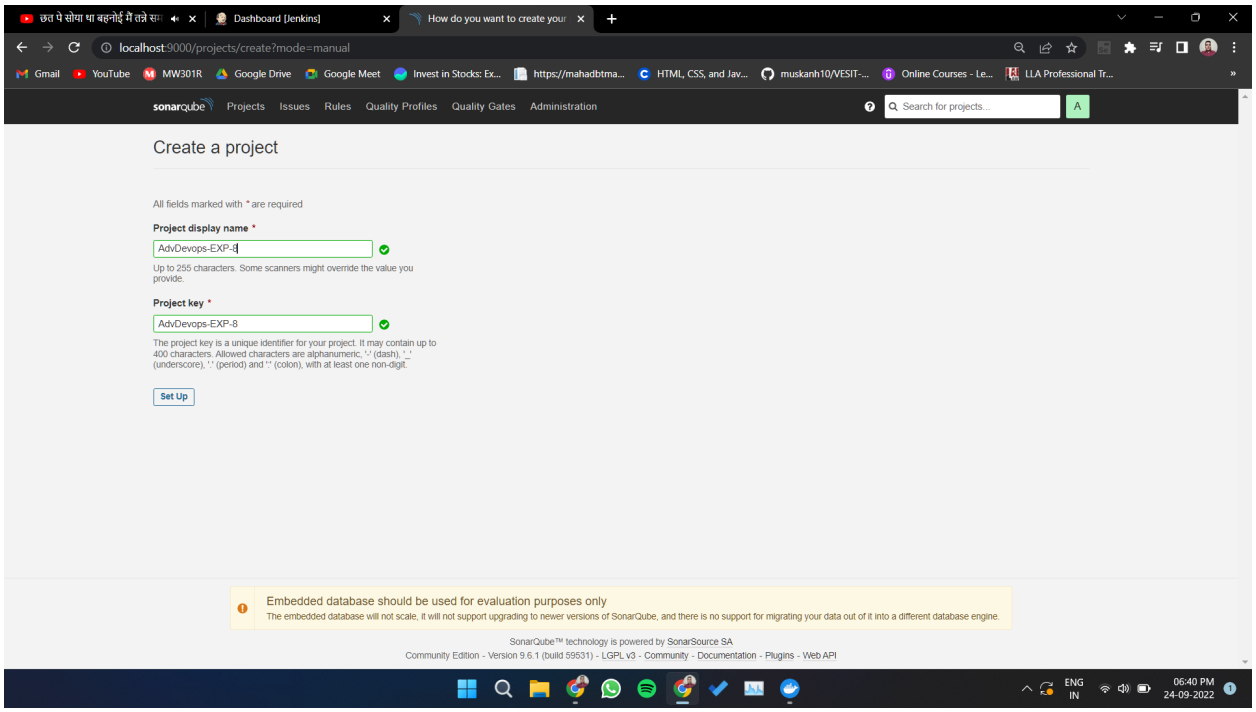
Step 2: Start the Docker Container in which the SonarQube image is stored.



Step 3: Once the container is up and running, you can check the status of SonarQube at localhost port 9000.

