Publish Code Coverage to SonarCloud

.NET

SonarCloud is a cloud-based code analysis service designed to detect code quality issues in 25 different programming languages, continuously ensuring the maintainability, reliability and security of your code.

HTTPS://SONARCLOUD.IO/DOCUMENTATION

First Steps

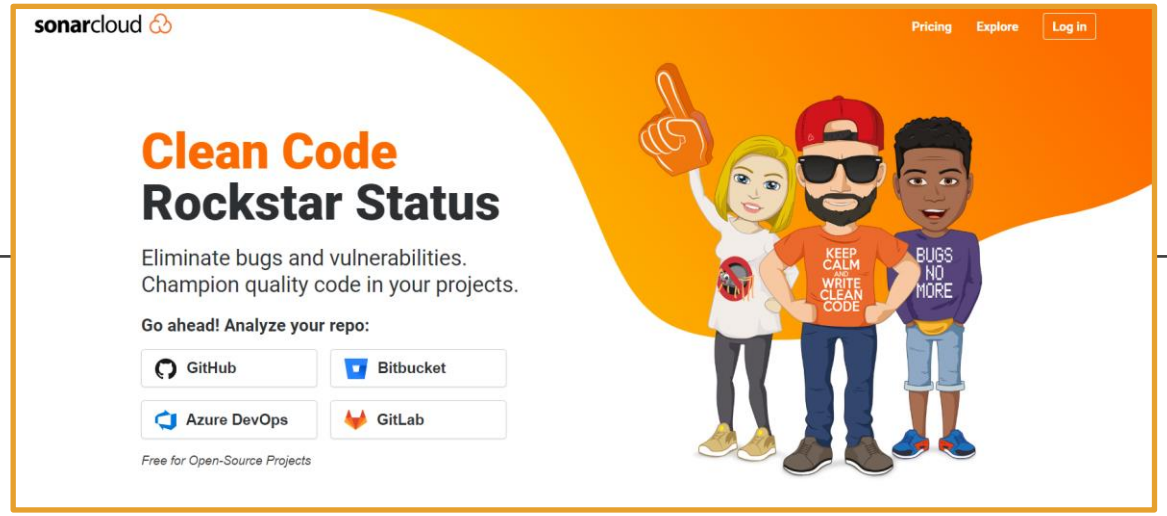
https://sonarcloud.io/

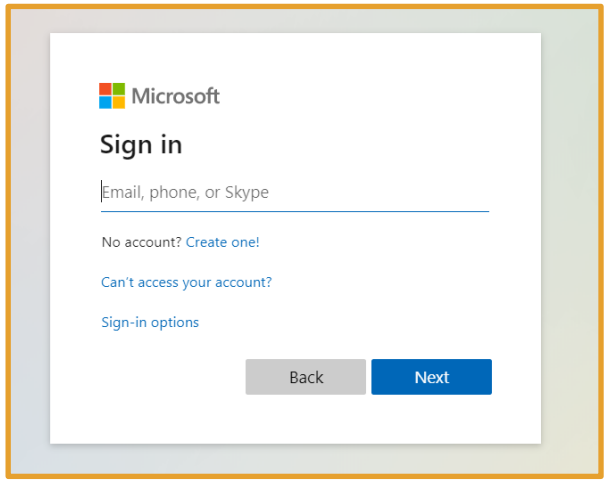
First, create a pipeline that successfully builds and deploys to your Website.

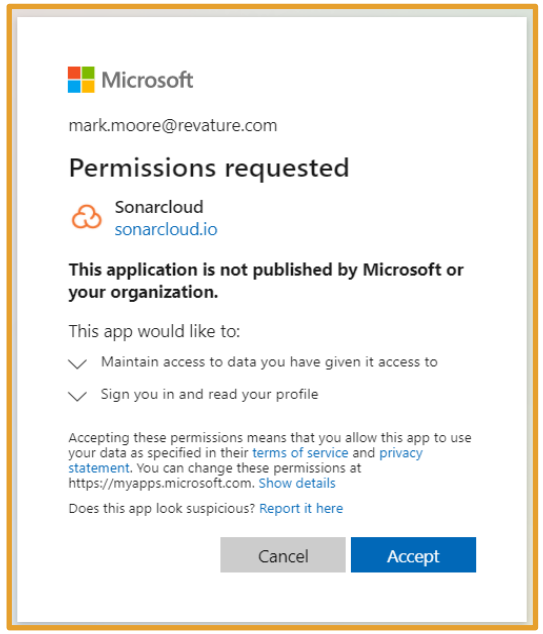
Next, log in to www.sonarcloud.io.

(Important) Use your Azure DevOps credentials.

If asked, accept any request for permission or access by Sonarcloud.

