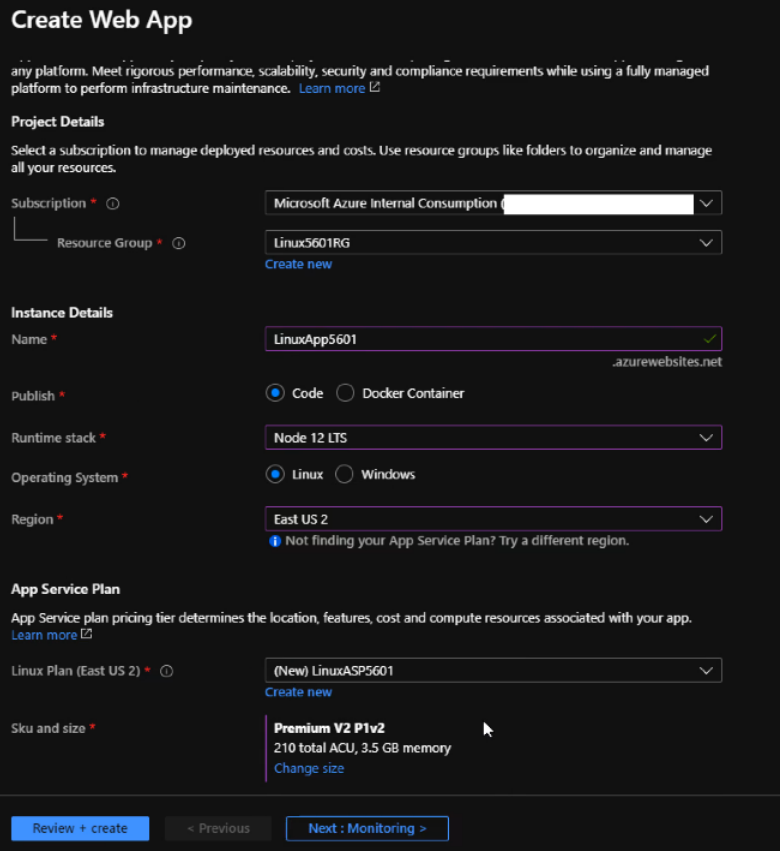
**Creating a CI/CD pipeline for a JavaScript app**

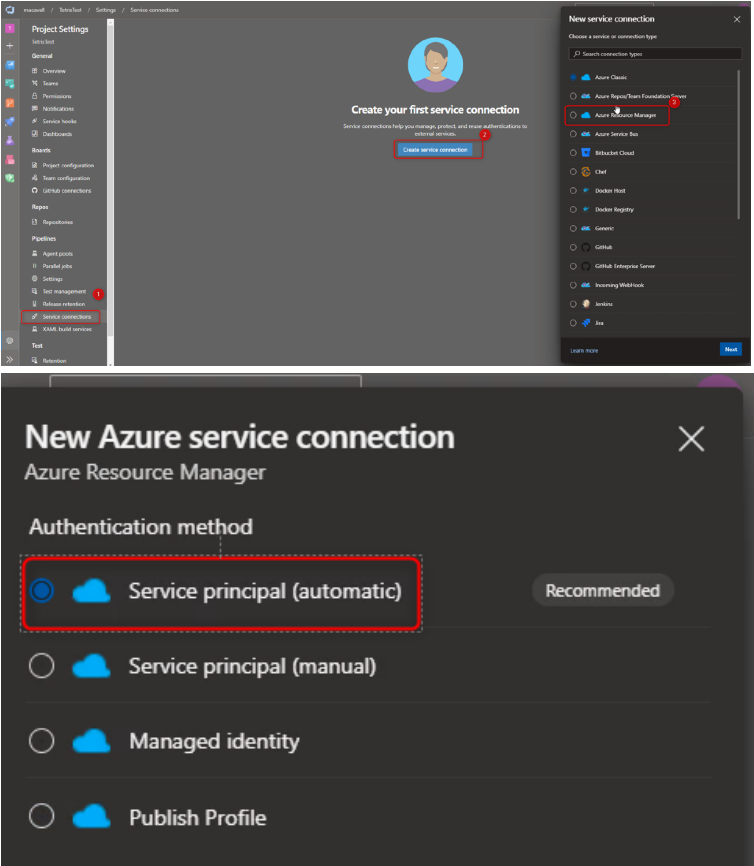
**1.** In Azure, create a Linux WebApp Service:



Make sure to use a clone of the Flatris-LAB GitHub repository:

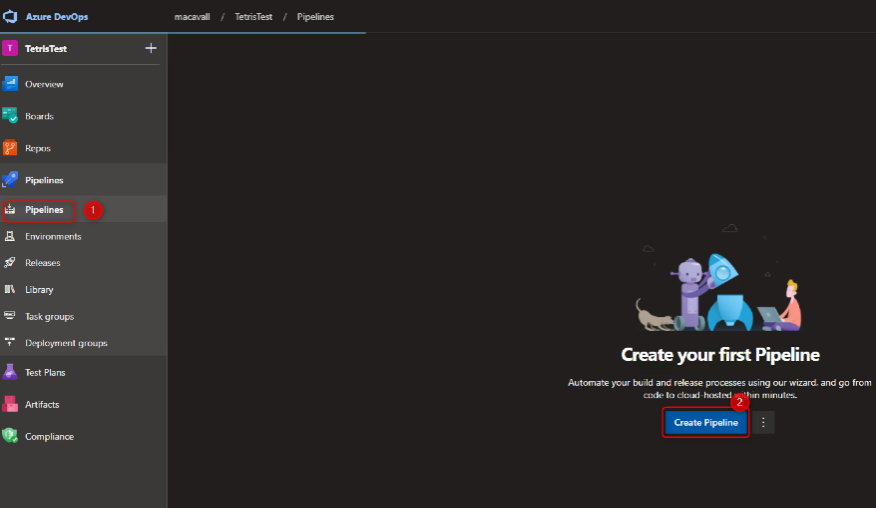
<https://github.com/hosniah/nodeapp-flatris>

**2.** Create a Service Connection to link DevOps to your Azure subscription.

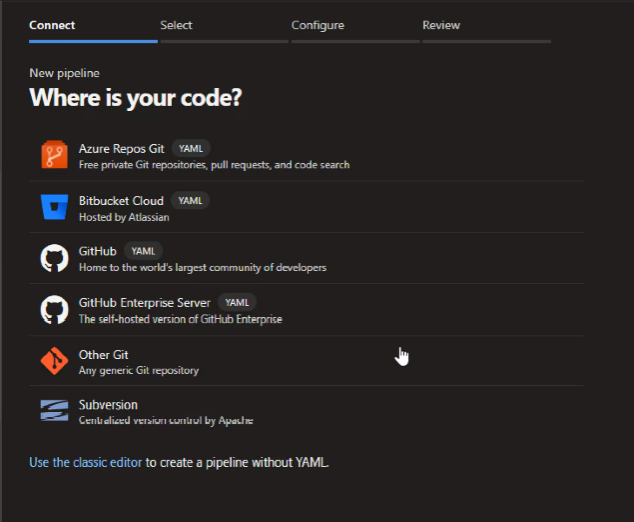


You’ll use this Service connection name in the YAML file.

**3.** Create the DevOps Pipeline.



Point to the Azure DevOps Repo hosting the Flatris DevOps app.



Use the first option, ‘Node.js to get started with a general Node.js project.

**4.** Use the following custom YAML file, replacing the first 4 variables with the Service Connection, App Service, and Resource Group Names that match your environment. (You should be able to copy and paste.

# Node.js Express Web App to Linux on Azure

# Build a Node.js Express app and deploy it to Azure as a Linux web app.

# Add steps that analyze code, save build artifacts, deploy, and more:

# https://docs.microsoft.com/azure/devops/pipelines/languages/javascript

trigger:

- master

variables:

# Azure Resource Manager connection created during pipeline creation

azureSubscription: 'ServiceConnectionName'

# Web app name

webAppName: 'LinuxApp5601'

# Resource group

resourceGroupName: 'Linux5601RG'

# Environment name

environmentName: 'LinuxApp5601'

# Agent VM image name

vmImageName: 'ubuntu-latest'

stages:

- stage: Archive

displayName: Archive stage

jobs:

- job: Archive

displayName: Archive

pool:

vmImage: $(vmImageName)

steps:

- task: AzureAppServiceSettings@1

inputs:

azureSubscription: $(azureSubscription)

appName: $(webAppName)

resourceGroupName: $(resourceGroupName)

appSettings: |

[

{

"name": "SCM\_DO\_BUILD\_DURING\_DEPLOYMENT",

"value": "true"

}

]

- task: ArchiveFiles@2

displayName: 'Archive files'

inputs:

rootFolderOrFile: '$(System.DefaultWorkingDirectory)'

includeRootFolder: false

archiveType: zip

archiveFile: $(Build.ArtifactStagingDirectory)/$(Build.BuildId).zip

replaceExistingArchive: true

- upload: $(Build.ArtifactStagingDirectory)/$(Build.BuildId).zip

artifact: drop

- stage: Deploy

displayName: Deploy stage

dependsOn: Archive

condition: succeeded()

jobs:

- deployment: Deploy

displayName: Deploy

environment: $(environmentName)

pool:

vmImage: $(vmImageName)

strategy:

runOnce:

deploy:

steps:

- task: AzureWebApp@1

displayName: 'Azure Web App Deploy: Matt-Test-NodeJS-Deploy'

inputs:

azureSubscription: $(azureSubscription)

appType: webAppLinux

appName: $(webAppName)

runtimeStack: 'NODE|10.14'

package: $(Pipeline.Workspace)/drop/$(Build.BuildId).zip

**5.** Save and run the pipeline, then open the link associated with your WebApp.