# **CI/CD Best practices using Azure:**

***Deployment on the VM using Docker containers:***

# **Requirements( resources ):**

1. Azure Devops
2. Azure Container Registry

***Azure Devops:***

1. **Repository service**:

We need this service for the version control of our code and to maintain the code safe and secured inside the repo. We can use any repo service such as Azure Repos, Github, Gitlab or Bitbucket. We just need to map when we are creating the pipelines.

1. **Build and Release Pipelines:**

We can have a single pipeline here which depends on the user or the architecture how it needs to be designed. As a **standard** practice pipelines are segregated as two parts one is **build** and another one as **release**.

These pipelines are triggered when any changes to the application is made and the code should be updated or deployed on the **VM’s** without any manual changes to the code on the **VM.**

**Build pipeline:** Usually we design the pipelines using YAML scripts. Whenever we are building pipelines, we should have different stages so that where we can perform check on code quality, authentication, testing, etc. As many stages as we desire basically predefined checks before completing the whole build process and **updating data**/**deploying** or **images** on the **container registry** and the same applies to code deployment on the **VM** and waits for the release.

**Release Pipeline:** So, here in this pipeline we usually wait for the approvals from the reviewers, business owners or anyone who validates the changes which needs to be deployed. We design this pipeline just to **pull the latest changes** which is waiting in **container registry** into the **VM**. Or triggering the updated code on **VM**. Here we can also maintain the release versions, release date, release features.

***Azure Container Registry(ACR):***

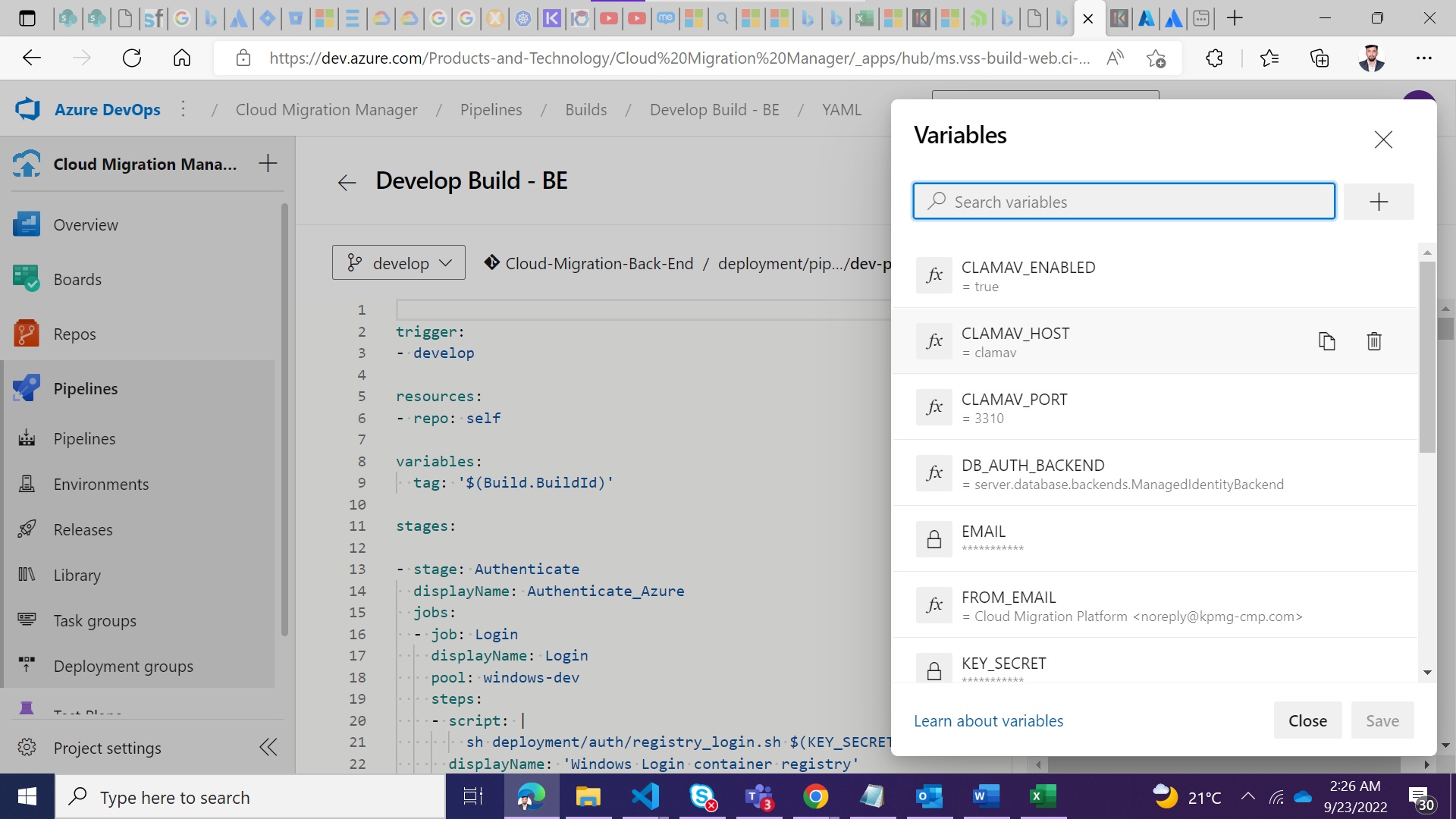
Container Registry allows you to build, store, and manage container images and artifacts in a private registry for all types of container deployments. As we are using Docker and container service to build the application, we are using the private container registry to store the containers. So, these images can be later used for the deployment.

