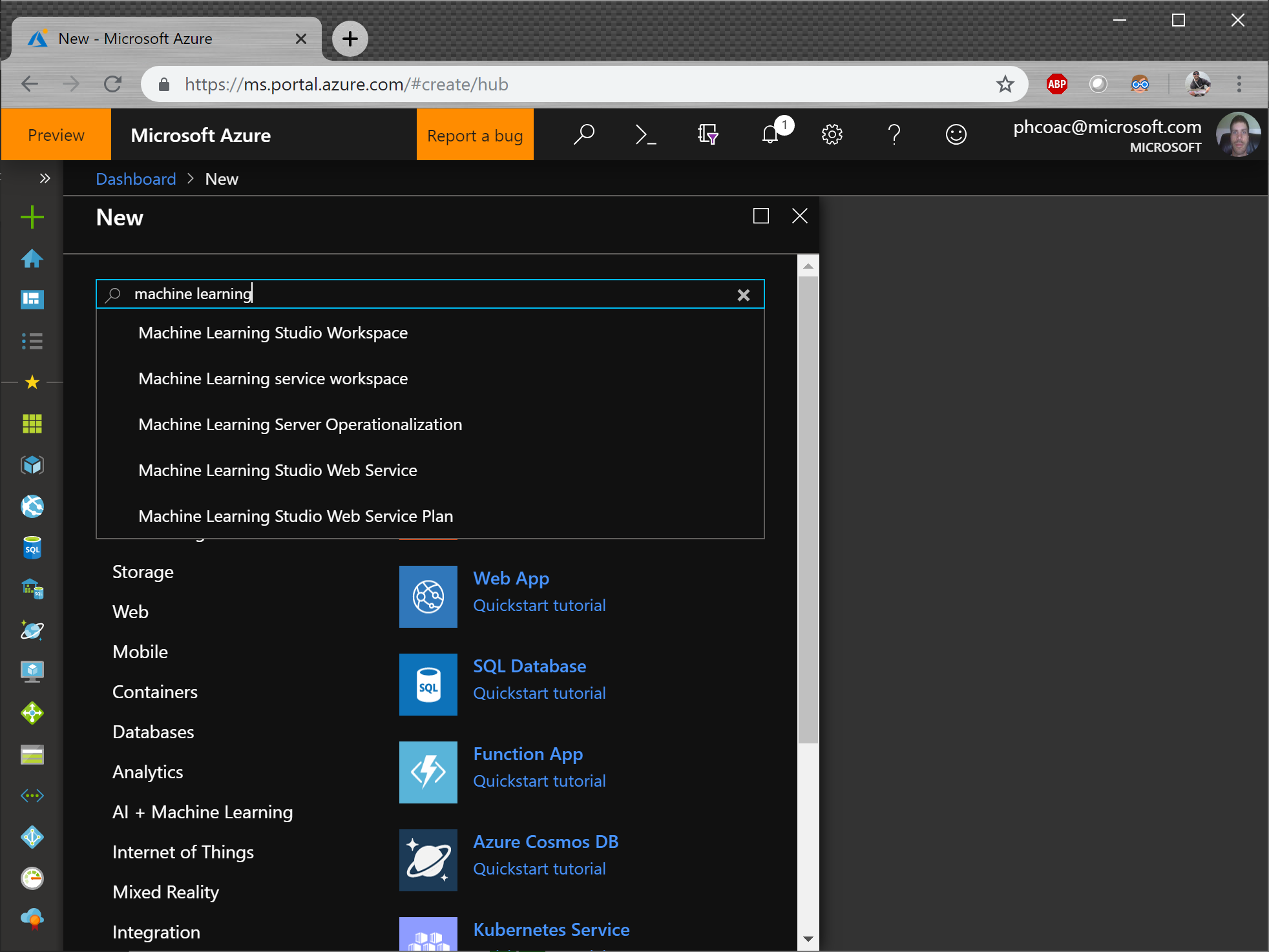
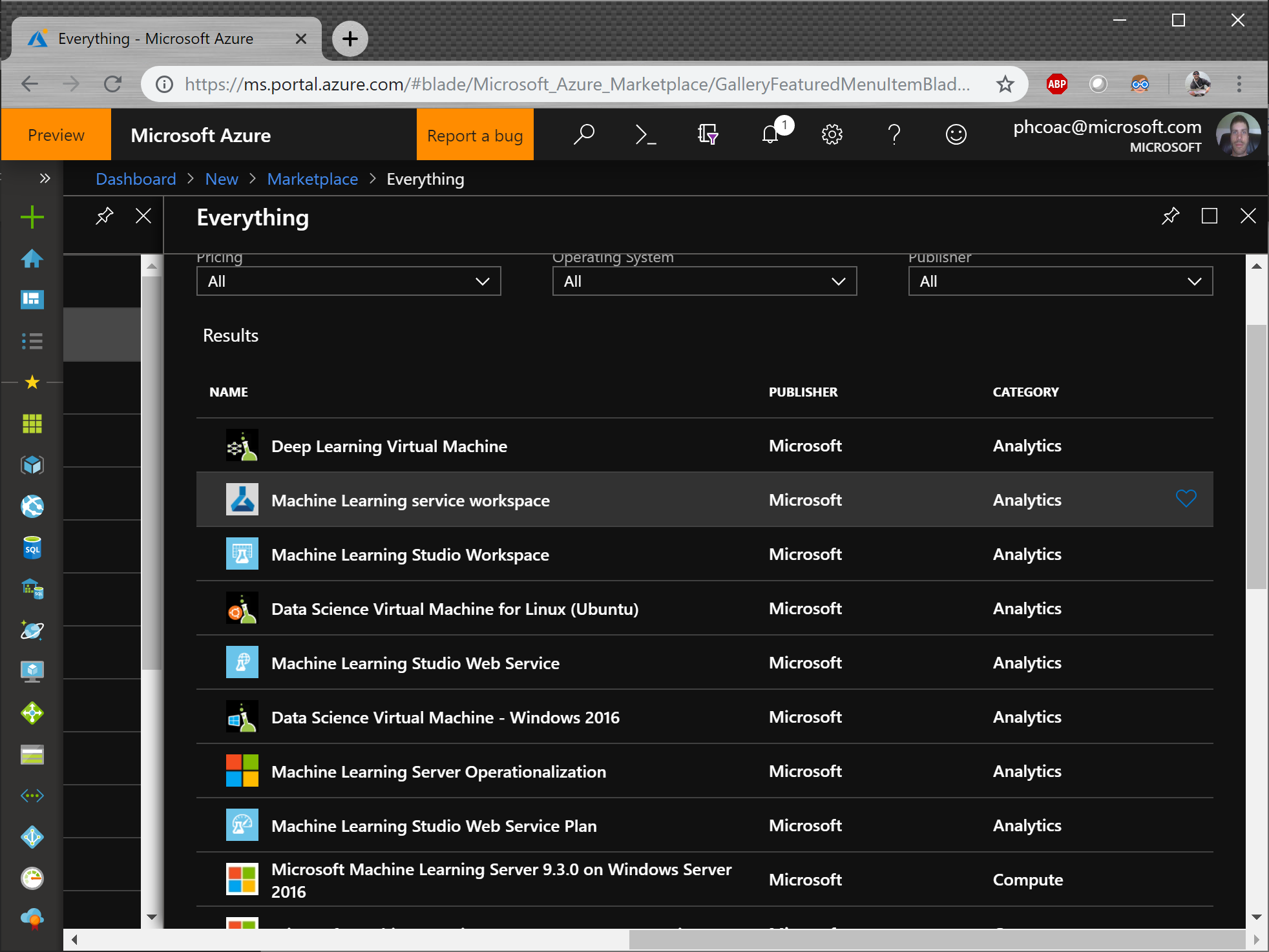
# Part 1 – Setup Environment

## Setup AzureML Services Workspace

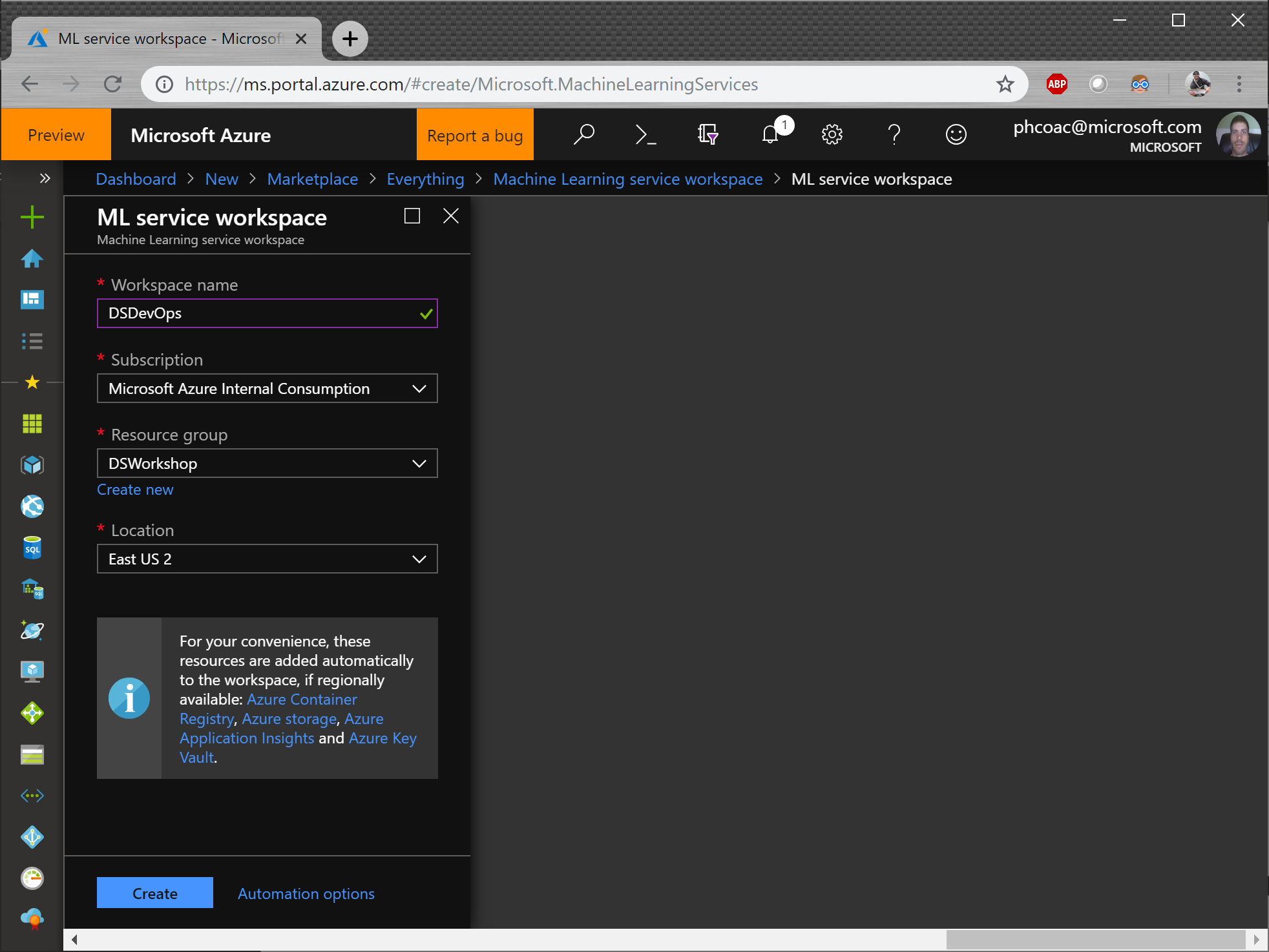
1. Navigate to <http://portal.azure.com>. Click Create a Resource and enter “machine learning” in the search box



