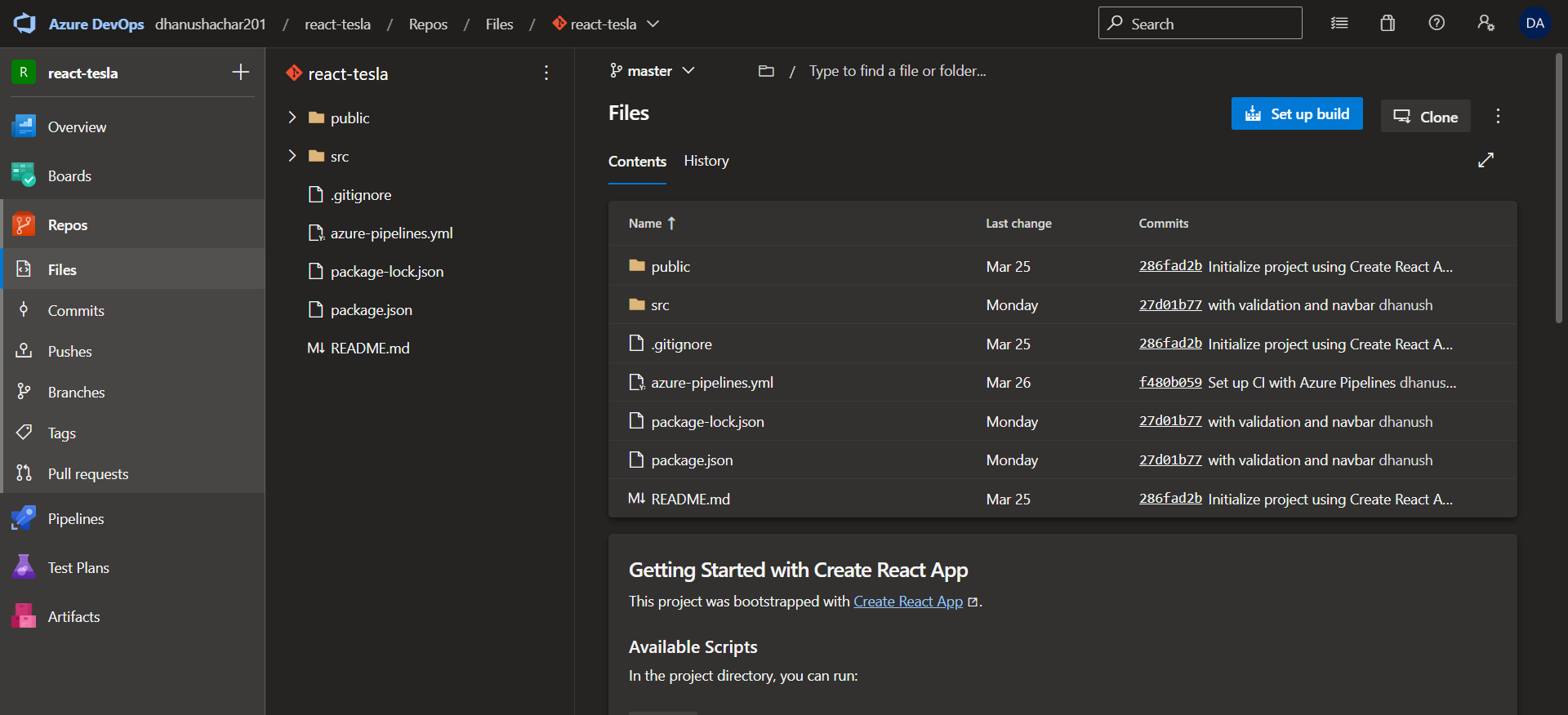
**DevOps Work Updates**

**Sprint 1**

**1 Creating CI/CD for react front-end in Azure DevOps**



Yaml file CI

# Define the pipeline name and trigger

name: ReactJS-CI

trigger:

- main

# Define the pool of agents to use

pool:

  vmImage: 'ubuntu-latest'

# Define the stages of the pipeline

stages:

- stage: Build

  jobs:

  - job: Build

    steps:

    - task: NodeTool@0

      inputs:

        versionSpec: '14.x'

      displayName: 'Install Node.js'

    - script: |

        npm install

        npm run build

      displayName: 'Install dependencies and build the app'

    - task: ArchiveFiles@2

      inputs:

        rootFolderOrFile: '$(System.DefaultWorkingDirectory)/build'

        includeRootFolder: false

        archiveFile: '$(Build.ArtifactStagingDirectory)/$(Build.BuildId).zip'

      displayName: 'Archive build artifacts'

