

Casus

Eén van onze Data Scientists heeft een nieuw ML model opgeleverd, in een Jupyter Notebook. Als Data Engineering-team nemen we deze in beheer. Het model voorspelt de kosten die we als verzekeraar gaan maken bij een patiënt die in het ziekenhuis opgenomen wordt.

In opdracht 2 hebben we een *containerized* versie van dit model gemaakt. Maar hoewel een container in theorie op elke *container host* zou kunnen draaien, staat deze nog steeds op onze ontwikkel-VM. Daarom brengen we het proces nu naar de cloud, zodat we de container daadwerkelijk kunnen gaan “uitrollen”

Globaal stappenplan

Het recept voor het maken van de container hebben we al slim vastgelegd door de Dockerfile. Om het nu centraal beschikbaar te stellen gebruiken we twee tools:

1. Een CI/CD tool die bij nieuwe ontwikkelingen volautomatisch nieuwe container-images maakt: Azure DevOps Pipelines
2. Een plek waarin de uitkomsten (container-images) historisch worden bijgehouden: Azure Container Registry¹

¹ Binnen Achmea gebruiken we hier Artifactory voor. Dit werkt voor het proces echter identiek: in beide gevallen zullen we de container na het maken ervan publiceren naar een registry, waarvandaan we de containers kunnen laten instantiëren: als losse container, of in een cluster als Kubernetes / OpenShift.

Uitwerking

Inloggen op Azure DevOps en Azure

Er is voor iedereen binnen Azure een account gemaakt met de benodigde rechten op Azure DevOps, Azure Container Registry, en een eigen Resource Group waarin als test een Container Instance gehost kan worden.

Inloggen doe je met de volgende gegevens:

Gebruikersnaam: voornaam.tussenvoegsel.achternaam@sigmadatalearning.onmicrosoft.com

Wachtwoord: SigmaDataLearning.nl

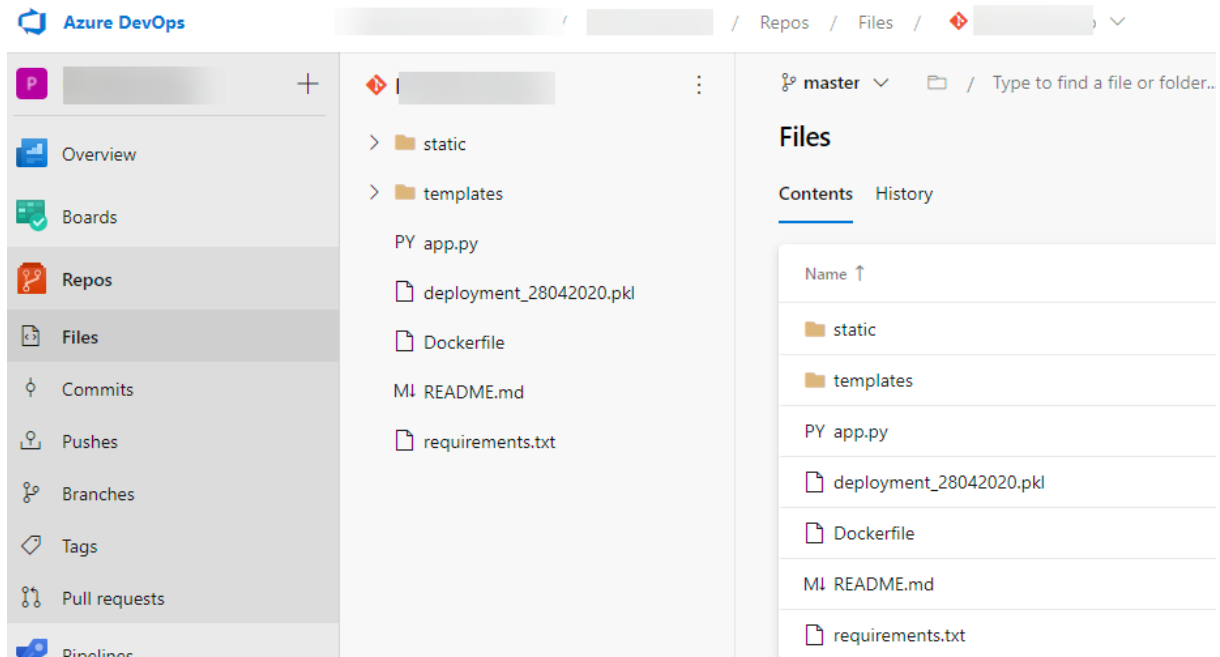
Je kunt het beste inloggen in een InPrivate-, Incognito- of Private-venster. Je hebt dan geen last van je Achmea-account waarmee je eventueel al ingelogd kan zijn.