**Setup CI/CD pipelines on Azure devops:**

1. Create Build Pipeline
2. Create Pipeline for release
3. Create Agent Pool on Azure Devops
4. Host the agent on the VM for the communication between the Azure devops and the VM.

**Steps to create Pipeline and setup agent on VM:**

1. **Create pipeline:**

****

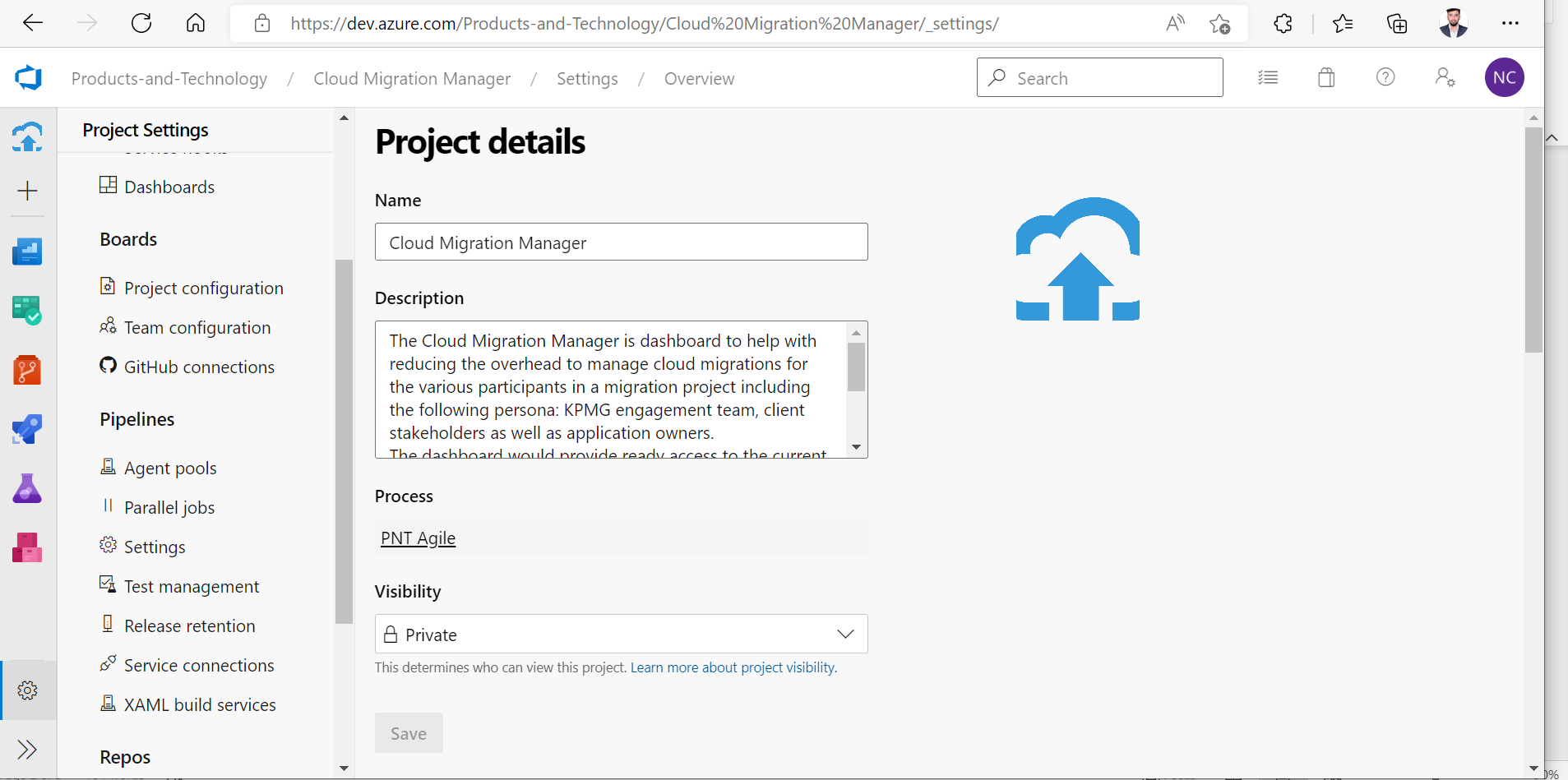
First step is to create a build for Backend or Frontend