    - publish: $(Build.ArtifactStagingDirectory)/$(Build.BuildId).zip

      artifact: build

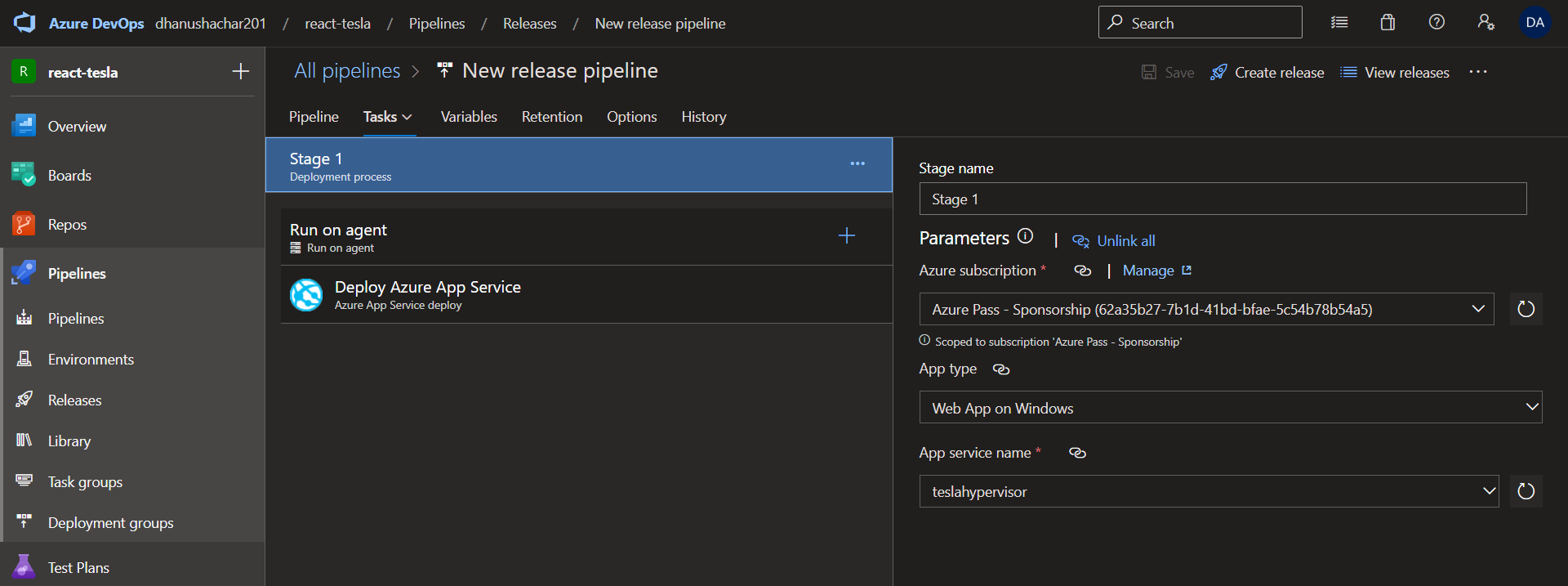
      displayName: 'Publish build artifacts'

In the yaml file as mentioned it creates build artifacts and publishes it in drop folder for release pipeline

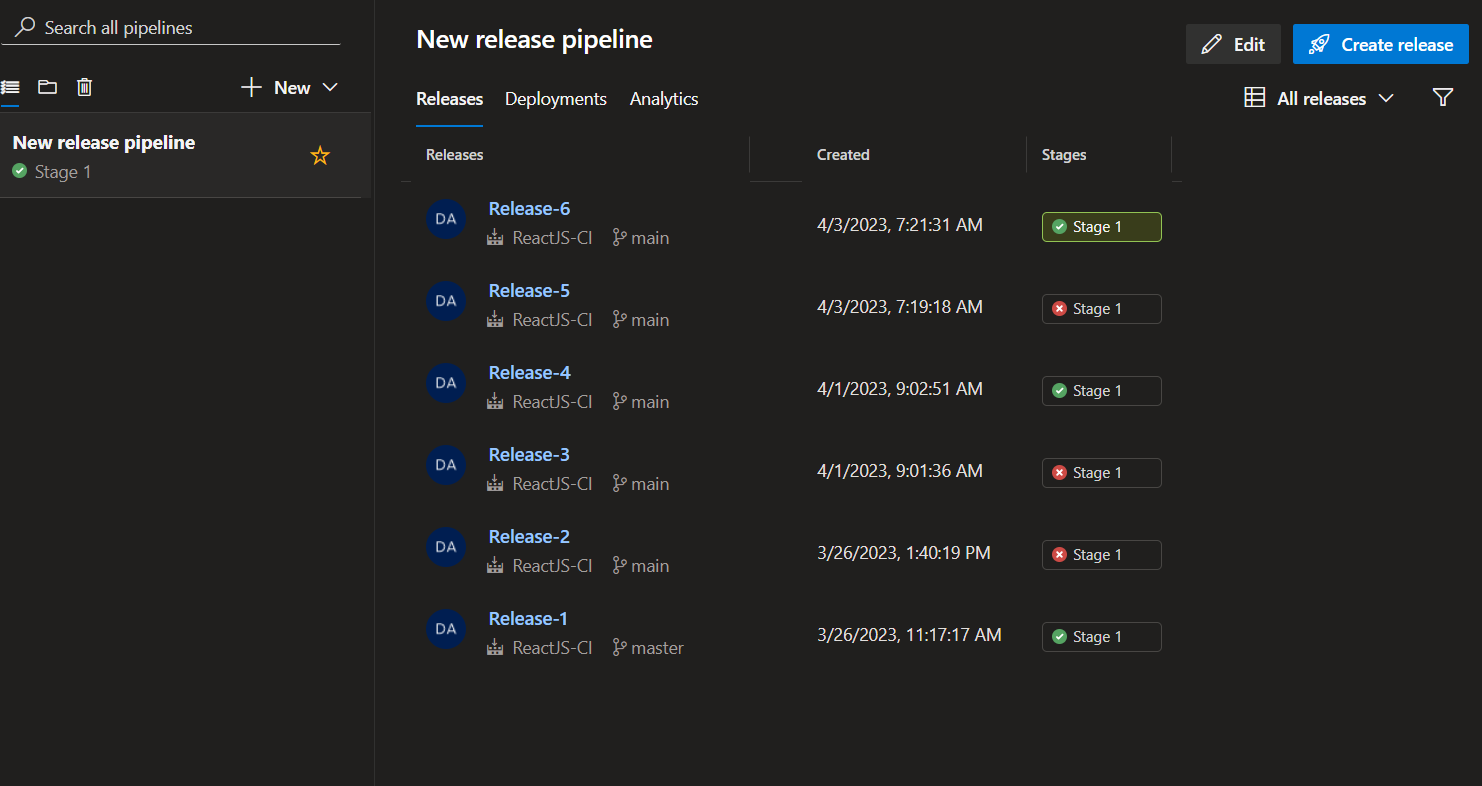
Graphical user interface, text, application, Teams

