**Coding Standards and Guidelines**

Different modules specified in the design document are coded in the Coding phase according to the module specification. The main goal of the coding phase is to code from the design document prepared after the design phase through a high-level language

**Purpose of Having Coding Standards:**

* A coding standard gives a uniform appearance to the codes written by different engineers.
* It improves readability, and maintainability of the code and it reduces complexity also.
* It helps in code reuse and helps to detect error easily.
* It promotes sound programming practices and increases efficiency of the programmers.

**Following are some of the coding standards:**

1. **Limited use of global variables:**  
   These rules talk about which types of data that can be declared global and the data that can’t be.
2. **Standard headers for different modules**:  
   For better understanding and maintenance of the code, the header of different modules should follow standard format and information.
3. **Naming conventions variables and functions:**

* Meaningful and understandable variables name help coders to understand the reason of using it.
* It is better to avoid the use of digits in variable names.
* The name of the function must describe the reason of using the function clearly and briefly.

1. **Indentation:**

* Proper indentation is very important to increase the readability of the code. For making the code readable, programmers should use white spaces properly
* There must be a space after giving a comma between two function arguments.
* Each nested block should be properly indented.
* Proper Indentation should be there at the beginning and at the end of each block in the program.
* All braces should start from a new line and the code following the end of braces also start from a new line.

1. **Avoid using a coding style that is too difficult to understand:**  
   Code should be easily understandable. The complex code makes maintenance and debugging difficult and expensive.
2. **Avoid using an identifier for multiple purposes:**  
   Each variable should be given a descriptive and meaningful name indicating the reason behind using it. This is not possible if an identifier is used for multiple purposes and thus it can lead to confusion to the reader. Moreover, it leads to more difficulty during future enhancements.
3. **Code should be well documented:**  
   The code should be properly commented for understanding easily. Comments regarding the statements increase the understandability of the code.
4. **Length of functions should not be very large:**  
   Lengthy functions are very difficult to understand. That’s why functions should be small enough to carry out small work and lengthy functions should be broken into small ones for completing small tasks.
5. **Try not to use GOTO statement:**  
   GOTO statement makes the program unstructured, thus it reduces the understandability of the program and also debugging becomes difficult.
6. **Use of atomic functions**:

Single function should have single responsibility

Atomic functions benefit us in several ways. Some of which are given below:

* They are simple to test. Because each function has as single responsibility, we can easily predict what the output of each function will be with a given input.
* Since each function is simple to test, they are simple to troubleshoot as well. Each function has so few lines that the errors have no room to hide.
* End result is easier to maintain

**Code coverage report using SonarQube**

SonarQube is an open-source platform developed by SonarSource for continuous inspection of code quality. Sonar does static code analysis, which provides a detailed report of bugs, code smells, vulnerabilities, code duplications.

Analyzing source code in important to ensure quality, reliability, and maintainability over the life-span of the project, a poorly written codebase is always more expensive to maintain.

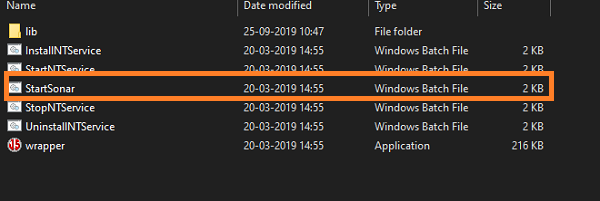
**Steps to generate code coverage report using SonarQube:**

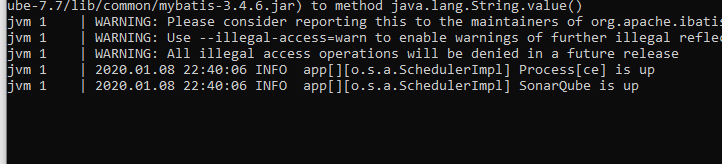
**Step 1:** Before installing and configuring the SonarQube, we need to install Java -JDK, because the SonarQube scanner requires version 8 or 11 of the JDK.

Use the link to download Java:[*https://www.java.com/en/download/*](https://www.java.com/en/download/)

**Step 2:** Download SonarQube Community edition using the below link: [*https://www.sonarqube.org/downloads/*](https://www.sonarqube.org/downloads/)

Once the download is completed, unzip the file. Go to sonarqube\bin\windows-x86-64 - >and run StartSonar





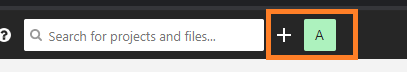
**Step 3:** Open command prompt and run the below command:

dotnet tool install –global dotnet-sonarscanner

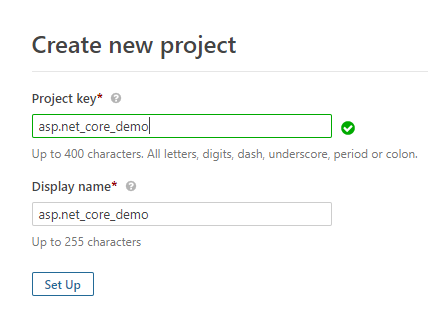
**Step 4:** Once the SonarQube is up and running, open the dashboard in the browser using <http://localhost:9000/>, and login as admin.

By default, the username: admin password: admin

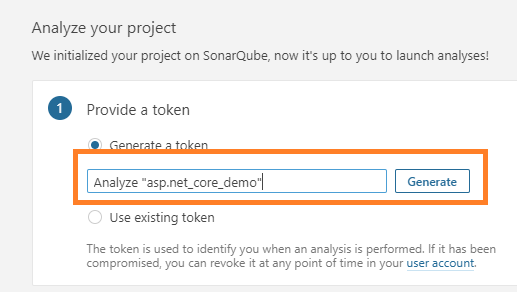
Click on New Project in SonarQube running in browser.



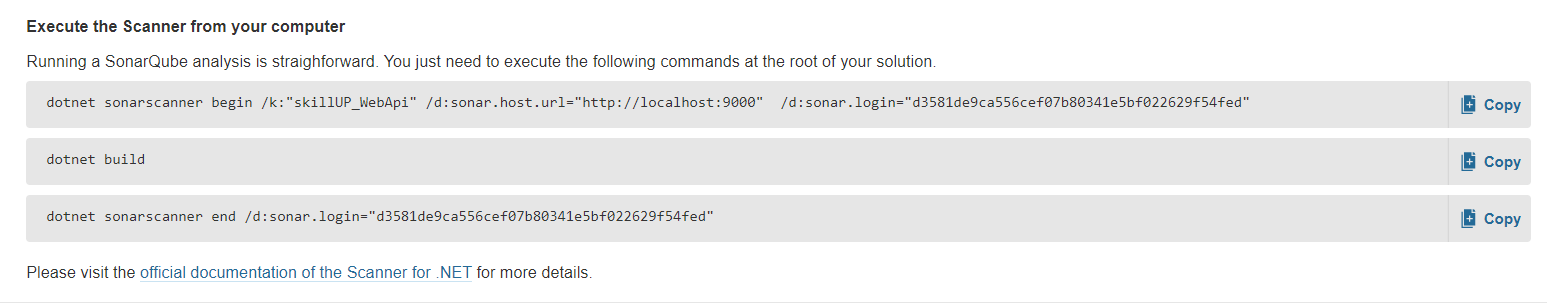
Enter the Project key and Display name



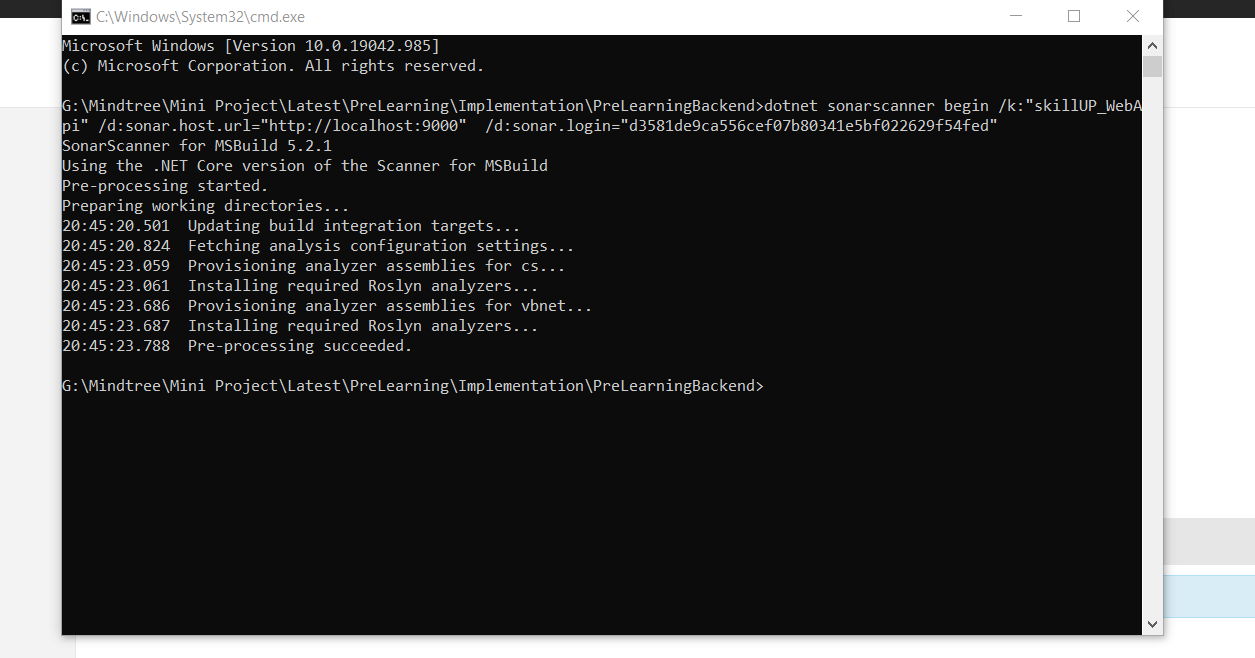
Click on Setup and provide a token



After successful generation of token below window pops up



**Step 5:** Open Visual Studio command prompt, switch to project path and provide the above listed commands sequentially



**Step 6:** Switch back to the browser where SonarQube is already running. Click on overview where you can see the overall code coverage report of your project.