1. Choose Machine Learning Service Workspace and click Create

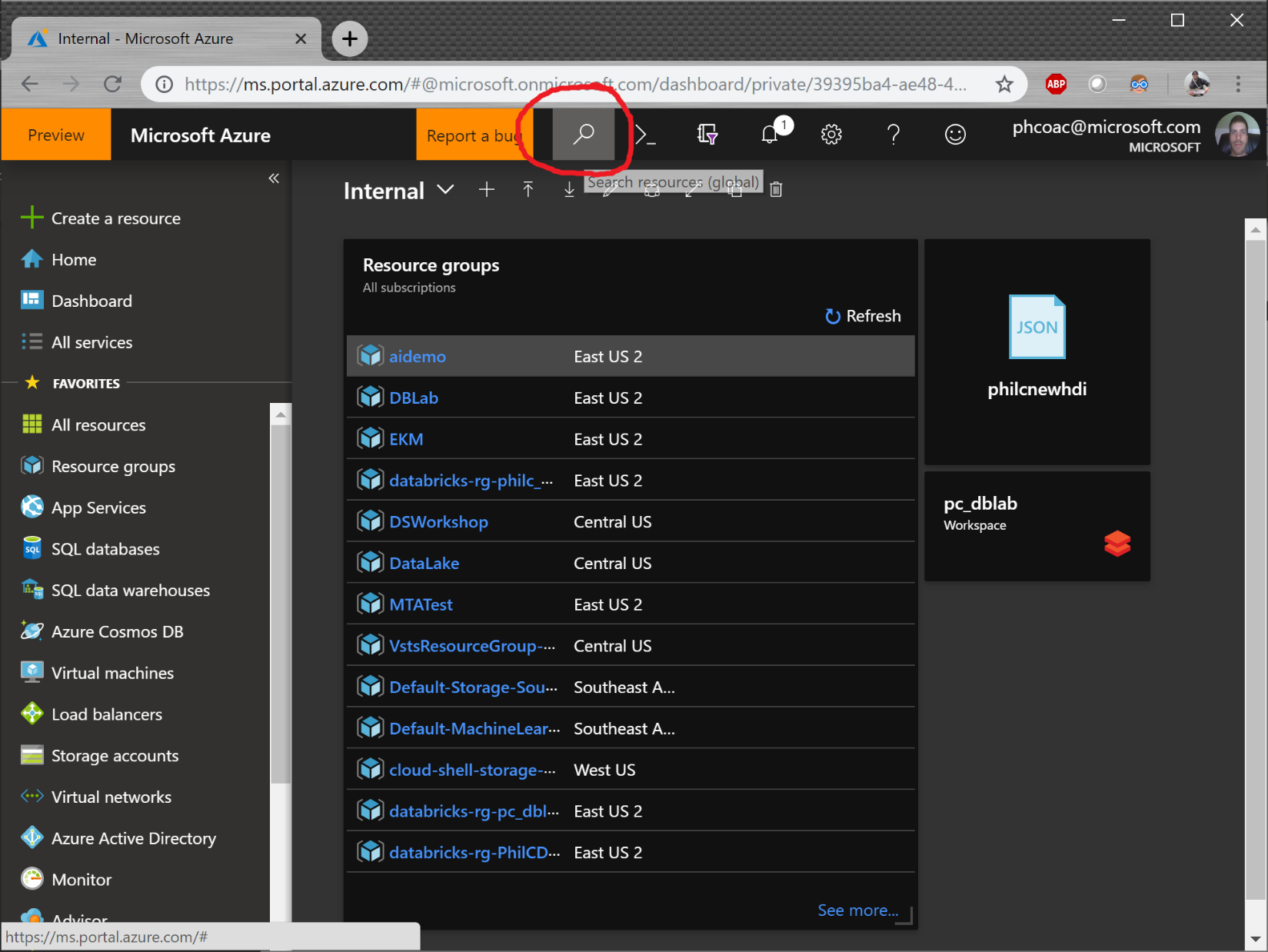


1. Enter a workspace name, the resource group and region, and click Create

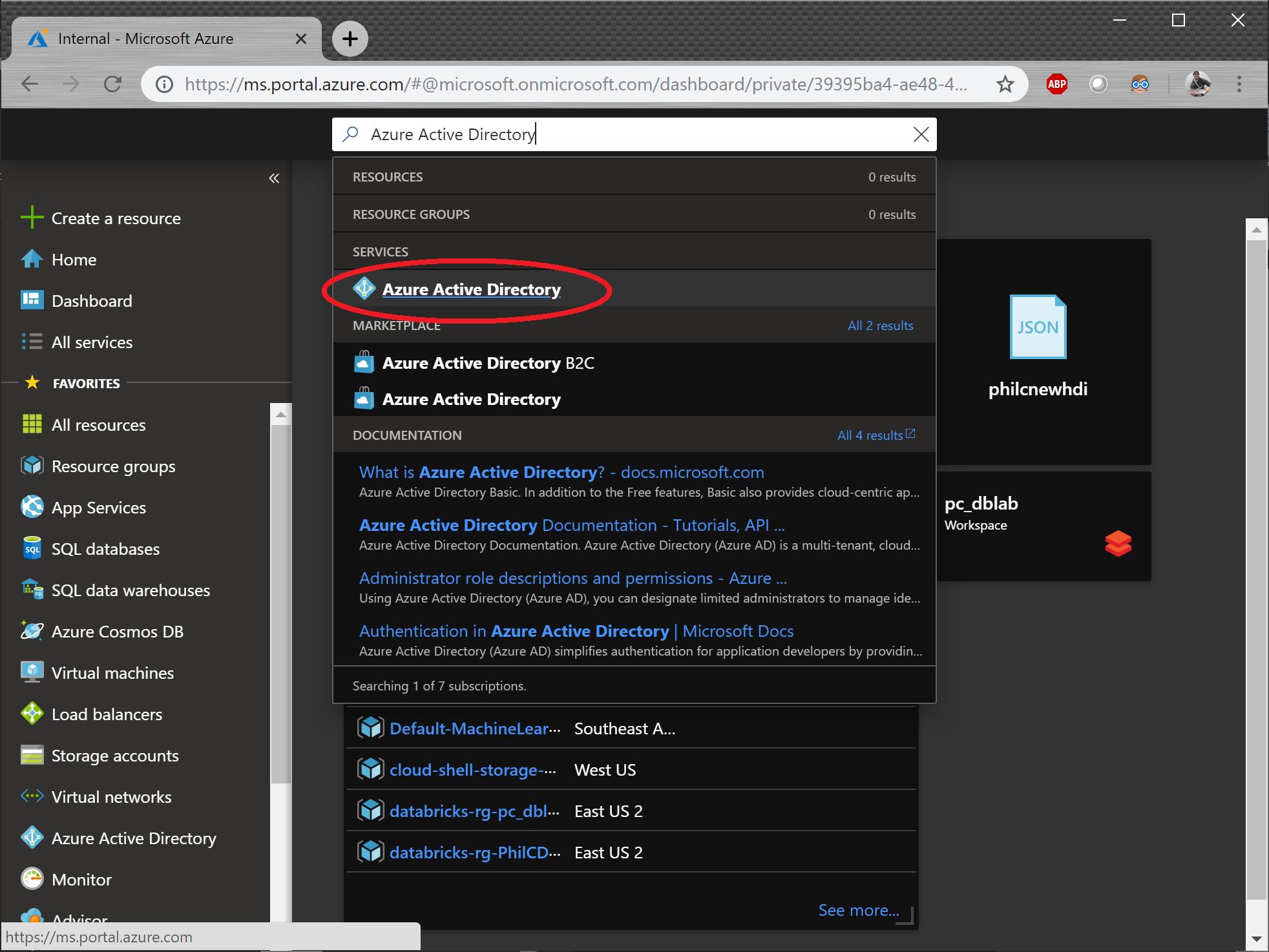


## Setup Service Principal

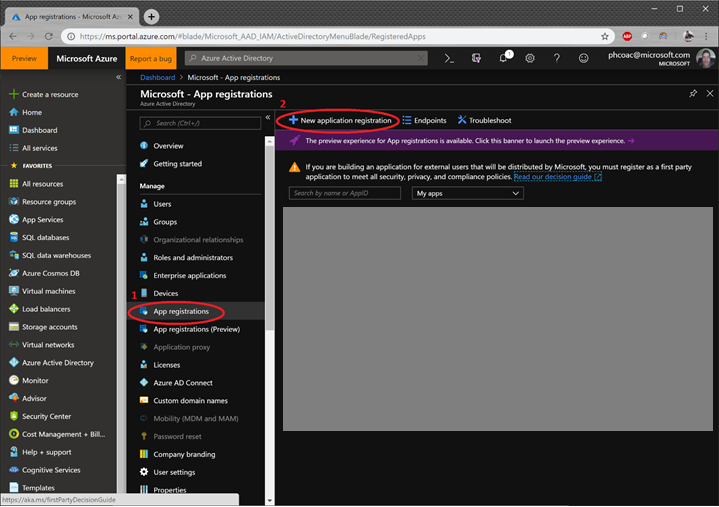
1. In the Azure Portal (<http://portal.azure.com> ), click the Search button on the top bar



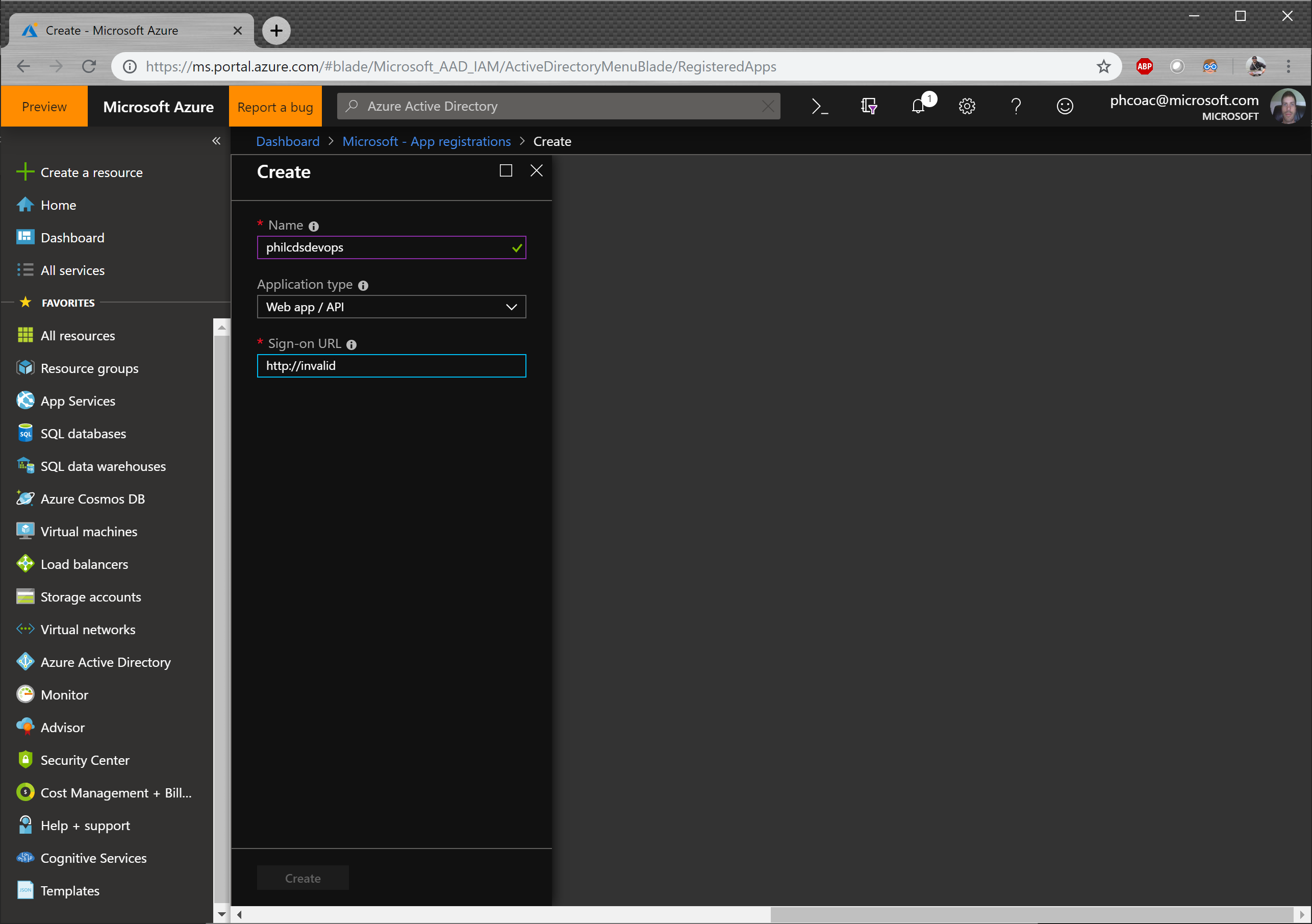
1. Search for “Azure Active Directory” and select the option under Services:



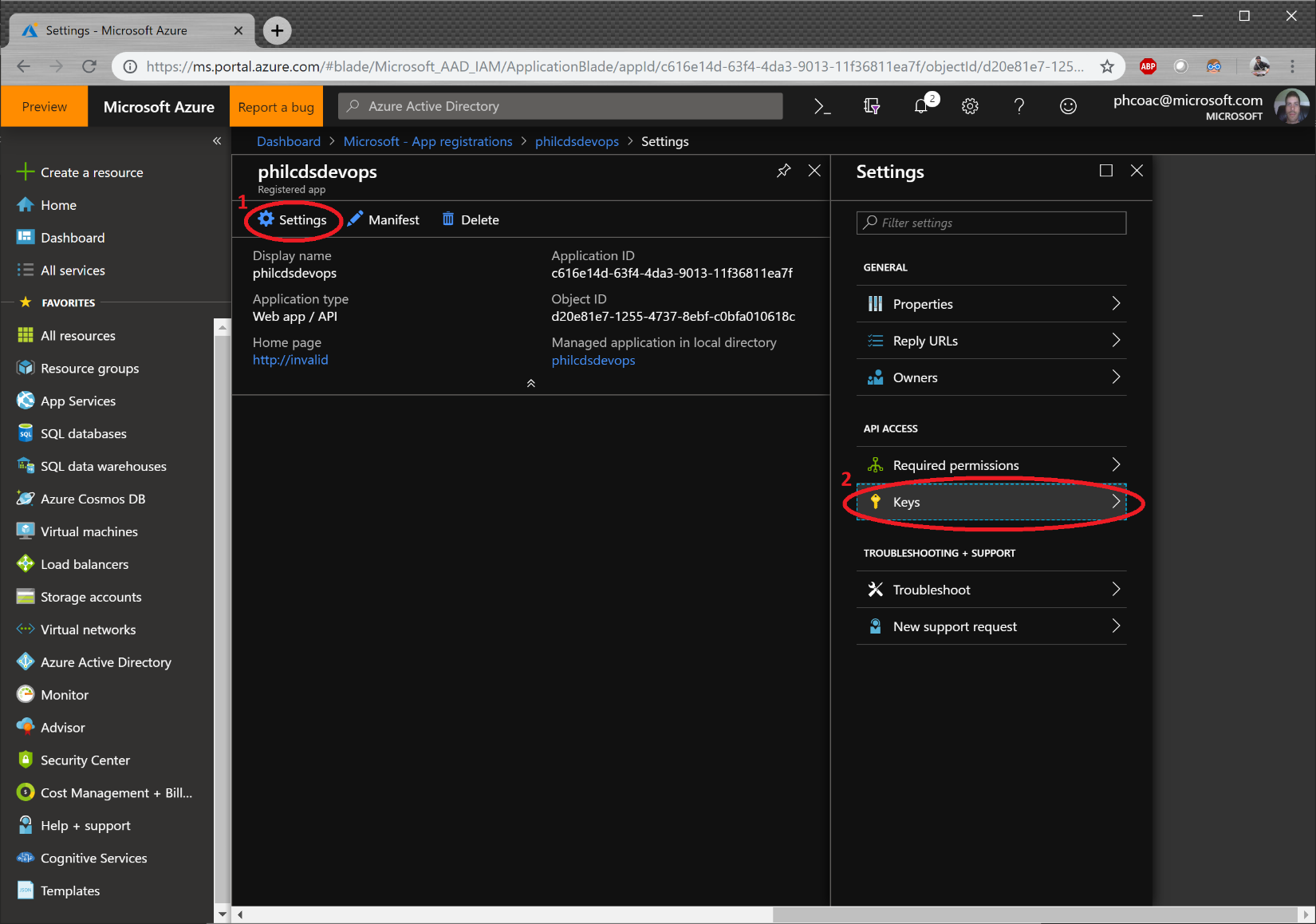
1. Click “App Registrations” on the left, then click “New application registration”