Description automatically generated

Release Pipeline



The release of zip file will be done in App service as shown below



In App service

Graphical user interface, application, email

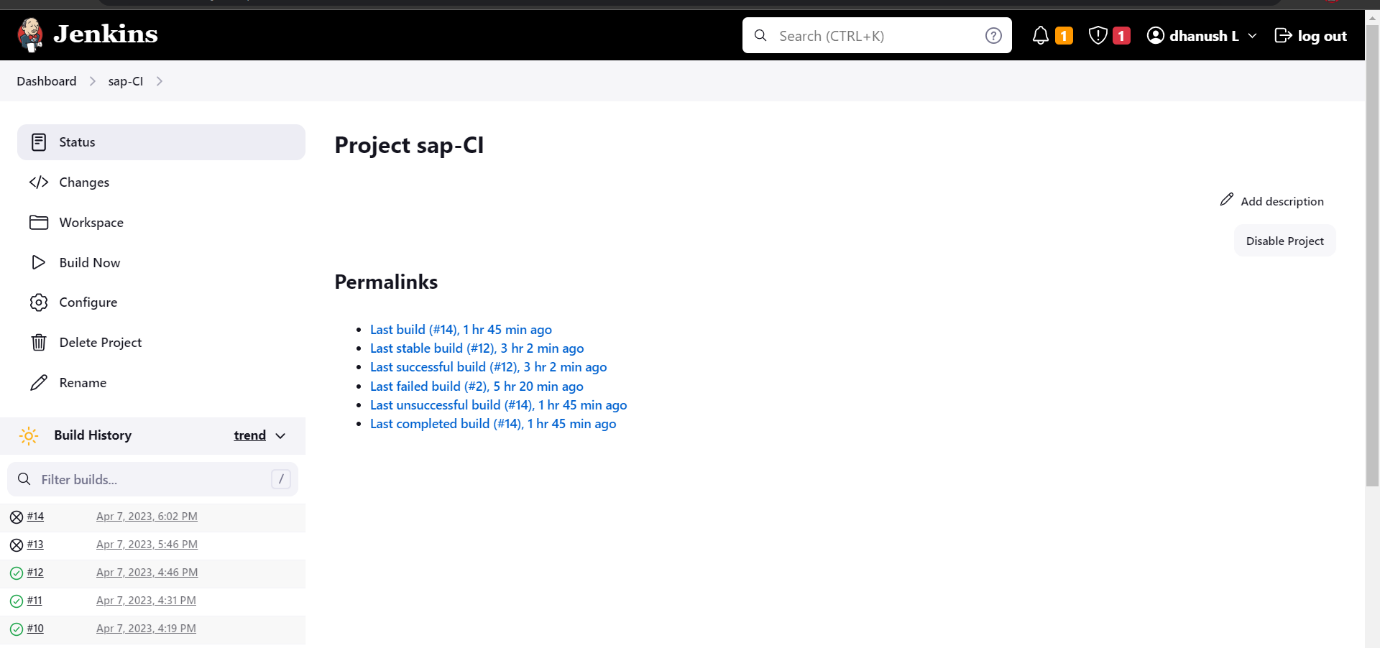
