CiCd pipeline

Matthijs Kotterink

# Inhoud

<…>

# CiCd

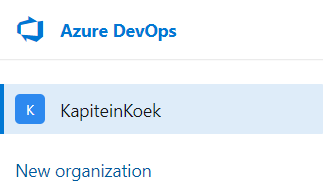
a CiCd pipeline (continuous integration, continuous deployment) pipeline is a type of integration where the code deploys as soon as you have made a change. There are a lot of common ways to create such a pipeline (Github, docker) but I am going to show you how to integrate is effectively in Azure using DevOps.

Azure DevOps is a platform that is used for pulling projects from github and pushing it to azure, there are a lot of different features inside of azure but I will start by focussing on the basics.

# Getting set up

In this document I will show you how to set everything up from scratch, I will use references from my own personal project so that I have visuals to substantiate my explanation.

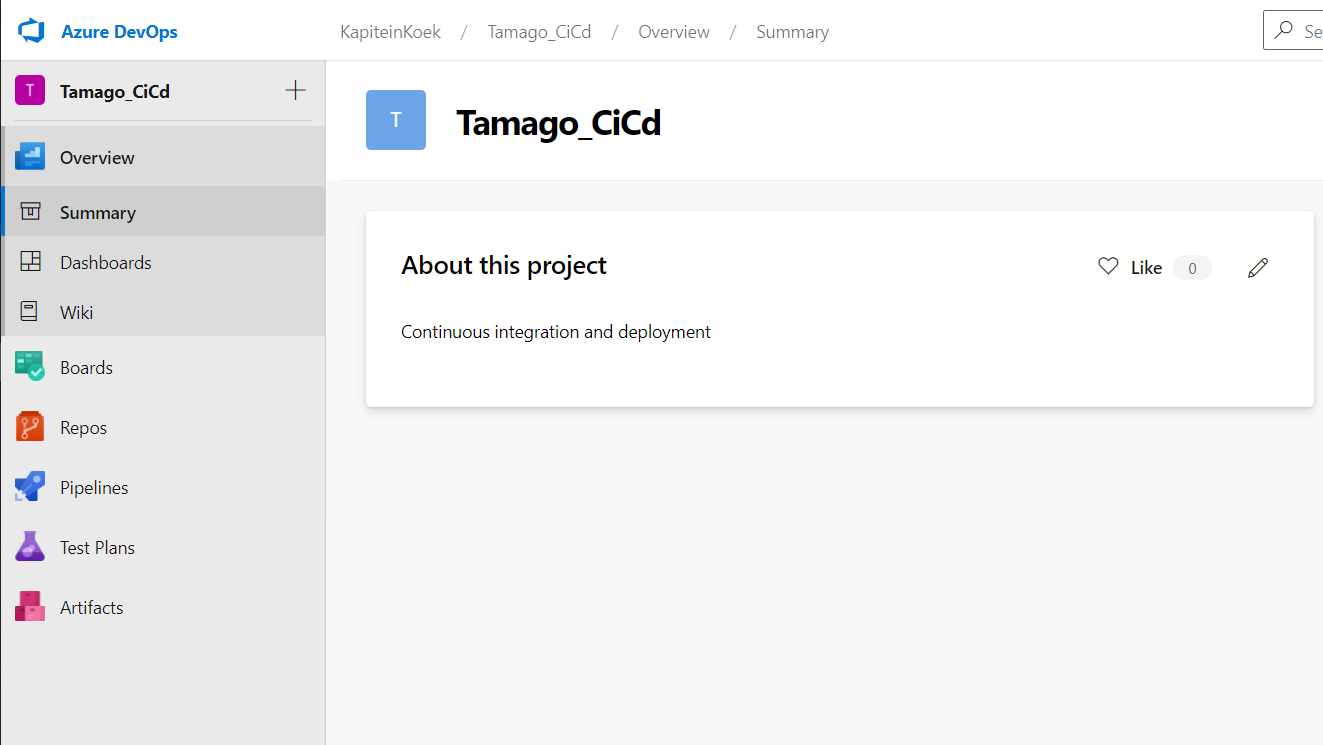
Start by going to <https://dev.azure.com/> and logging in with your school account.



Once logged in you can see your organisations on the left, I have already created one but you can create your own by clicking the ‘new organisation’ button on the left (just follow the steps given to create an organisation).

Once created you might also want to add a new project on the right of the screen which is also pretty straight forward.

