



Vivekanand Education Society's Institute of Technology

An Autonomous Institute Affiliated to University of Mumbai,, Approved by AICTE & Recognized by Govt. of Maharashtra
Hashu Advani Memorial Complex, Collector Colony, Chembur East, Mumbai - 400074.

Department of Information Technology

A.Y. 2024-25

Advance DevOps Lab Experiment 07

Aim: To understand Static Analysis SAST process and learn to integrate Jenkins SAST to SonarQube/GitLab.

Roll No.	22
Name	Sarthak Harade
Class	D15B
Subject	Advance DevOps Lab
LO Mapped	LO1: To understand the fundamentals of Cloud Computing and be fully proficient with Cloud based DevOps solution deployment options to meet your business requirements. LO4: To identify and remediate application vulnerabilities earlier and help integrate security in the development process using SAST Techniques.
Grade:	

AIM : To understand Static Analysis SAST process and learn to integrate Jenkins SAST to SonarQube/GitLab.

THEORY :

What is SAST?

Static Application Security Testing (SAST) is a method that analyzes source code to find security weaknesses before the code is compiled. This is often called white box testing.

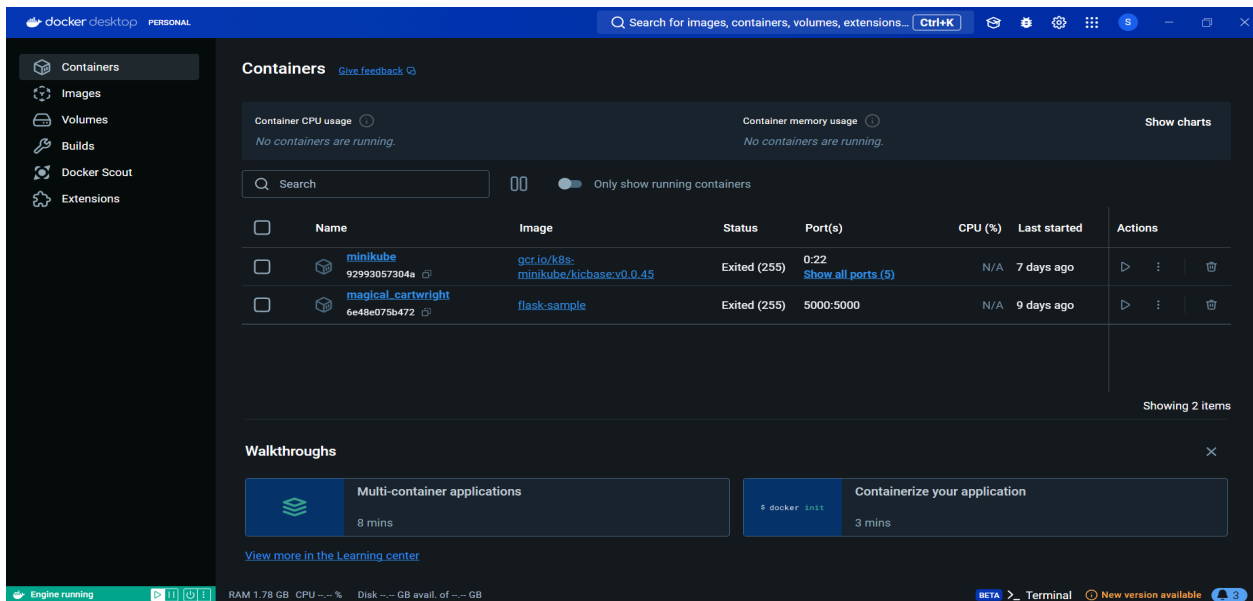
What Problems Does SAST Solve?

SAST occurs early in the Software Development Life Cycle (SDLC) and does not require a working application. It helps developers find and fix vulnerabilities during development, preventing issues from reaching the final product. SAST tools provide real-time feedback, allowing developers to address problems before moving on. They also give visual representations of issues and highlight the exact locations of vulnerabilities, guiding developers on how to fix them without needing extensive security knowledge. It's crucial to run SAST tools regularly, such as during daily builds or code releases.