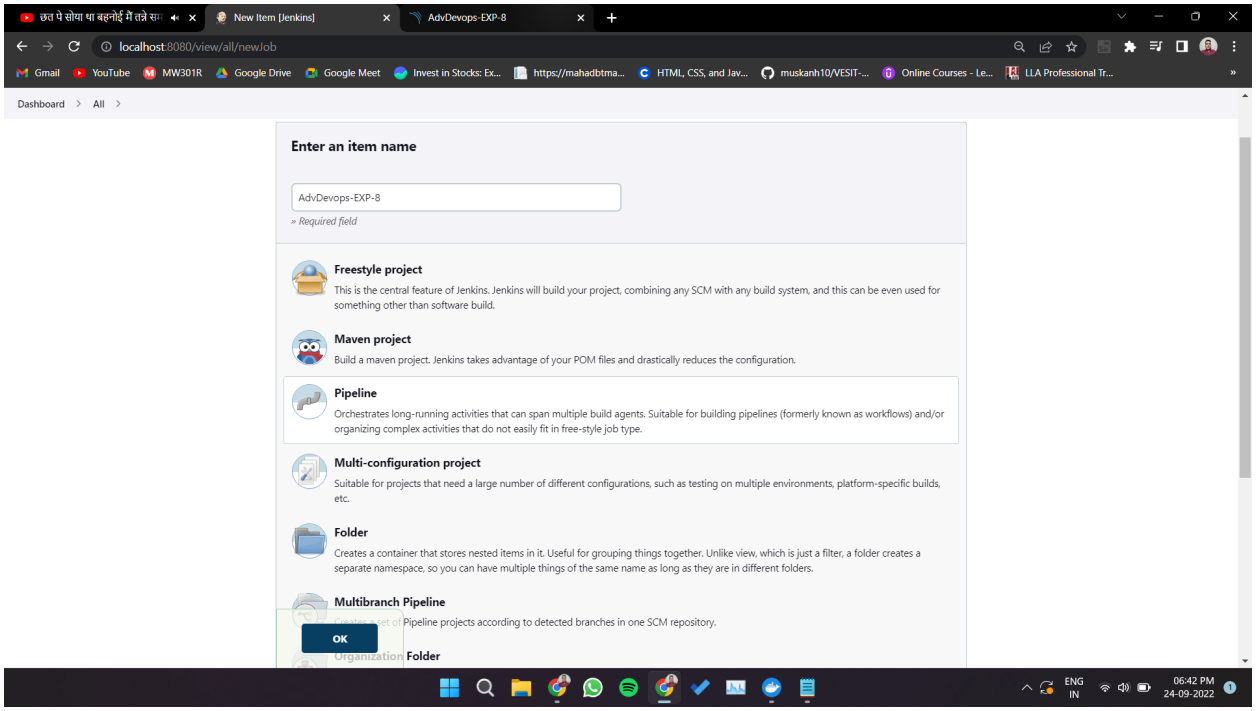


Step 4: Login to SonarQube using username and password and then create a manual project in SonarQube with the name **“AdvDevops-EXP-8”**



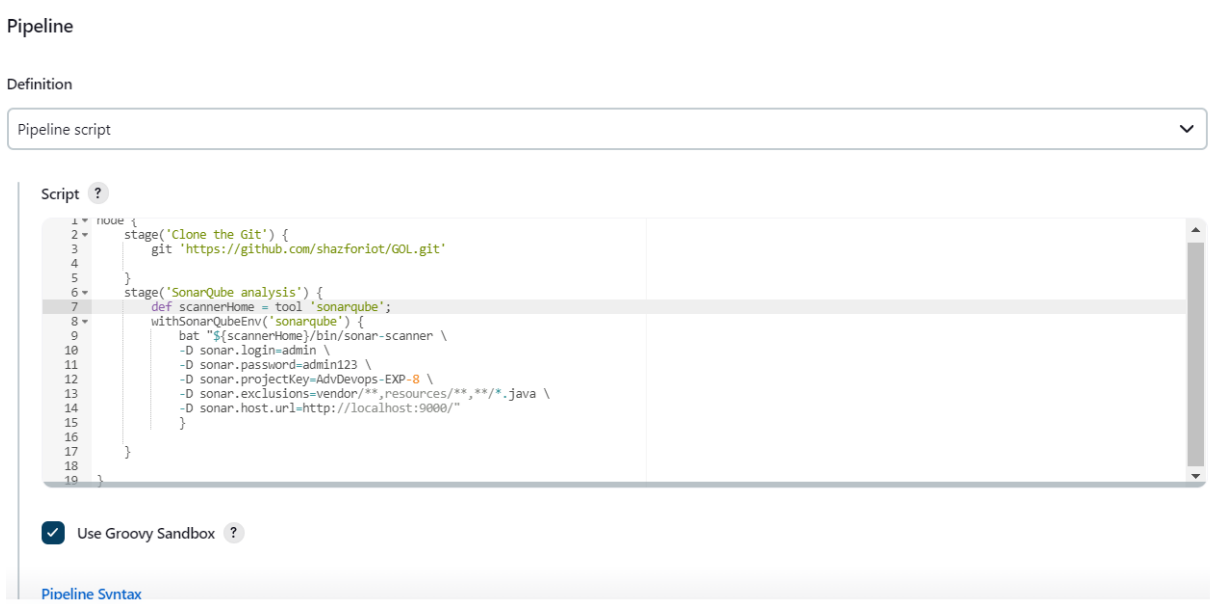
Setup the project and come back to Jenkins Dashboard.

