**How to implement SonarQube in CICD pipeline for Application pipeline**

**SonarQube** : SonarQube is a popular tool used for continuous code quality inspection.

**Code Quality Analysis**: SonarQube analyzes code to detect issues related to bugs, vulnerabilities, and code smells. This helps improve the maintainability, reliability, and security of software.

**Automated Code Review**: It provides automated feedback during the development process, which can catch issues early before they become larger problems.

**Technical Debt Management**: SonarQube helps manage technical debt by identifying areas of the codebase that need improvement and offering suggestions for refactoring.

**Standardization**: It enforces coding standards and guidelines, ensuring consistency across a codebase, which is especially valuable in larger teams.

**Integration**: It integrates with CI/CD pipelines, allowing for continuous analysis of code as part of the build process, thus ensuring that code quality is maintained throughout the development lifecycle.

**Historical Tracking**: It tracks code quality over time, helping teams to monitor improvements or regressions and make informed decisions about codebase health.

**Customizable Rules**: Teams can customize rules and quality profiles to match their specific needs and industry standards.

**Reporting**: SonarQube provides detailed reports and dashboards that help teams visualize code quality metrics and track progress.

Step1: Go to docker hub and search SonarQube and then go inside SonarQube, copy the docker pull sonarqube command in your docker machine.(Need to login our docker machine, and then need to run docker pull command)

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Step2: Check docker images and then create container from this sonar Qube image.

A screen shot of a computer

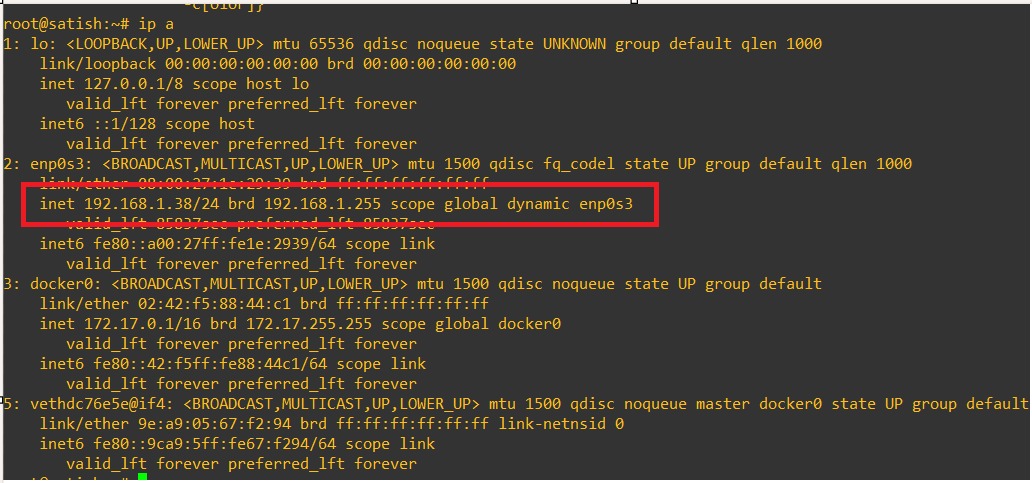
Description automatically generated

Docker ps

Docker run -d -p 9000:9000 –name sonar sonarQube

1st port no : host port, 2nd port no: container port, we can check the sonar Qube port details from docker hub

Step3: Check Docker system IP and run in our browser:



A screenshot of a computer

Description automatically generated

First time login with admin and admin, then change the password and then it login with new password.

Step4: Now create one project for CICD implementation, and fill required attributes.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Step5: Choose our repository for generate credential to be used in to create service principle in CICD pipeline azure devOps portal.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