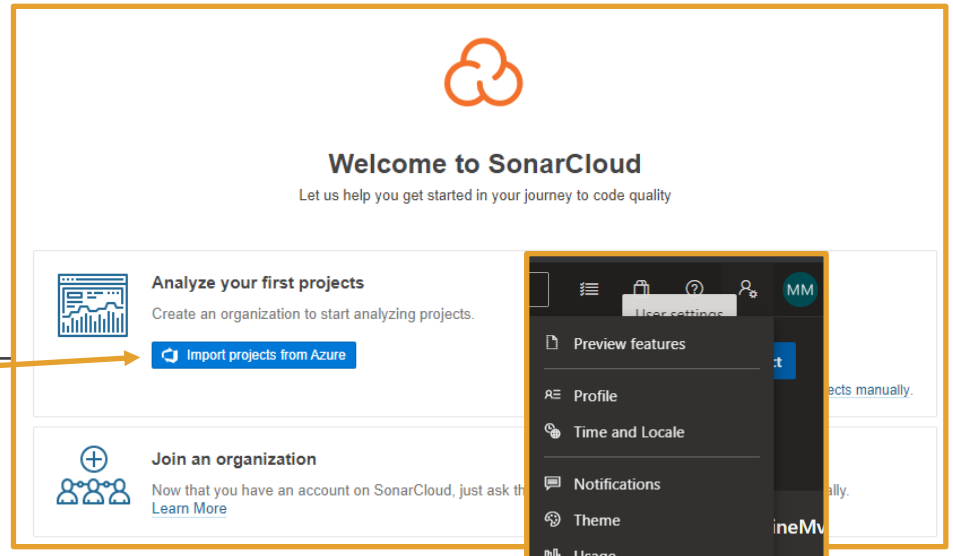


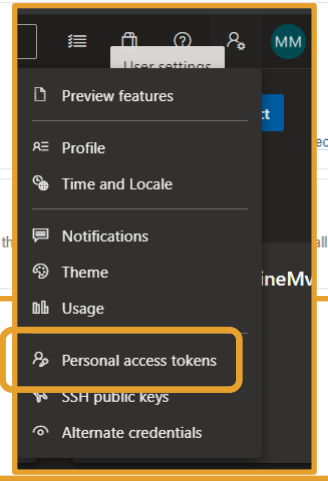


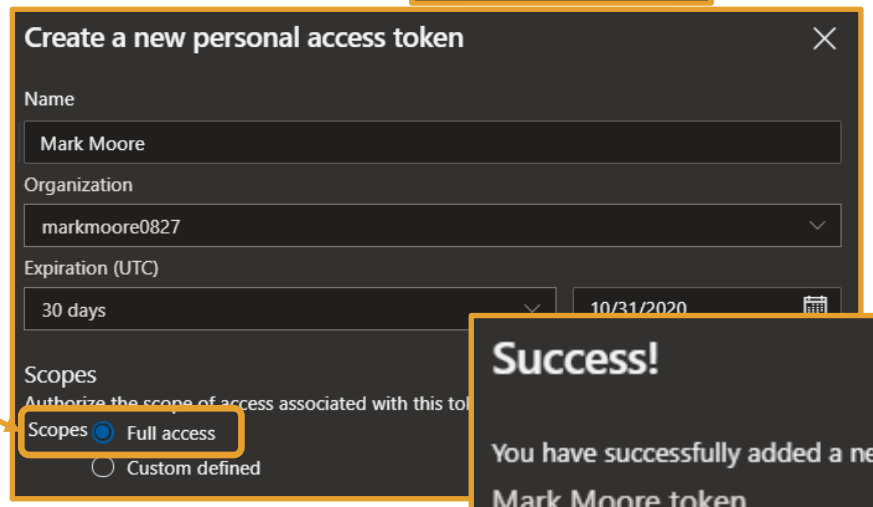


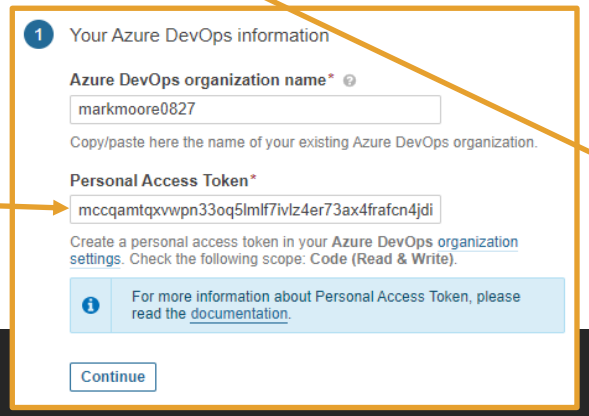
Create SonarCloud Organization

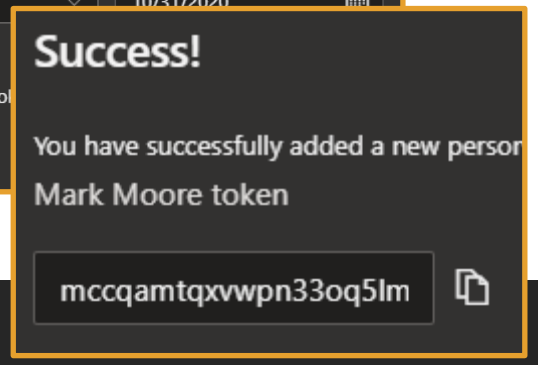
1. Create an organization
2. Enter the correct name of your Azure Devops Organization and a Personal Access Token (P.A.T.).
   1. Log into your Azure Account.
   2. In the upper right, click on User Settings>Personal Access tokens.
   3. Click ‘New Token’.
   4. Enter your name and select ‘Full Access’
   5. Select ‘Create’
   6. Copy your P.A.T.
3. Enter the P.A.T.
4. Click ‘Continue’