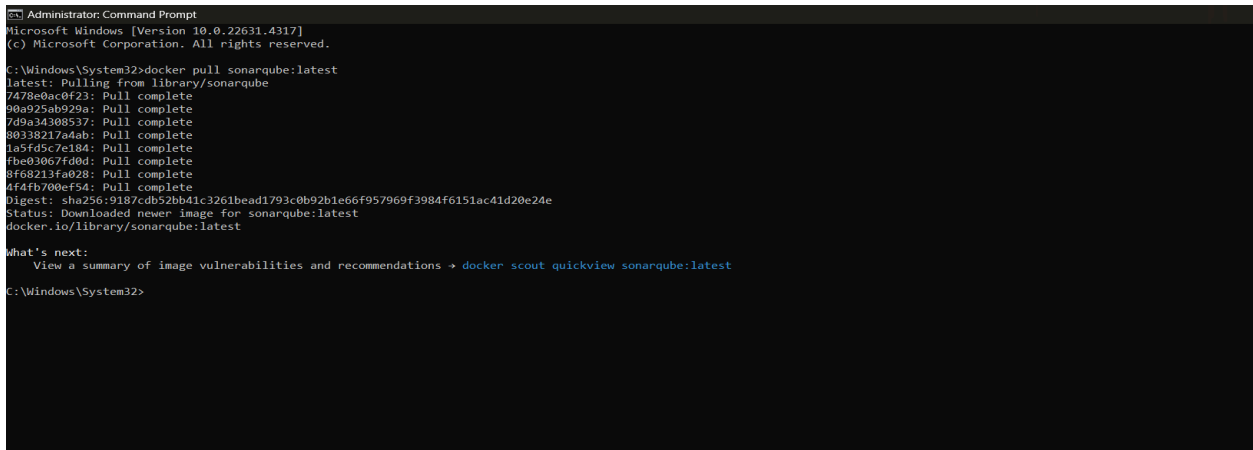
Key Steps to Run SAST Effectively:

1. Choose the Right Tool: Select a SAST tool that supports the programming languages and frameworks you use.
2. Set Up the Infrastructure: Handle licensing, access control, and resources needed to deploy the tool.
3. Customize the Tool: Fine-tune the tool to meet your organization's needs, reducing false positives and adding rules for better vulnerability detection.
4. Onboard Applications: Prioritize scanning high-risk applications first, then gradually onboard all applications for regular scans.
5. Analyze Results: Review the scan results, remove false positives, and ensure issues are tracked for timely fixing.
6. Provide Governance and Training: Ensure development teams use the tools properly and integrate SAST into the application development process.