Binnen Edge of Google Chrome start je dit met de sneltoets Ctrl+Shift+N

Azure DevOps inrichten

We hebben Azure DevOps al twee keer eerder voor ons laten werken. Waar we toen echter wel achter kwamen, is dat we vaak voor data-producten als SQL-databases en Data Factory-pipelines een extra stap (of twee) moeten zetten.

1. Navigeer naar dev.azure.com, en log in met je SigmaDataLearning.onmicrosoft.com-account
Je ziet nu een project dat je eigen naam draait
2. Open dit project, en ga naar Repos
Bekijk de bestanden die hier staan. Zoals je ziet staan hier de bestanden waarmee eerder de container gebouwd werd alvast voor je klaar



3. Navigeer nu naar Pipelines.
Zoals je ziet zijn hier nog geen pipelines aanwezig
4. Klik **Create Pipeline** om de eerste pipeline te maken. Kies voor Azure Repos Git als de plek waar je code momenteel staat.

Azure DevOps ziet nu in je repository twee mogelijkheden:

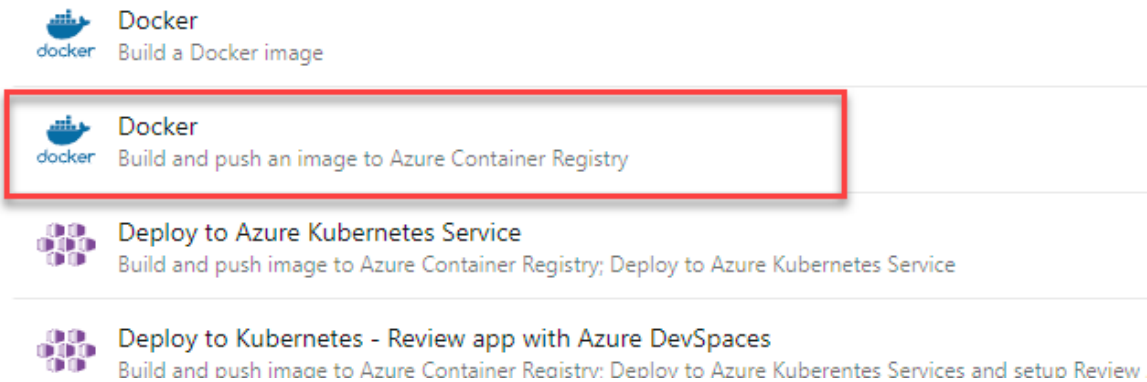
- a. Het is een Python-script
- b. Het is een container

De container kan vervolgens naar keuze worden:

- Gebouwd
- Gepubliceerd naar een *registry* zodat deze door anderen gebruikt kan worden
- Uitgerold op een cluster (Kubernetes of OpenShift)

We kiezen voor nu voor de build + push naar Azure Container Registry:

Configure your pipeline



The screenshot shows the 'Configure your pipeline' interface with three tasks listed:

- Build a Docker image** (Docker icon)
- Build and push an image to Azure Container Registry** (Docker icon, highlighted with a red box)
- Deploy to Azure Kubernetes Service** (Kubernetes icon)

Below these, there is a section for 'Deploy to Kubernetes - Review app with Azure DevSpaces'.

Kies nu achtereenvolgens:

- **Subscription:** Visual Studio Enterprise Subscription – MPN
 - **Container registry:** ACR (gevolgd door je naam)
 - **Image name:** MLOpsContainers
 - **Dockerfile:** niet aanpassen, laat deze staan op
\$(Build.SourcesDirectory)/Dockerfile
5. Klik hierna op **Validate and Configure**, en bekijk de resulterende pipeline.
 6. Klik als laatste op **Save and Run** en wacht tot de pipeline voltooid is

Elke keer dat er een wijziging plaatsvindt in je broncode, zorgt dit nu automatisch voor een *trigger* van deze pipeline – én dus voor een nieuw container-image dat in Azure een plek krijgt.