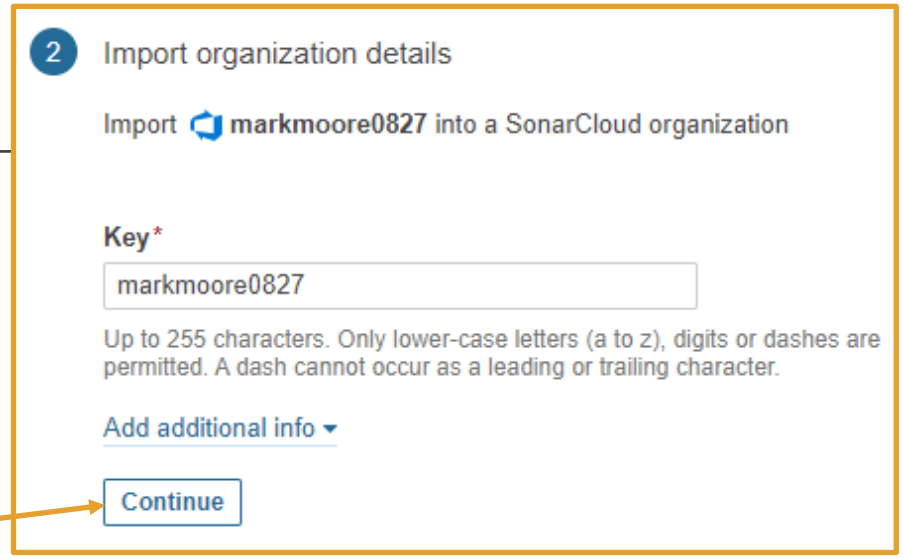


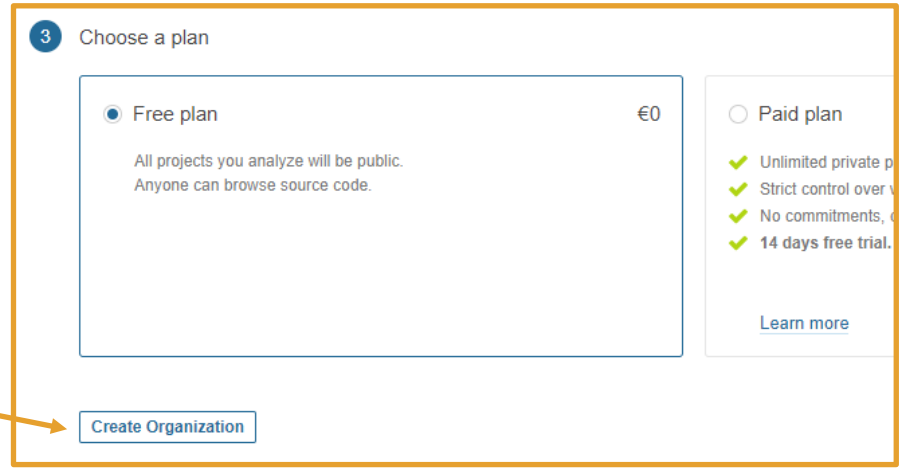




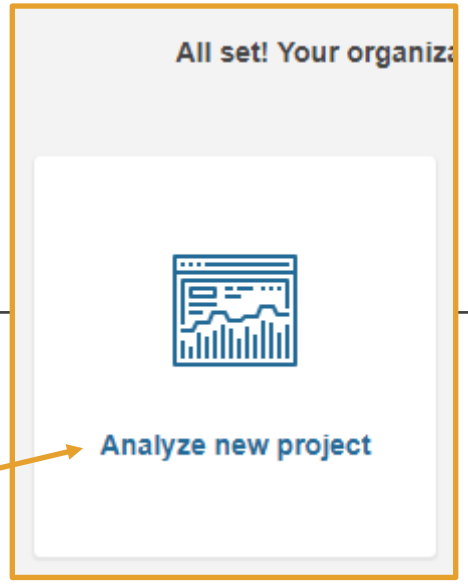
Create SonarCloud Organization

1. Click ‘Continue’ to Import your organization details
2. Select ‘Free plan’
3. Click “Create Organization”