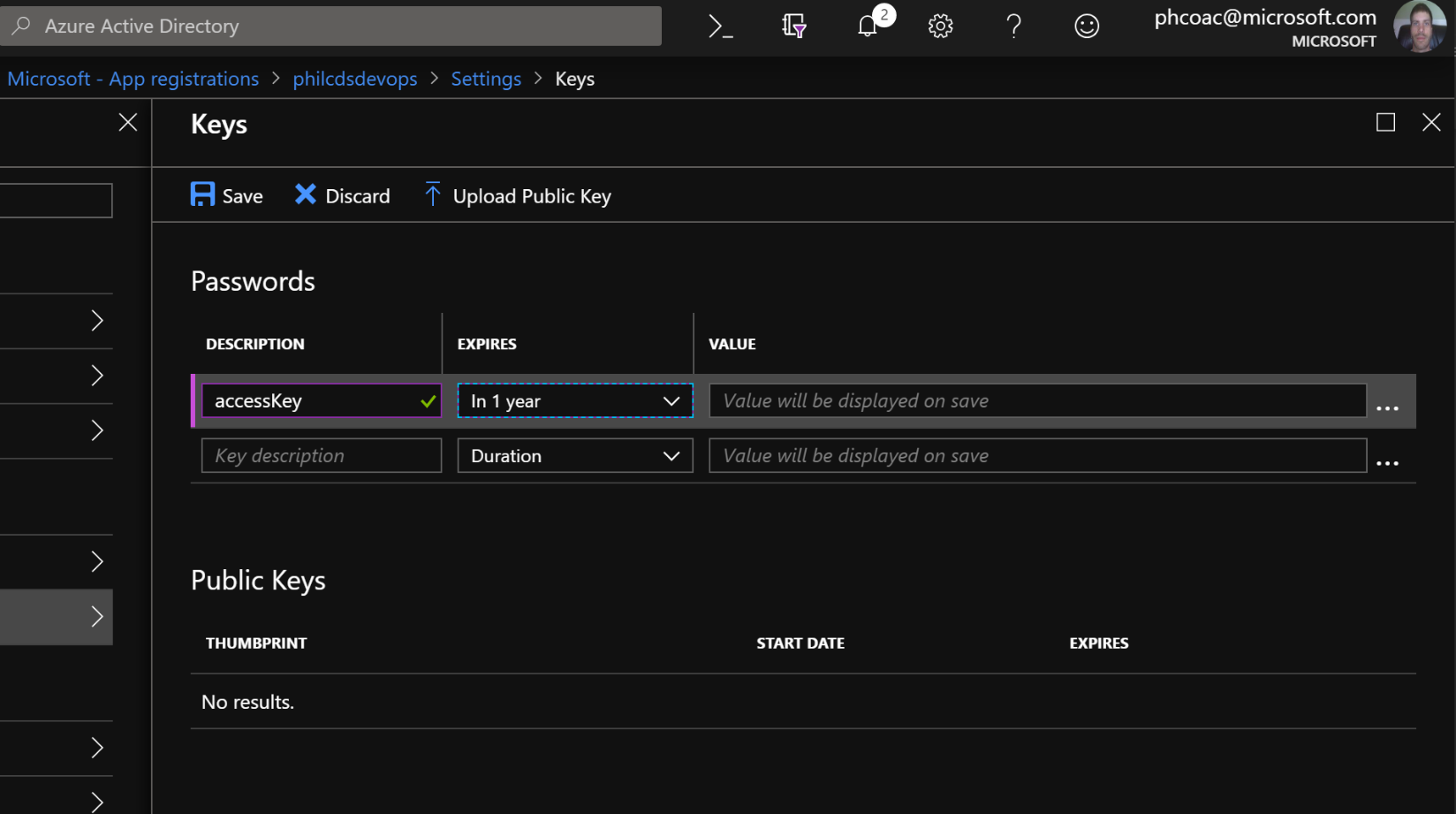
1. Enter the name of your application, and a URL. For convenience and ease of tracking, we suggest you use you’re a unique prefix to your service principals. The URL doesn’t have to be a real URL, you can enter <http://invalid>



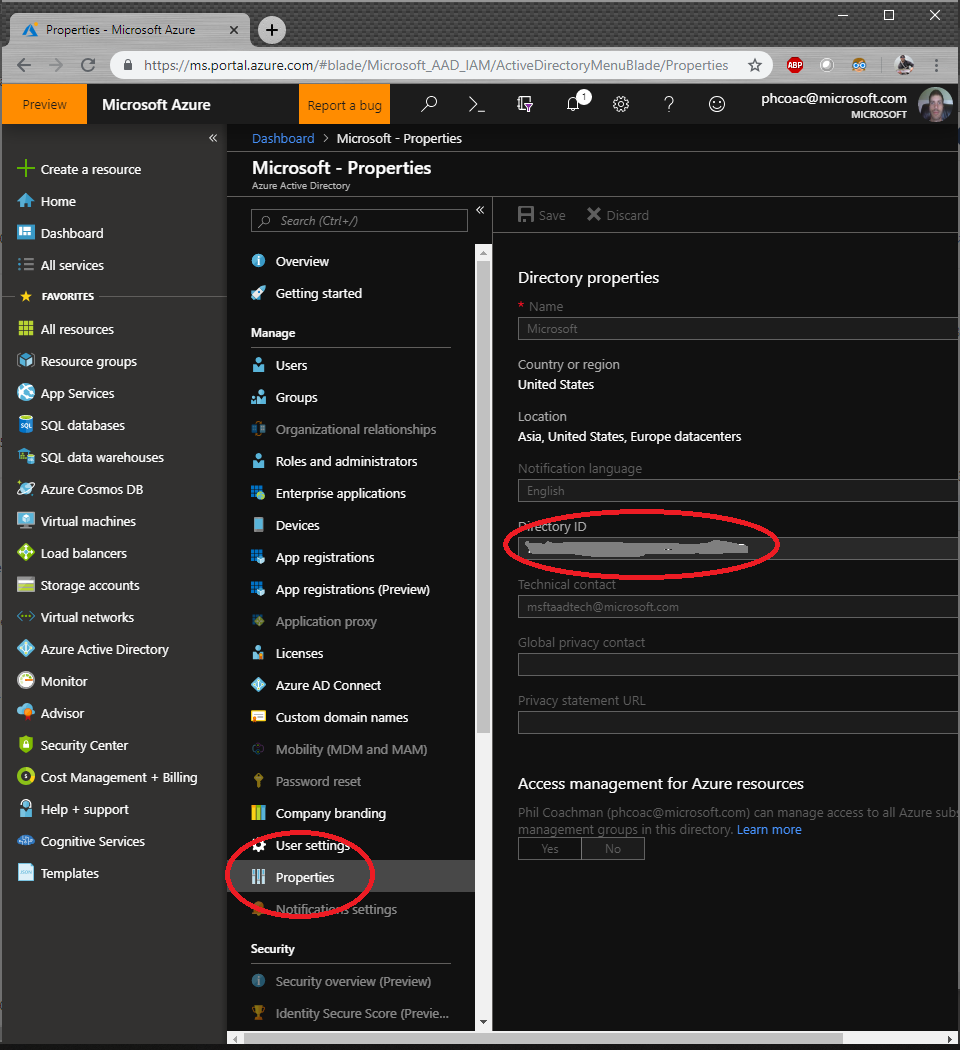
1. Once complete, click on the Settings button for your service principal then click Keys



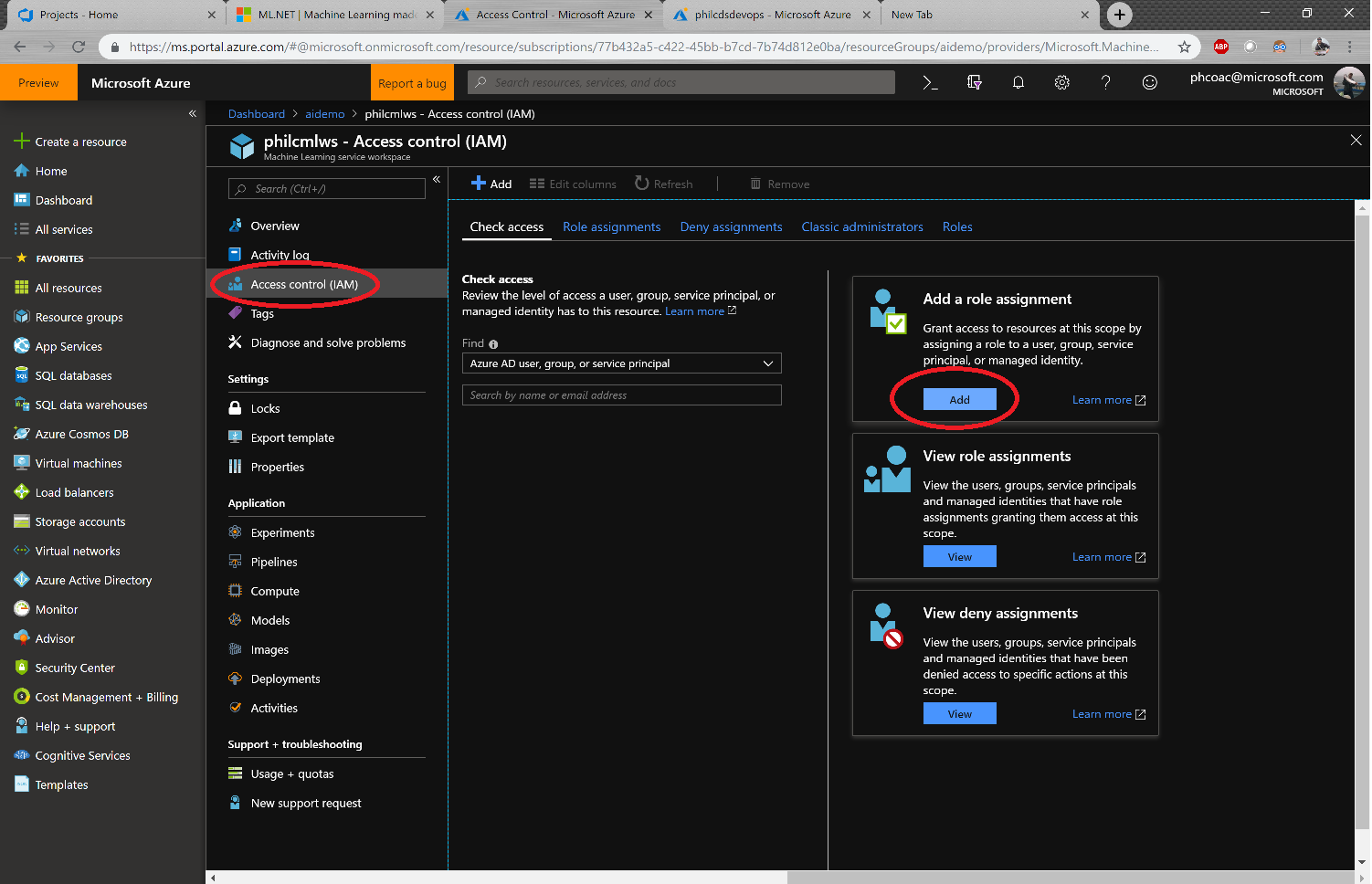
1. Here we will create a password for your service principal. Enter “accessKey” for the description, and 1 year for the expiration. Then click Save. Once you click save, the password will be displayed only once. Copy the password before leaving the blade, otherwise you won’t be able to see it again. Be sure to copy the Application ID as well, as we’ll need that later on.



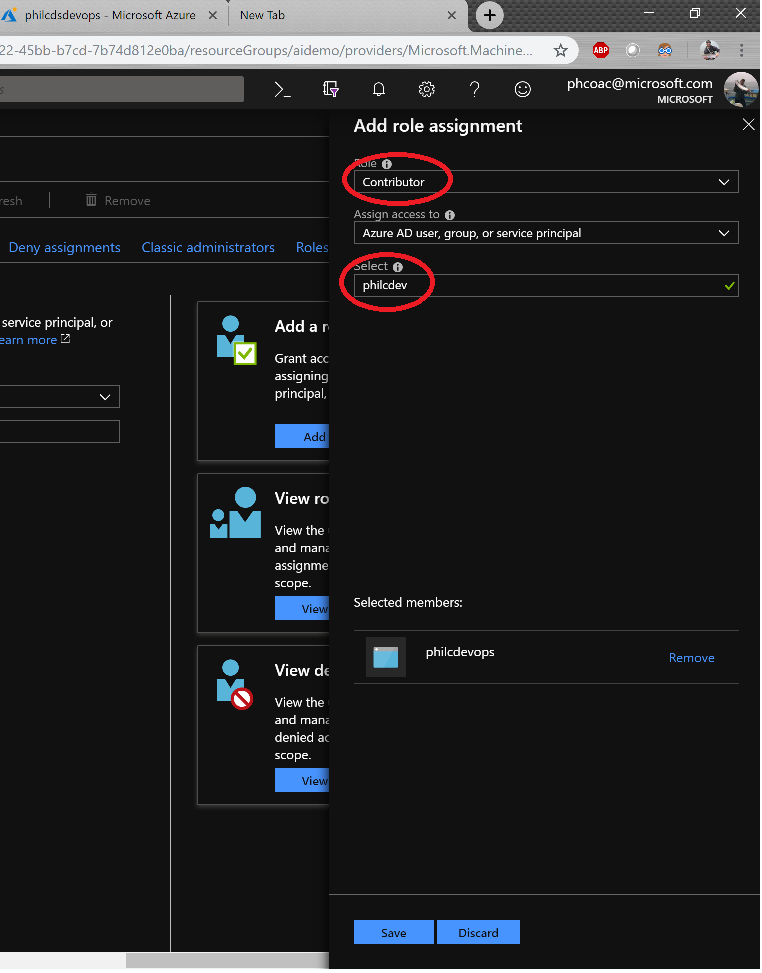
1. Retrieve the Tenant ID. We’ll need this later. On the left-hand blade, click “Properties” to show the properties of your Azure Active Directory. Copy the Directory ID. This is your Tenant ID.