Description automatically generated

URL of AppService

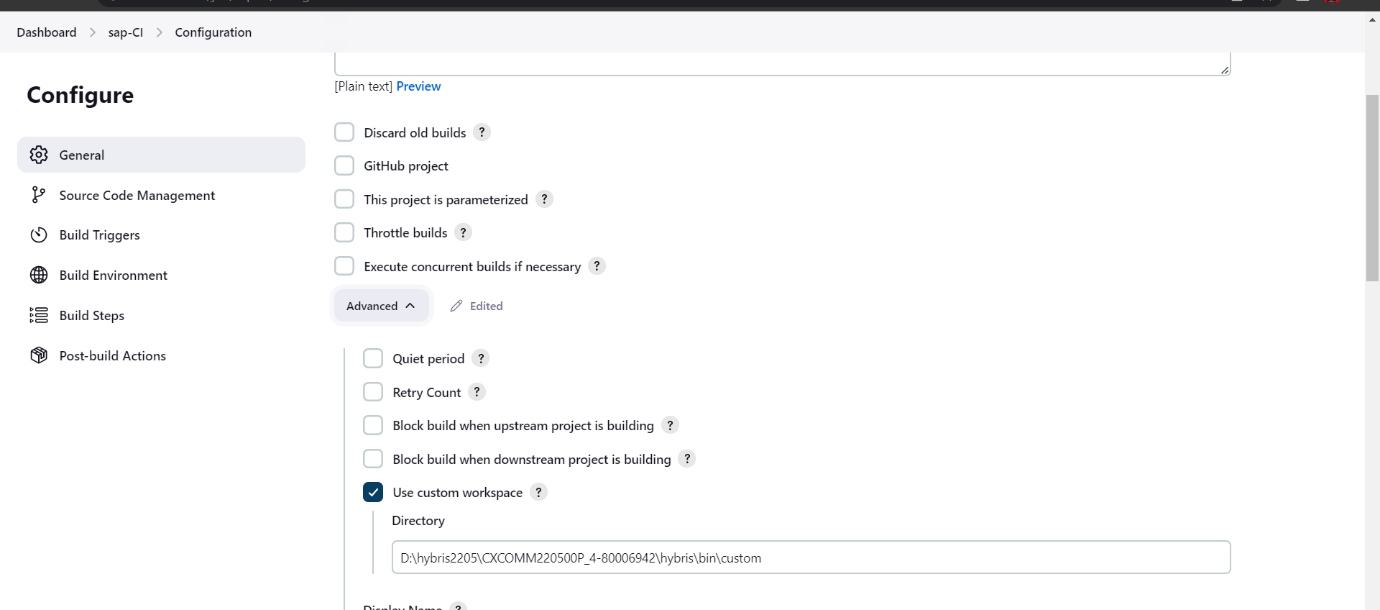
<https://teslahypervisor.azurewebsites.net/>

**Sprint 2**

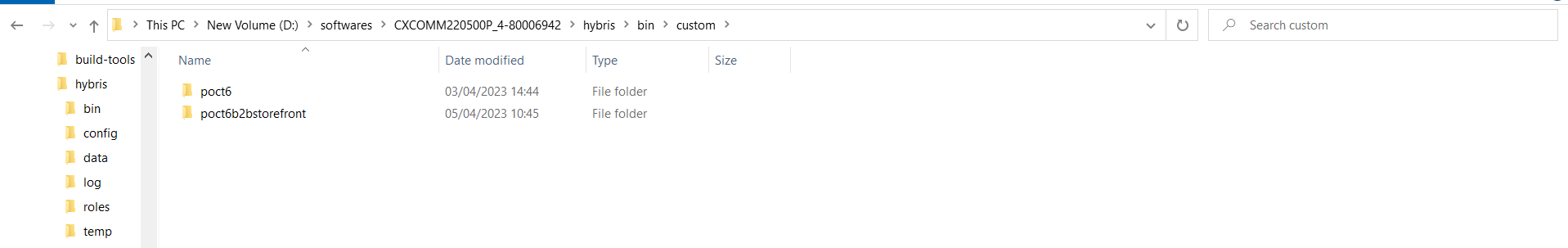
**2. implementing Jenkins CI for SAP Hybris**

****

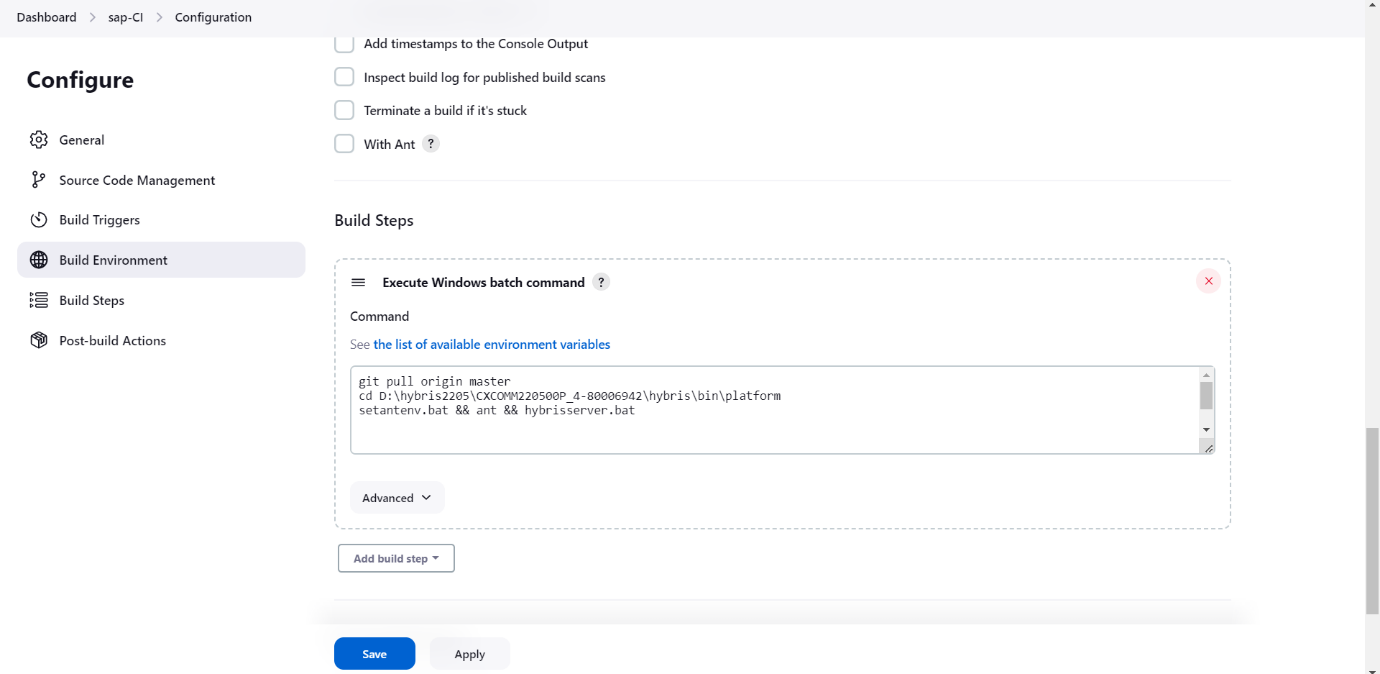
**Setting up the workspace directory**

****

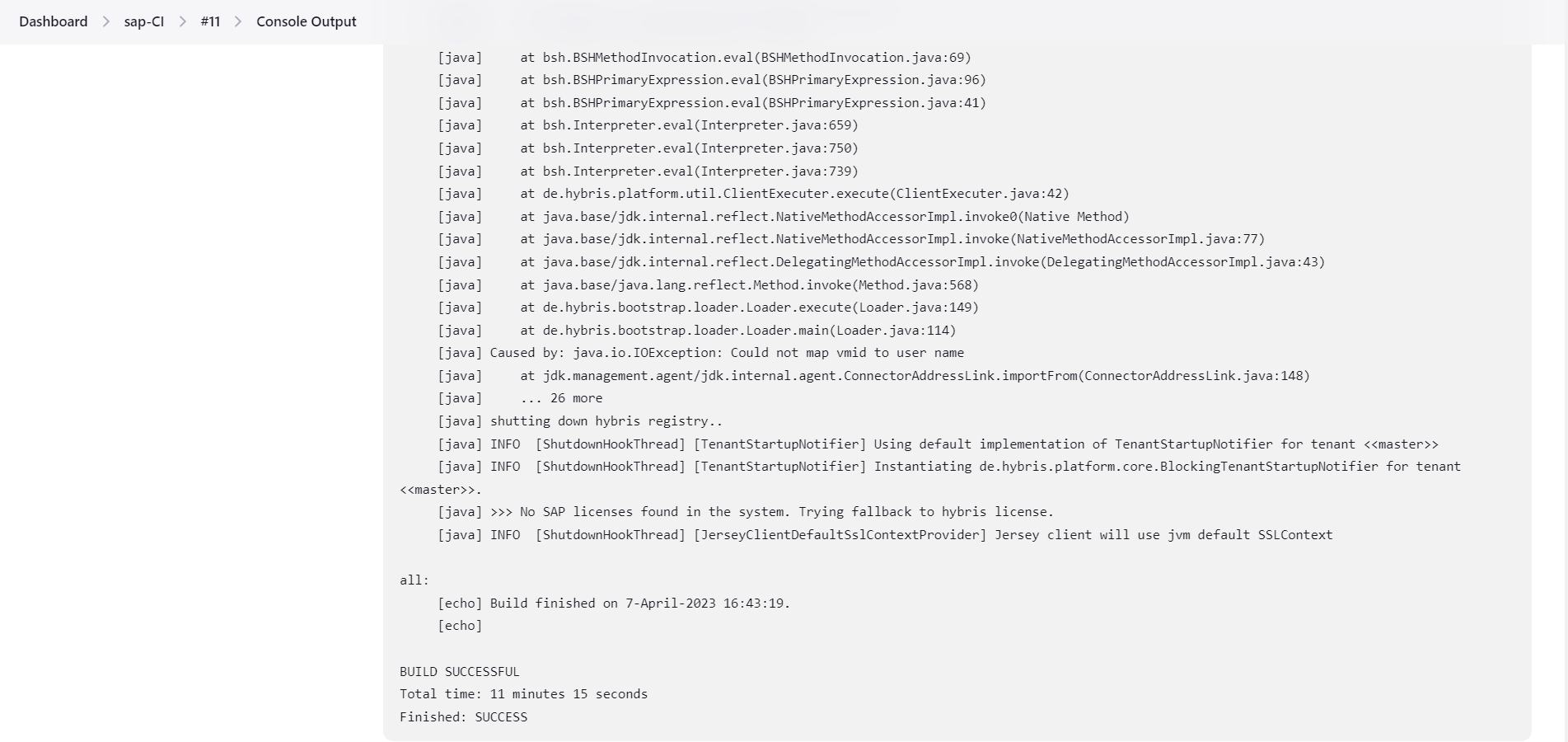
**This is our work space directory from where git pull will be done on one hour of interval and build’s a production code**