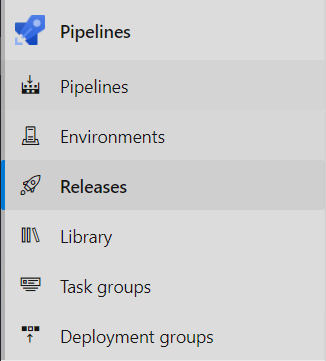
Once you have created your project you can click on it and you should get a screen similar to this.



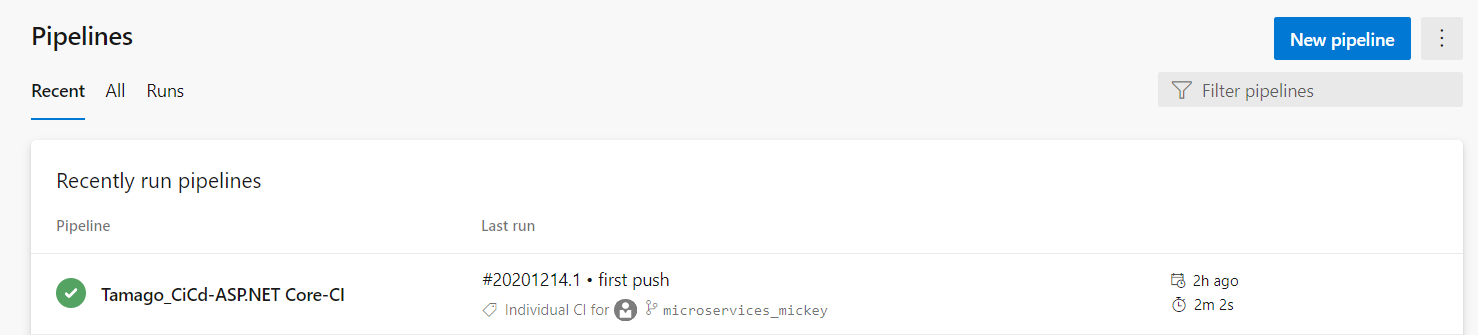
In here we will be creating our first pipeline.

# Creating a pipeline

Inside of your newly created pipeline you can explore the pipeline tap on the left. Inside of this tab you will find a ton of different options which will help us in the process of creating a CiCd pipeline.

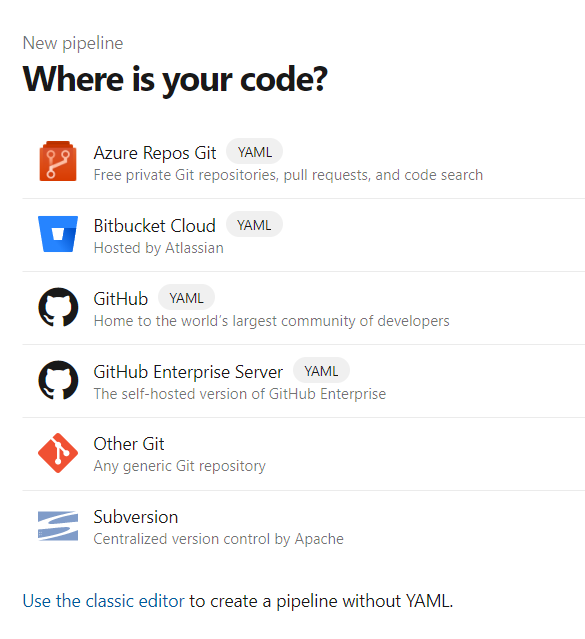


There is a lot in here but for the first step we will just go to the Pipelines tab inside Pipelines.



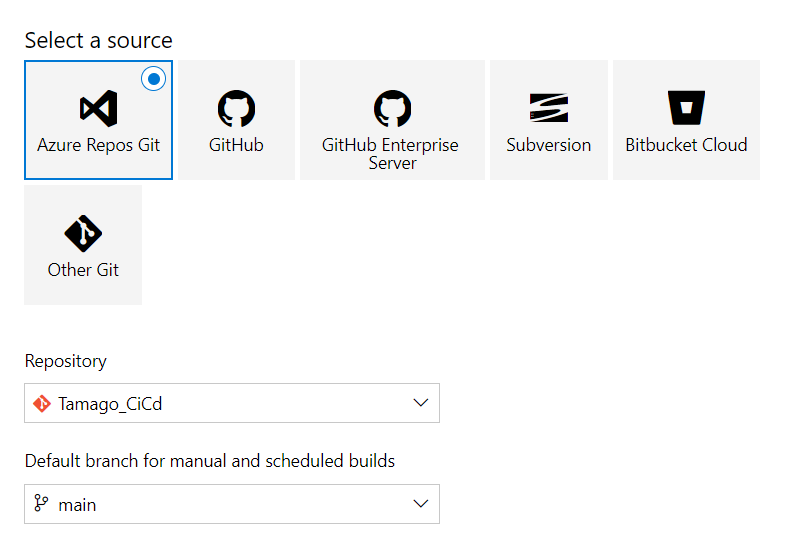
In here you can see the recently run pipelines as well as the outcome on the left of the run and the last time you ran the pipeline on the right, if you want to create a new pipeline you can just press the ‘new pipeline’ button on the top right corner of your screen.

