Azure Devops CI PART

Azure DevOps Overview

Azure DevOps is a powerful set of development tools and services by **Microsoft** that helps teams plan, build, test, and deliver software faster and more efficiently.
It offers a complete DevOps toolchain for developing and deploying software, and it integrates seamlessly with popular tools like **GitHub**, **Docker**, **Kubernetes**, and more.

🌟 Key Features

- 📋 **Azure Boards** – Agile project management with Kanban boards, backlogs, and dashboards.
- 💻 **Azure Repos** – Unlimited, cloud-hosted private Git repositories.
- ⚙️ **Azure Pipelines** – CI/CD for any language, platform, and cloud.
- 📦 **Azure Artifacts** – Package management for Maven, npm, NuGet, and Python.
- 🛠️ **Azure Test Plans** – Manual and exploratory testing tools.

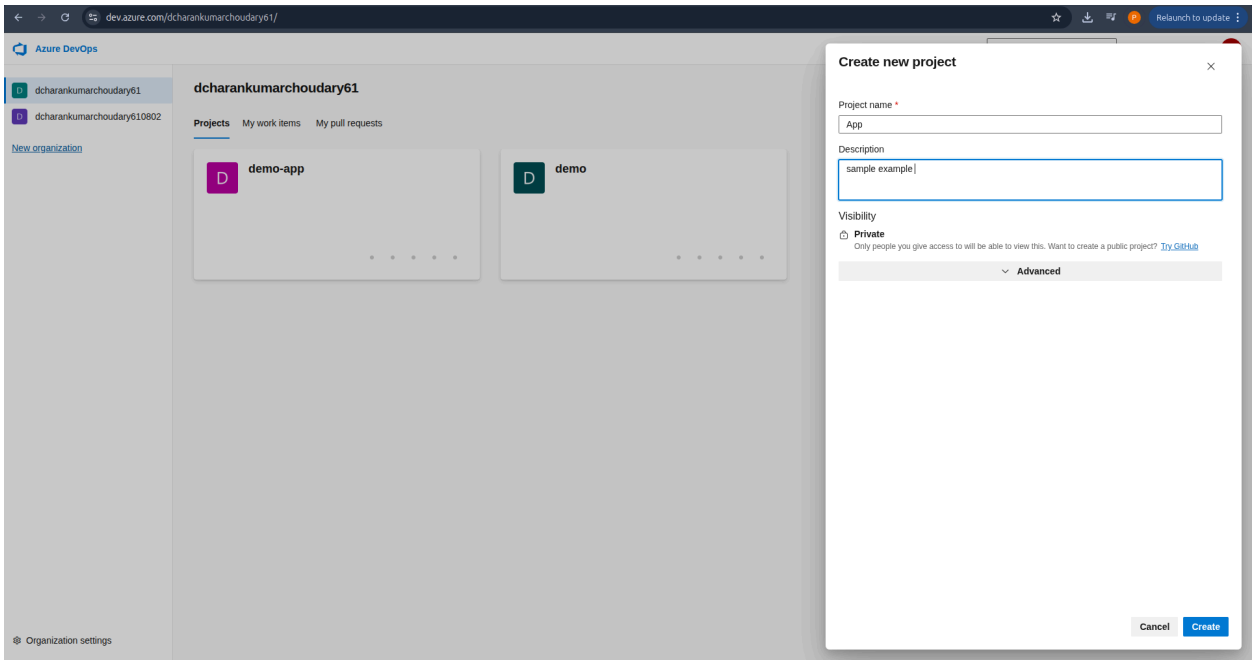
🔗 Benefits

- 🌍 Cross-platform support (Windows, Linux, macOS).
- 🔄 Seamless integration with GitHub, Bitbucket, and other repos.
- ☁️ Deploy to **Azure**, **AWS**, **GCP**, or on-premises servers.
- 🚀 Faster delivery with automated builds and releases.