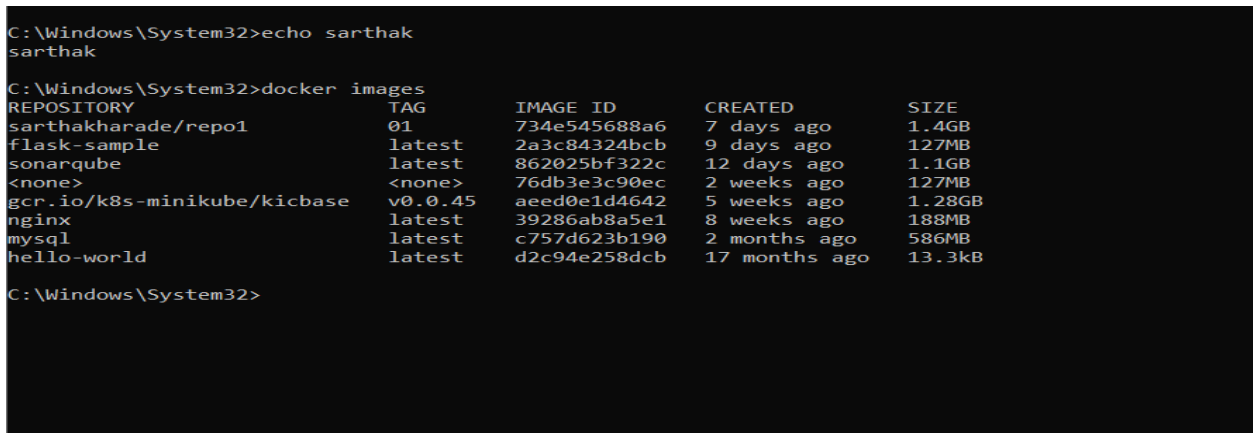
Ensure Docker is running



Open Command Prompt or PowerShell and execute the following command to pull the SonarQube image from Docker Hub. `docker pull sonarqube:latest`



Verify that the image has been downloaded successfully by running. `docker images`



Execute the following command in your terminal to run the SonarQube Container. `docker run -d --name sarthak-sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest`

```
C:\Windows\System32>docker run -d --name sarthak-sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
58d161dceb7a62e29395e0a78cd37190919c18345abdc1f9c792c9b72192cdd9

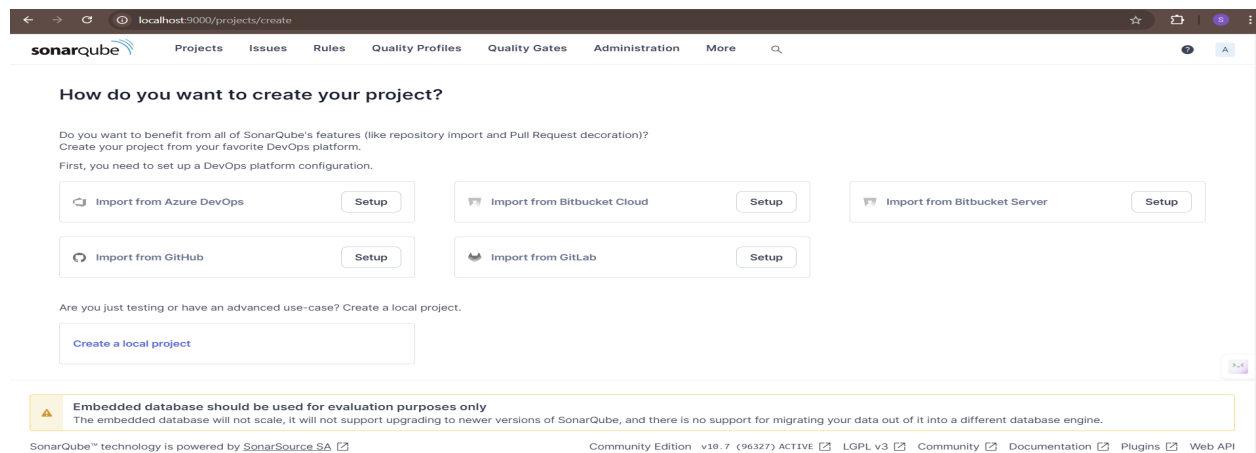
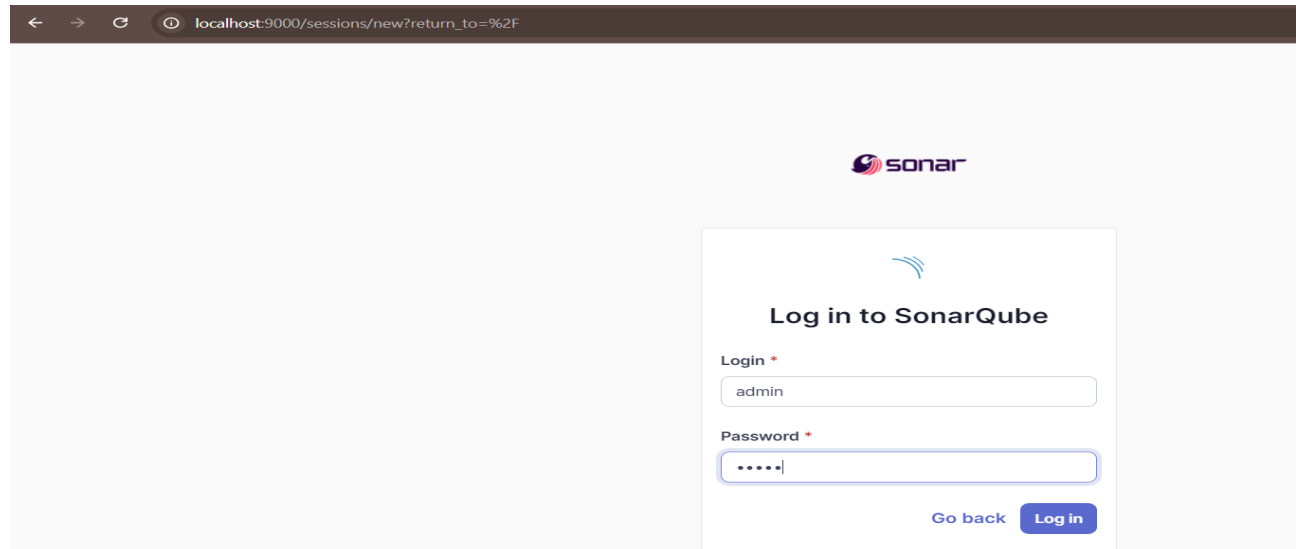
C:\Windows\System32>
```

```
C:\Windows\System32>docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                    NAMES
58d161dceb7a   sonarqube:latest  "/opt/sonarqube/dock..."  About a minute ago  Up About a minute  0.0.0.0:9000->9000/tcp    sarthak-sonarqube

C:\Windows\System32>
```