Creating a build pipeline on azure devops. This can be configured having the pipeline.yml file configured. We describe all the commands and stages the build should run to complete the whole deployment process.

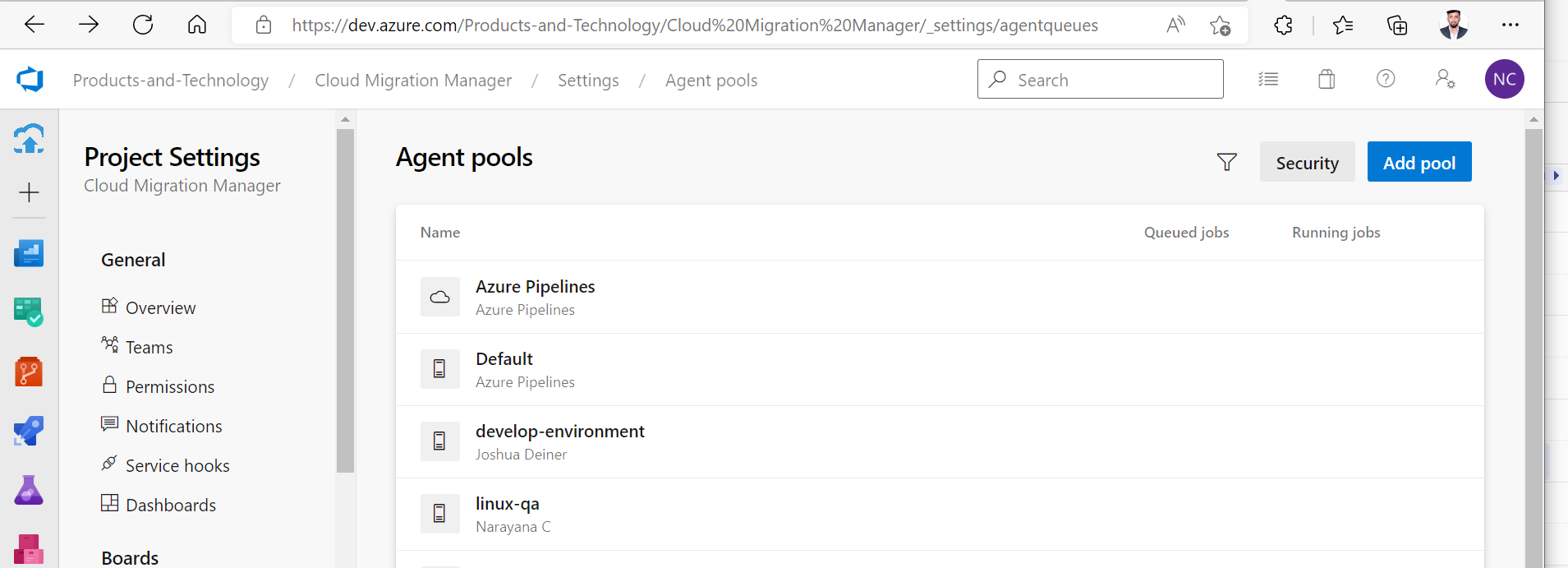
Also, we can set the environment variables which is required by the project as shown in the above image.

1. **Create a agent pool:**

Go to Project settings and click on Agent Pools on the sidebar menu as show in the below screenshot.