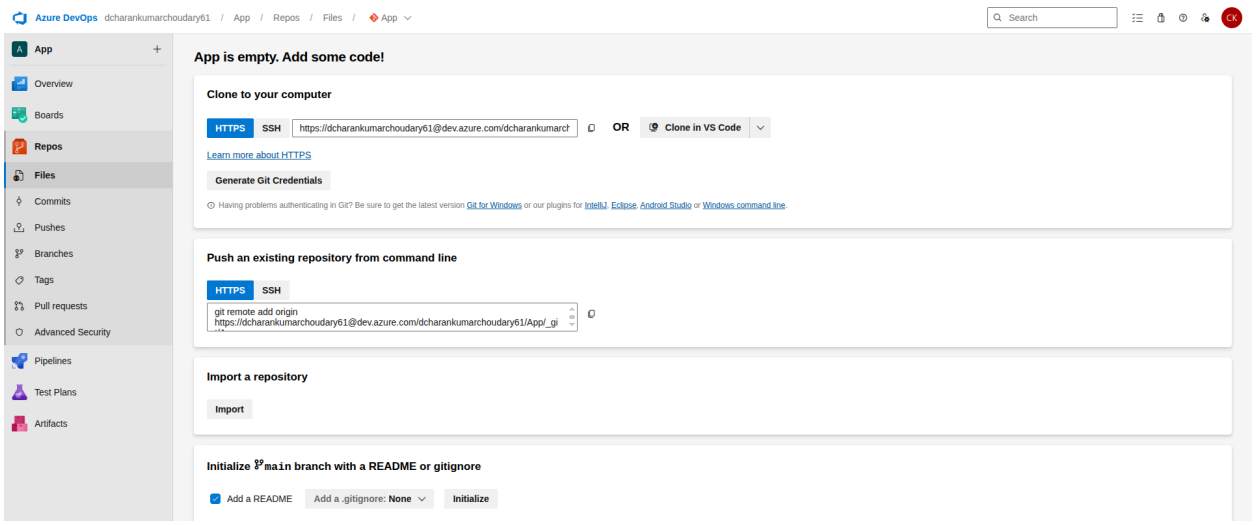
Creation Of Azure Pipelines

Step 1: Login into the azure Devops

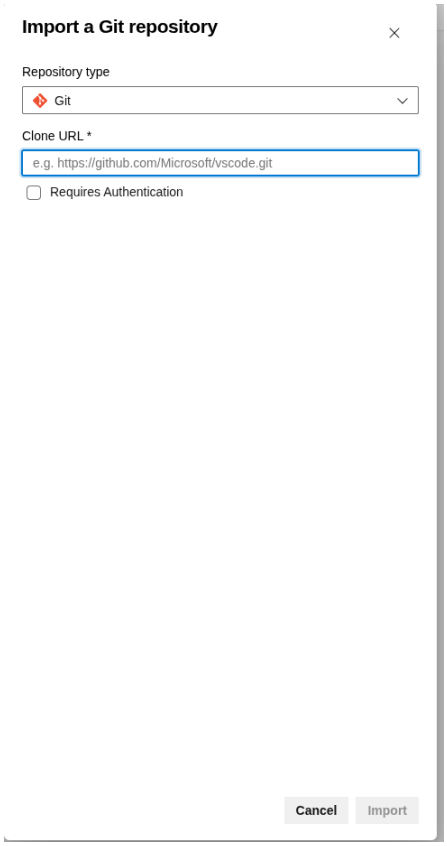
Step 2: Create a Project



Step 4: Click on Repos And click on import My Repo is in Github.