Login to SonarQube: Use the default credentials:

- Username: admin • Password: admin



Click on Create Project. Name the project and follow the prompts to set it up.

localhost:9000/projects/create?mode=manual

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration More

1 of 2

Create a local project

Project display name *

sarthak-sonarqube ✓

Project key *

sarthak-sonarqube ✓

Main branch name *

main

The name of your project's default branch [Learn More](#)

Cancel Next

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration More

2 of 2

Set up project for Clean as You Code

The new code definition sets which part of your code will be considered new code. This helps you focus attention on the most recent changes to your project and follow the Clean as You Code methodology. Learn more: [Defining New Code](#)

Choose the baseline for new code for this project

☒ Use the global setting

Previous version

Any code that has changed since the previous version is considered new code.
Recommended for projects following regular versions or releases.

☐ Define a specific setting for this project

☐ Previous version

Any code that has changed since the previous version is considered new code.
Recommended for projects following regular versions or releases.

☐ Number of days

Any code that has changed in the last x days is considered new code. If no action is taken on a new issue after x days, this issue will become part of old code.

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration More

☆ sarthak-sonarqube / main

Overview Issues Security Hotspots Measures Code Activity Project Settings Project Information

Analysis Method

Use this page to manage and set-up the way your analyses are performed.

How do you want to analyze your repository?

With Jenkins

With GitHub Actions

With Bitbucket Pipelines

With GitLab CI

With Azure Pipelines

Other CI
SonarQube integrates with your workflow no matter which CI tool you're using.

Locally
Use this for testing or advanced use-case. Other modes are recommended to help you set up your CI environment.

Open Jenkins Dashboard: Go to <http://localhost:8080> in your web browser (or the port where Jenkins is running).

Jenkins Search (CTRL+K) Sarthak Harade log out

Dashboard

+ New Item

Build History

Project Relationship

Check File Fingerprint

Manage Jenkins

My Views

Build Queue

No builds in the queue.

Build Executor Status

Build-In Node

1 Idle

2 Idle

adityanode (offline)

Build History

All

S	W	Name	Last Success	Last Failure	Last Duration
✓	☁	exp6	15 days #12	19 days #10	8.2 sec
✗	☁	exp6 maven	15 days #10	9 days 8 hr #15	25 sec
✗	☁	newexp5	9 days 13 hr #5	9 days 8 hr #8	10 sec
✓	☀	newexp6	9 days 14 hr #4	N/A	2.7 sec
✓	☀	pro8	9 days 7 hr #3	9 days 11 hr #1	15 sec
✓	☀	project1	9 days 16 hr #9	23 days #3	20 sec

Icon: S M L

Manage Jenkins → Manage Plugins → Available tab, search for SonarQube Scanner. Check the box next to it and click Install without Restart.