****

**Script for CI using windows batch commands**

****

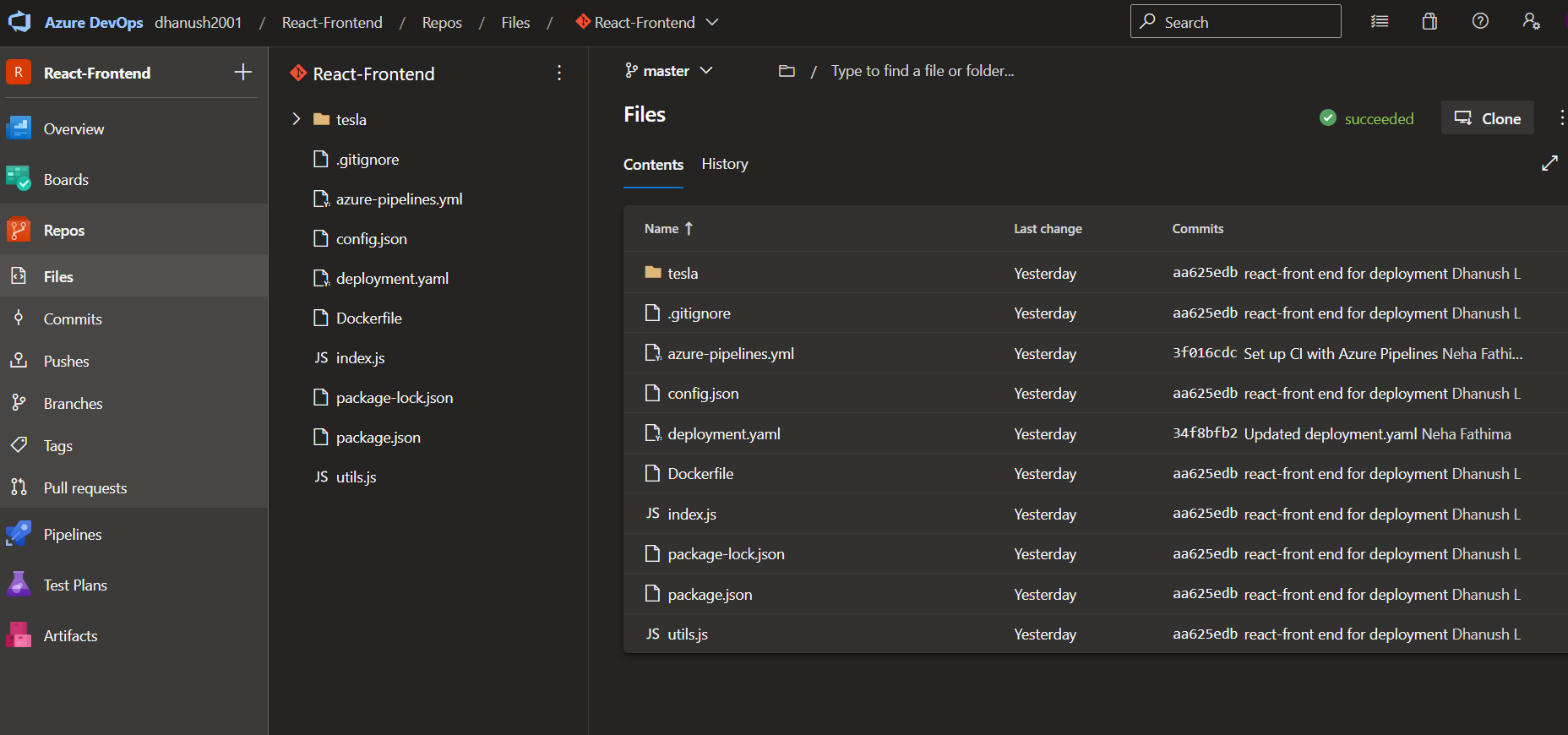
**Final build output**

****

**3.creating docker image of front-end code and hosting it on kubernities using load balancer service**

**Code repository url :**

<https://github.com/dhanush-dev01/react-deployment.git>

****

**Yaml script for CI**

pool:

  name: Azure Pipelines

steps:

- task: Docker@0

  displayName: 'Build an image'

  inputs:

    azureSubscription: 'Azure Pass - Sponsorship (cbdeaee3-372d-4773-b1b1-9806183b8557)'

    azureContainerRegistry: '{"loginServer":"hypertesla.azurecr.io", "id" : "/subscriptions/cbdeaee3-372d-4773-b1b1-9806183b8557/resourceGroups/react-dhanush-deployment/providers/Microsoft.ContainerRegistry/registries/hypertesla"}'

    dockerFile: Dockerfile

    includeLatestTag: true

- task: Docker@0

  displayName: 'Push an image'

  inputs:

    azureSubscription: 'Azure Pass - Sponsorship (cbdeaee3-372d-4773-b1b1-9806183b8557)'

    azureContainerRegistry: '{"loginServer":"hypertesla.azurecr.io", "id" : "/subscriptions/cbdeaee3-372d-4773-b1b1-9806183b8557/resourceGroups/react-dhanush-deployment/providers/Microsoft.ContainerRegistry/registries/hypertesla"}'

    action: 'Push an image'

    includeLatestTag: true

- task: CopyFiles@2

  displayName: 'Copy Files to: $(build.artifactstagingdirectory)'

  inputs:

    Contents: deployment.yaml

    TargetFolder: '$(build.artifactstagingdirectory)'

- task: PublishBuildArtifacts@1

  displayName: 'Publish Artifact: drop'