Step 5: Create a New Item in Jenkins, choose **Pipeline**.



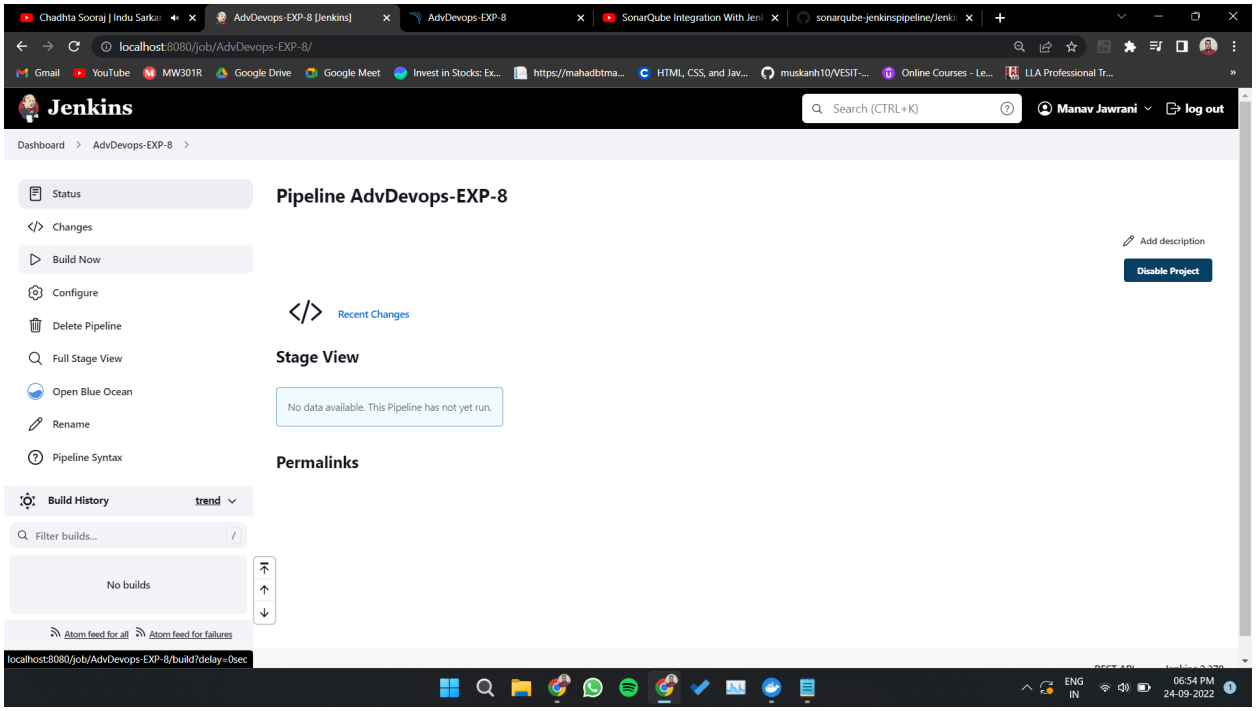
Step 6: Under Pipeline Script, enter the following:

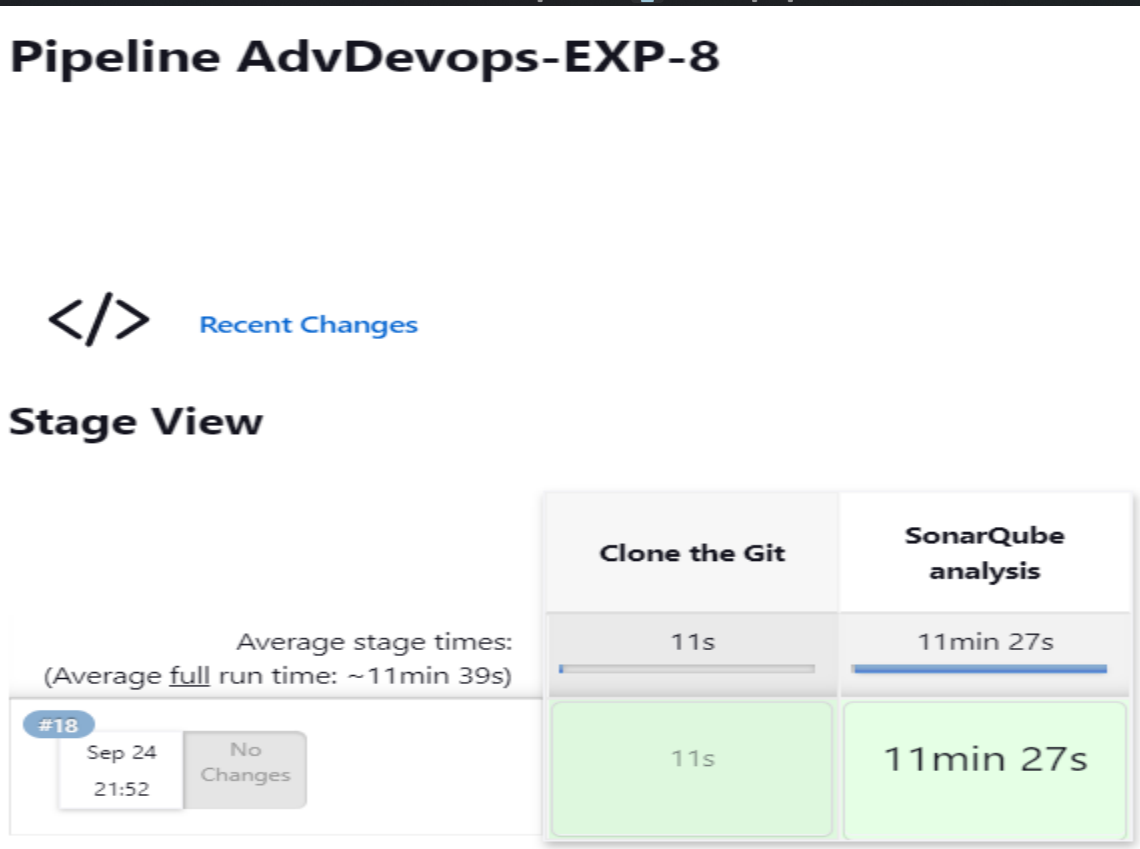
```
node {
    stage('Clone the Git') {
        git 'https://github.com/shazforiot/GOL.git'

    }
    stage('SonarQube analysis') {
        def scannerHome = tool 'sonarqube';
        withSonarQubeEnv('sonarqube') {
            bat "${scannerHome}/bin/sonar-scanner \
            -D sonar.login=*your username* \
            -D sonar.password=*your password* \
            -D sonar.projectKey=*your projectkey* \
            -D sonar.exclusions=vendor/**,resources/**,**/*.java \
            -D sonar.host.url=http://localhost:9000/"
            }

    }

}
```