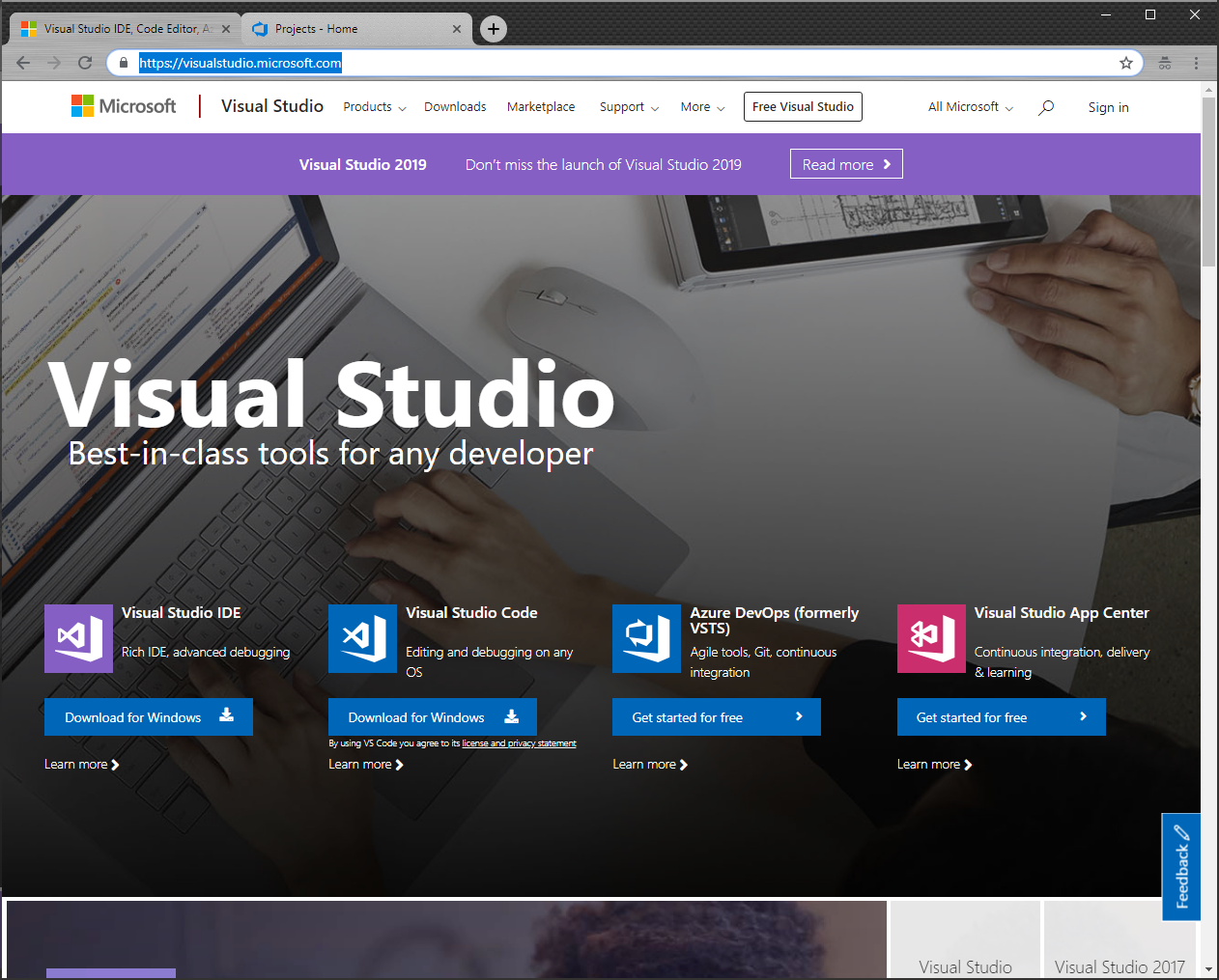
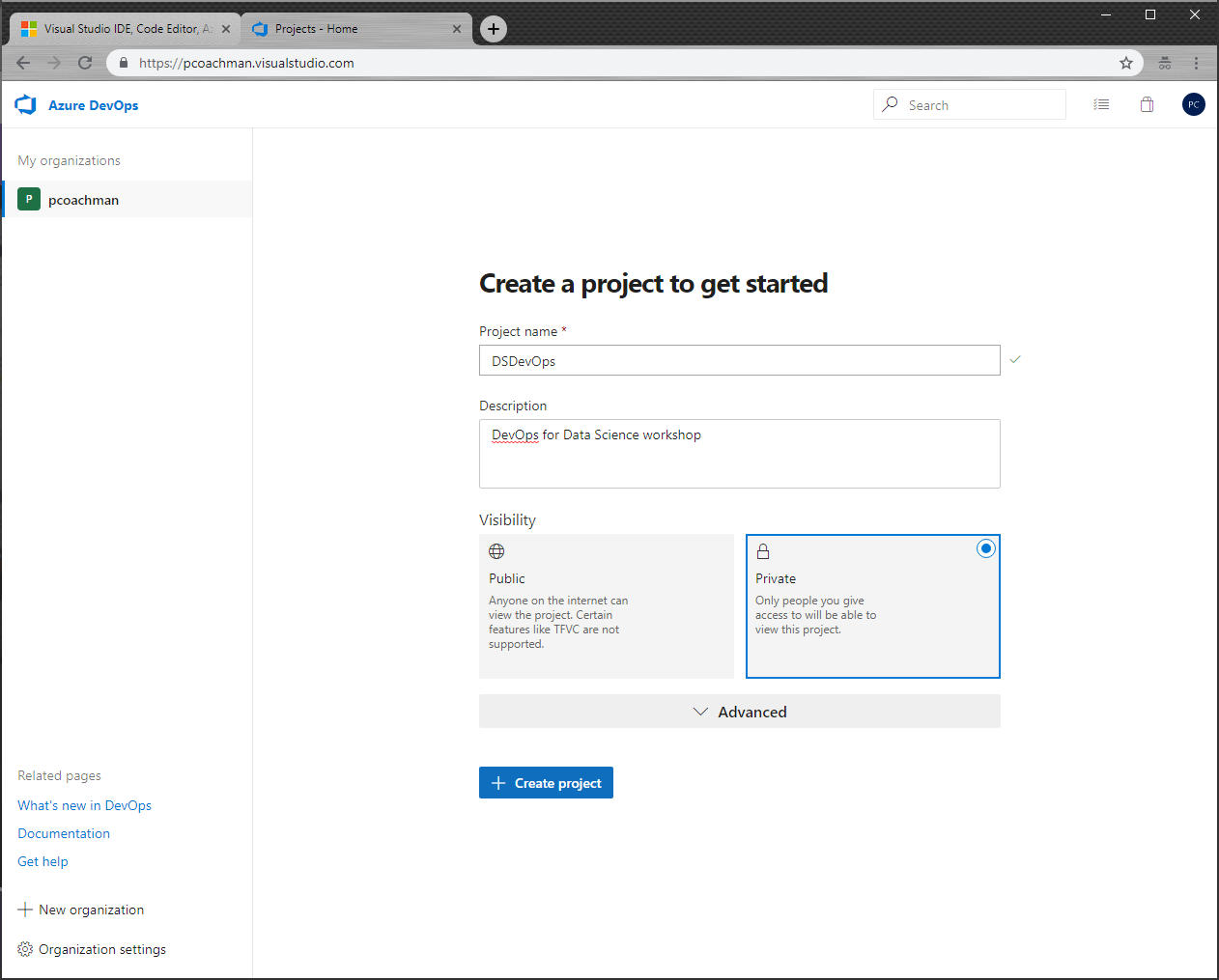
1. Now we need to give access for the Service Principal to use the AzureML Workspace. Open the AzureML Workspace you created, click Access control (IAM) and Add a role assignment to add a user to a role.

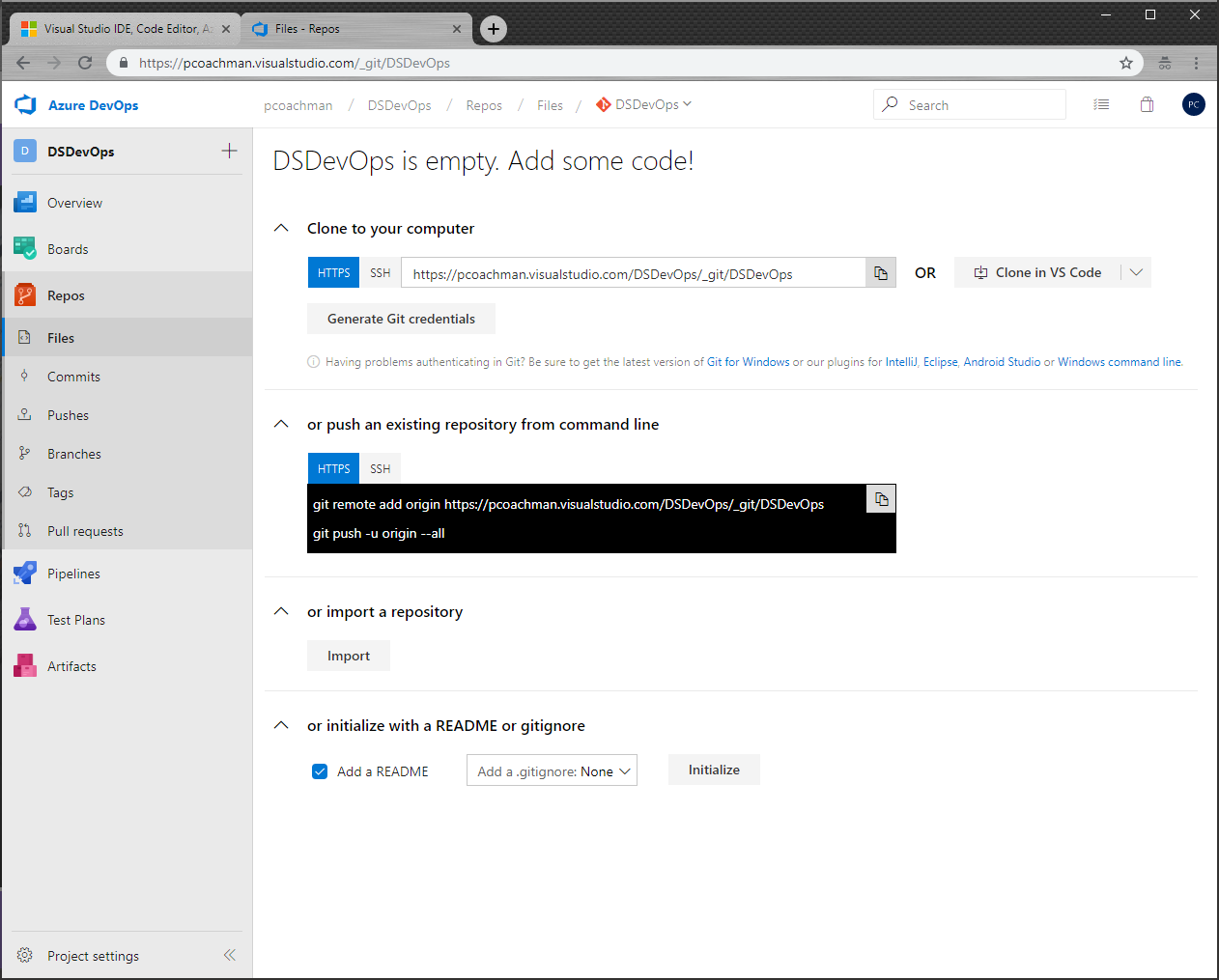


1. Choose Contributor for the role and enter the name of the service principal you created, then click save.

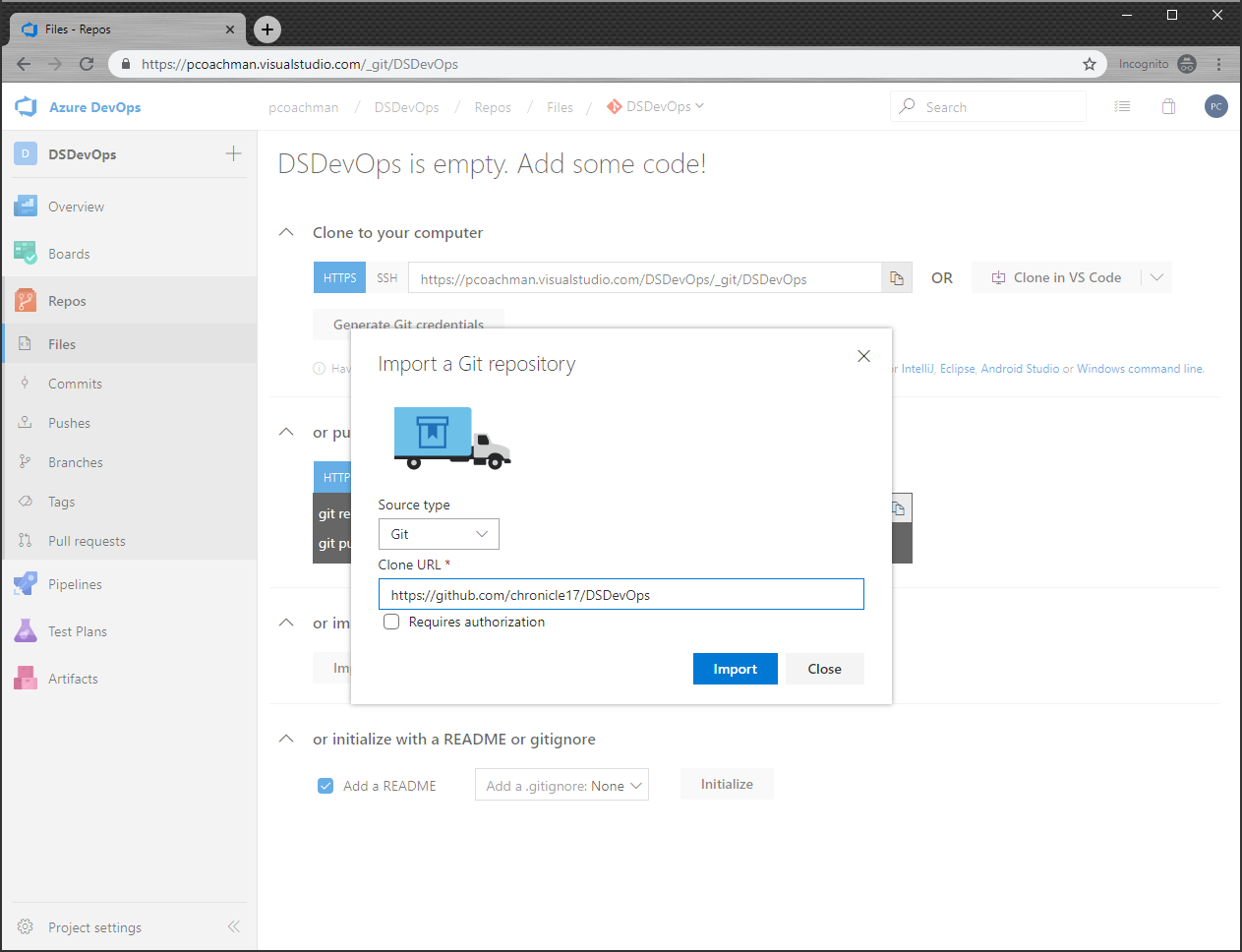


## Build Azure DevOps Pipeline

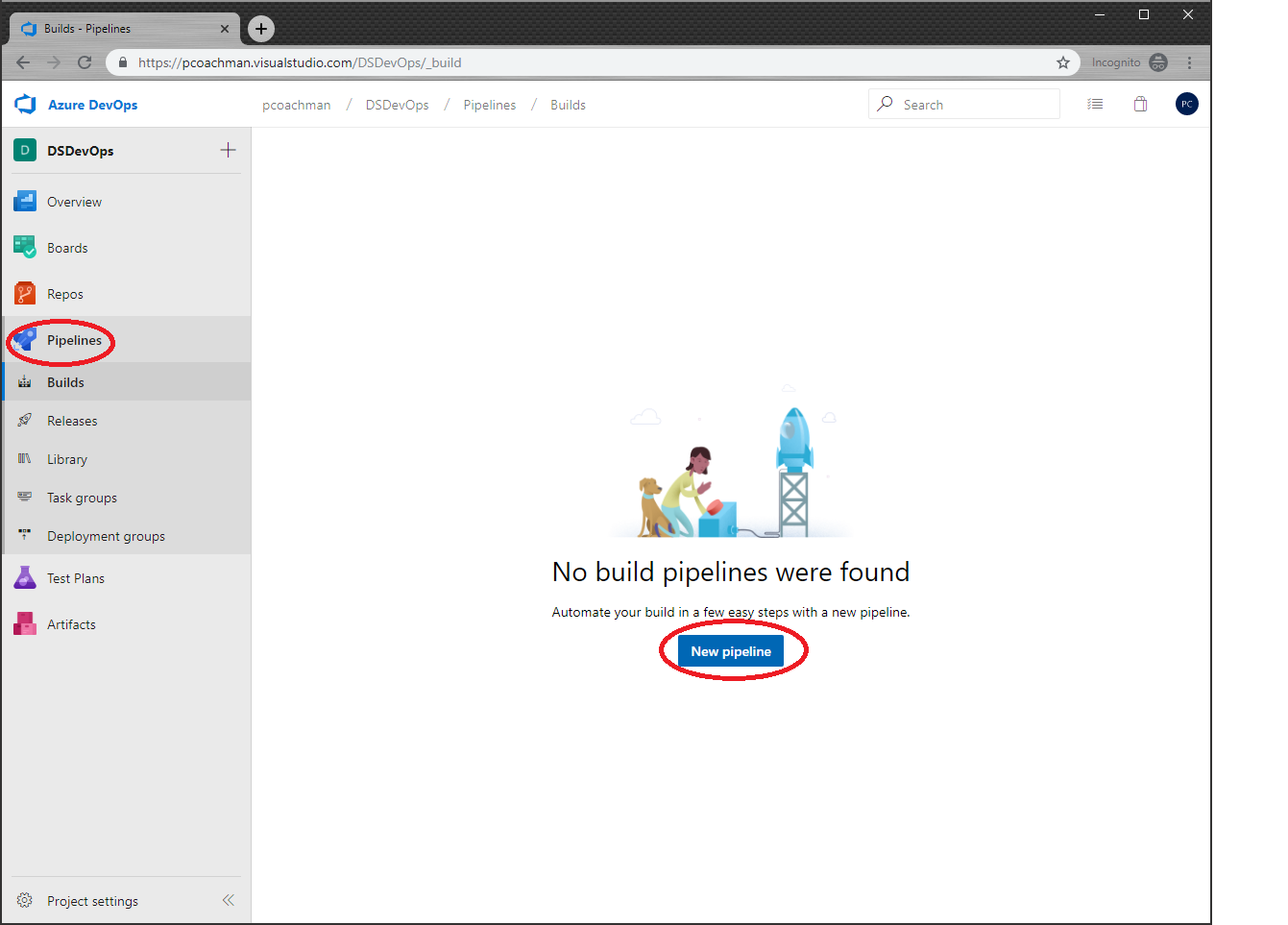
1. Open a browser window and navigate to <https://visualstudio.microsoft.com/>
2. Under Azure DevOps (formerly VSTS) click “Get started for free”
3. Once you sign in with a Microsoft user account, enter a project name and choose Private to create a new project
4. Once the project is created, click Repos on the left navigation bar