**docker code**

FROM node:14-alpine AS base

WORKDIR /app

COPY package\*.json ./

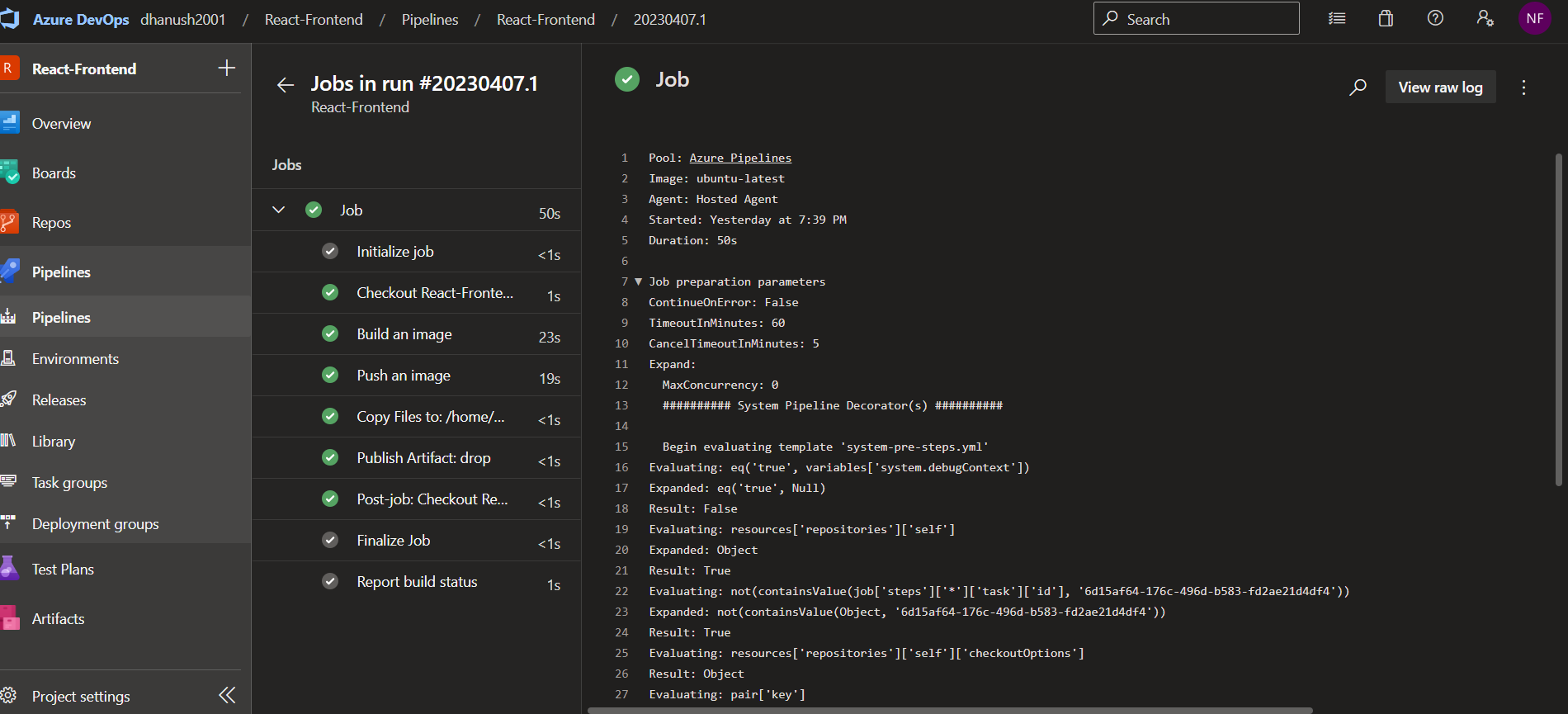
RUN npm install mongoose cors express

COPY . .

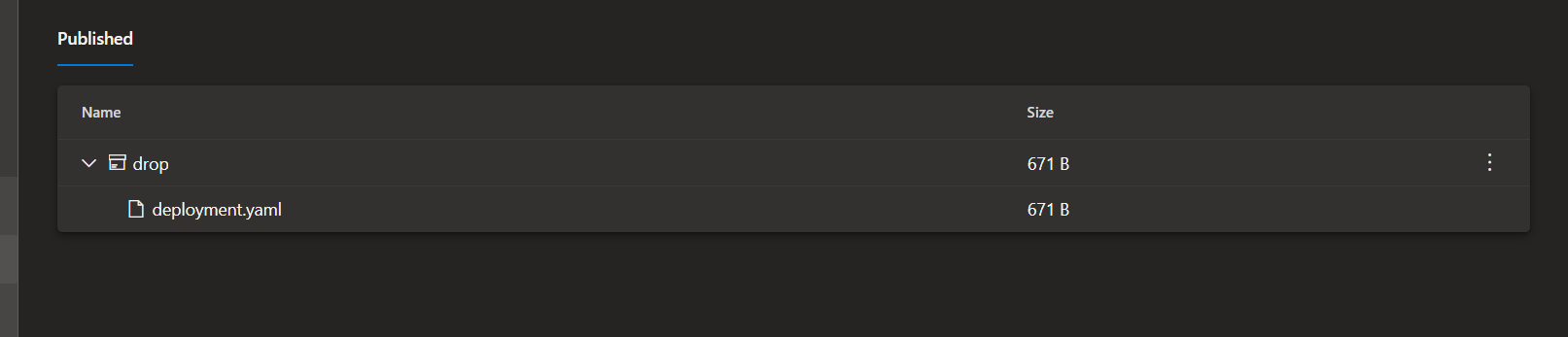
EXPOSE 7080

CMD ["node","index.js"]