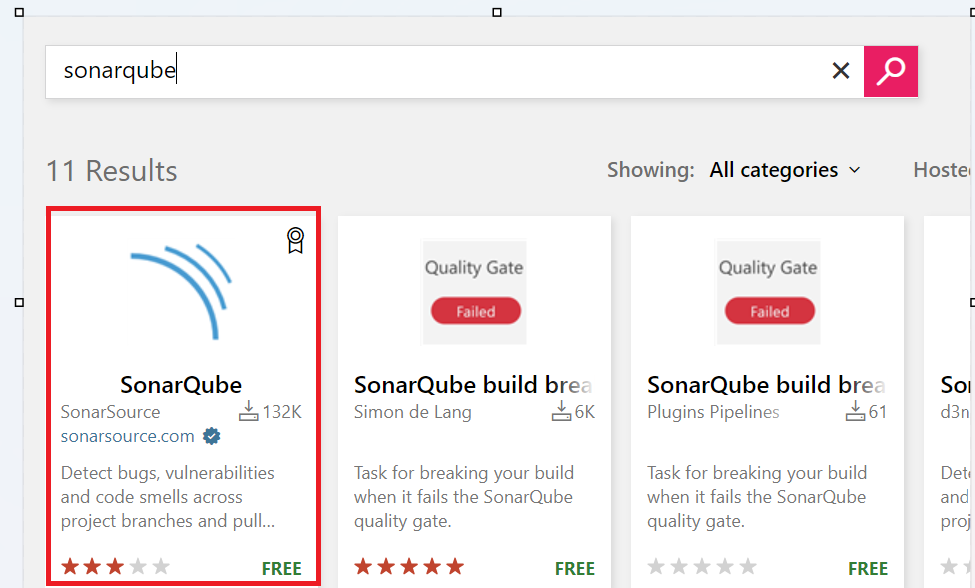
A screenshot of a screenshot of a computer

Description automatically generated

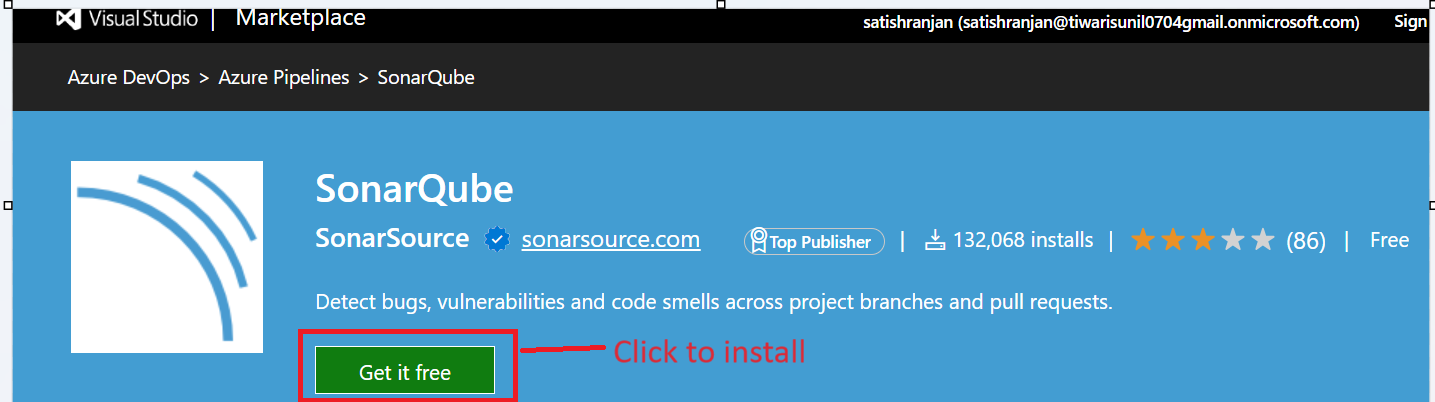
Step6: Login to dev.azure.com portal, go to our project and then clicked on marketplace to search SonarQube.