Azure Container Registry

In de Azure Portal staat voor iedere deelnemer een eigen Azure Container Registry gereed. Dit is vergelijkbaar met Artifactory binnen Achmea: Data Engineers die toegang hebben tot een registry hebben de mogelijkheid om een kant-en-klare container te gebruiken.

Dat kan in de basis op drie manieren:

- a. Door de container op een lokale machine of ontwikkelserver “los” uit te voeren (bijvoorbeeld met *docker run*)
- b. Door een deployment te doen naar een Kubernetes-cluster, bijvoorbeeld AKS of OpenShift.
- c. Door een losse Azure Container Instance op te spinnen

Optie 1: containers los uitvoeren

De eerste optie hebben we feitelijk al gezien in de vorige opdrachten: we hebben daar een bestaande container gepakt en lokaal uitgevoerd. Door de manier waarop een container werkt kan dit enorm handig zijn om een stukje ontwikkelwerk te doen: alle afhankelijkheden en juiste softwareversies voor dat stukje software zijn altijd gegarandeerd aanwezig.

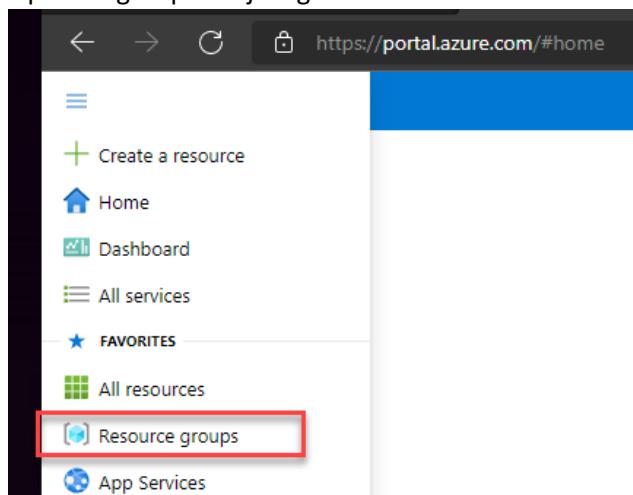
Optie 2: Kubernetes-cluster

Eén van de grote voordelen van containers, is dat een verzameling van servers flexibel inzetbaar is voor een hele diverse workload. Dat is weliswaar waar we naartoe willen, maar niet wat we binnen deze opdracht al gaan doen.

Optie 3: een losse Azure Container Instance

Om te bewijzen dat “onze” ML app goed werkt, willen we voor nu alleen maar een kleine container instance “los” draaien. Dat doen we op Azure met een zogenaamde *Container Instance*. Dit is niet automatisch schaalbaar, kan alleen handmatig beheerd, maar is wel heel flexibel en klein op te zetten. En dat is precies wat we zoeken.

7. Navigeer naar portal.azure.com
8. Zoek je eigen Resource Group op (in het hamburgermenu kies je voor Resource Groups). Open de groep met je eigen naam.