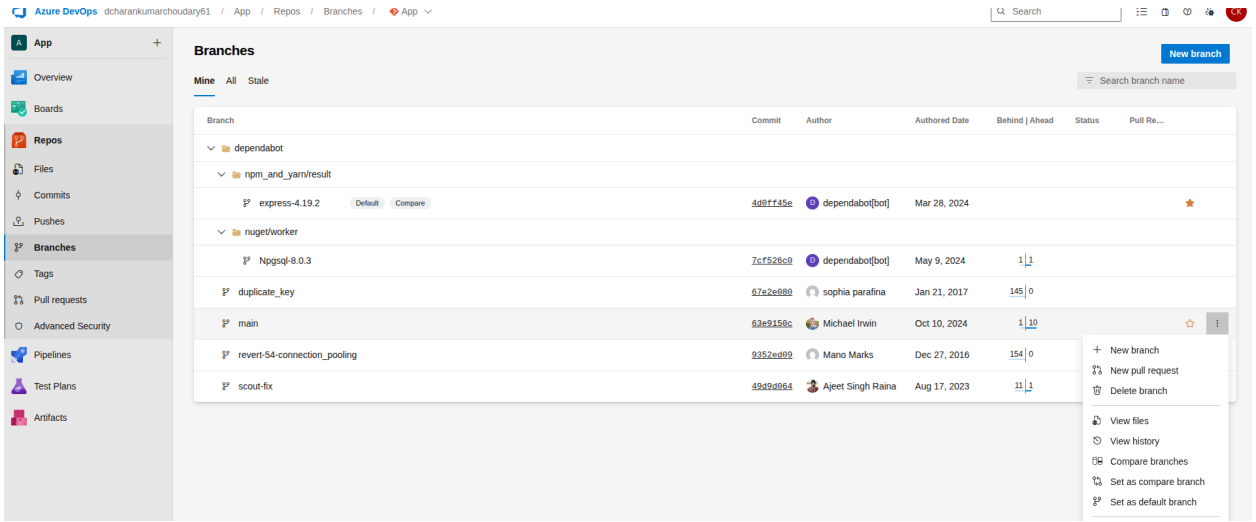


- Provide the Github Repo url and click on import



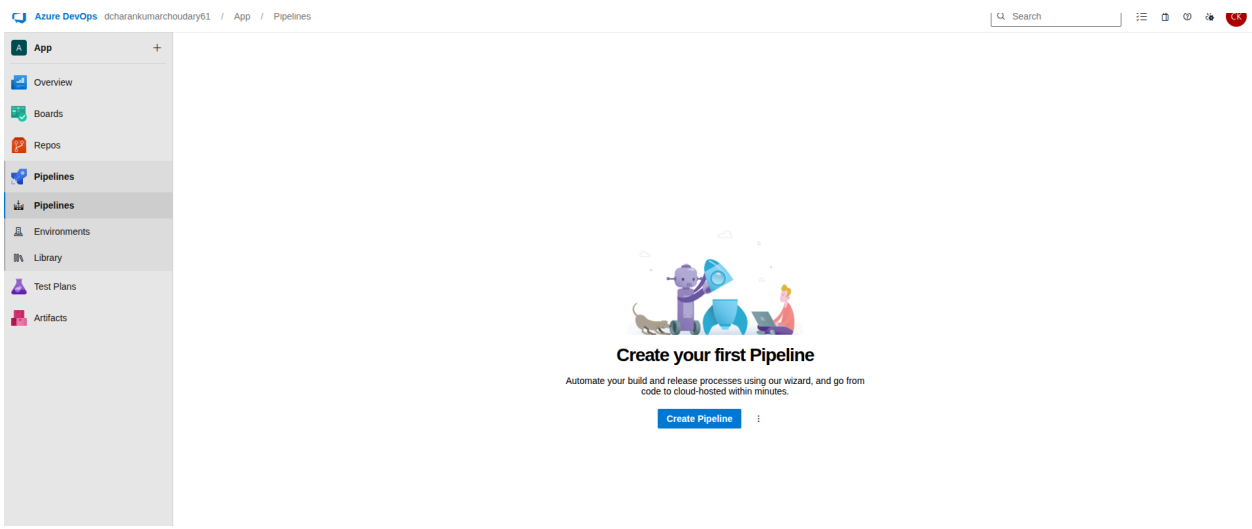
Step 5: Need to check the Default branch as the default branch is main.

- Go branches and click on 3 dots and make the default branch

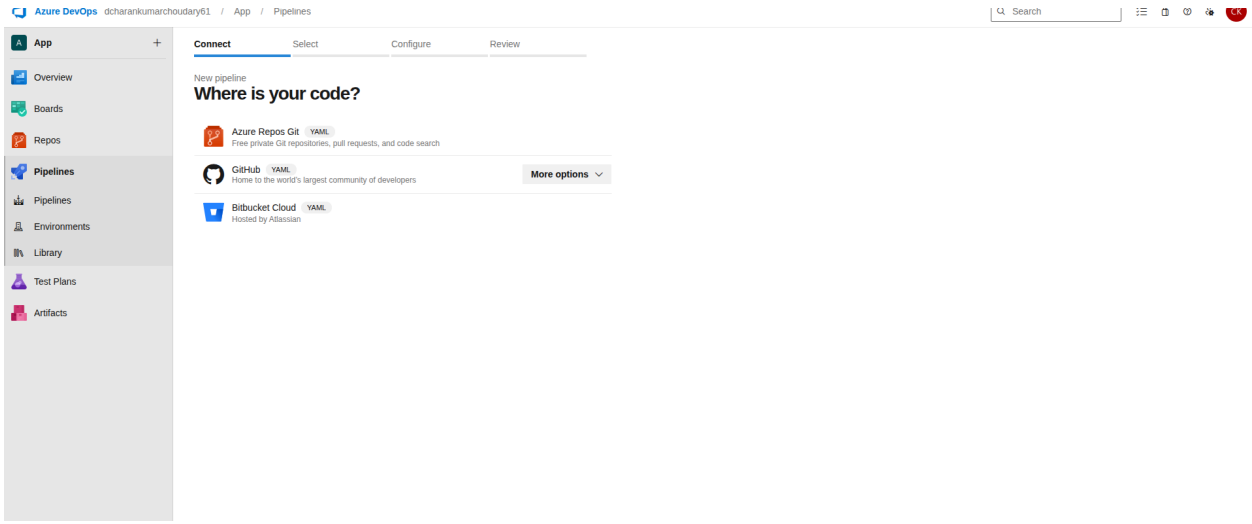


Step 6: Create A pipeline I have 3- micro services one is worker,result,vote, I will create 3 separate pipeline for 3 services

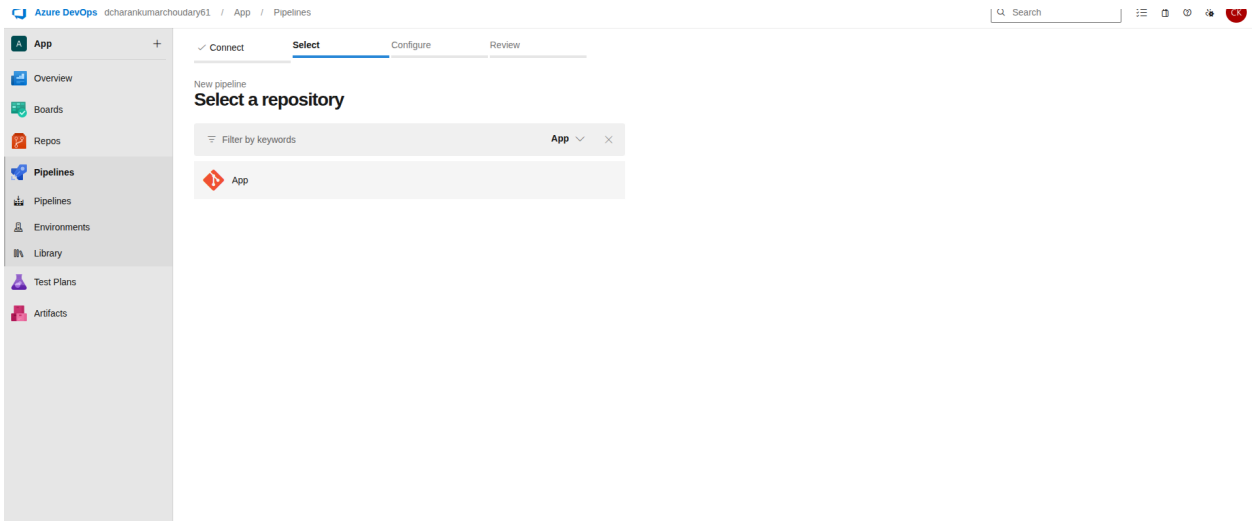
Click on pipeline. Setup of the pipeline:



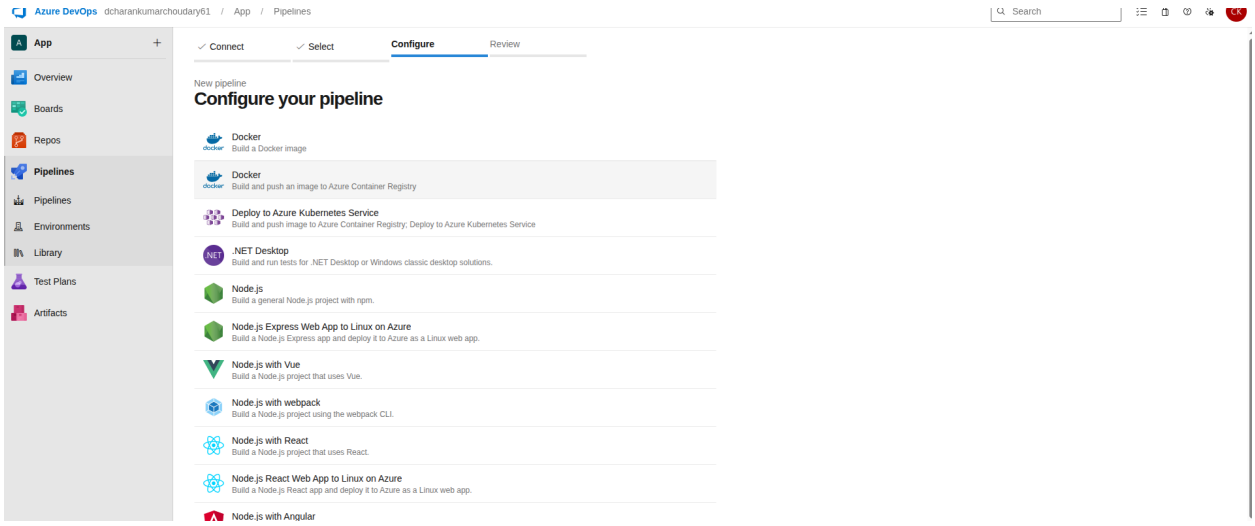
Select the Azure Repo



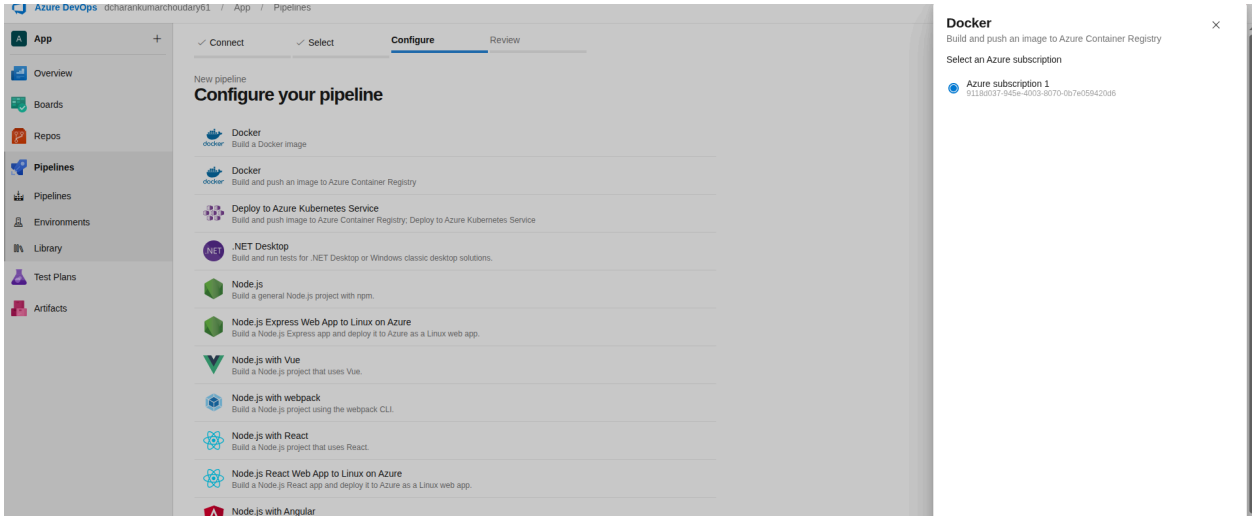
- I selected Previous created repo



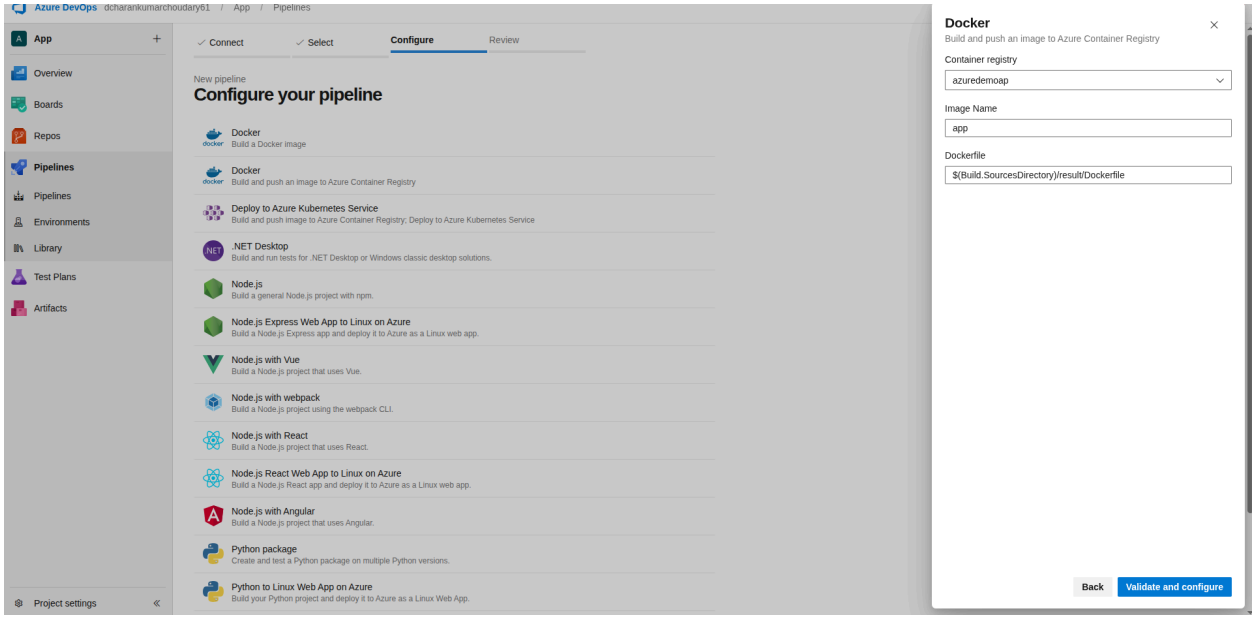
- Select the Docker build and Push image to Azure container Registry



Need to select the azure subscription i have azure container regirsey so this is for authentication to azure container registry and pipeline