## Creation

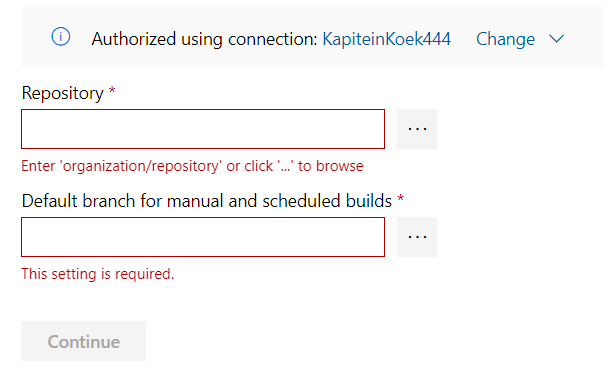


Once in the creator menu there are a lot of different options to choose from, notice there is Yaml written behind almost all of them, you don’t want that as it is truly a pain to implement it that way.

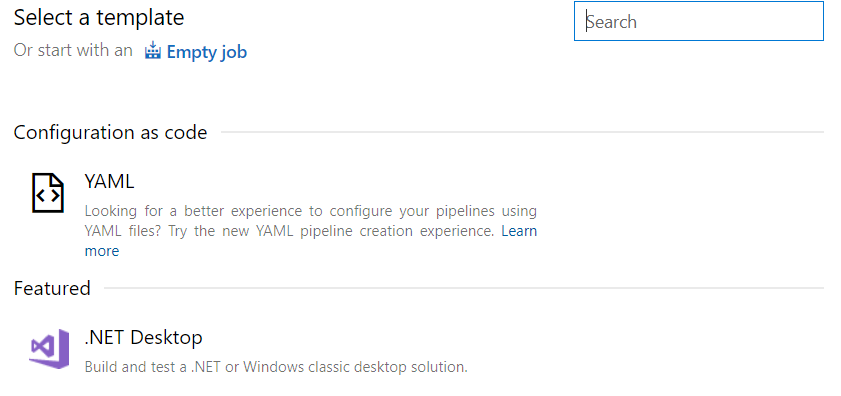
A way easier method is to use the classic editor in all the way at the bottom of the page, press that hyperlink and you will be redirected to the proper link to start your pipeline properly.



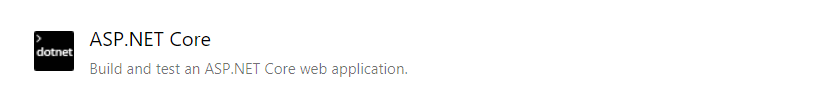
Once there you will be welcomed by this screen, in here there are all the different methods of pulling your data from an external source. The most common is Github itself, but you can use any you like.