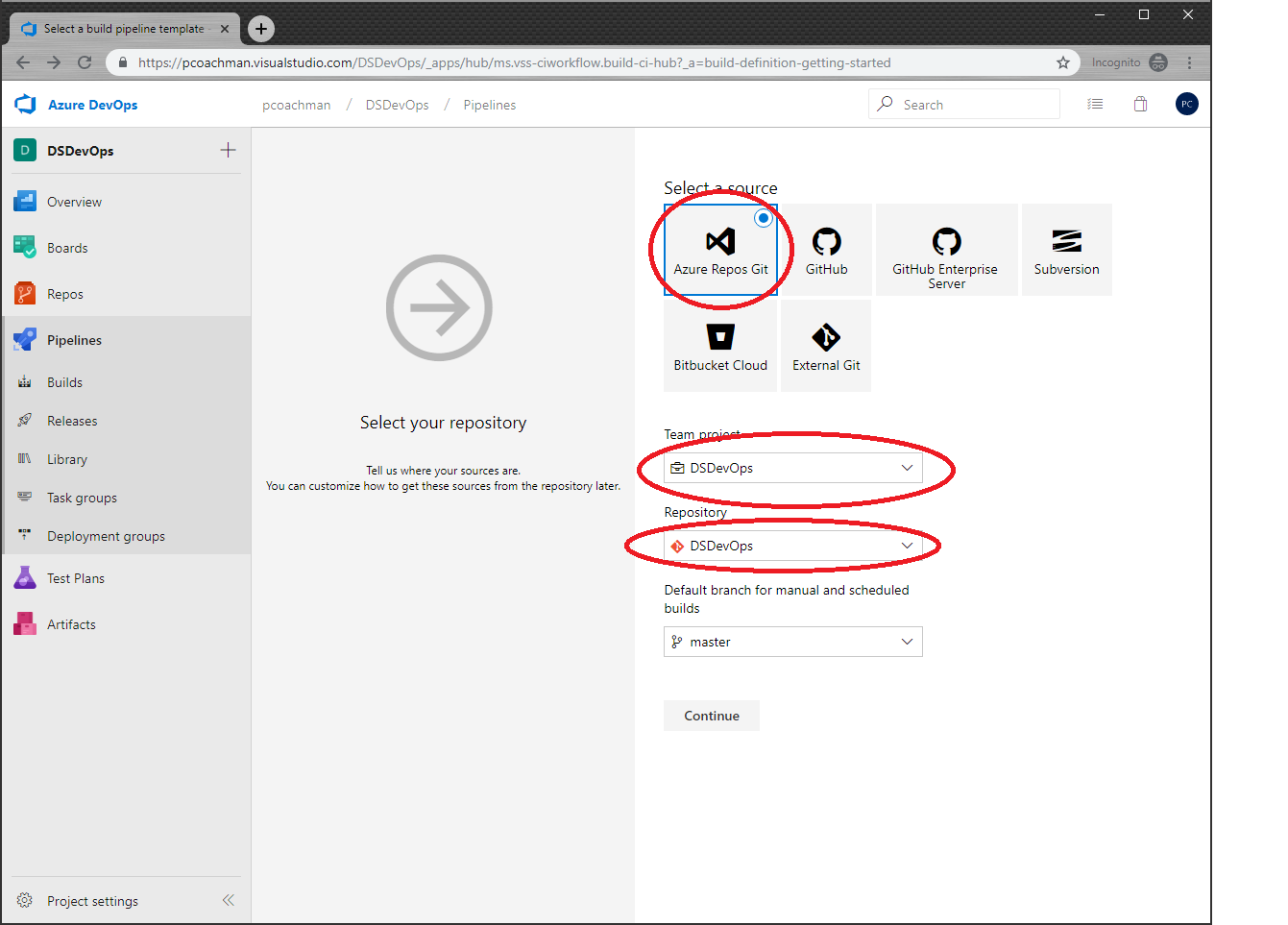
1. Click import to import a Git registry, enter <https://github.com/chronicle17/DSDevOps>, and click Import



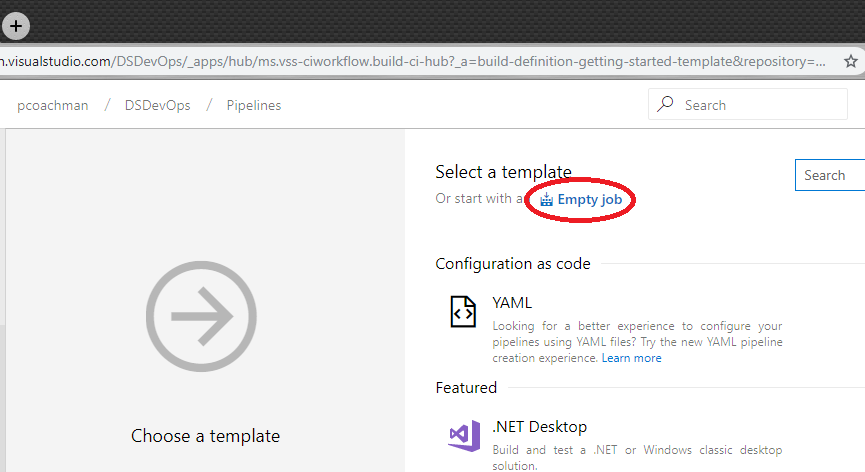
1. Click Pipelines and then New Pipeline to create a new build



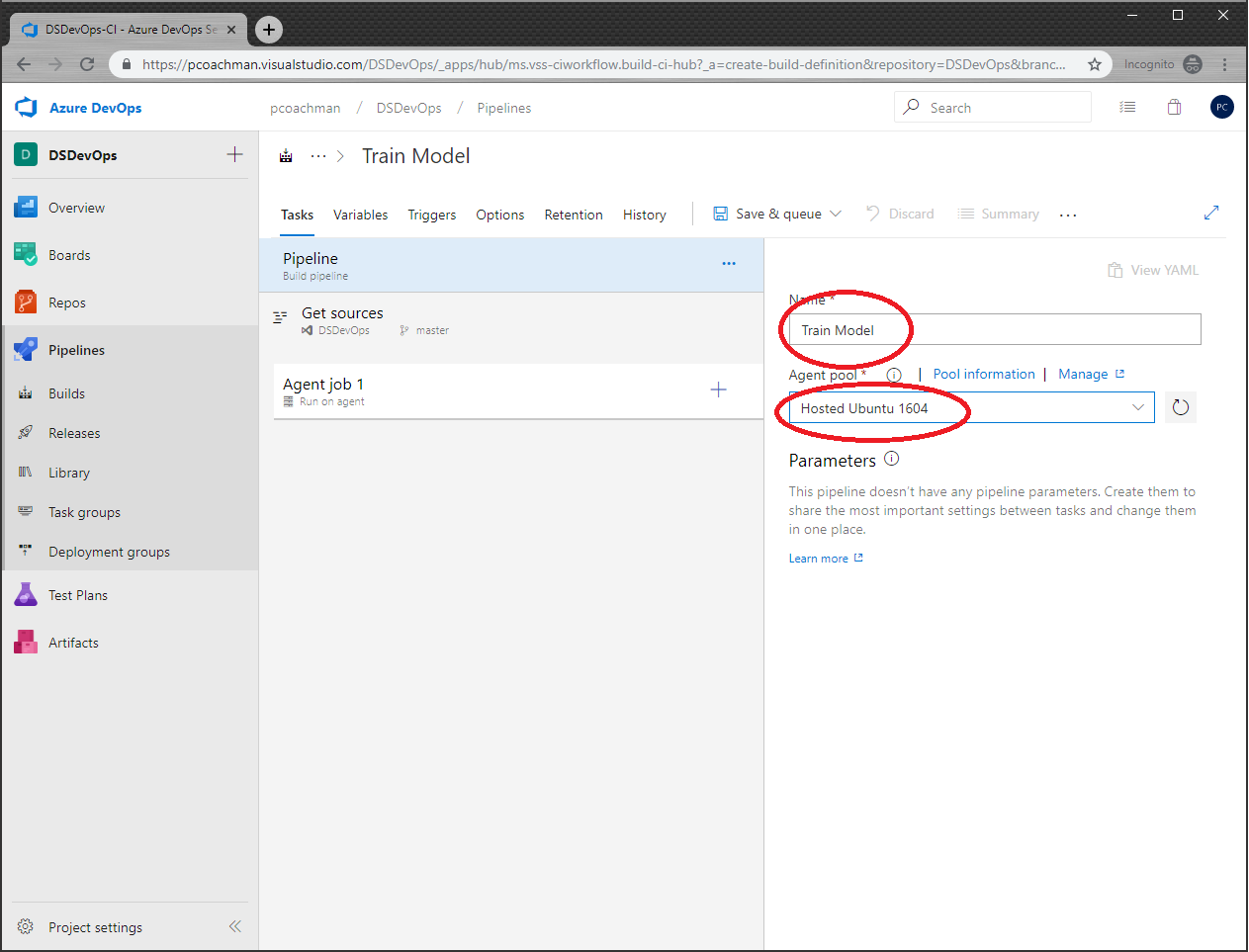
1. Select Azure Repos Git, then select the project and repository that you just created and click Continue



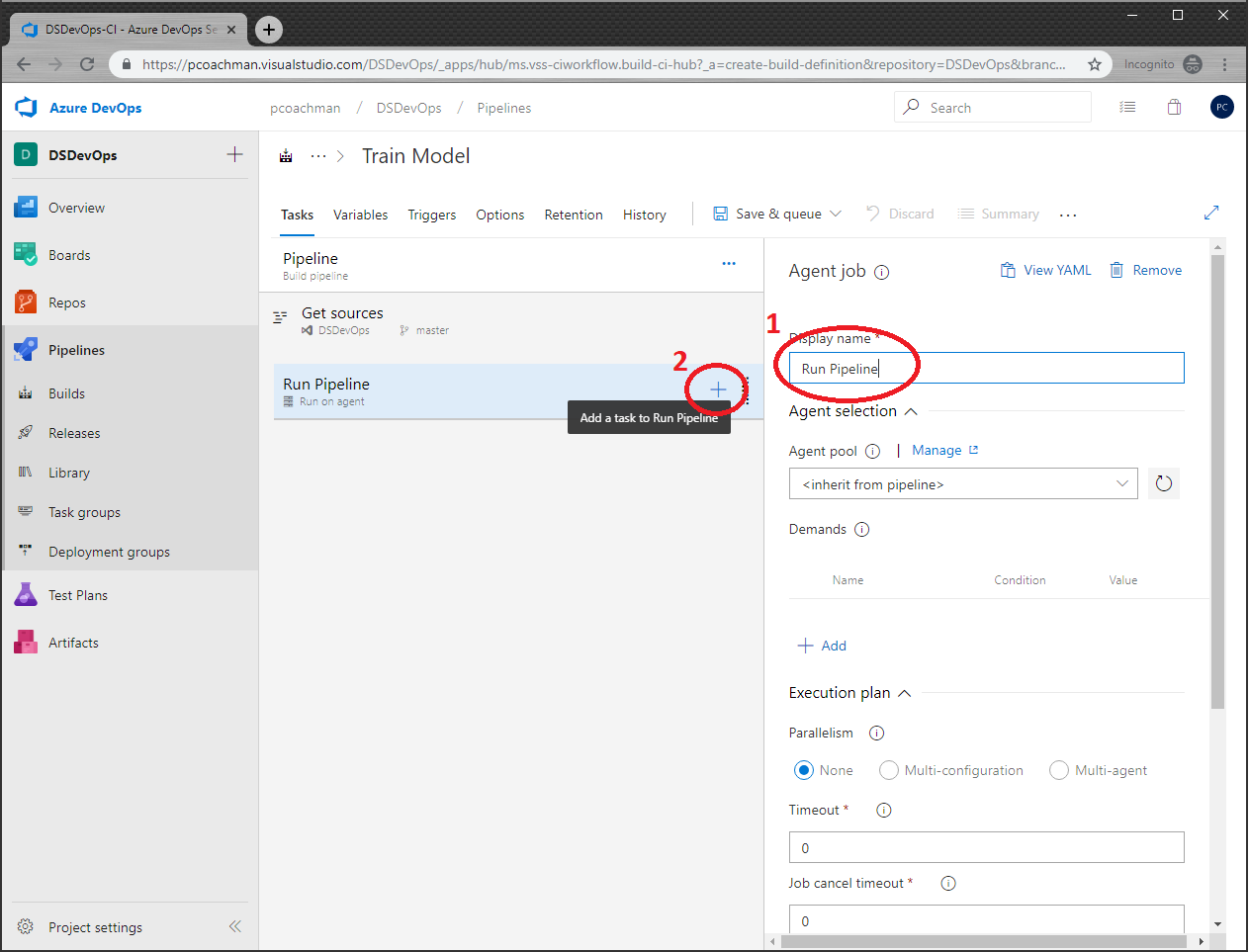
1. When asked to select a template, choose Empty job