Jenkins Search (CTRL+K) Sarthak Harade log out

Dashboard Manage Jenkins Plugins

Plugins

sonarqube scanner

Install

Install	Name	Released
<input checked="" type="checkbox"/>	SonarQube Scanner 2.17.2 External Site/Tool Integrations Build Reports This plugin allows an easy integration of SonarQube , the open source platform for Continuous Inspection of code quality.	7 mo 26 days ago

The screenshot shows the Jenkins 'Plugins' page. On the left, a sidebar lists 'Updates' (19), 'Available plugins', 'Installed plugins', 'Advanced settings', and 'Download progress' (selected). The main area is titled 'Download progress' and shows the status of various plugins. Under 'Preparation', it lists 'Checking internet connectivity', 'Checking update center connectivity', and 'Success'. Under 'SonarQube Scanner', it shows 'Success'. Under 'Loading plugin extensions', it shows 'Success'. At the bottom, there are two links: 'Go back to the top page' (with a note 'you can start using the installed plugins right away') and 'Restart Jenkins when installation is complete and no jobs are running' (with a checkbox).

Manage Jenkins → Configure System. Scroll down to the SonarQube Servers section and add a new SonarQube server.

- Name: Give it a name (e.g., SonarQube).
- Server URL: Enter `http://localhost:9000`.

The screenshot shows the 'Configure System' page in Jenkins, specifically the 'SonarQube servers' section. It includes a checkbox for 'Environment variables' which is checked. Below this, there is a section for 'SonarQube installations' with a list of installations. A new installation is being added with the following details: Name: 'sonarqube', Server URL: 'http://localhost:9000' (with a default value of 'http://localhost:9000' shown above the input), and Server authentication token: '- none -'. At the bottom, there are 'Save' and 'Apply' buttons.

Manage Jenkins → Global Tool Configuration. Find SonarQube Scanner and choose the latest version. Check the Install automatically option.

Dashboard > Manage Jenkins > Tools

SonarQube Scanner installations

Add SonarQube Scanner

☰ **SonarQube Scanner**

Name

☒ Install automatically ?

☰ **Install from Maven Central**


Version

Add Installer ▾

Add SonarQube Scanner

Save Apply

On the Jenkins dashboard, click on New Item. Enter a name for your project. Choose the Freestyle project and click OK.


 **Jenkins**


Dashboard > All > New Item


New Item


Enter an item name

Select an item type

 **Freestyle project**
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

 **Maven project**
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

 **Pipeline**
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

 **Multi-configuration project**
Suitable for projects that need a large number of different configurations, such as testing on multiple environment platform-specific builds, etc.

OK

In the project configuration, look for the Source Code Management section. Choose Git and enter the repository URL. https://github.com/shazforiot/MSBuild_firstproject.git

The screenshot shows the 'Configure' page for a SonarQube project. The left sidebar has a 'Configure' header and a list of sections: General, Source Code Management (selected), Build Triggers, Build Environment, Build Steps, and Post-build Actions. The main content area is titled 'Git' and contains the following fields:

- Repositories**: A section with a 'Repository URL' field containing 'https://github.com/shazforiot/MSBuild_firstproject.git'.
- Credentials**: A dropdown menu showing '- none -' and an '+ Add' button.
- Advanced**: A dropdown menu.
- Add Repository**: A button.
- Branches to build**: A section with a 'Branch Specifier (blank for 'any')' field containing '*/master'.

At the bottom of the configuration area are 'Save' and 'Apply' buttons.

Scroll down to the Build section. Click on Add build step and select Execute SonarQube Scanner.