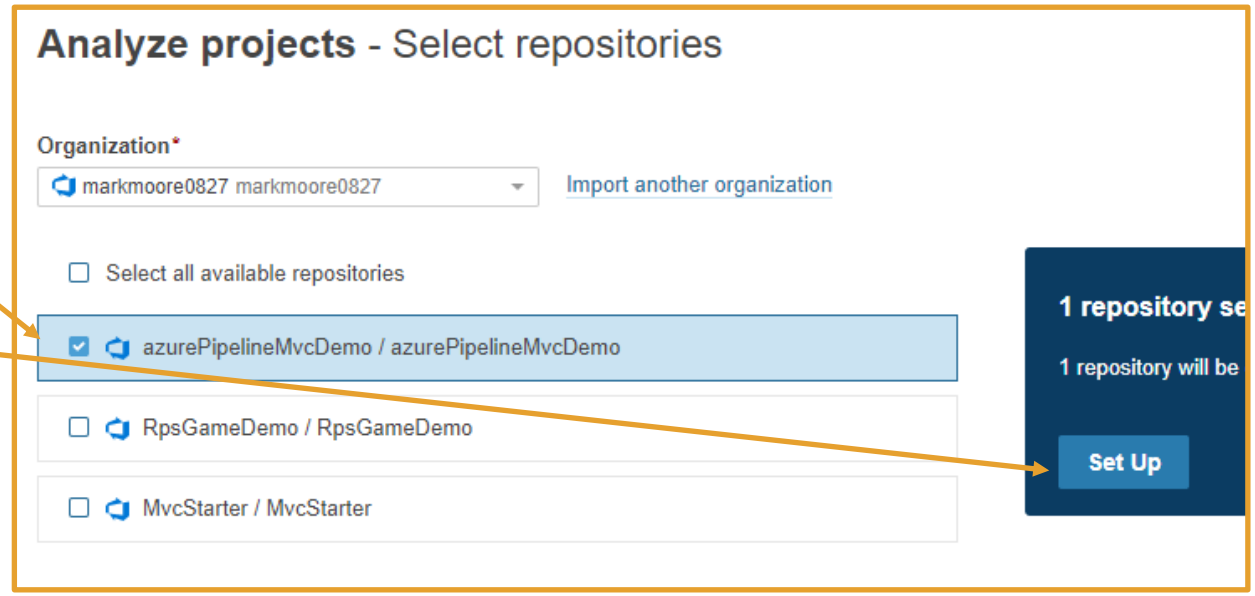




Analyse a New Project

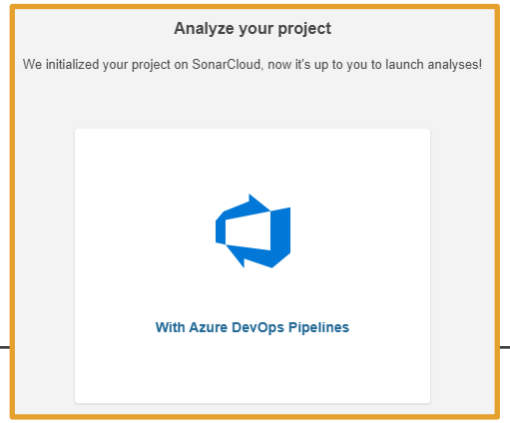


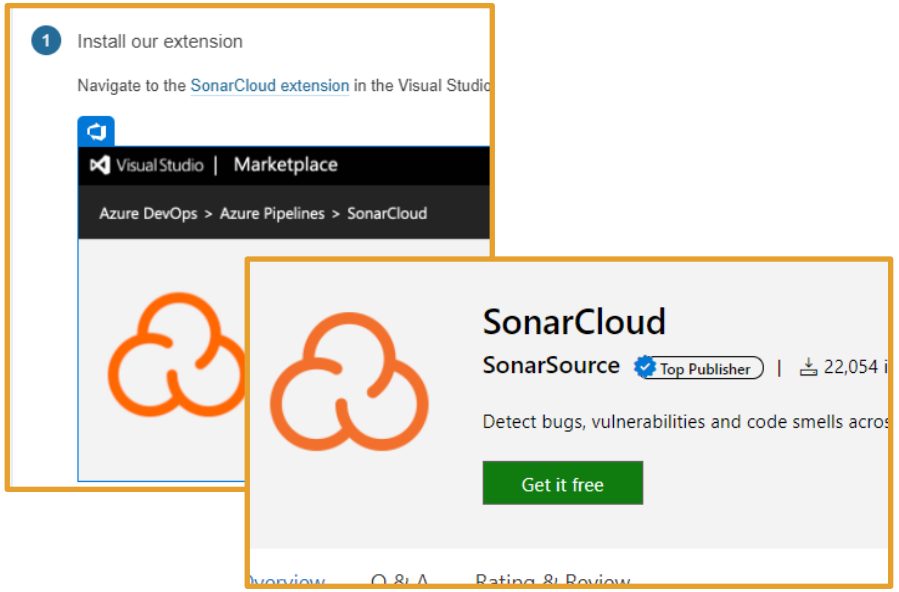
4. Click ‘analyse new project’.
5. Choose the pipeline you want to analyse.
6. Click ‘Set Up’.