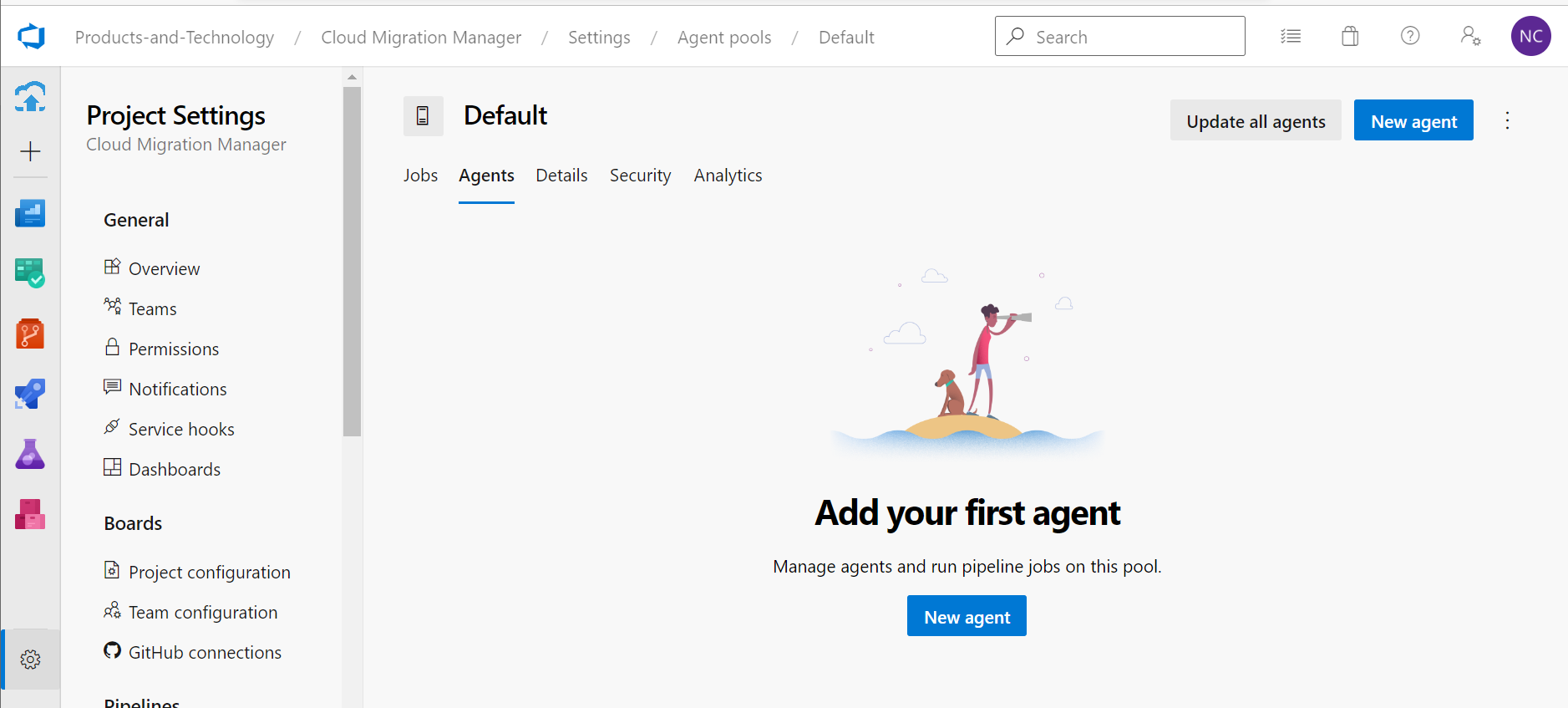
****

You can use the **default** agent pool or you can create a new pool which is dedicated for a single project. So, click on the default agent or create a new agent pool by clicking add pool on the top right.