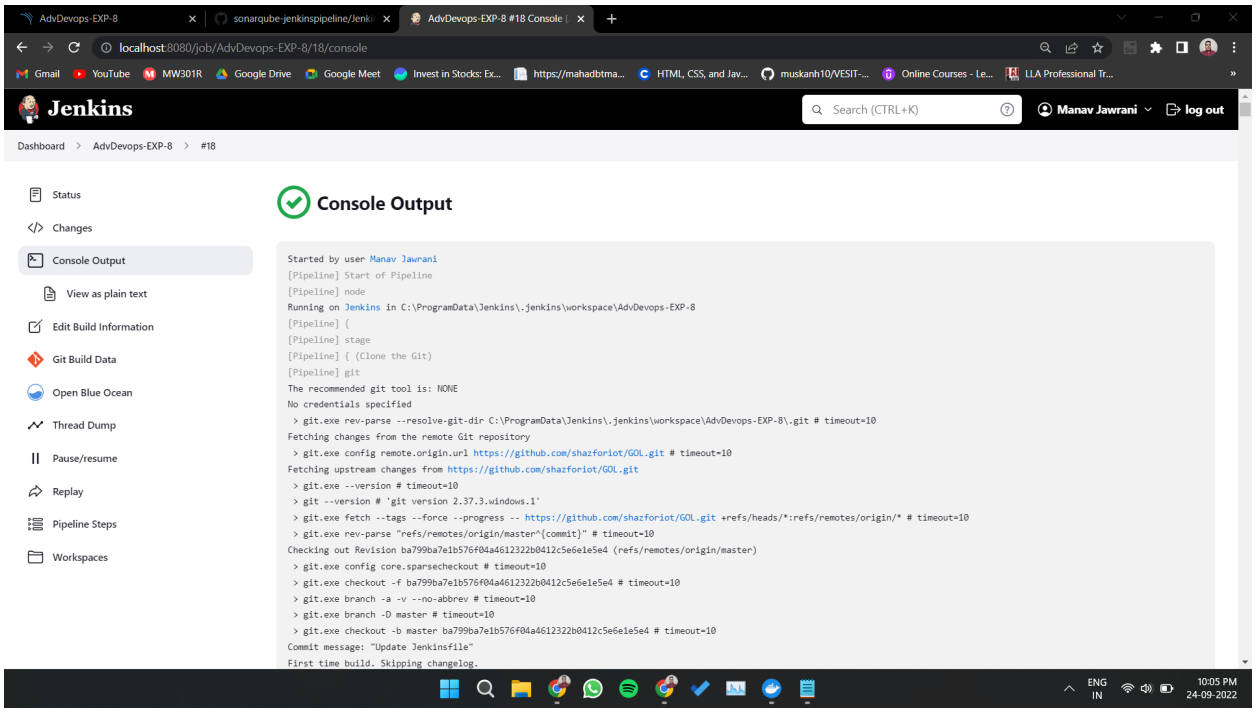
It is a java sample project which has a lot of repetitions and issues that will be detected by SonarQube.

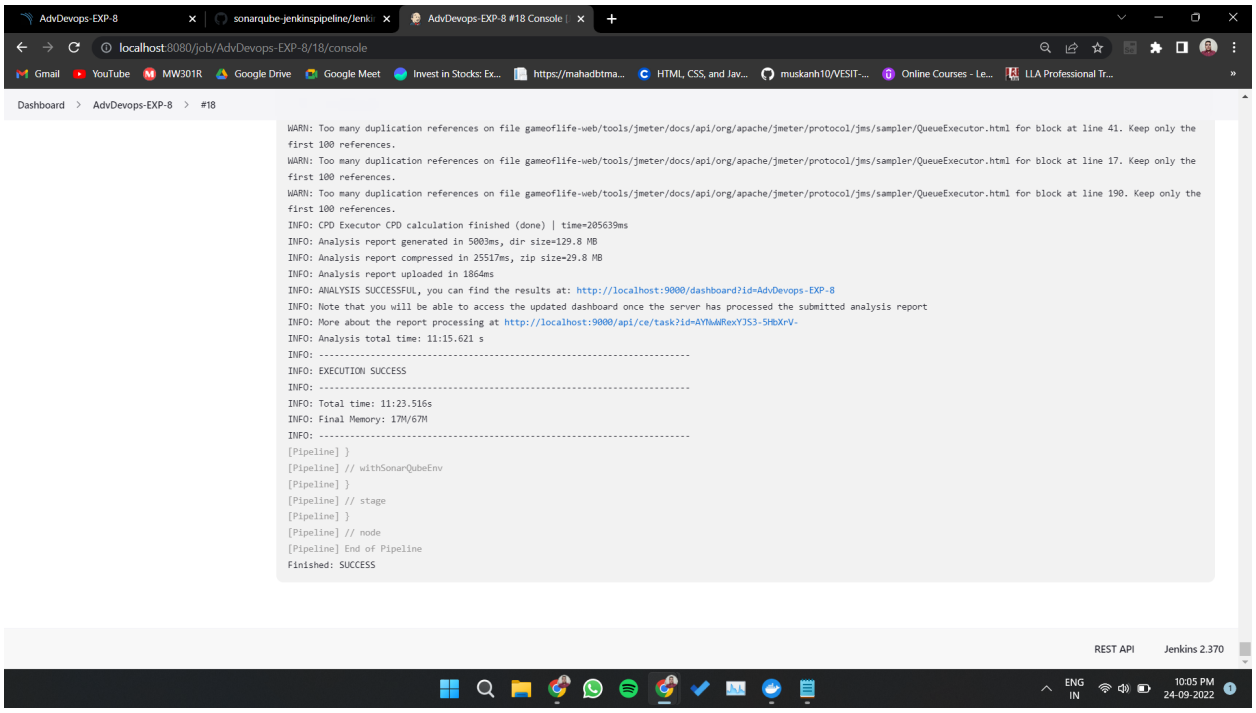
Step 7: Run The Build.