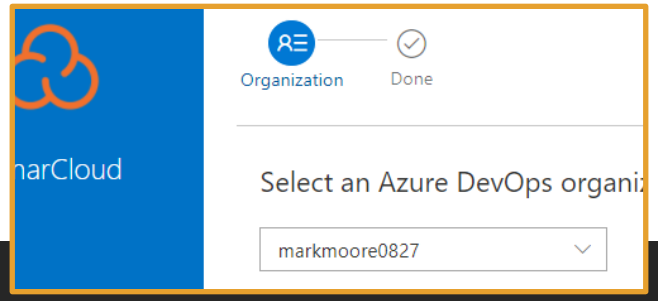


Configure Code Analysis

7. Click on ‘With Azure DevOps Pipelines’
8. Click to download the SonarCloud Extension from the from the Visual Studio Marketplace.
9. Make sure to choose the correct organization when installing in the Marketplace.







Configure Azure Pipeline

2 Configure Azure Pipeline

What option best describes your build?

Maven | Gradle | .NET | Other (for JS, TS, Go, Python, PHP, ...)

Create a new pipeline

Follow the steps on Azure to initialize your pipeline and link it to your repository.

Add a new SonarCloud Service Endpoint

1. Go to **Project settings > Service connections**
2. Add a new service connection of the type SonarCloud
3. Use this token: `f3cf46527df3aec0546d71961a7358d9ecfcfb29`
4. Click on **Verify** to check that everything is linked correctly.

Add the following three tasks to your pipeline

1. Prepare Analysis Configuration

1. Select the SonarCloud endpoint.
2. Select the SonarCloud organization `markmoore0827`
3. In Choose the way to run the analysis, select **Integrate with MSBuild**.
4. In the Project Key field, enter `markmoore0827_azurePipelineMvcDemo`
5. In the Project Name field, enter `azurePipelineMvcDemo`

Please ensure this task runs before your build step.

2. Run Code Analysis

This task needs to run after your build step.

The steps on the following pages will guide you through setup on the ***Azure DevOps*** side.

See the Step-By-Step on the following slides.

2. Run Code Analysis

This task needs to run after your build step.

3. Publish Quality Gate Result

This task is not mandatory but will allow you to decorate your Pull Request.

If you plan not to use such a feature, you can omit it. Be aware that this task may increase your build time.

Enable the pipeline on push

Enable Continuous Integration for your pipeline to make sure your code will be analysed after every update.

Enable the PR Decoration

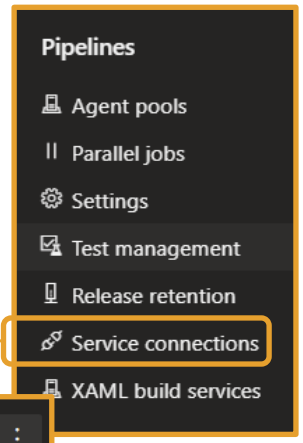
Enable PR trigger to make sure your code will be analysed after every update.

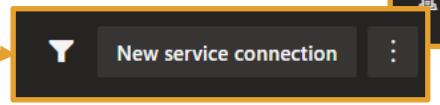
Add a new SonarCloud Service Endpoint

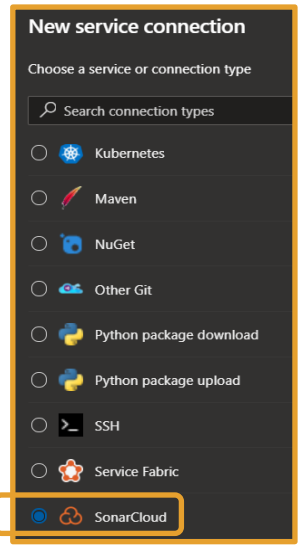
Add a new SonarCloud Service Endpoint

1. Go to **Project settings > Service connections**
2. Add a new service connection of the type SonarCloud
3. Use this token: `f3cf46527df3aec0546d71961a7358d9ecfcfb29`
4. Click on **Verify** to check that everything is linked correctly.

1. In your Azure DevOps account, click on the project you will be adding the Code Analysis to.
2. Click Project settings at the bottom left of your project home page.
3. Click the ‘back’ arrow next to Project details.
4. Click Service Connections under Pipelines.
5. Click ‘New service connection’ in the upper right.
6. Select ‘SonarCloud’ in the ‘New Service Connection’ List
7. Click Next.

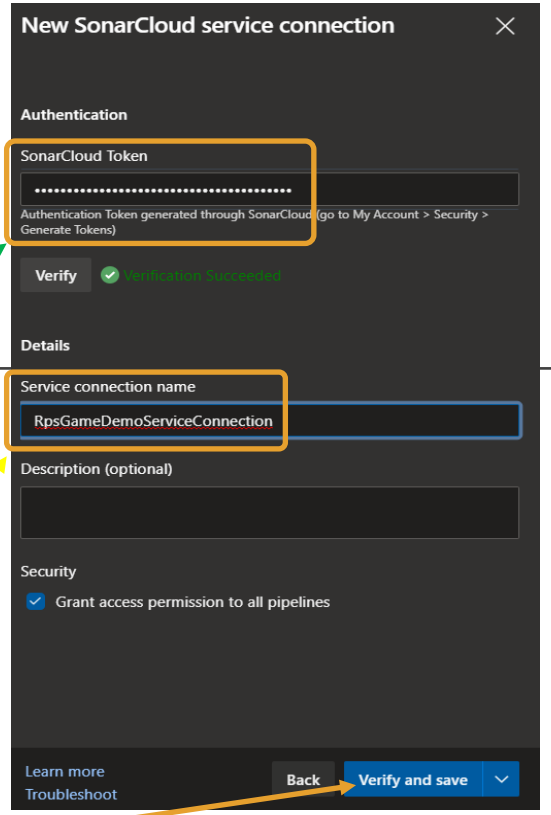


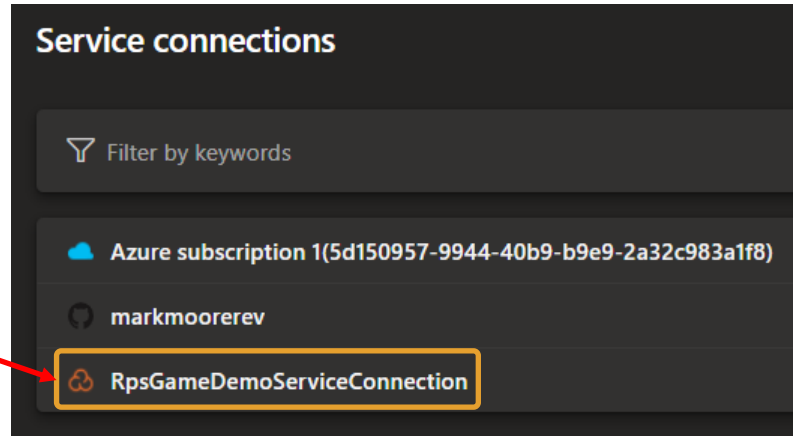




New SonarCloud Connection

- Paste in the Personal Access Token under ‘SonarCloud Token’.
- Create a name for your connection under ‘Service connection name’.
- Click ‘Verify and Save’.
- Verify the connection was successful.





Add publish task to your pipeline YAML

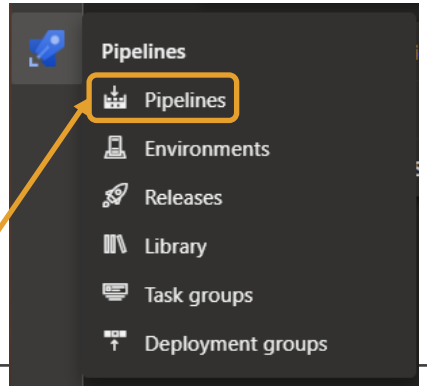
Add the following three tasks to your pipeline.

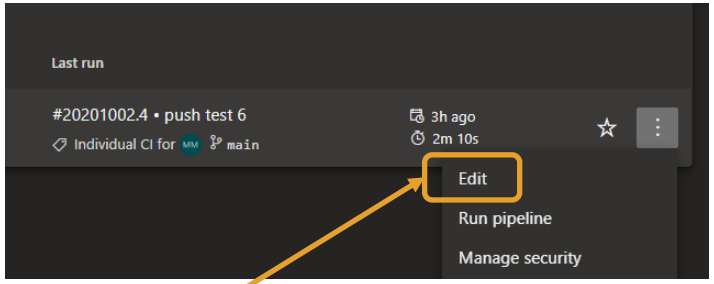
1. Prepare Analysis Configuration

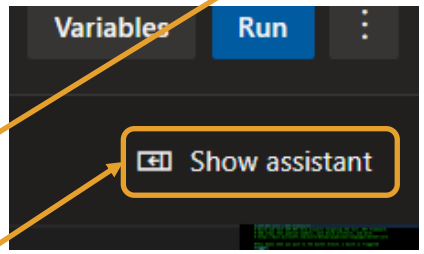
1. Select the SonarCloud endpoint.
2. Select the SonarCloud organization `markmoore0827`
3. In Choose the way to run the analysis, select **Integrate with MSBuild**.
4. In the Project Key field, enter `markmoore0827_azurePipelineMvcDemo`
5. In the Project Name field, enter `azurePipelineMvcDemo`

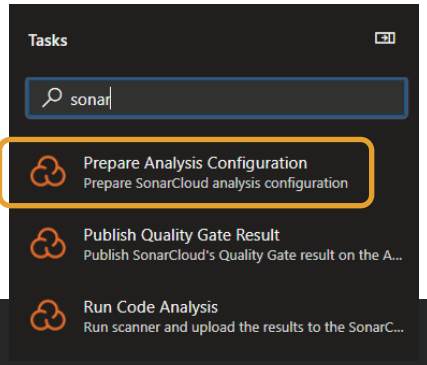
Please ensure this task runs before your build step.

1. Click Pipelines under the Rocket (Pipelines) avatar in the left column.
2. Click Edit under the pipeline you want to add code analysis to.
3. Click 'Show Assistant' in the upper left.
4. Select 'Prepare Analysis Configuration'.









Add Code Analysis configuration task to your pipeline YAML

1. Prepare Analysis Configuration

1. Select the SonarCloud endpoint.
2. Select the SonarCloud organization `markmoore0827`
3. In Choose the way to run the analysis, select **Integrate with MSBuild**.
4. In the Project Key field, enter `markmoore0827_azurePipelineMvcDemo`
5. In the Project Name field, enter `azurePipelineMvcDemo`

Please ensure this task runs before your build step.

Prepare Analysis Configuration

SonarCloud Service Endpoint *
RpsGameDemoServiceConnection

Organization *
markmoore0827 (markmoore0827)

Choose the way to run the analysis *
- (•) Integrate with MSBuild
- ( ) Integrate with Maven or Gradle
- ( ) Use standalone scanner

Project Key *
markmoore0827_azurePipelineMvcDemo

Project Name
azurePipelineMvcDemo

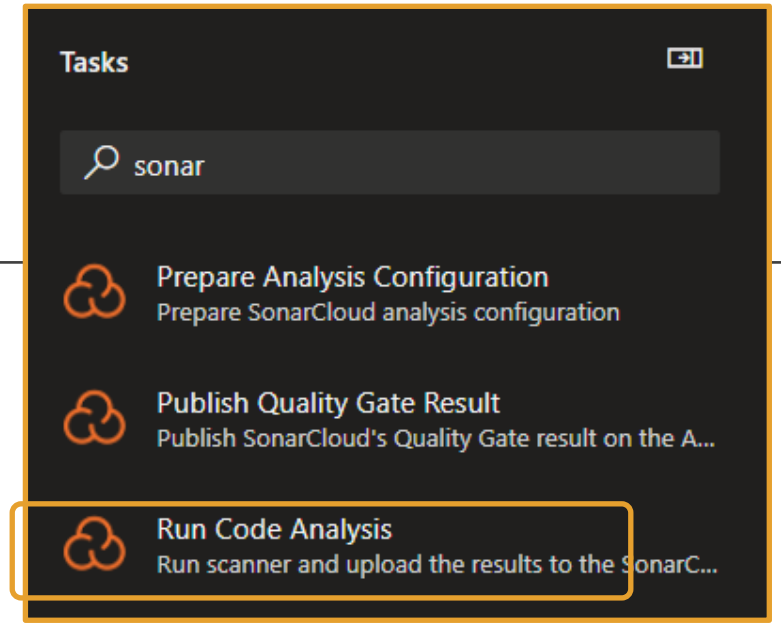
Project Version
1.0

Settings
```yaml
- task: SonarCloudPrepare@1
  inputs:
    SonarCloud: 'RpsGameDemoServiceConnection'
    organization: 'markmoore0827'
    scannerMode: 'MSBuild'
    projectKey: 'markmoore0827_azurePipelineMvcDemo'
    projectName: 'azurePipelineMvcDemo'
```

Settings
```yaml
- task: DotNetCoreCLI@2
  displayName: Build
  inputs:
```

You don't have to change anything under 'Advanced'.
Place the curser on the line above your 'build' Task.
Click 'Add'