- Select the container registry and click on validate and configure.

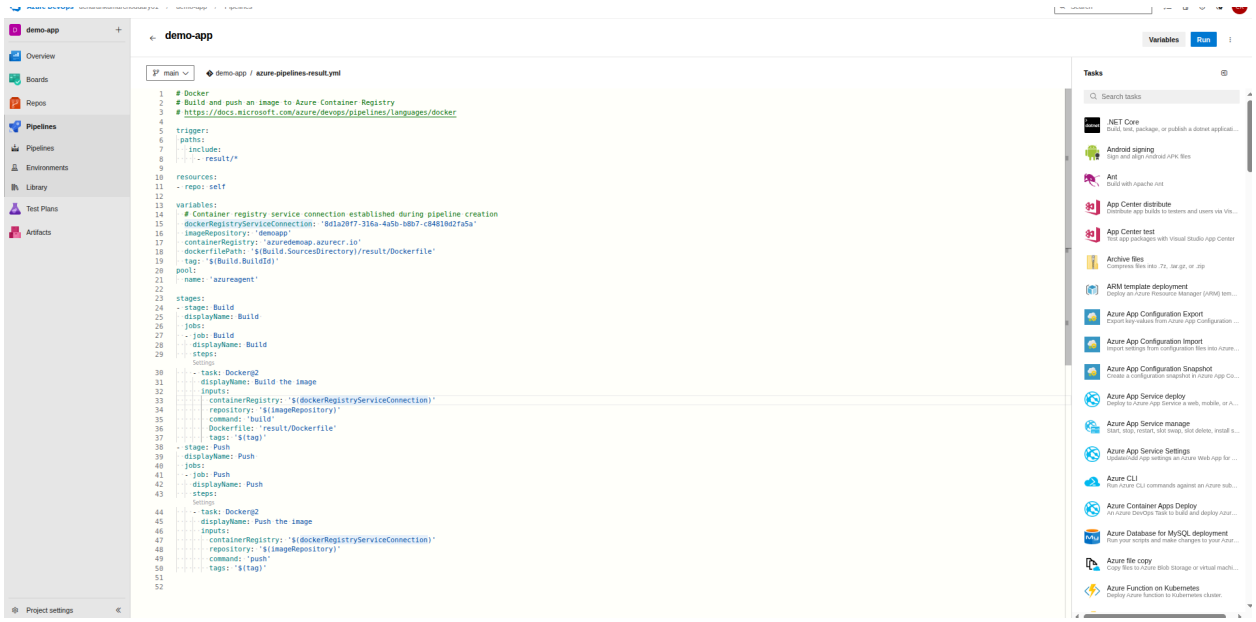


Azure Pipeline structure:

`Pipeline → Stage(s) → Job(s) → Step(s) → Task(s)/Script(s)`

- **Stage**: Logical grouping of jobs (e.g., *Build*, *Test*, *Deploy*).
- **Job**: Runs on an agent (e.g., Ubuntu, Windows).