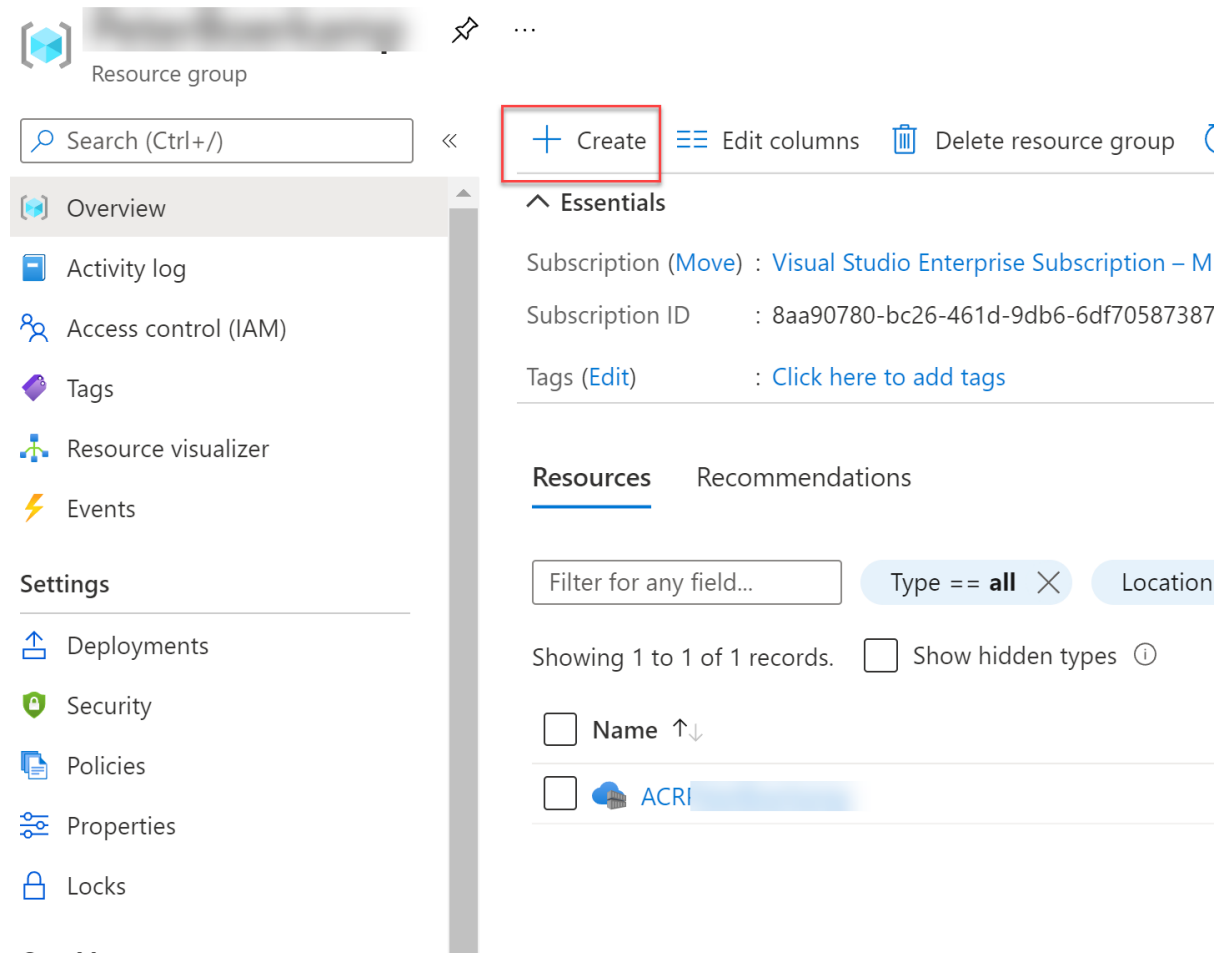
Je ziet nu één resource: een Azure Container Registry, genaamd *ACRjouweweigennaam* (dus bijvoorbeeld ACRJanJansen).

9. Bekijk de Container Registry, en kijk of je de container die zojuist hierheen *gepusht* is kunt terugvinden.

Als je het leuk vindt, kun je de container ook in Docker binnen je VM *pullen* en vanuit de cloud ophalen. Dit is echter optioneel en verdiepend!

Om nu de container als *instance* neer te zetten, doe je de volgende stappen:

10. Navigeer nu opnieuw naar je eigen Resource Group, en klik linksboven op Create.



11. Zoek naar **Container Instances**, en voeg er hier één van toe. Geef de volgende details mee:

- Container name: mlopstest
- Region: West Europe
- Image Source: Azure Container Registry
- Laat de overige instellingen staan zoals ze standaard worden ingevuld:

Subscription * ⓘ Visual Studio Enterprise Subscription – MPN

Resource group * ⓘ
[Create new](#)

Container details

Container name * ⓘ mlopstest ✓

Region * ⓘ (Europe) West Europe ✓

Image source * ⓘ
☐ Quickstart images
☒ Azure Container Registry
☐ Docker Hub or other registry

Registry * ⓘ ACR
 Image * ⓘ
 Image tag * ⓘ

OS type Linux

Size * ⓘ 1 vcpu, 1.5 GiB memory, 0 gpus
[Change size](#)

12. Klik op **Next: Networking** en vul hier de volgende zaken in:

- **DNS name label:** mlopsjouwnaam
- **Ports:** voeg de poort **5000** toe, met het **TCP** protocol

Basics **Networking** Advanced Tags Review + create

Choose between three networking options for your container instance:

- **'Public'** will create a public IP address for your container instance.
- **'Private'** will allow you to choose a new or existing virtual network for your container instance. This is not yet available for Windows containers.
- **'None'** will not create either a public IP or virtual network. You will still be able to access your container logs using the command line.