The screenshot shows the 'Configure' page for a SonarQube project, specifically the 'Build Environment' section. The left sidebar has a 'Configure' header and a list of sections: General, Source Code Management, Build Triggers, Build Environment (selected), Build Steps, and Post-build Actions. The main content area is titled 'Build Environment' and contains a list of checkboxes:

- ☐ Provide Configuration files
- ☐ Add timestamps to the Console Output
- ☐ Inspect build log for published build scans
- ☐ Prepare SonarQube Scanner environment
- ☐ Terminate a build if it's stuck
- ☐ With Ant

Below this list is the 'Build Steps' section, which includes an 'Add build step' button and a dropdown menu. The dropdown menu is open, showing a list of build steps:

- Execute SonarQube Scanner (highlighted)
- Execute Windows batch command
- Execute shell
- Invoke Ant
- Invoke Gradle script
- Invoke top-level Maven targets
- Provide Configuration files
- Run with timeout
- Set build status to "pending" on GitHub commit

Enter Analysis Properties
sonar.projectKey=sonarqube

sonar.login=admin sonar.
password=admin
sonar.sources=.
sonar.host.url=http://localhost:9000

Dashboard > sarthak-sonarqube > Configuration

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps**
- Post-build Actions

Execute SonarQube Scanner

JDK ?
JDK to be used for this SonarQube analysis
(Inherit From Job)

Path to project properties ?

Analysis properties ?
sonar.projectKey=sonarqube
sonar.login=admin
sonar.password=admin
sonar.sources=.
sonar.host.url=http://localhost:9000

Additional arguments ?

JVM Options ?

Save Apply

Go to <http://localhost:9000/admin/permissions> and give the Admin user execute permissions.

localhost:9000/admin/permissions

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration More

Administration

Configuration Security Projects System Marketplace

Grant and revoke permissions to make changes at the global level. These permissions include editing Quality Profiles, executing analysis, and performing global system administration.

All Users Groups Search for users or groups...

	Administer System ?	Administer ?	Execute Analysis ?	Create ?
sonar-administrators System administrators	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> Quality Gates <input checked="" type="checkbox"/> Quality Profiles	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> Projects
sonar-users Every authenticated user automatically belongs to this group	<input type="checkbox"/>	<input type="checkbox"/> Quality Gates <input type="checkbox"/> Quality Profiles	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> Projects
Anyone DEPRECATED Anybody who browses the application belongs to this group. If authentication is not enforced, assigned permissions also apply to non-authenticated users.	<input type="checkbox"/>	<input type="checkbox"/> Quality Gates <input type="checkbox"/> Quality Profiles	<input type="checkbox"/>	<input type="checkbox"/> Projects
A Administrator admin	<input type="checkbox"/>	<input type="checkbox"/> Quality Gates <input type="checkbox"/> Quality Profiles	<input type="checkbox"/>	<input type="checkbox"/> Projects

4 of 4 shown

Save the project configuration and click Build Now

Dashboard > sarthak-sonarqube >

Status

</> Changes

Workspace

▶ Build Now

⚙️ Configure


🗑️ Delete Project

🌐 GitHub

🌊 SonarQube

✎️ Rename

✓ sarthak-sonarqube


Permalinks

- [Last build \(#5\), 37 sec ago](#)
- [Last stable build \(#5\), 37 sec ago](#)
- [Last successful build \(#5\), 37 sec ago](#)
- [Last failed build \(#4\), 6 min 3 sec ago](#)
- [Last unsuccessful build \(#4\), 6 min 3 sec ago](#)
- [Last completed build \(#5\), 37 sec ago](#)

Build History

trend ▾

Filter...

✓ #5

Oct 13, 2024, 8:44 AM

✗ #4

Oct 13, 2024, 8:38 AM

Click on the build number in the build history.