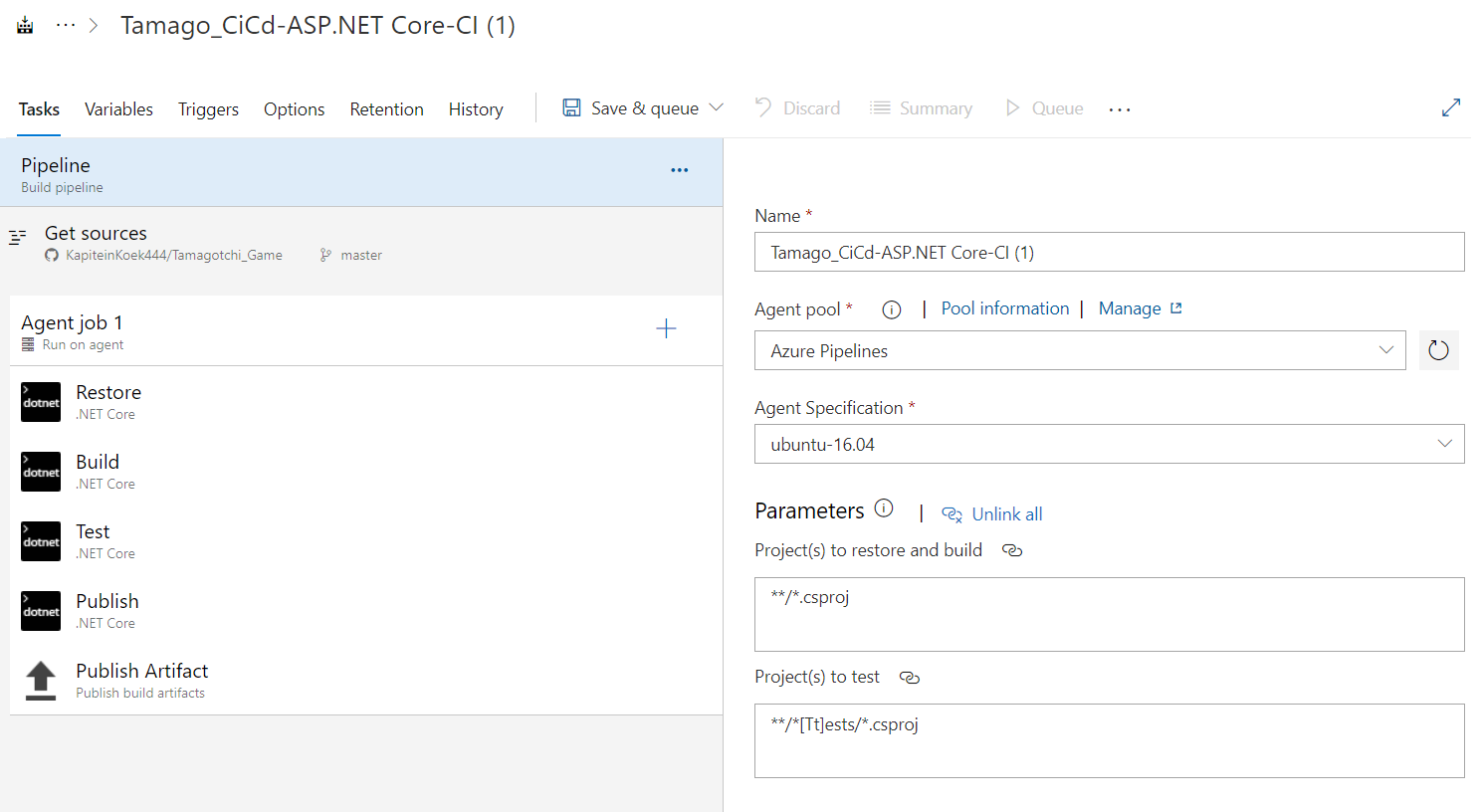


In Github you have to authorize yourself with your Github account and choose a repository as well as a branch of your choise. This way DevOps knows where it has to get its data from.



Next up you are asked to choose a template for your project, you could use Yaml, but this is definitively not the way to go as we can just let azure do everything for you. If you use .Net core you can use the template for Core, otherwise you just need to look up which one is the best for you.



