Documentation:

1) Create and set up a new Azure DevOps Account
2) Create a New Project
3) Set up a new Azure Repository and push source code from local repository to Azure Repos using Git commands.

git init – initialize working directory

git clone <Azure repository-url> - Clone the Azure Repository url to local repository

git add . - Stage all files

git commit -m "Your descriptive commit message here" - Commit stage changes with message

git push origin <branch-name> - Push commit changes to Azure Repos branch

4) Set up CICD Pipeline
   A) Go to Pipeline -> New Pipeline
   B) Select Azure Repos to get the latest code
   C) Select the Repository
   D) Configure Build and Deploy Pipeline (i.e. Build & Deploy to AKS)
   E) Set up variables and service connections with ACR, Docker Registry etc...
   F) Configure both Build and Deploy stage in a single CICD yaml pipeline file
   G) Build Artifacts produced by build pipeline are used in Deploy stage

```yaml
trigger: # trigger the main branch
- main

variables: # define service connections to Azure Portal
  dockerRegistryServiceConnection: 'e9c630f6-f62d-4eb8-8100-196437b7648b'
  imageRepository: 'helloworld'
  containerRegistry: 'congre17.azurecr.io'
  dockerfilePath: '**/Dockerfile'
  tag: '$(Build.BuildId)'
  imagePullSecret: 'congre17cf44-auth'

  vmImageName: 'ubuntu-latest' # Microsoft hosted agent to run the CICD pipeline


stages: #Logical Boundary in Azure Pipeline
- stage: Build # Build Stage
  displayName: Build stage
  jobs: # Consists of one or more steps
  - job: Build
```

```yaml
    displayName: Build #Build job
    pool: # Uses Microsoft hosted Agent to run pipeline
      vmImage: $(vmImageName)
    steps: # Contains one or more task
    - task: Docker@2 #Docker task to build & Push image to ACR
      displayName: Build and push an image to container registry
      inputs:
        command: buildAndPush
        repository: $(imageRepository)
        dockerfile: $(dockerfilePath)
        containerRegistry: $(dockerRegistryServiceConnection)
    - task: PublishBuildArtifacts@1 #Build Artifacts are used in Deploy stage
      inputs:
        PathtoPublish: '$(Build.ArtifactStagingDirectory)'
        ArtifactName: 'drop'
        publishLocation: 'Container'


- stage: Deploy #Deploy stage
  jobs:
    - job: Deploy
      displayName: Deploy
      steps:
      - task: Kubernetes@1 #Uses Kubernetes v1 to deploy application AKS
        inputs:
          connectionType: 'Kubernetes Service Connection'
          kubernetesServiceEndpoint: 'Kubecluster1-default'
          command: 'apply' # applies the config files using kubectl apply
          useConfigurationFile: true
          secretType: 'dockerRegistry'
          containerRegistryType: 'Azure Container Registry'

      - task: DownloadBuildArtifacts@1 #Download build artifacts
        inputs:
          buildType: 'current'
          downloadType: 'single'
          artifactName: 'drop'
          downloadPath: '$(System.ArtifactsDirectory)'
```

5) Save and commit to master branch in-order to trigger the pipeline