Add Code Analysis to your YAML



1. Click ‘Show Assistant’ to return to the Assistant.
2. Make sure the curser is immediately below the ‘build’ task.
3. Click ‘Run Code Analysis’.

```yaml
- task: DotNetCoreCLI@2
  inputs:
    command: 'test'
    projects: 'Demos/JavaScriptWithDotnetAPI/memesaver/memesaver.sln'
    #tell the compiler to collect code coverage. This artifact will be saved in a default location
    arguments: '--configuration $(buildConfiguration) --collect "Code Coverage"'
    testRunTitle: 'Dotnet tests running'
    workingDirectory: 'Demos/JavaScriptWithDotnetAPI/memesaver'

Settings
- task: PublishCodeCoverageResults@1
  inputs:
    #tell the compiler what code coverage tool you used.
    codeCoverageTool: 'Cobertura'
    # this is the path to the default location of the coverage artifact file.
    summaryFileLocation: '**/coburtura/coverage.xml'
```

Complete Pipeline YAML file (1/2)

```yaml
trigger:
  branches:
    include:
    - main
#the below is required only if you have multiple .sln files in your repo.
  paths:
    include:
    - Demos/JavaScriptWithDotnetAPI


pool:
  vmImage: 'windows-latest'

variables:
  solution: 'Demos/JavaScriptWithDotnetAPI/memesaver/memesaver.sln'
  buildPlatform: 'Any CPU'
  buildConfiguration: 'Release'


#this installs the newest SDK for this build
steps:
Settings
- task: UseDotNet@2
  displayName: 'Install .NET Core SDK'
  inputs:
    packageType: 'sdk'
    version: '5.0.x'
    # performMultiLevelLookup: true
    # includePreviewVersions: false # Required 'true' to use preview versions

```

```yaml
Settings
- task: UseDotNet@2
  displayName: 'Install .NET Core Runtime'
  inputs:
    packageType: 'runtime'
    version: '2.x'
    # performMultiLevelLookup: true


Settings
- task: NuGetToolInstaller@1

Settings
- task: NuGetCommand@2
  inputs:
    restoreSolution: '$(solution)'

#this must go before the 'build' task.
Settings
- task: SonarCloudPrepare@1
  inputs:
    SonarCloud: '03012021BatchServiceToken1'
    organization: '03012021batch'
    scannerMode: 'MSBuild'
    projectKey: '03012021Batch_03012021BatchPipelineDemo'
    projectName: '03012021BatchPipelineDemo'

#this will restore and build the directory.
Settings
- task: DotNetCoreCLI@2
  displayName: 'building'
  inputs:
    command: 'build'
    #path to project .sln
    projects: 'Demos/JavaScriptWithDotnetAPI/memesaver/*.sln'
    #path to root of project
    workingDirectory: 'Demos/JavaScriptWithDotnetAPI/memesaver'
    arguments: '--configuration $(buildConfiguration)'
```

Complete Pipeline YAML file (2/2)

```yaml

#
Settings
- task: DotNetCoreCLI@2
  inputs:
    command: 'test'
    projects: 'Demos/JavaScriptWithDotnetAPI/memesaver/memesaver.sln'
    #tell the compiler to collect code coverage. This artifact will be saved in a default location
    arguments: '--configuration $(buildConfiguration) --collect "Code Coverage"'
    testRunTitle: 'Dotnet tests running'
    workingDirectory: 'Demos/JavaScriptWithDotnetAPI/memesaver'

Settings
- task: PublishCodeCoverageResults@1
  inputs:
    #tell the compiler what code coverage tool you used.
    codeCoverageTool: 'Cobertura'
    # this is the path to the default location of the coverage artifact file.
    summaryFileLocation: '**/coburtura/coverage.xml'

Settings
- task: SonarCloudAnalyze@1

Settings
- task: SonarCloudPublish@1
  inputs:
    pollingTimeoutSec: '300'

Settings
- task: DotNetCoreCLI@2
  displayName: 'Publishing'
  inputs:
    command: 'publish'
    publishWebProjects: false
    # modifyOutputPath: false
    # workingDirectory: 'Demos/JavaScriptWithDotnetAPI/memesaver'
    projects: '$(solution)'
    zipAfterPublish: true
```

```yaml

#publish to Azure AppService
Settings
- task: AzureRmWebAppDeployment@4
  inputs:
    ConnectionType: 'AzureRM'
    azureSubscription: 'Azure subscription 1(5d150957-9944-40b9-b9e9-2a32c983a1f8)'
    appType: 'webApp'
    WebAppName: 'memesaver'
    packageForLinux: '$(System.DefaultWorkingDirectory)/**/*.zip'
```

A Microsoft Docs Lab Tutorial

https://azuredevopslabs.com/labs/vstsextend/sonarcloud//