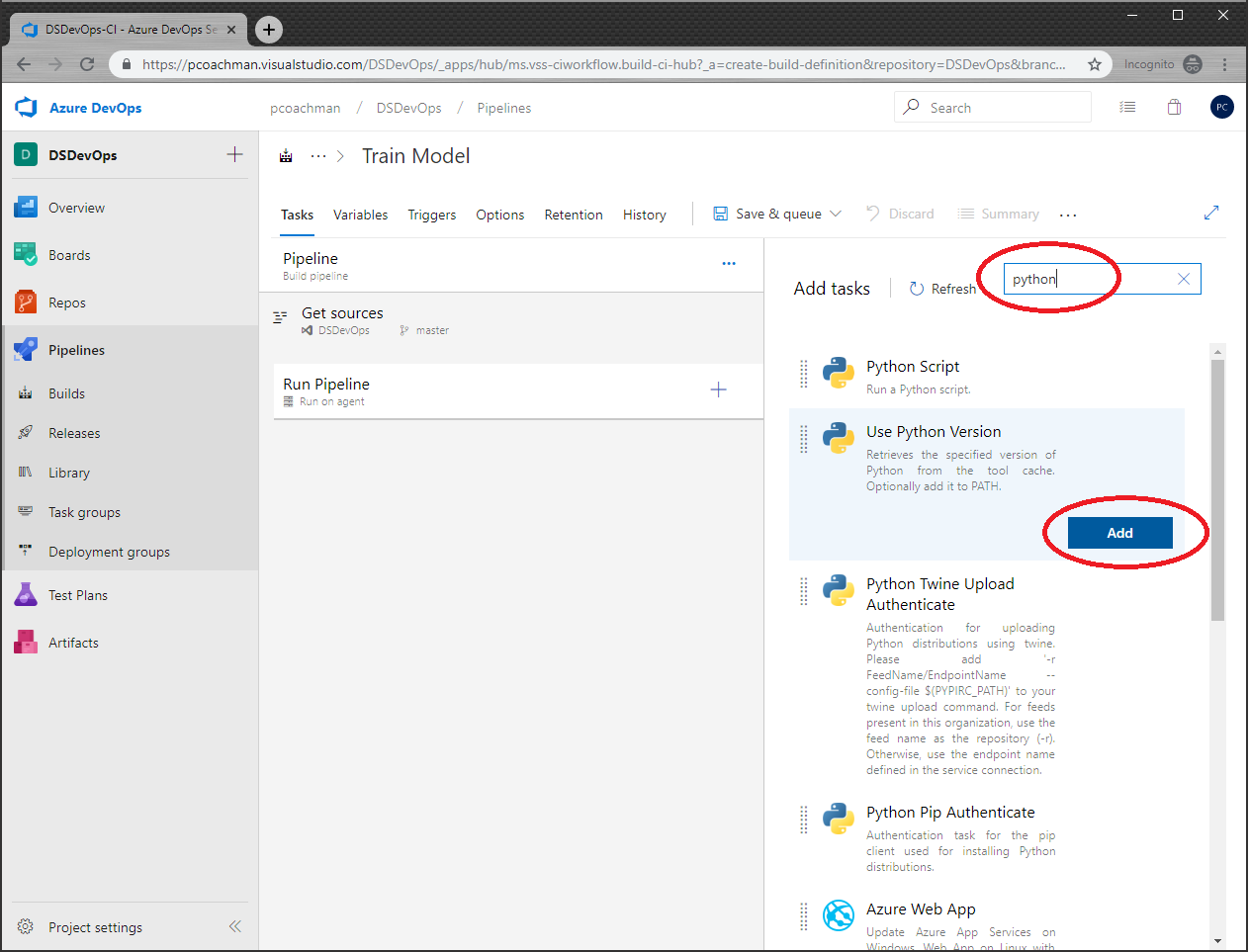
1. Enter “Train Model” for the name and choose “Hosted Ubuntu 1604” for the agent pool



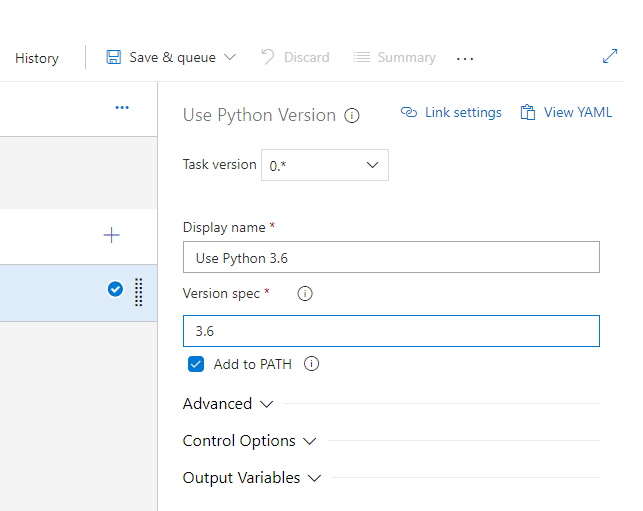
1. Click on Agent job 1. Change the name to Run Pipeline and click the plus sign to add a task



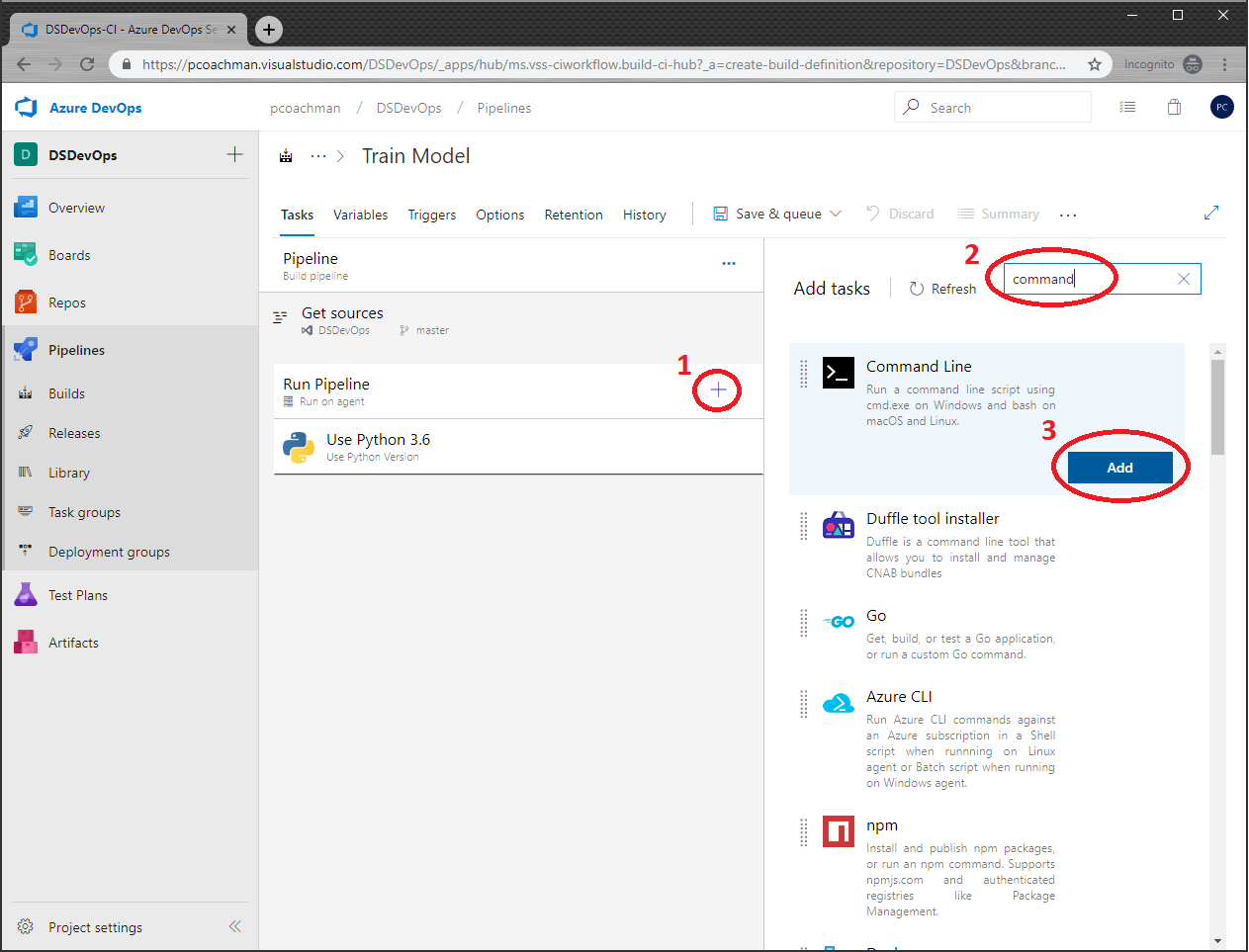
1. In the Add tasks blade, search for python in the search box and add Use Python Version



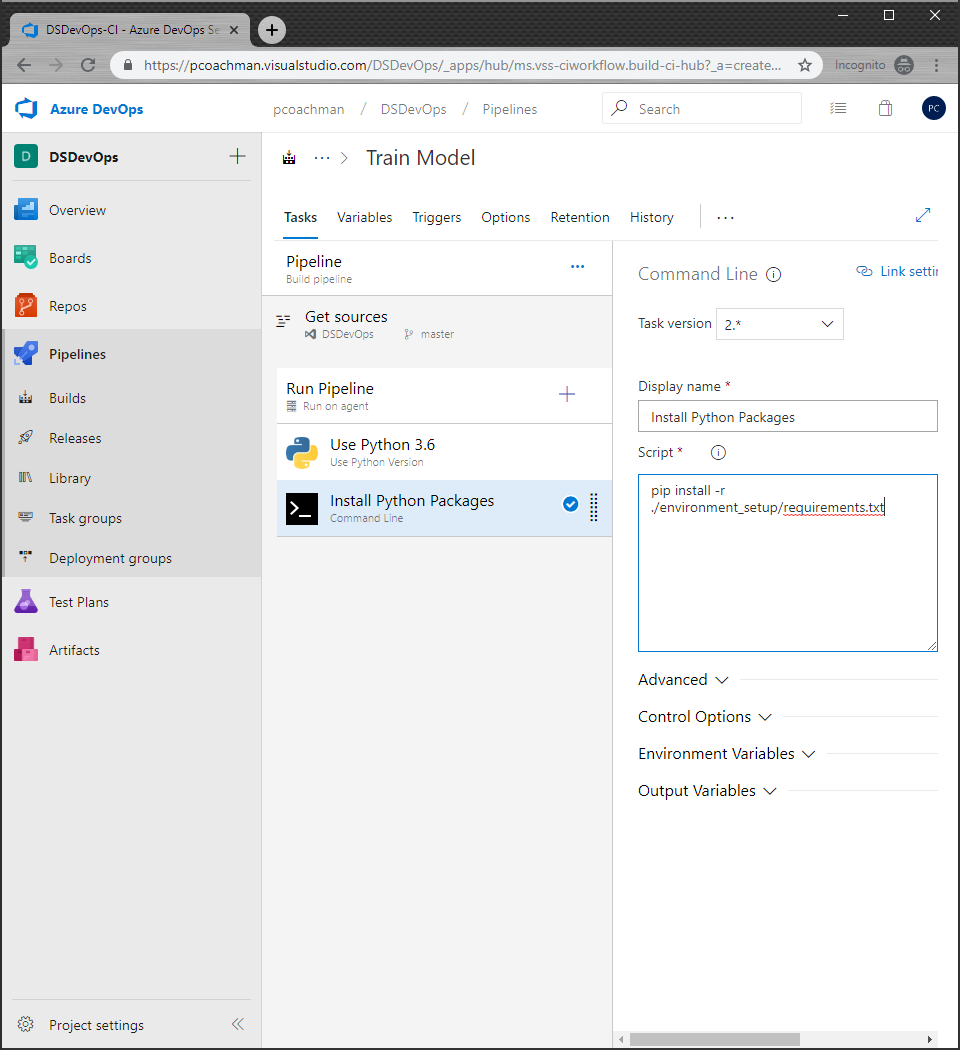
1. Change the name to Use Python 3.6 and enter 3.6 in the Version spec



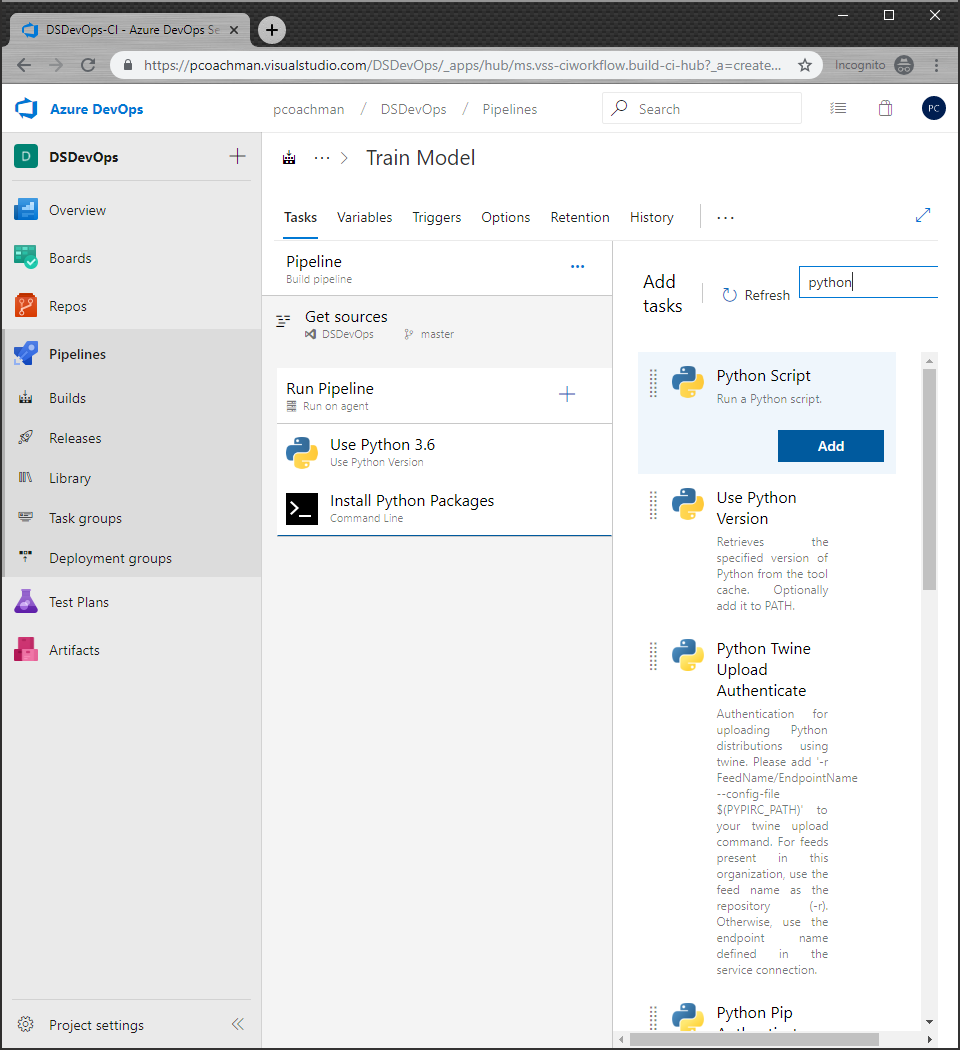
1. Click the plus sign to add a task to the pipeline, search for command, and add Command line