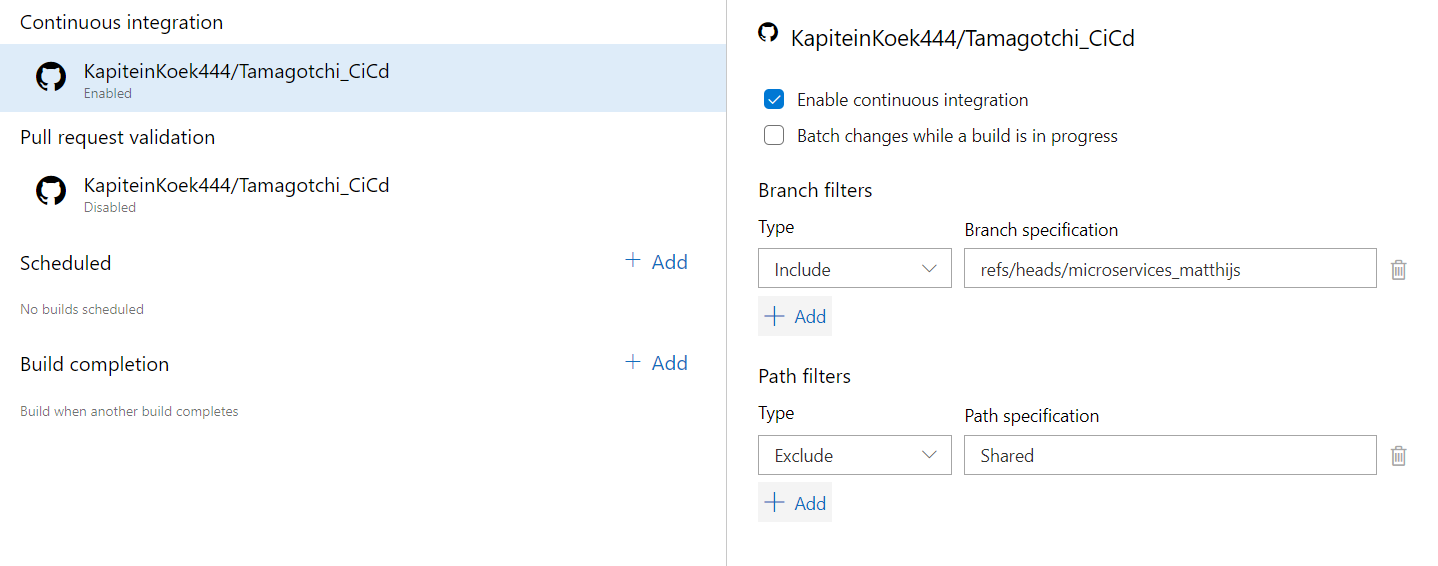


Once you have chosen your template you will be granted with a screen similar to this one, in here there are a lot of different task already built in, you can configure this if you would like but this is not necessary, after you have configured it you need to go to triggers and enable continuous integration. This will make it so that the pipeline activates once something is pushed.

## Exceptions

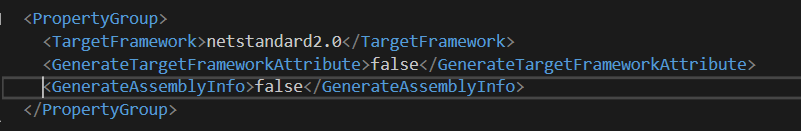
I had some personal problems because I have a solution with multiple projects.

* The first problem was that my shared folder tried to build which would fail automatically, the solution to this is that you can exclude a folder upon building the application.



On the bottom right you can select an in or exclusion if you want.

* The second problem I encountered with my multiproject solution was a duplicate references, but you can solve this problem quite easily by adding a single line to your propertygroup inside of the project itself.



If you add the assemblyInfo line to the group, this problem will be solved.

# Release

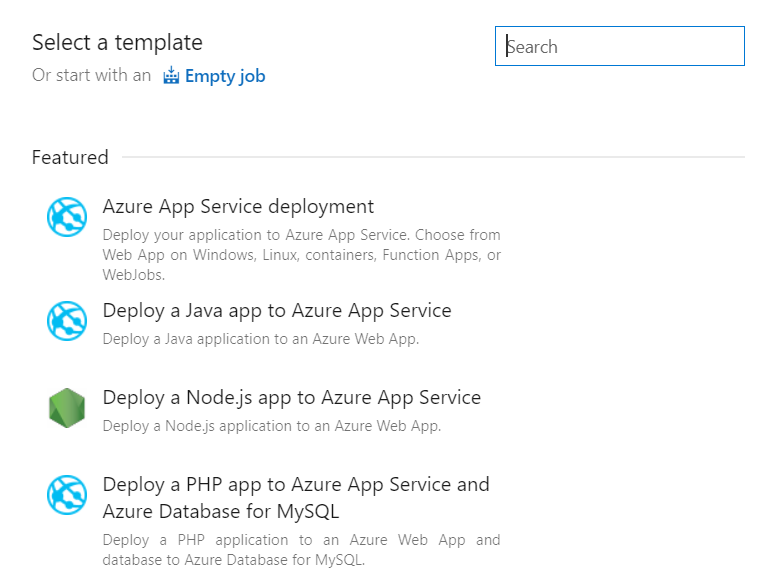
After you created the integration part your application will be pushed automatically to devops when published on a site of your choise. In the Integration part, tests and a build will be covered for you, the only thing left for us to do is to publish the application.

## Adding a release

To add a release you need to go back to the main menu and click on the release menu item, in here you will find a sublink that says ‘releases’. In this page you can make your connection between azure and DevOps.



In here you can see your current release pipelines, at the left under the search bar you can create a new release pipeline.



Once you clicked to create a new pipeline you will be granted with the window above. Here you will see a list with all the different options for publishing a release, we will choose the azure app deployment as this is our way of deploying to azure.

After you choose the action of your choise you will be granted with the framework of your release pipeline, I have an application with multiple projects so mine looks a bit more complicated.