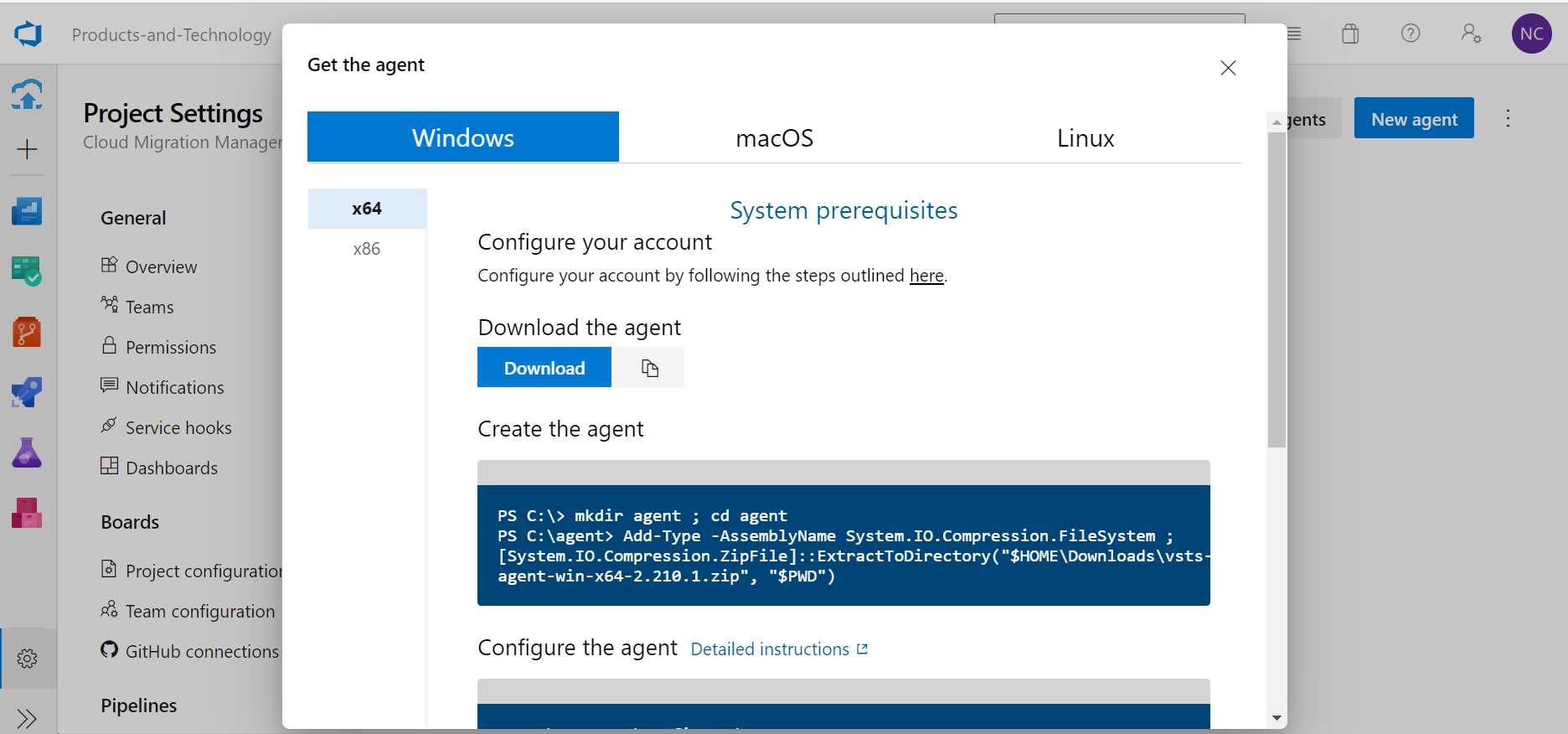
1. **Create an agent:**

Now after getting into one of the agent pool. Now you can navigate to **agents** tab as shown in the below screenshot and click on the **New Agent** button



Follow the steps as directed, you should be successfully able to create an agent on a virtual machine. There is two ways in creating the agent on Virtual machine. One is as shown below depends on the OS which you are using OR if you are using a container service such as docker you can create a docker agent which serves the same purpose but more stable compared to normal agent. To create a self hosted dockeragent refer the below link, on how to setup the docker agent on virtual machine.

<https://learn.microsoft.com/en-us/azure/devops/pipelines/agents/docker?view=azure-devops>



You can refer this link to create a CI/CD pipeline on azure devops which shows how it is done.

[Tutorial: Using Azure DevOps to setup a CI/CD pipeline and deploy to Kubernetes (microsoft.com)](https://cloudblogs.microsoft.com/opensource/2018/11/27/tutorial-azure-devops-setup-cicd-pipeline-kubernetes-docker-helm/)

1. **Use the created agent pools and agents in the Yaml file of the Project:**

As now the complete setup of agent pools and agents are done. Now we should map the created pipeline resources in our project on a pipeline.yml file.