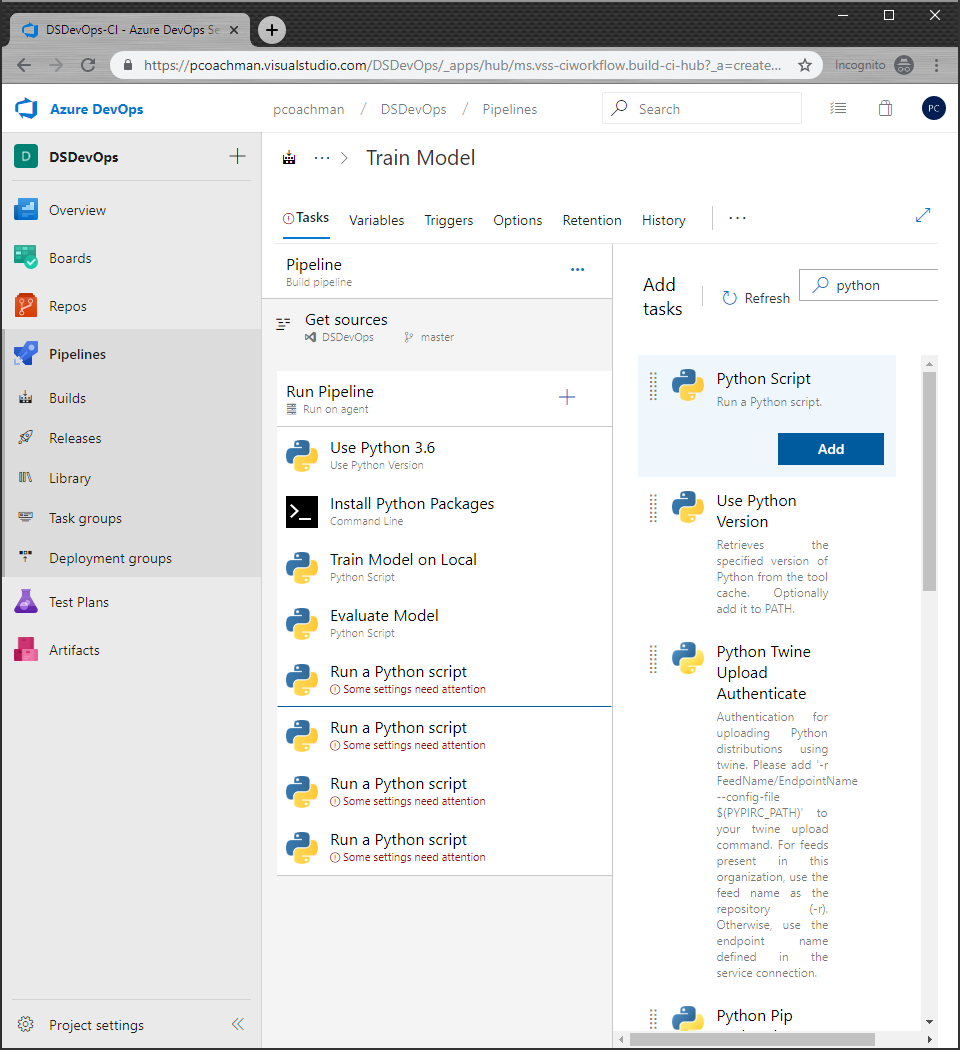
1. Change the Display name to “Install Python Packages”. In the Script box, enter  
   “pip install -r ./environment\_setup/requirements.txt”  
   This will ensure that our build server has all of the packages that we need to build our model.



1. Add a task to the pipeline. Search for python and choose Python Script



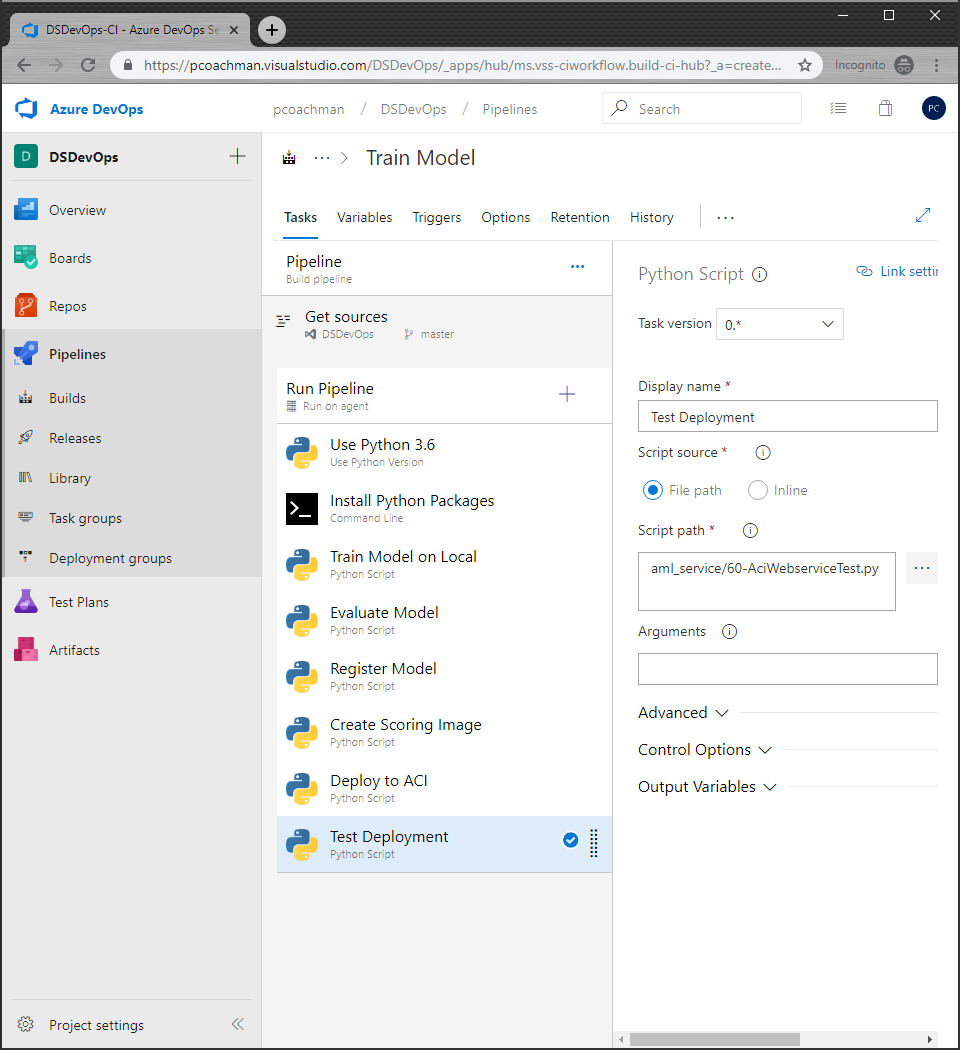
1. Change the name to “Train Model on Local” and choose the script path “aml\_service/10-TrainOnLocal.py”
2. Add another Python script to the pipeline. Change the name to Evaluate Model and the script path to “aml\_service/15\_EvaluateModel.py”
3. Add 4 more Python scripts to the pipeline (hint, you can click Add multiple times)



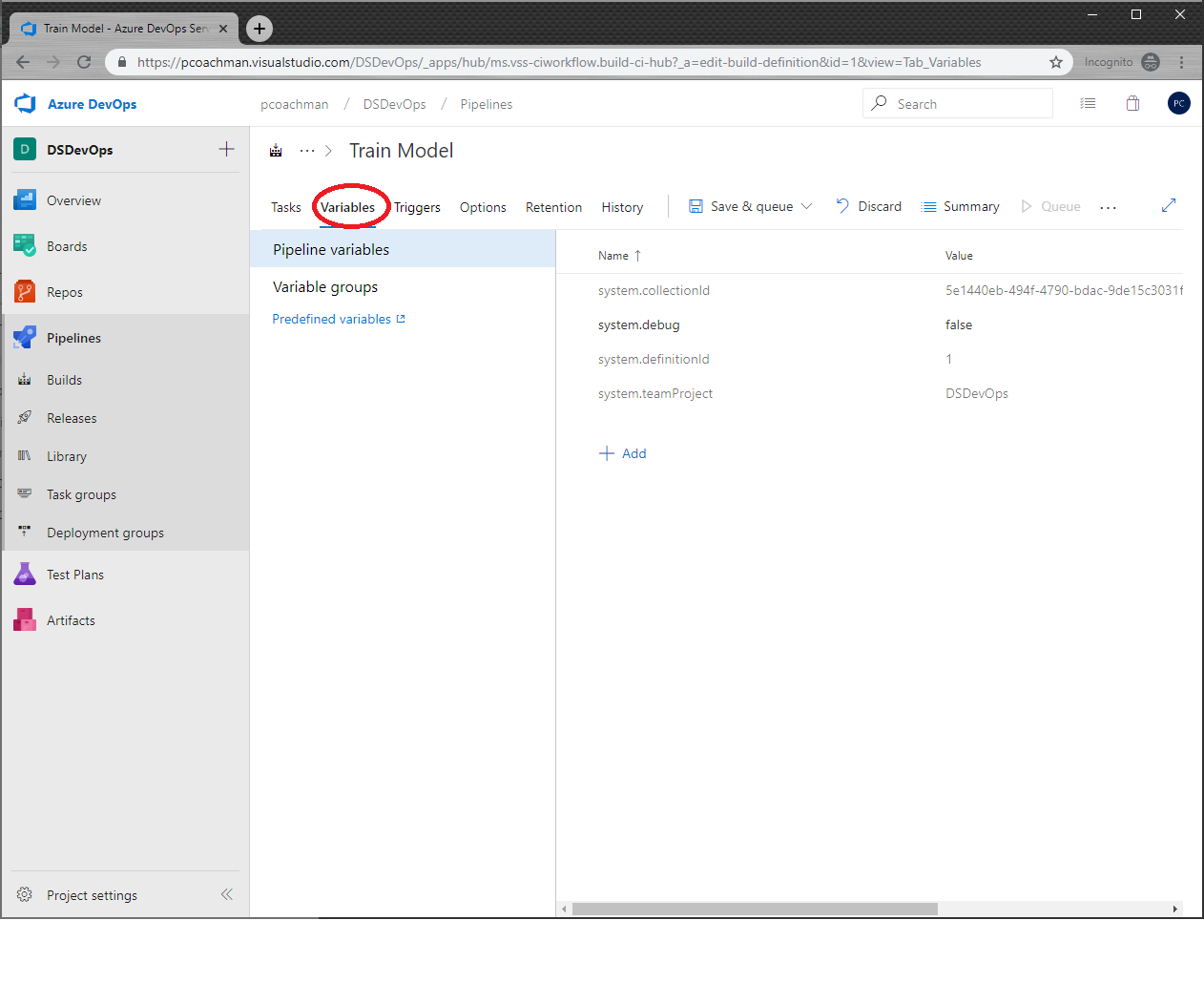
1. Fill out the remaining python scripts with following names and script path

|  |  |
| --- | --- |
| Register Model | aml\_service/20-RegisterModel.py |
| Create Scoring Image | aml\_service/30-CreateScoringImage.py |
| Deploy to ACI | aml\_service/50-deployOnAci.py |
| Test Deployment | aml\_service/60-AciWebserviceTest.py |

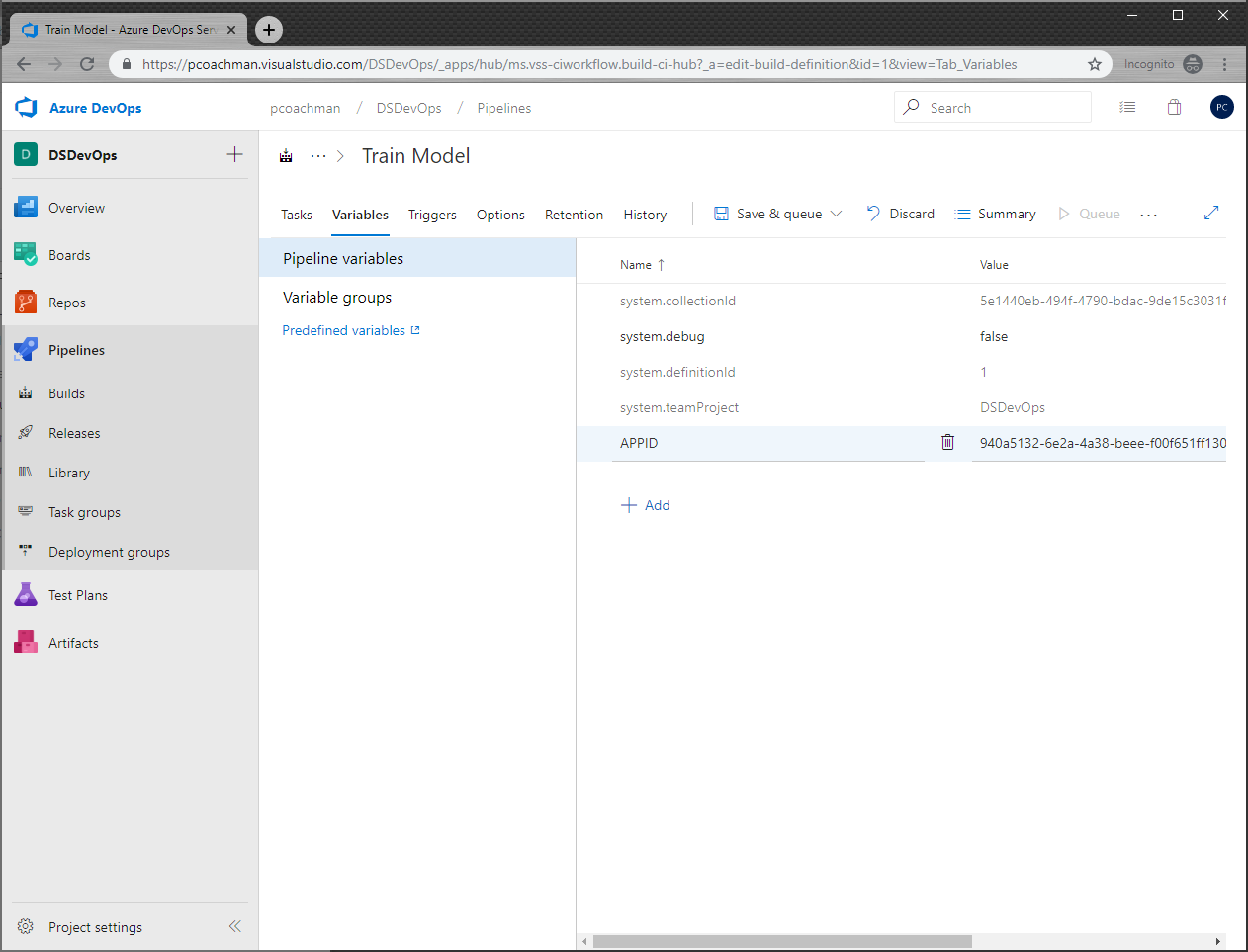
1. Your final pipeline should look like below:



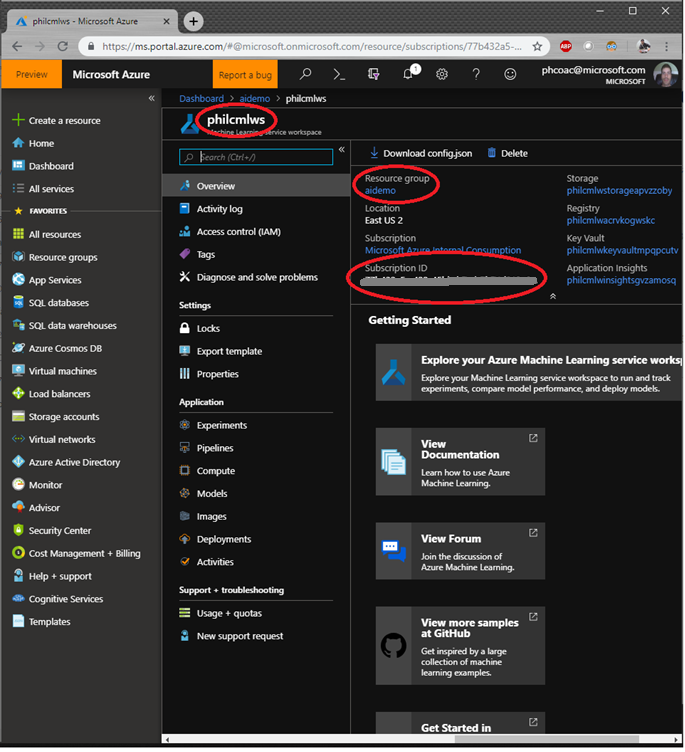
1. Now we have to set our environment variables. At the top, click Variables.



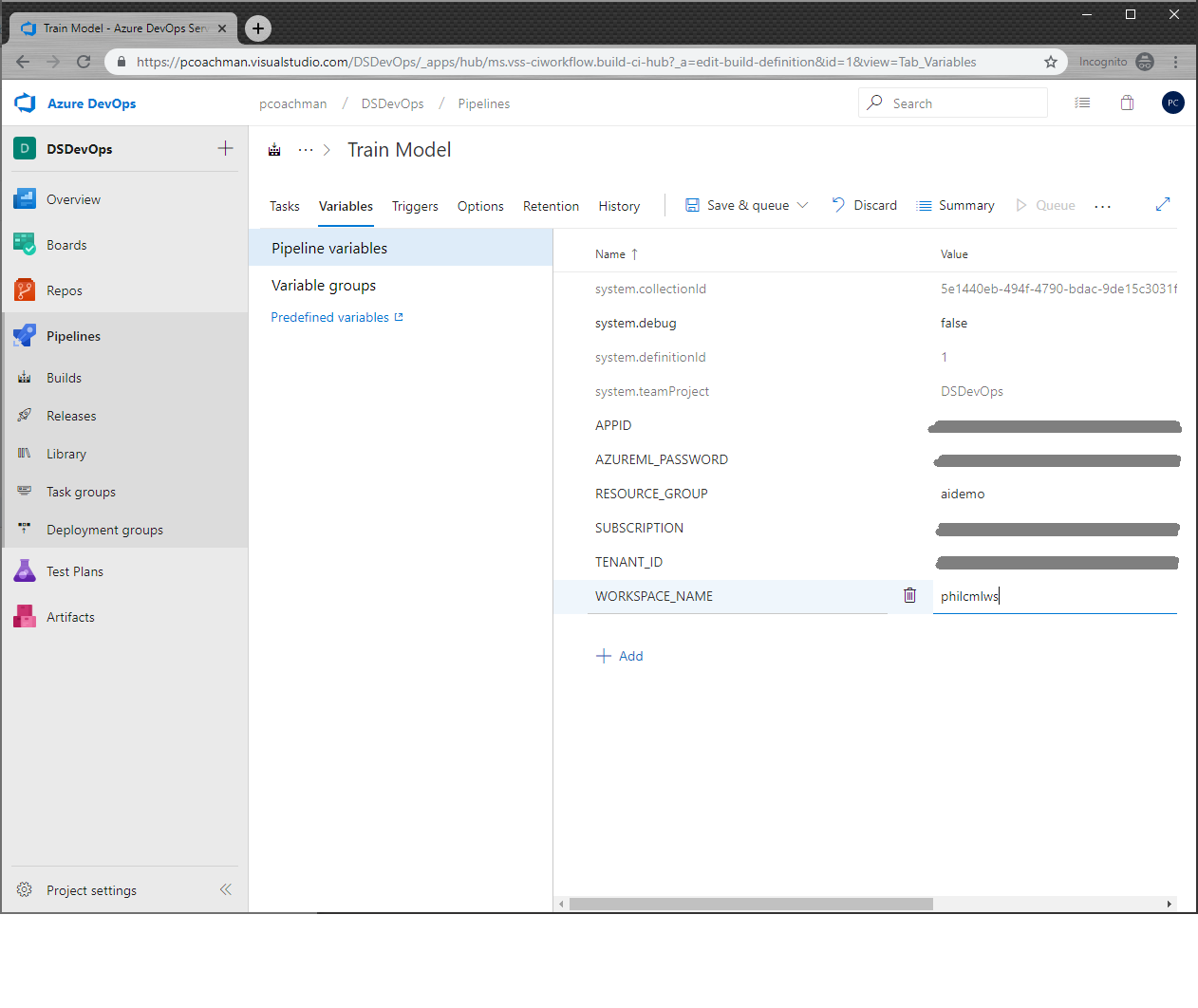
1. In our scripts, we have environment variables for APPID, AZUREML\_PASSWORD, RESOURCE\_GROUP, SUBSCRIPTION, TENANT\_ID, and WORKSPACE\_NAME. The APPID and AZUREML\_PASSWORD are for the service principal that you created earlier on.
2. Click Add and set the name to “APPID” and paste in the application id for the service principal we created.



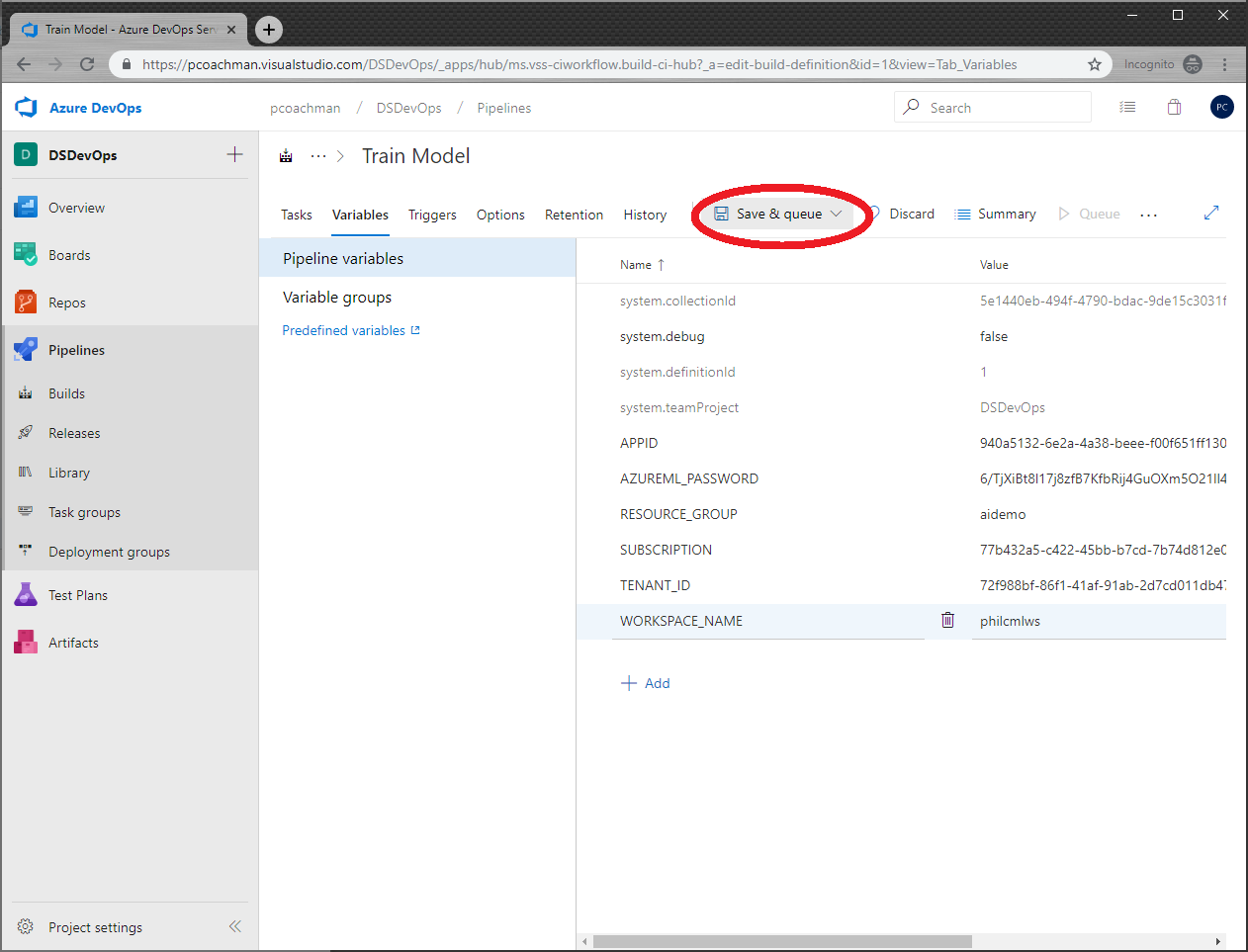
1. Add a variable “AZUREML\_PASSWORD” and paste the password that you copied when you created the service principal
2. Add a variable for “TENANT\_ID” and paste in the Directory ID that you copied above when creating the service principal
3. Add environment variable for “RESOURCE\_GROUP”, “WORKSPACE\_NAME”, and “SUBSCRIPTION”. These are the resource group that your AzureML Workspace is in, the name of your AzureML Workspace, and your subscription ID.



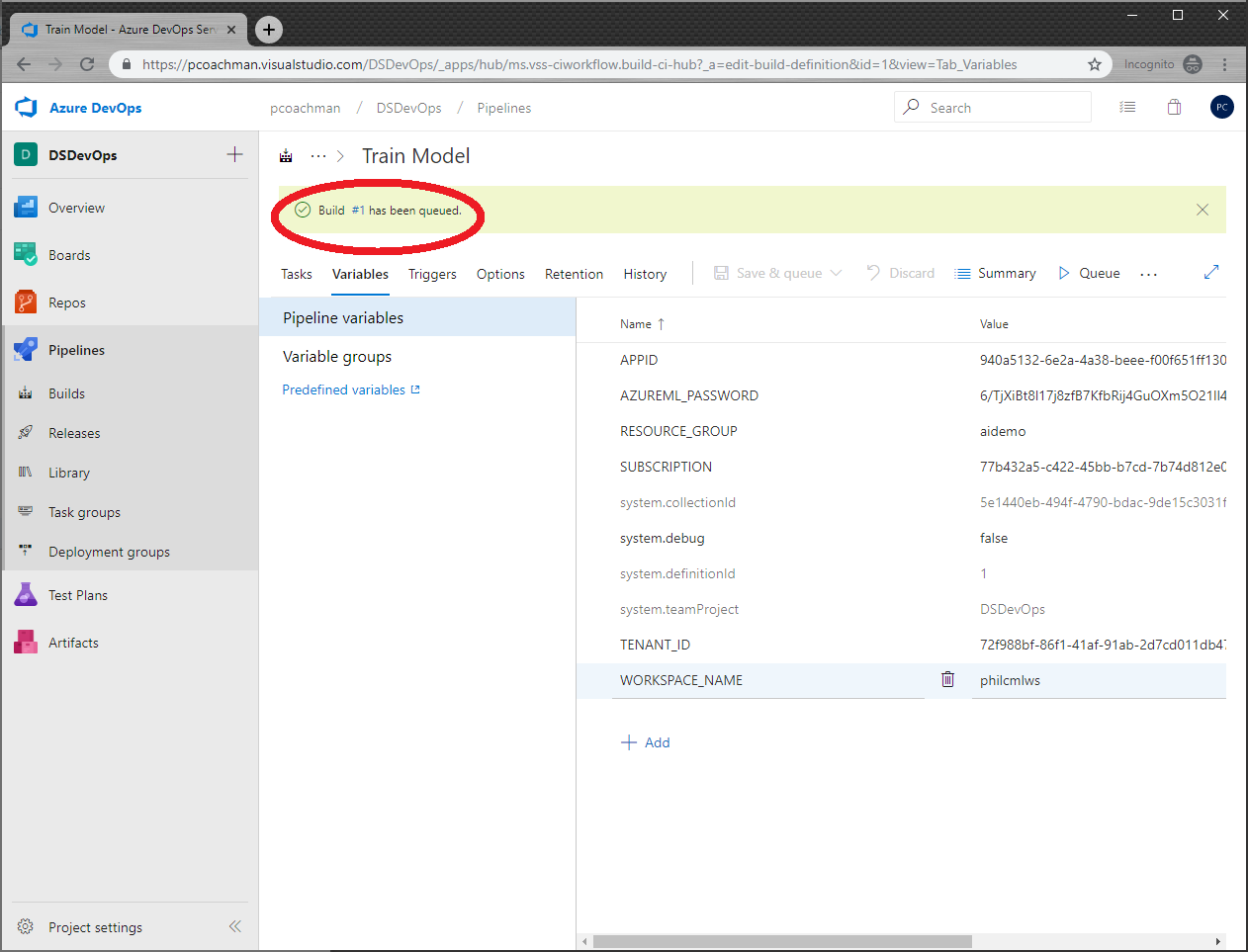
1. Your environment variables should look like below



1. At the top, click Save & queue, then Save & queue from the dropdown



1. On the popup, click Save & queue
2. To monitor the build, click on the build number in the top



Your pipeline run should look like the following:



Congratulations, you just built a devops pipeline to build & deploy a machine learning model!

## Post Workshop Challenge

Most devops pipelines have a dev build and a release to production. Treat the build pipeline that you just created as your dev build. Create a release that uses your development pipeline, triggers a user approval for, then pushes the image and trained model to a production release. (hint, in Azure DevOps, explore the Releases tab)  
<https://docs.microsoft.com/en-us/azure/devops/pipelines/release/?view=azure-devops>