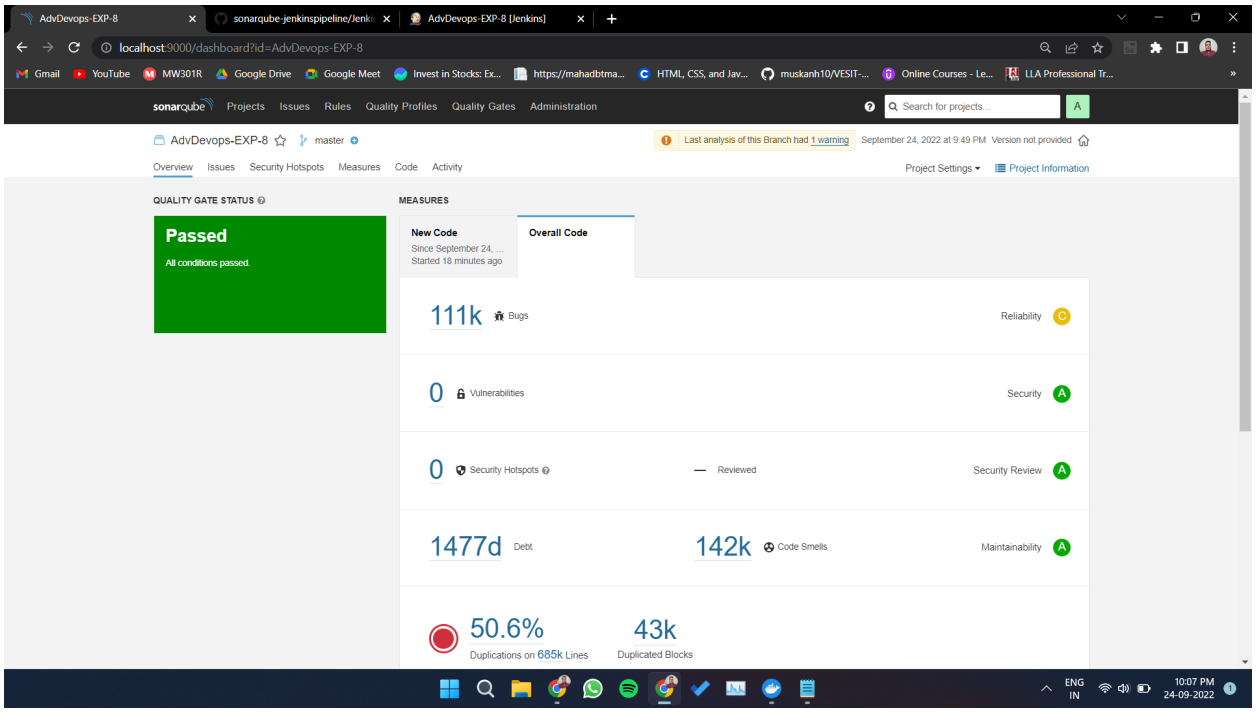


Step 8: Check the console output once the build is complete.