Networking type ☒ Public ☐ Private ☐ None

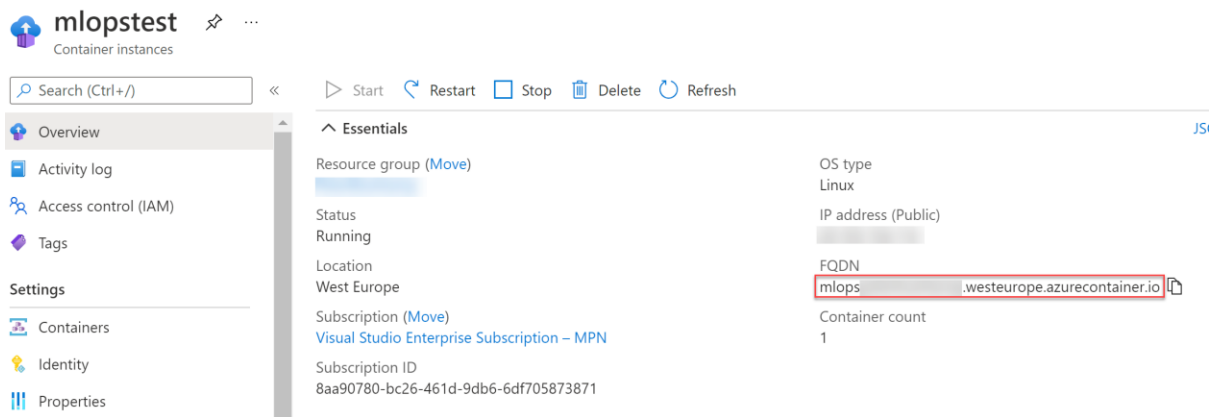
DNS name label ⓘ mlops ✓
 .westeurope.azurecontainer.io

Ports ⓘ

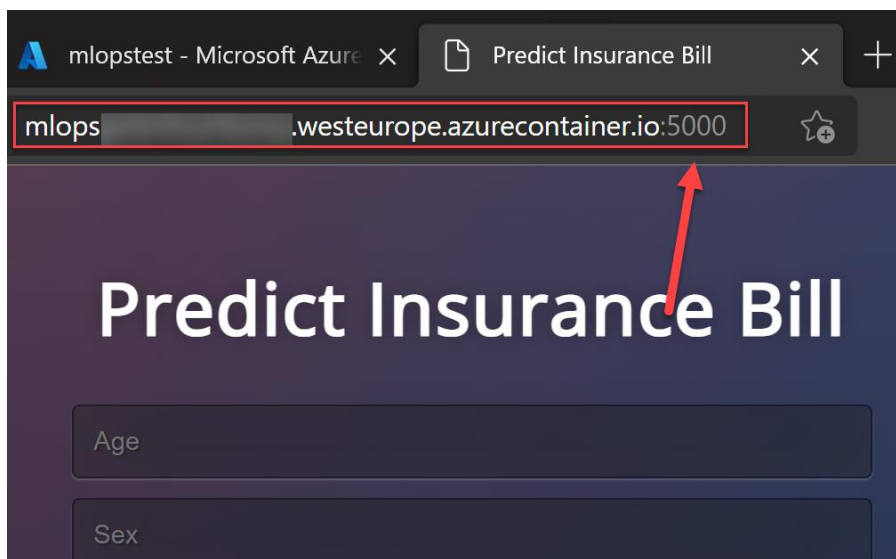
Ports	Ports protocol	
80	TCP	
5000 ✓	TCP ✓	

13. Klik op **Review + Create**, en maak de container aan

Wacht tot de melding **Your deployment is complete** er staat, en klik op **Go to resource** om de Container Instance te bekijken in de Azure Portal



14. Kopieer nu de FQDN (Fully Qualified Domain Name), en open deze in een nieuwe tab in je browser **gevolgd door de tekst :5000**



15. Verifieer dat de app inderdaad werkt zoals verwacht door het formulier in te vullen en de resultaten te bekijken.