A screenshot of a computer

Description automatically generated

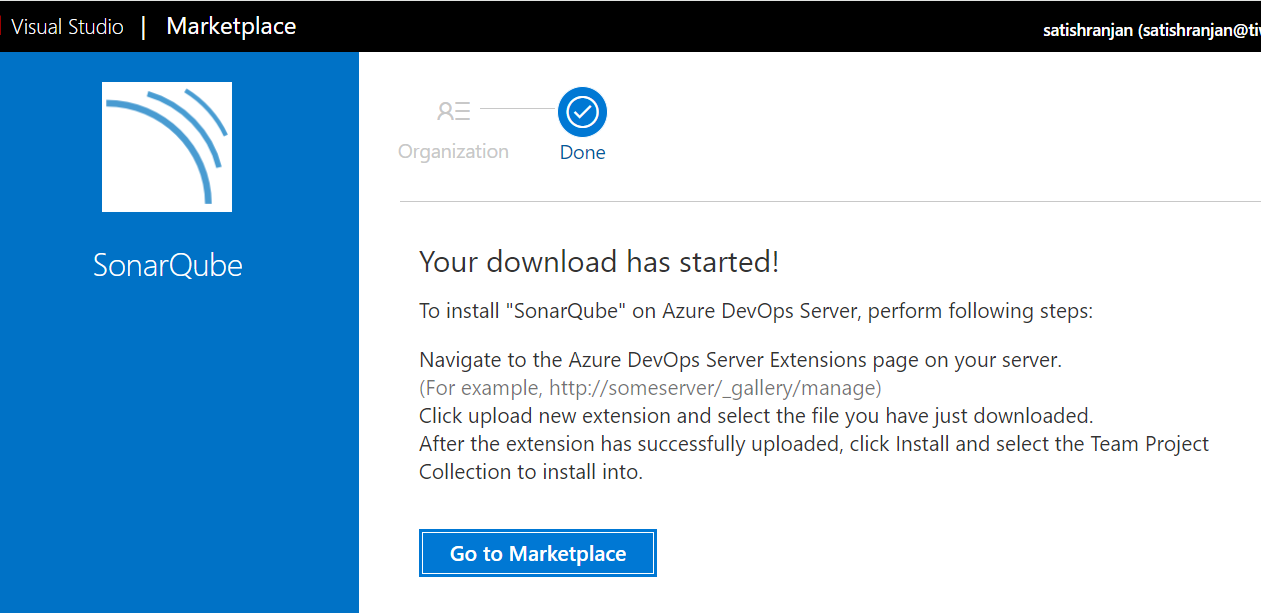


Step7: Install SonarQube in CICD Pipeline.



A screenshot of a computer

Description automatically generated



Step8: Now create service principle for connectivity between SonarQube to our azure DevOps.

Go to project setting 🡪 click on service connection

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a software menu

Description automatically generated

A screenshot of a login page

Description automatically generated

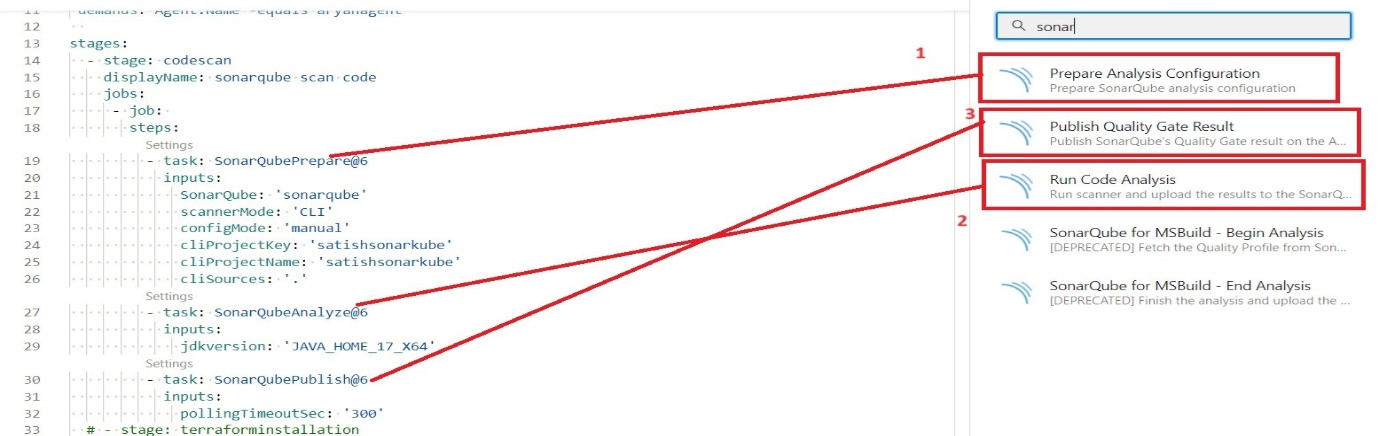
A close-up of a person

Description automatically generated

Step9: Now Implement this SonarQube in Application pipeline.

There are three step to implement SonarQube in application pipeline:

1. Prepare analysis Configuration
2. Run Code Analysis
3. Publish Quality Gate Result

Step10: Run Self hosted Agent and then run the pipeline.

Step11 : All logs present in below path in self agent host:

