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.

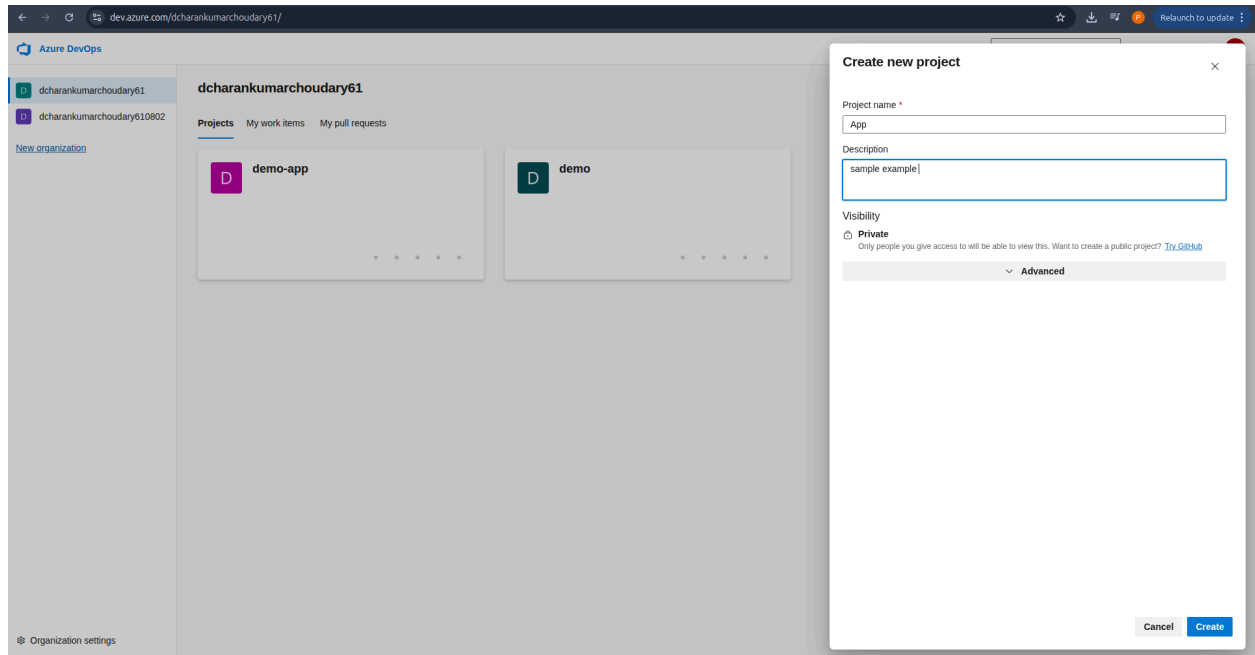- 🛠️ **Azure Test Plans** – Manual and exploratory testing tools.

🔗 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.
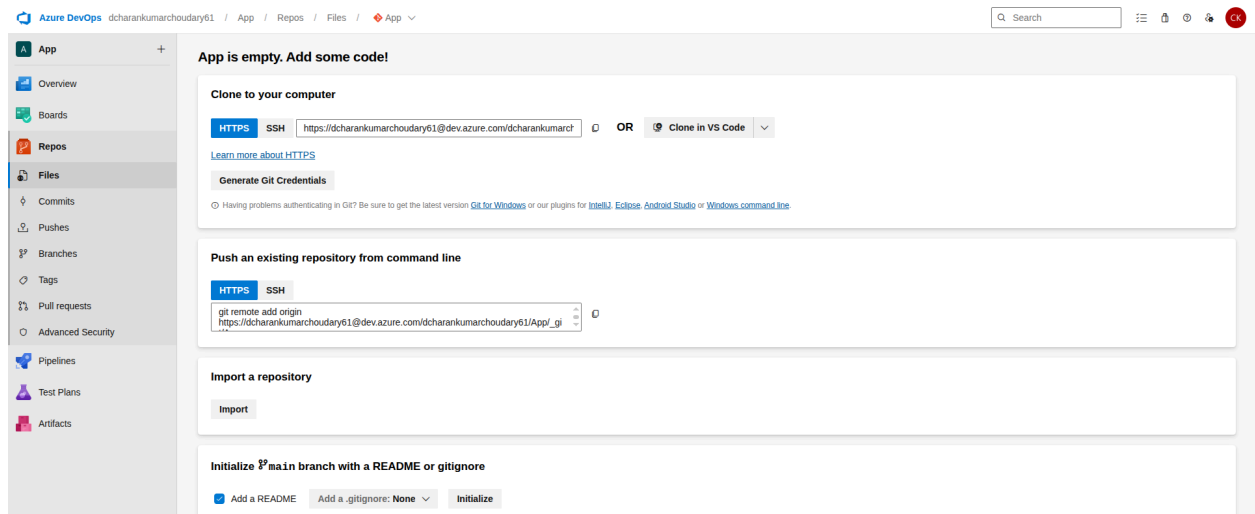
Creation Of Azure Pipelines

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
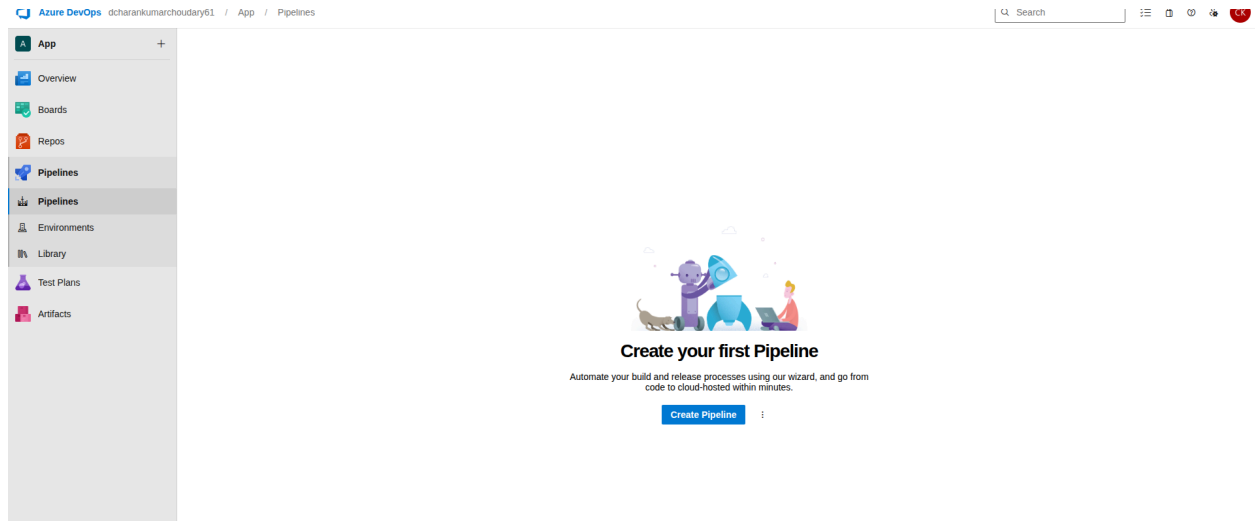

Step 6: Create A pipeline I have 3- micro services one is worker,result,vote, I will create 3 separate pipeline for 3 services

Click on pipeline. Setup of the pipeline:

Select the Azure Repo
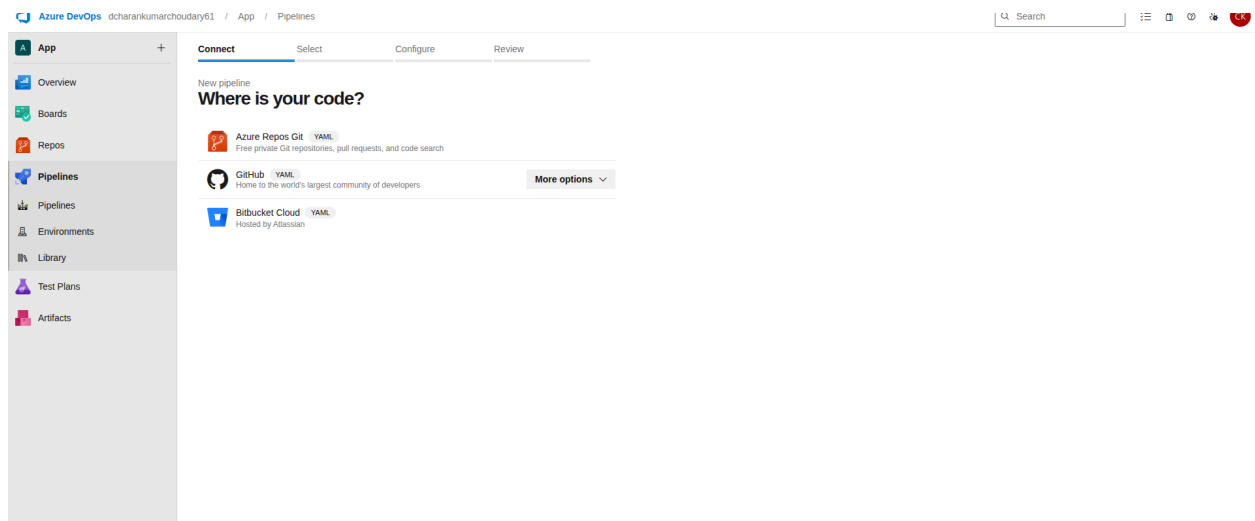
- I selected Previous created repo

● Select the Docker build and Push image to Azure container Registry
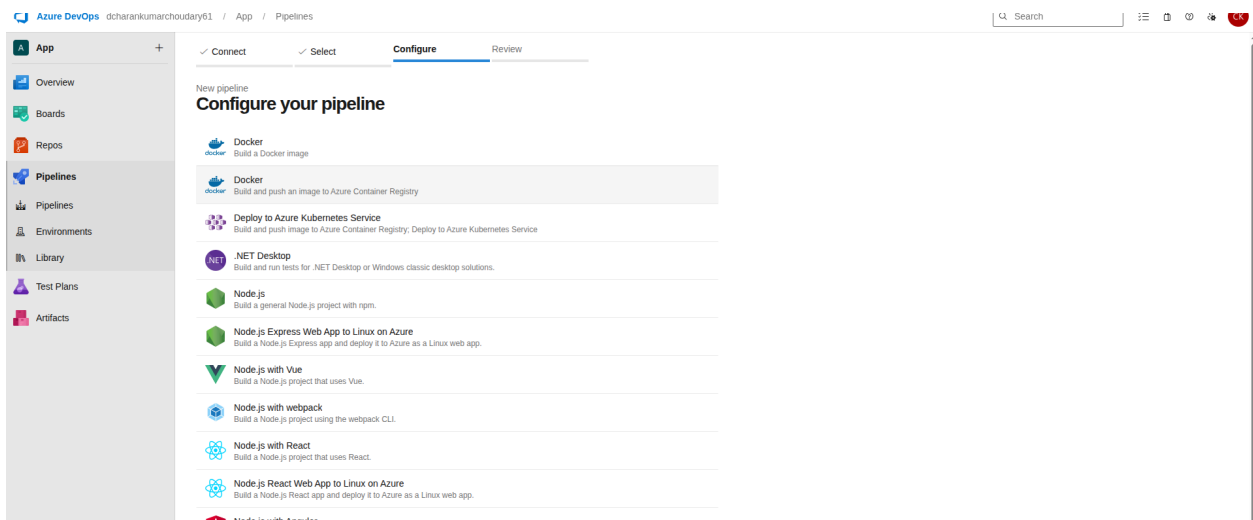
Need to select the azure subscription i have azure container regirsey so this is for authentication to azure container registry and pipeline
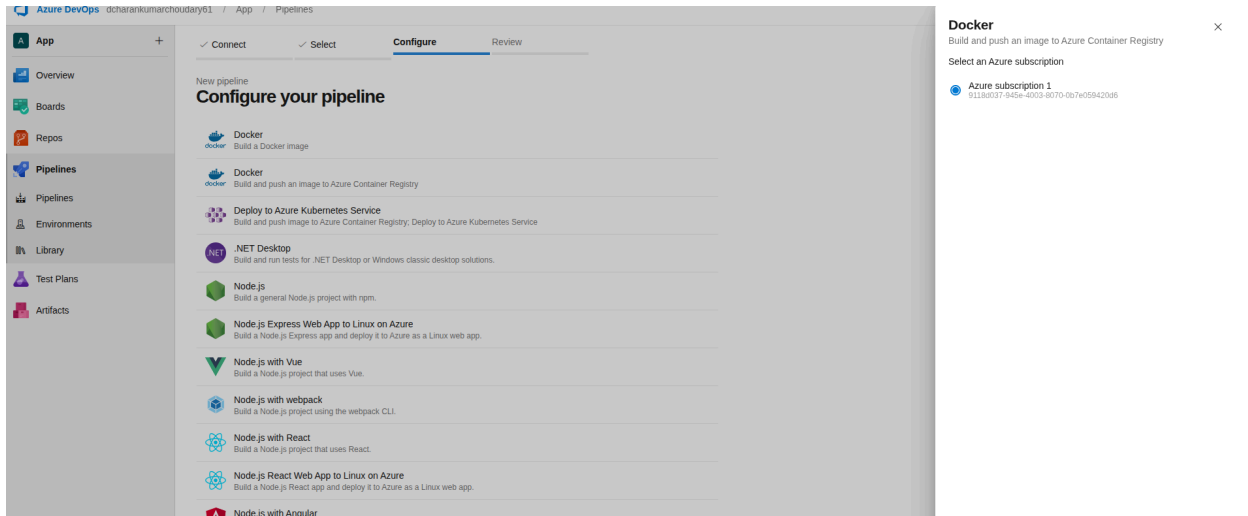
- Select the container registry and click on validate and configure.
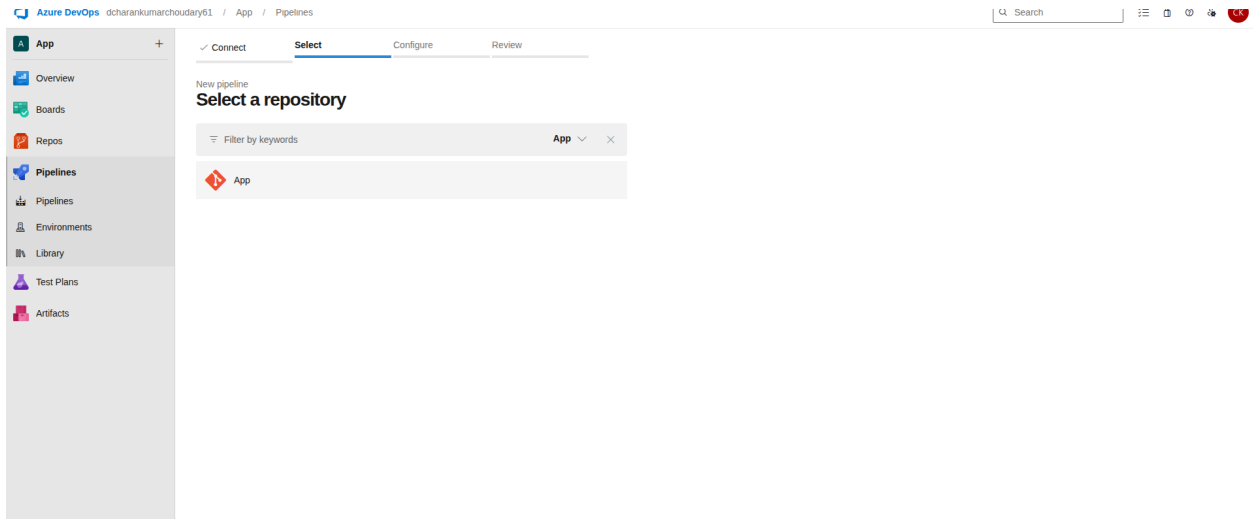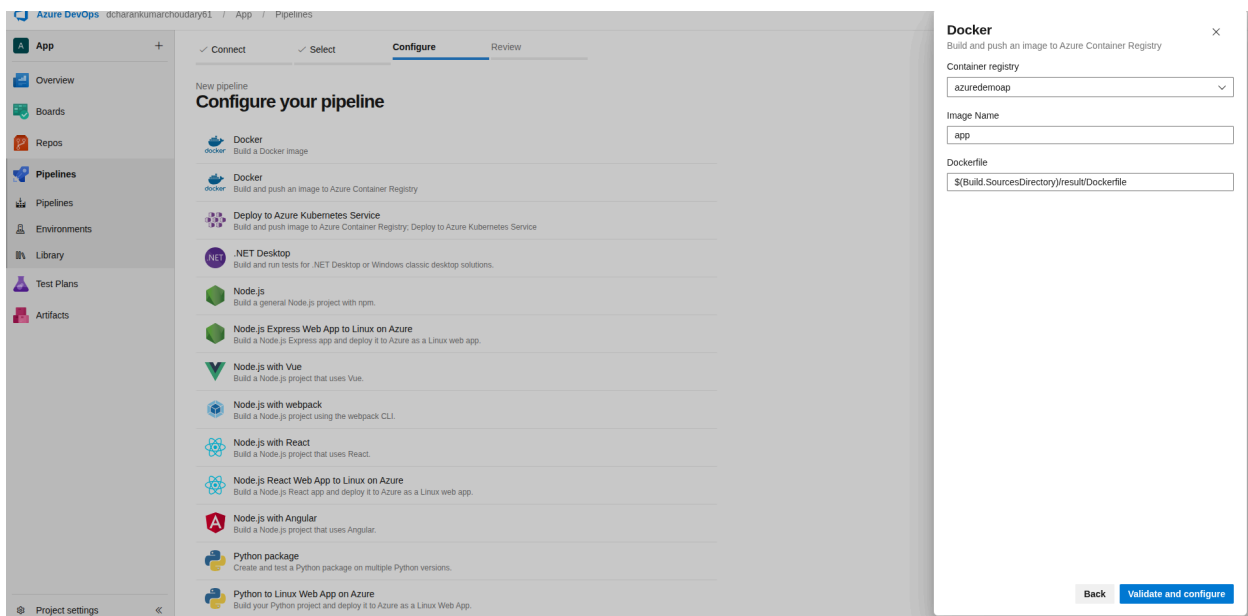

Azure Pipeline structure:

```
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).
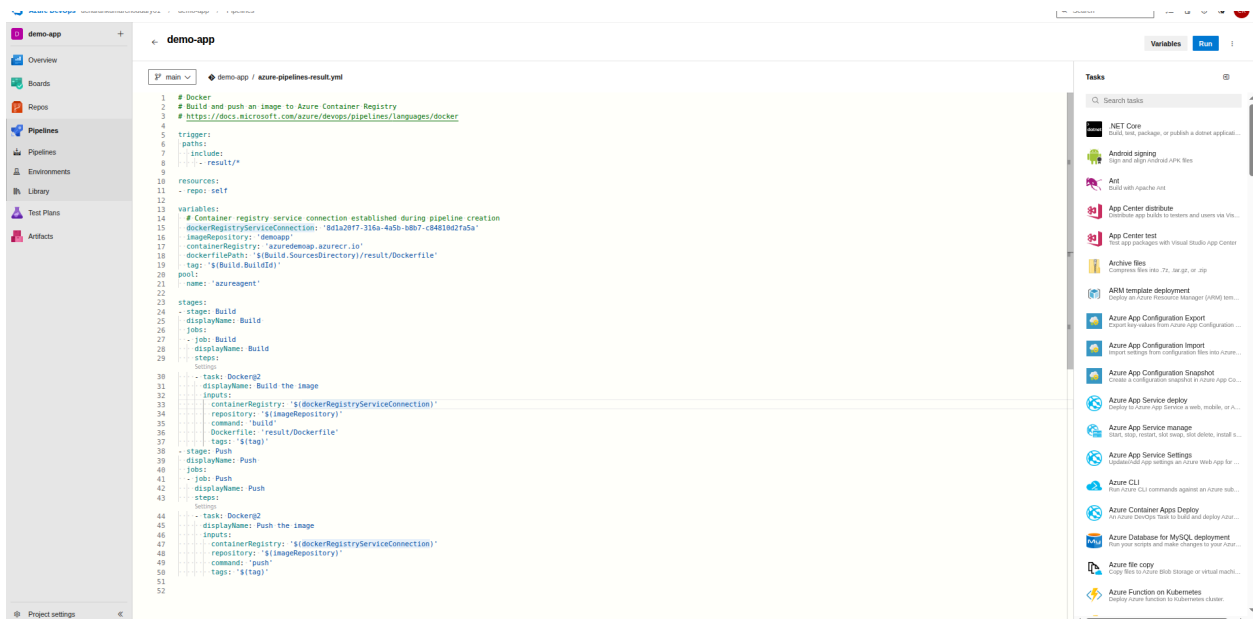

```
trigger:

paths:

  include:

    - result/*

resources:

- repo: self

variables:

  # Container registry service connection established during pipeline
creation

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

  imageRepository: 'demoapp'
```

```yaml
    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.

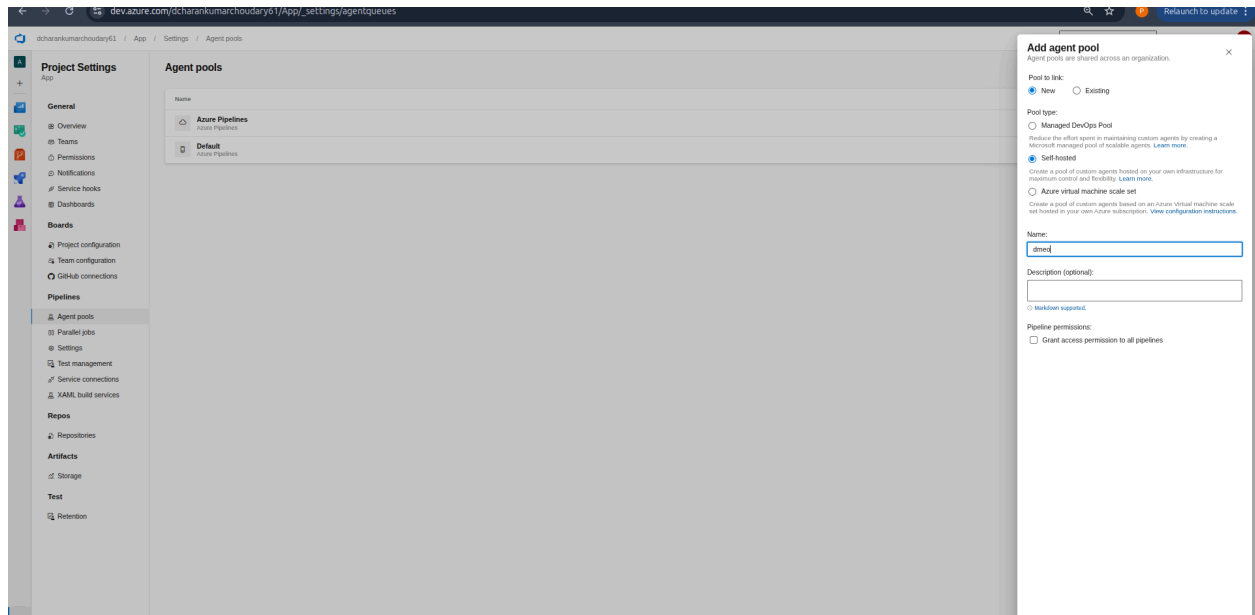**Install prerequisites** on the VM

- Update packages.
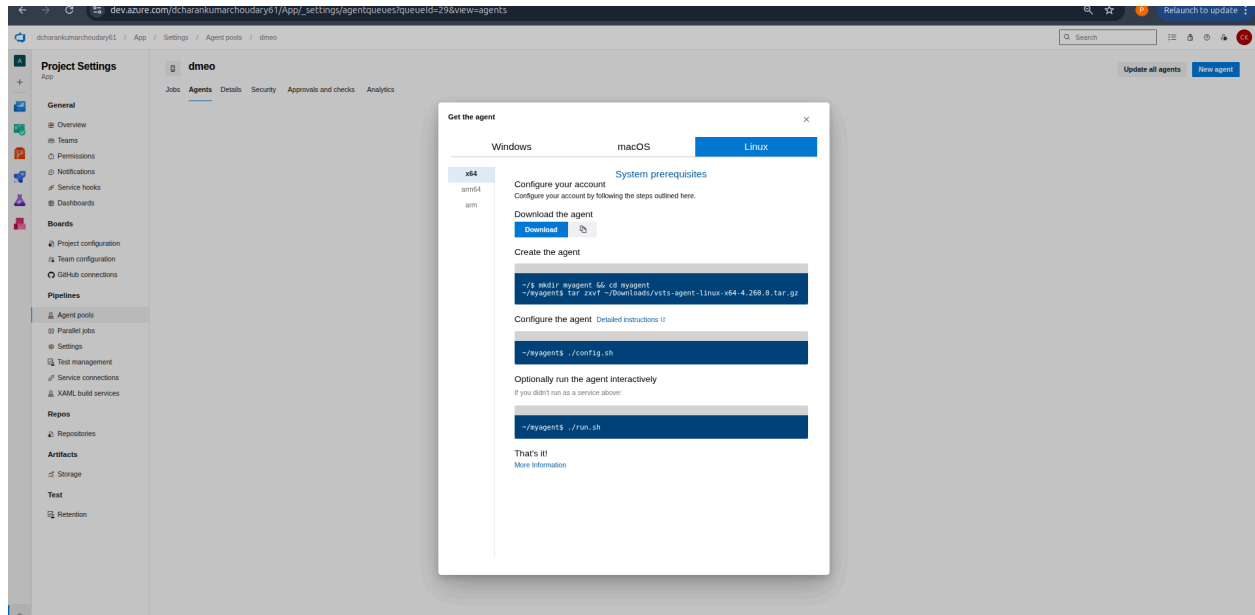
Create a Personal Access Token (PAT) in Azure DevOps

- 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

dcharankumarchoudary61 / App / Settings / Agent pools / dmeo

Search

Project Settings
App

General
- Overview
- Teams
- Permissions
- Notifications
- Service hooks
- Dashboards

Boards
- Project configuration
- Team configuration
- GitHub connections

Pipelines
- Agent pools
- Parallel jobs
- Settings
- Test management
- Service connections
- XAML build services

Repos
- Repositories

Artifacts
- Storage

Test
- Retention

dmeo

Jobs   Agents   Details   Security   Approvals and checks   Analytics

Update all agents   New agent

Get the agent   ✕

| Windows | macOS | Linux |

System prerequisites

Configure your account
Configure your account by following the steps outlined here.

Download the agent
Download

Create the agent

```
~/$ mkdir myagent && cd myagent
~/myagent$ tar zxvf ~/Downloads/vsts-agent-linux-x64-4.260.0.tar.gz
```

Configure the agent   Detailed instructions

```
~/myagent$ ./config.sh
```

Optionally run the agent interactively
If you didn't run as a service above:

```
~/myagent$ ./run.sh
```

That's it!
More information

```
./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) > 3eXJwdecPUxpLXSEWOFGbbCehL33mBZTSFYip3FOF3U90OEdFYzsJQQJ99BHACAAAAAAAAAAAAASAZDO44KD
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.
```

```
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) > 3eXJwdecPUxpLXSEWOFGbbCehL33mBZTSFYip3FOF3U90OEdFYzsJQQJ99BHACAAAAAAAAAAAAASAZDO44KD
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.
Enter work folder (press enter for _work) >
2025-08-13 07:59:38Z: Settings Saved.
azureuser@azureagent:~/myagent$ ./config.sh

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

Error reported in diagnostic logs. Please examine the log for more details.
   - /home/azureuser/myagent/_diag/Agent_20250813-080006-utc.log
Cannot configure the agent because it is already configured. To reconfigure the agent, run 'config.cmd remove' or './config.sh remove' first.
azureuser@azureagent:~/myagent$ ./run.sh
Scanning for tool capabilities.
Connecting to the server.
2025-08-13 08:00:29Z: Listening for Jobs
```

- 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

CD-Part:

STEP 1: Creat a EKS CLUSTER

STEP 2: INSTALL ARGOCD AND ACCESS THE ARGOCD DASHBOARD

STEP 3: SETUP-AZURE REPO
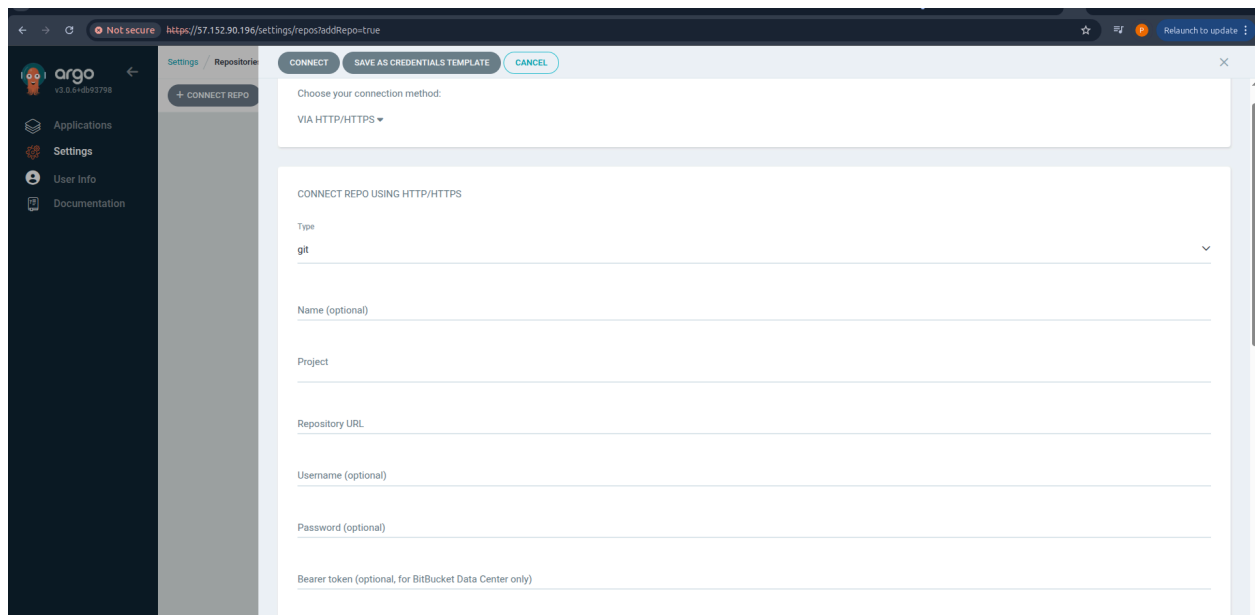
- Need to Creat a Token in azure repo

STEP 4: WRITE SHELL SCRIPT TO AUTOMATICALLY UPDATE THE BUILD NUMBER IN DEPLOYMENT.YAML-FILE

- NEED TO CREATE FOLDER "scripts/updateK8SManifests.sh

```bash
#!/bin/bash

set -x

# Set the repository URL
REPO_URL="https://FciEkLDKq6OHNA15kX9U9snjie7mJNRppwcVKCc2EZWfgzG7UnfhJQQJ99BHA
CAAAAAAAAAAAAASAZDOsmNP@dev.azure.com/dcharankumarchoudary61/demo-app/_git/demo
-app"

# Clone the git repository into the /tmp directory
git clone "$REPO_URL" /tmp/temp_repo
```

```bash
# Navigate into the cloned repository directory
cd /tmp/temp_repo

# Make changes to the Kubernetes manifest file(s)
# For example, let's say you want to change the image tag in a deployment.yaml
file
sed -i "s|image:.*|image: azuredemoap.azurecr.io/$2:$3|g"
k8s-specifications/$1-deployment.yaml

# Add the modified files
git add .

# Commit the changes
git commit -m "Update Kubernetes manifest"

# Push the changes back to the repository
git push

# Cleanup: remove the temporary directory
rm -rf /tmp/temp_repo
```
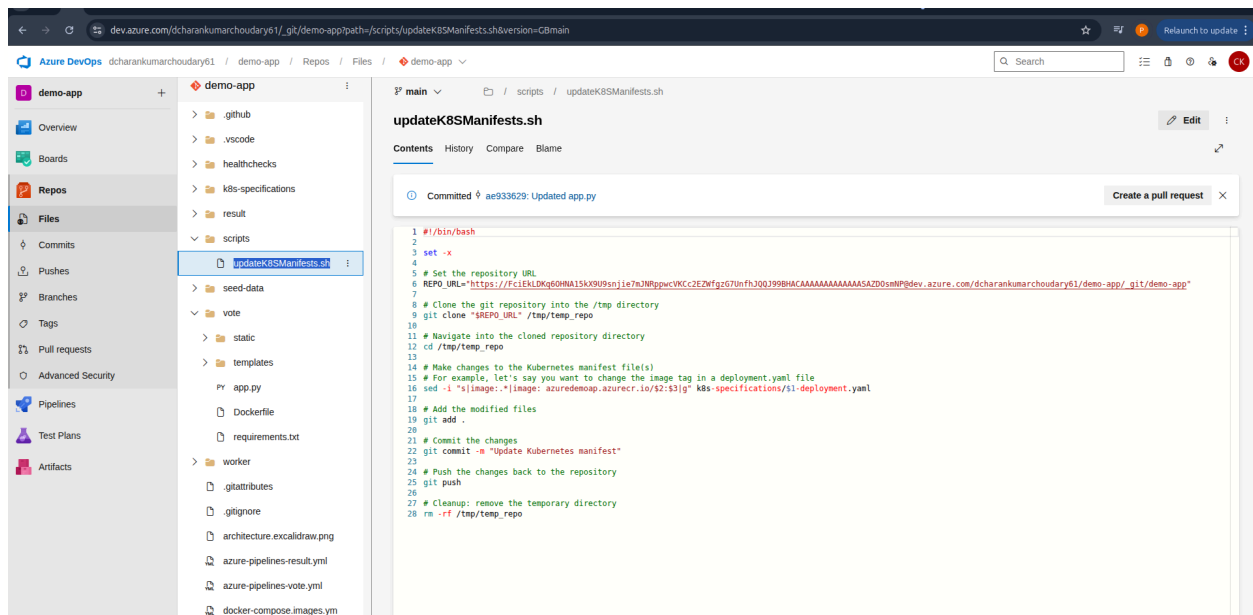