- **Step**: Single unit of execution inside a job (can be a task or a script).




```
trigger:

paths:

 include:

  - result/*

resources:

- repo: self


variables:

 # Container registry service connection established during pipeline
creation

 dockerRegistryServiceConnection: '8d1a20f7-316a-4a5b-b8b7-c84810d2fa5a'

 imageRepository: 'demoapp'
```

```
 containerRegistry: 'azuredemoap.azurecr.io'
 dockerfilePath: '$(Build.SourcesDirectory)/result/Dockerfile'
 tag: '$(Build.BuildId)'
pool:
 name: 'azureagent'
stages:
- stage: Build
 displayName: Build
 jobs:
 - job: Build
  displayName: Build
  steps:
  - task: Docker@2
   displayName: Build the image
   inputs:
    containerRegistry: '$(dockerRegistryServiceConnection)'
    repository: '$(imageRepository)'
    command: 'build'
    Dockerfile: 'result/Dockerfile'
    tags: '$(tag)'
- stage: Push
 displayName: Push
 jobs:
```

```
- job: Push
  displayName: Push
  steps:
  - task: Docker@2
    displayName: Push the image
    inputs:
      containerRegistry: '$(dockerRegistryServiceConnection)'
      repository: '$(imageRepository)'
      command: 'push'
      tags: '$(tag)'
```

Now We have Mentioned the agent Self hosted agent so that we need to create agent as vm.

Create a VM in Azure

- Use Azure Portal or CLI to create a Linux/Windows VM.
- Ensure it has internet access.

Install prerequisites on the VM

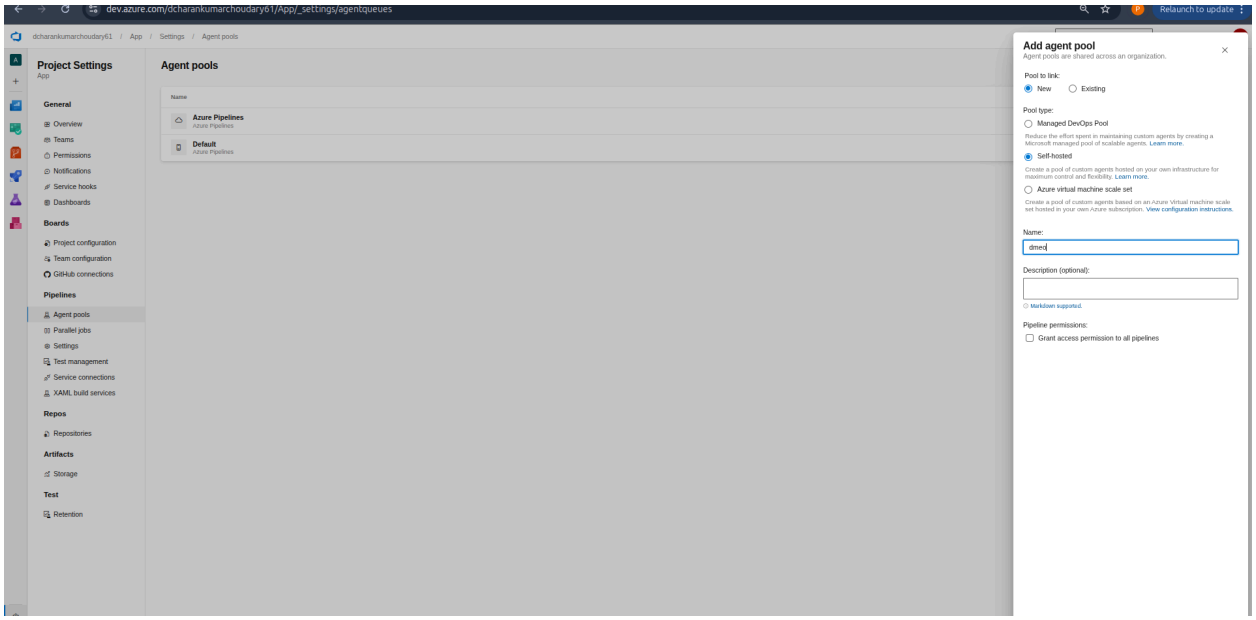
- Update packages.