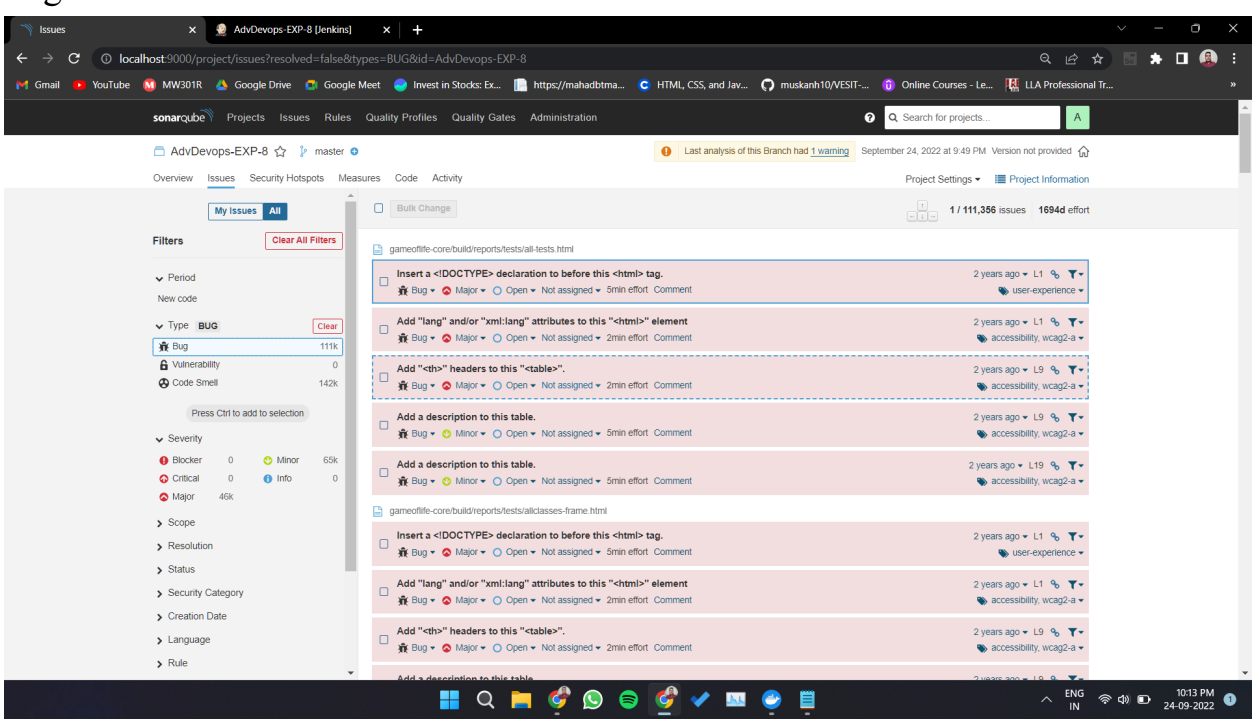
Step 9: After that, check the project in SonarQube.