Jenkins

Search (CTRL+K)

🔔 🔔 🔒 🔴 👤 Sarthak Harade ▾

Dashboard > sarthak-sonarqube > #5

Status

</> Changes

📄 Console Output

📄 Edit Build Information

🗑️ Delete build '#5'

🕒 Timings

🔗 Git Build Data

← Previous Build

✓ #5 (Oct 13, 2024, 8:44:14 AM)

✎️ Add description

Keep this build

🕒 Started by user [Sarthak Harade](#)

Started 1 min 5
Took 30 sec on

🕒 This run spent:

- 6 ms waiting;
- 30 sec build duration;
- 30 sec total from scheduled to completion.

🔗 git

Revision: f2bc042c04c6e72427c380bcae6d6fee7b49adf
Repository: https://github.com/shazforiot/MSBuild_firstproject.git

- refs/remotes/origin/master

</>

No changes.

Click on Console Output to see the build logs.

The screenshot shows the Jenkins web interface. The top navigation bar includes the Jenkins logo, a search bar, and a user profile for Sarthak Harade. The breadcrumb trail indicates the path: Dashboard > sarthak-sonarqube > #5 > Console Output. On the left sidebar, the 'Console Output' tab is selected. The main area displays the build log, which starts with 'Started by user Sarthak Harade' and shows the execution of various git commands to fetch and checkout a repository. The log concludes with a commit message 'updated' and a warning about the 'sonar.host.url' property being overridden.

```
Started by user Sarthak Harade
Running as SYSTEM
Building on the built-in node in workspace C:\ProgramData\Jenkins\jenkins\workspace\sarthak-sonarqube
The recommended git tool is: NONE
No credentials specified
> git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\jenkins\workspace\sarthak-sonarqube\.git # timeout=10
Fetching changes from the remote Git repository
> git.exe config remote.origin.url https://github.com/shazforiot/MSBuild_firstproject.git # timeout=10
Fetching upstream changes from https://github.com/shazforiot/MSBuild_firstproject.git
> git.exe --version # timeout=10
> git --version # 'git version 2.46.0.windows.1'
> git.exe fetch --tags --force --progress -- https://github.com/shazforiot/MSBuild_firstproject.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
Checking out Revision f2bc042c04c6e72427c380bcaee6d6fee7b49adf (refs/remotes/origin/master)
> git.exe config core.sparsecheckout # timeout=10
> git.exe checkout -f f2bc042c04c6e72427c380bcaee6d6fee7b49adf # timeout=10
Commit message: "updated"
> git.exe rev-list --no-walk f2bc042c04c6e72427c380bcaee6d6fee7b49adf # timeout=10
[sarthak-sonarqube] $ C:\ProgramData\Jenkins\jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\sonarqube\bin\sonar-scanner.bat -
Dsonar.host.url=http://localhost:9000 -Dsonar.projectKey=sonarqube -Dsonar.login=admin -Dsonar.host.url=http://localhost:9000 -Dsonar.sources
Dsonar.password=Sarthak@1234 -Dsonar.projectBaseDir=C:\ProgramData\Jenkins\jenkins\workspace\sarthak-sonarqube
08:44:17.047 WARN Property 'sonar.host.url' with value 'http://localhost:9000' is overridden with value 'http://localhost:9000'
```

Go back to <http://localhost:9000> and navigate to your project. Review the results of the analysis.

The screenshot displays the SonarQube web interface for a project named 'main'. The top navigation bar includes the SonarQube logo and links to Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, and More. The breadcrumb trail shows 'sonarqube / main'. The 'Overview' tab is selected. The main area shows a 'Passed' status with a green checkmark. A warning message states: 'The last analysis has warnings. See details'. Below this, there are two tabs: 'New Code' and 'Overall Code'. The 'Overall Code' tab is active, showing a dashboard with six metrics: Security (0 Open issues, A), Reliability (0 Open issues, A), Maintainability (0 Open issues, A), Accepted issues (0), Coverage (On 0 lines to cover), and Duplications (0.0%, On 86 lines).

CONCLUSION : Integrating Static Application Security Testing (SAST) into the software development process helps identify and fix security vulnerabilities early, improving the overall quality of applications. By regularly running SAST tools, organizations can ensure safer code and reduce the risk of security issues in their final products.