Create a Personal Access Token (PAT) in Azure DevOps

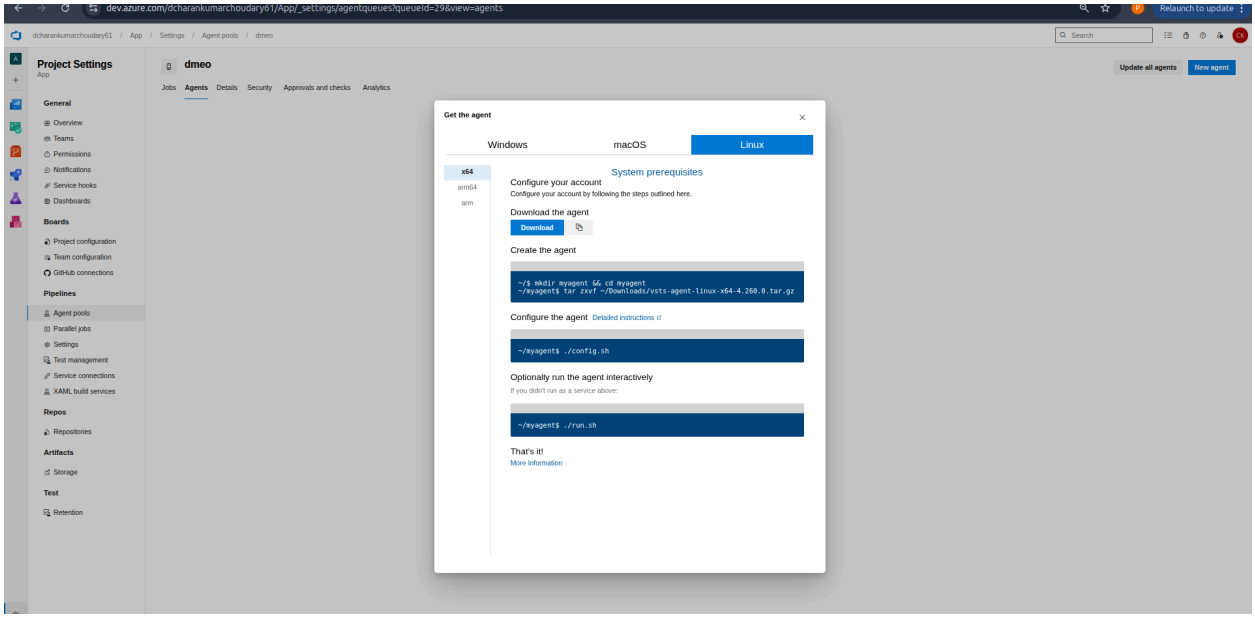
- Go to **User settings → Personal Access Tokens**.
- Scope: **Agent Pools → Read & manage**.

Download the Azure Pipelines agent

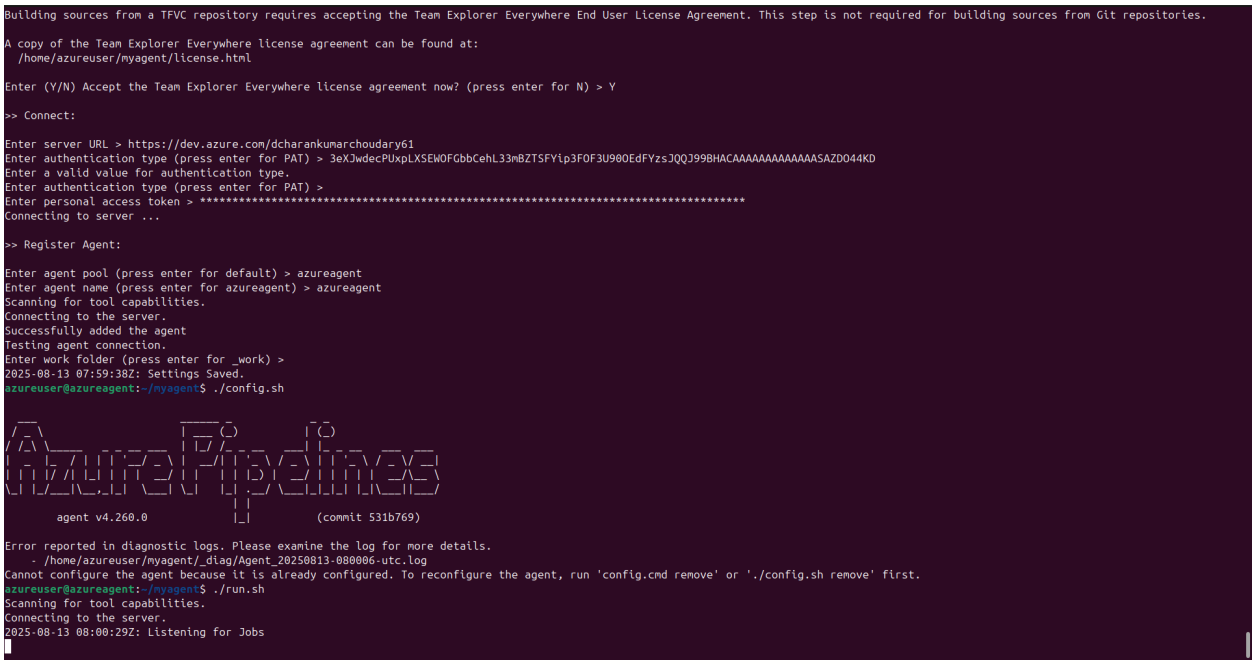
- Go to **Project Settings → Agent Pools → Default → New Agent**.