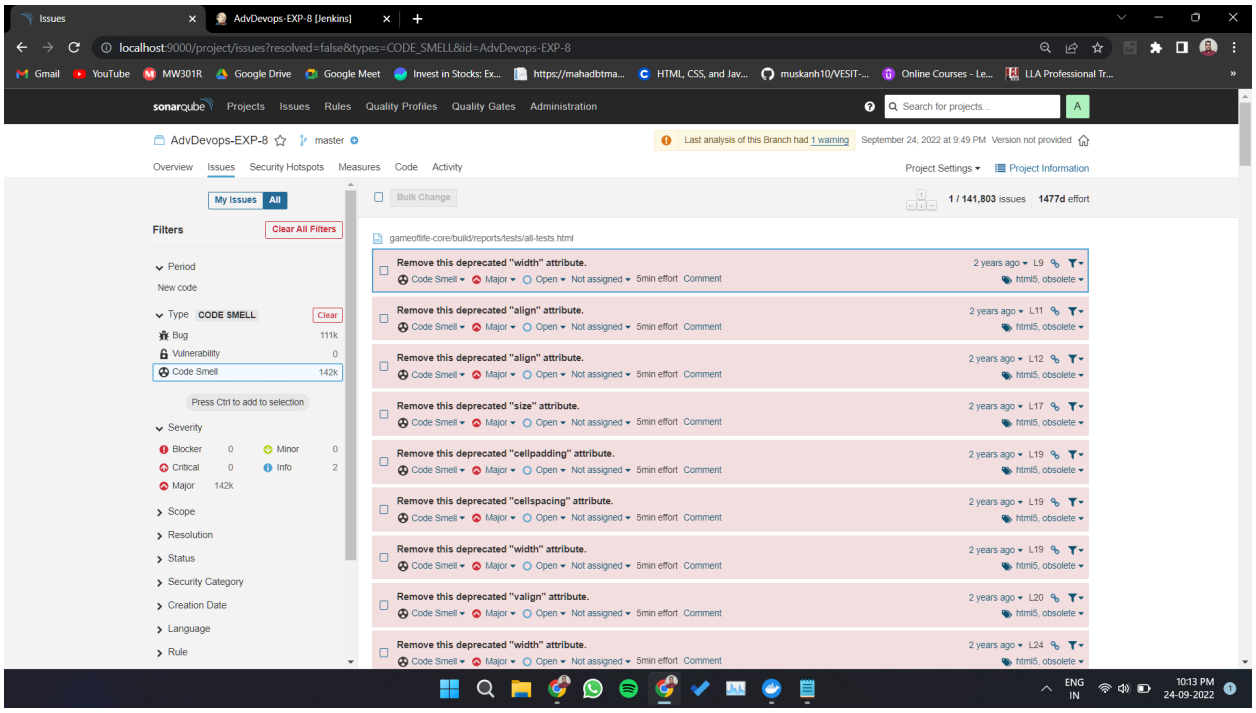
Under different tabs, check all different issues with the code.

Step 10: Code Problems

Bugs:



Code Smell:


Issues
AdvDevops-EXP-8 [Jenkins]
localhost:9000/project/issues?resolved=false&types=CODE_SMELL&id=AdvDevops-EXP-8
sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
AdvDevops-EXP-8
master
Last analysis of this Branch had 1 warning
September 24, 2022 at 9:49 PM
Version not provided
Overview
Issues
Security Hotspots
Measures
Code
Activity
Project Settings
Project Information
My Issues
All
Bulk Change
1 / 141,803 issues
1477d effort
Filters
Clear All Filters
Period
New code
Type
CODE SMELL
Clear
Bug
111k
Vulnerability
0
Code Smell
142k
Press Ctrl to add to selection
Severity
Blocker
0
Minor
0
Critical
0
Info
2
Major
142k
Scope
Resolution
Status
Security Category
Creation Date
Language
Rule
gameoflife-core/build/reports/tests/all-tests.html
Remove this deprecated "width" attribute.
Remove this deprecated "align" attribute.
Remove this deprecated "align" attribute.
Remove this deprecated "size" attribute.
Remove this deprecated "cellpadding" attribute.
Remove this deprecated "cellspacing" attribute.
Remove this deprecated "width" attribute.
Remove this deprecated "valign" attribute.
Remove this deprecated "width" attribute.
Code Smell
Major
Open
Not assigned
5min effort
Comment
2 years ago
L9
L11
L12
L17
L19
L19
L20
L24
html5, obsolete
ENG
IN
10:13 PM
24-09-2022

Conclusion:

As we have already demonstrated how to integrate Jenkins previously, with this experiment we will now move forward and build a pipeline that will aid in SAST analysis. The pipeline will investigate, review and produce a static analysis report using the Sonar Qube web application's built-in features. We discovered bugs, code smells in the sample code, and some duplications in the code after executing the pipeline. At the conclusion, we learned how to use SonarQube tool going forward to analyse code and aid in the creation of better software in future.