STEP 4: NEED TO ADD THE STAGE IN PIPELINE TO RUN THE SCRIPT.

```yaml
- stage: update

  displayName: update

  jobs:

  - job: update

    displayName: update

    steps:

      - task: ShellScript@2

        inputs:

          scriptPath: 'scripts/updateK8SManifests.sh'

          args: 'vote $(imageRepository) $(tag)'
```
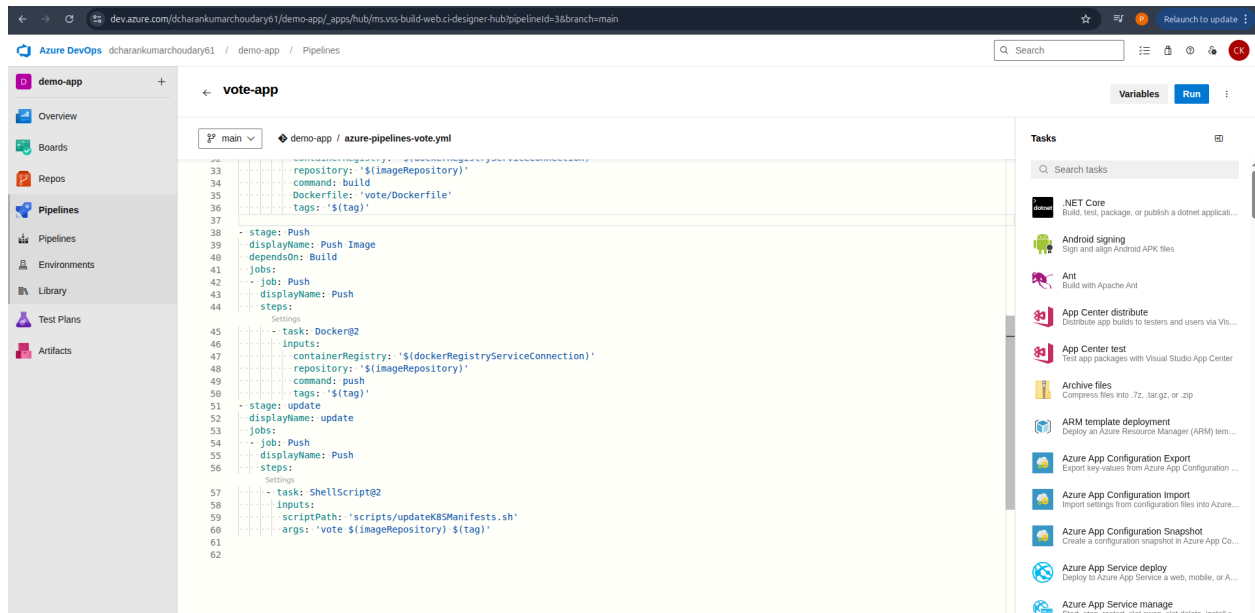
The above Stage is automatically update the build number with tag deployment file argocd will pickup this changes.

Here we will get error imagepullbackoff as the azurecontainer register is private so we need create a secret a need mention that in deployment file with Imagepull secret

```
kubectl create secret docker-registry <secret-name> \
    --namespace <namespace> \
    --docker-server=<container-registry-name>.azurecr.io \
    --docker-username=<service-principal-ID> \
    --docker-password=<service-principal-password>
```

Here Container registry name

Need to generate the username and password

- Go to access key Need create a password .

With this create a secret and mention the secret name in deployment.yaml file
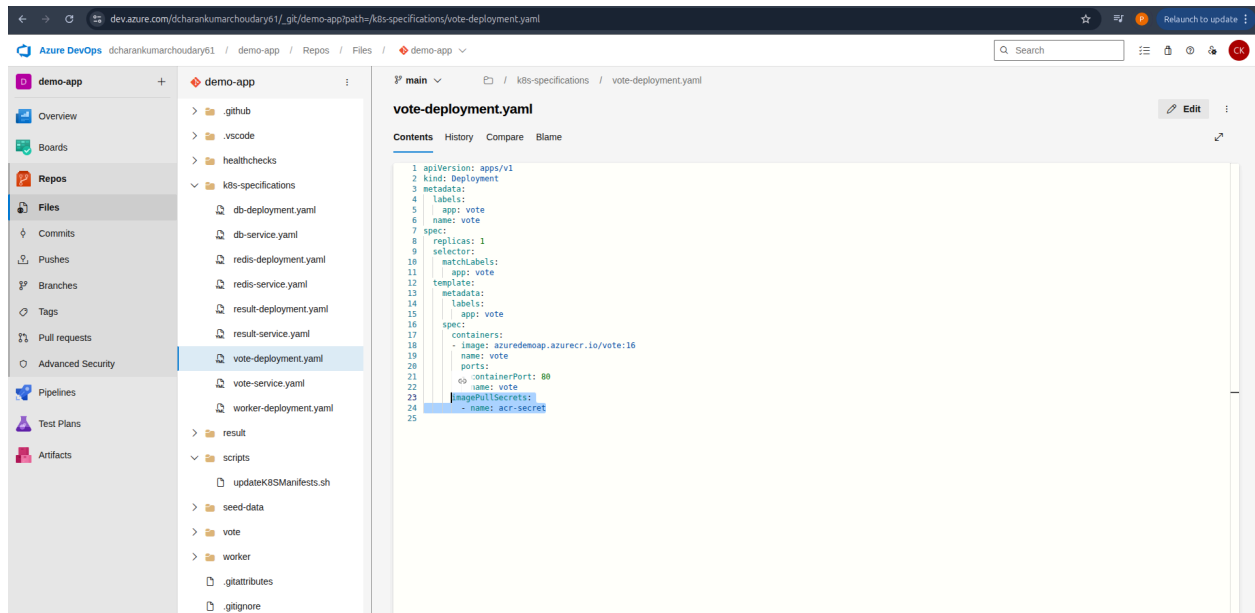
I expose the svc as LB access the application using LB