Configure the agent



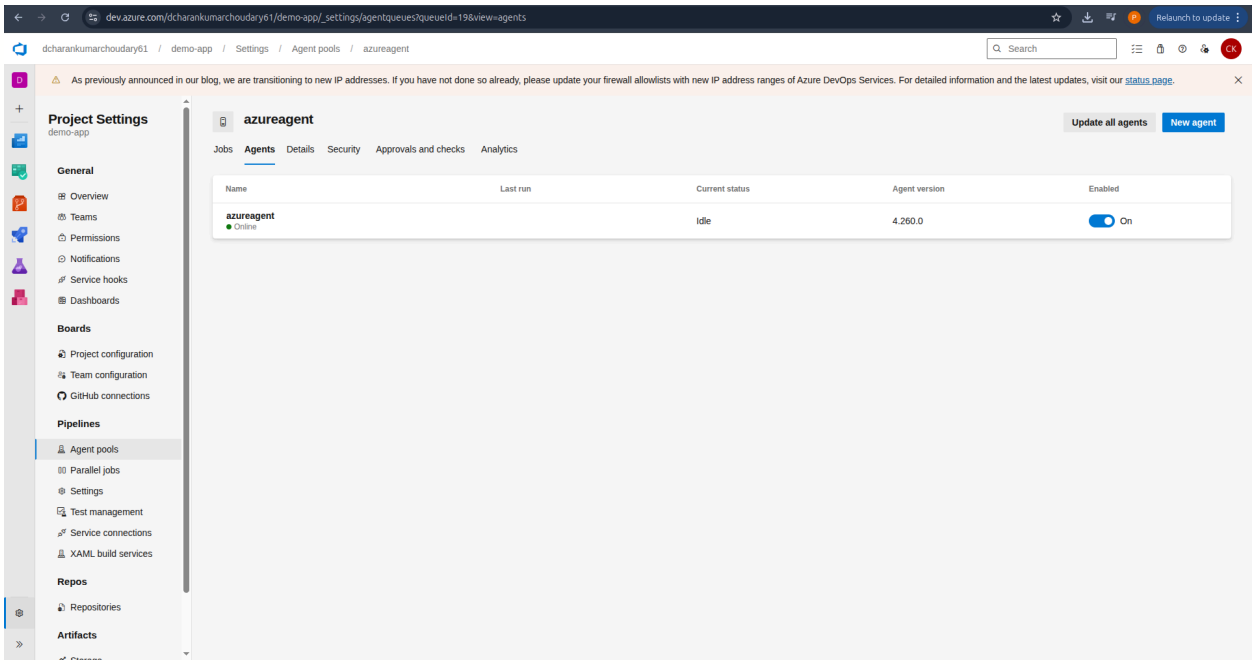

```
./run-docker.sh
./env.sh
azureuser@azureagent:~/myagent$ ls
bin  config.sh  env.sh  externals  license.html  reauth.sh  run-docker.sh  run.sh  vsts-agent-linux-x64-4.260.0.tar.gz
azureuser@azureagent:~/myagent$ sudo /config.sh
sudo: /config.sh: command not found
azureuser@azureagent:~/myagent$ ./config.sh

  ___                      ______ _           _ _
 / _ \                     | ___ (_)         | (_)
/ /_\ \_____  _ _ __ ___   | |_/ /_ _ __   ___| |_ _ __   ___  ___
|  _  |_  / | | | '__/ _ \ |  __/| | '_ \ / _ \ | | '_ \ / _ \/ __|
| | | |/ /| |_| | | |  __/ | |   | | |_) |  __/ | | | | |  __/\__ \
\_| |_/___|\__,_|_|  \___| \_|   |_| .__/ \___|_|_|_| |_|\___||___/
                                   | |
        agent v4.260.0             |_|          (commit 531b769)


>> End User License Agreements:

Building sources from a TFVC repository requires accepting the Team Explorer Everywhere End User License Agreement. This step is not required for building sources from Git repositories.

A copy of the Team Explorer Everywhere license agreement can be found at:
  /home/azureuser/myagent/license.html

Enter (Y/N) Accept the Team Explorer Everywhere license agreement now? (press enter for N) > Y

>> Connect:

Enter server URL > https://dev.azure.com/dcharankumarchoudary61
Enter authentication type (press enter for PAT) > 3eXJwdecPUxpLXSEWOFGbbCehL33mBZTSFYip3FOF3U90OEdFYzsJQQJ99BHACAAAAAAAAAAAAAASAZDO44KD
Enter a valid value for authentication type.
Enter authentication type (press enter for PAT) >
Enter personal access token > ********************************************************************************
Connecting to server ...

>> Register Agent:

Enter agent pool (press enter for default) > azureagent
Enter agent name (press enter for azureagent) > azureagent
Scanning for tool capabilities.
Connecting to the server.
Successfully added the agent
Testing agent connection.
```



- Run `./config.sh` (Linux) or `config.cmd` (Windows).
- Enter Azure DevOps URL, choose authentication type `PAT`, paste token, select agent pool, and name the agent.

Verify in Azure DevOps

- Go to **Agent Pools** and confirm your agent is **Online**.



Now Run the pipeline