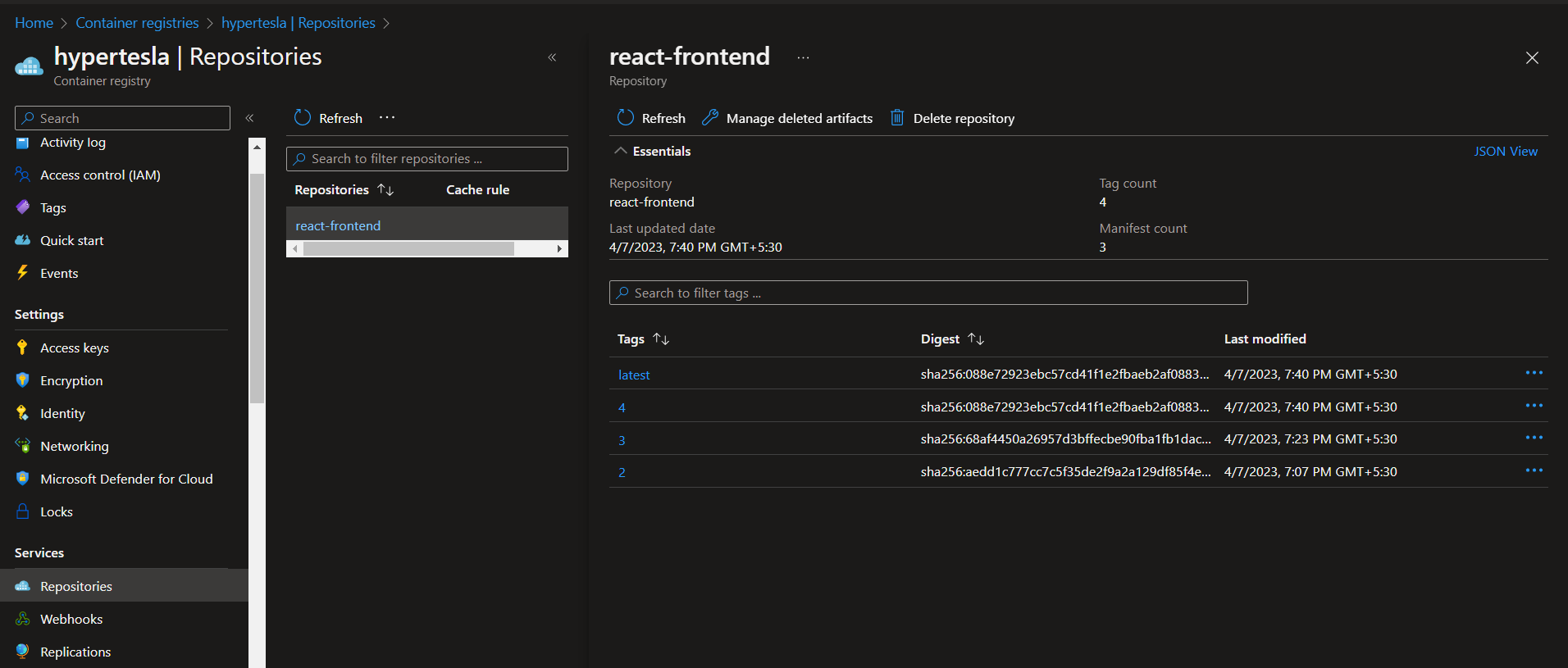
**CI Job**



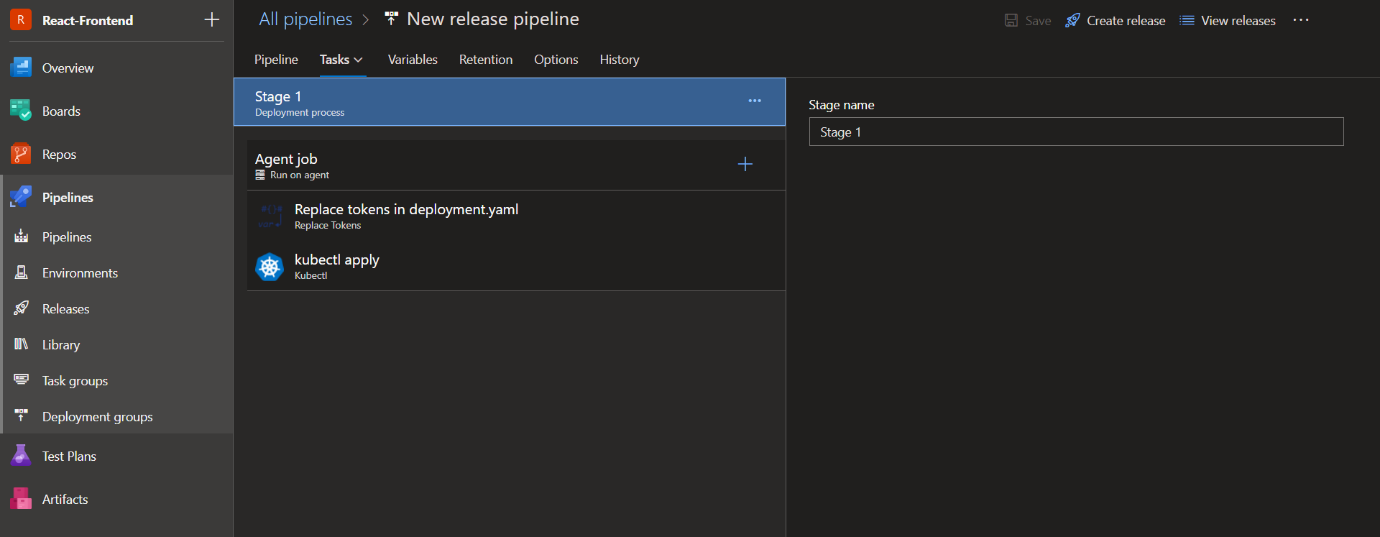
**Drops deployment.yaml file for aks deployment**

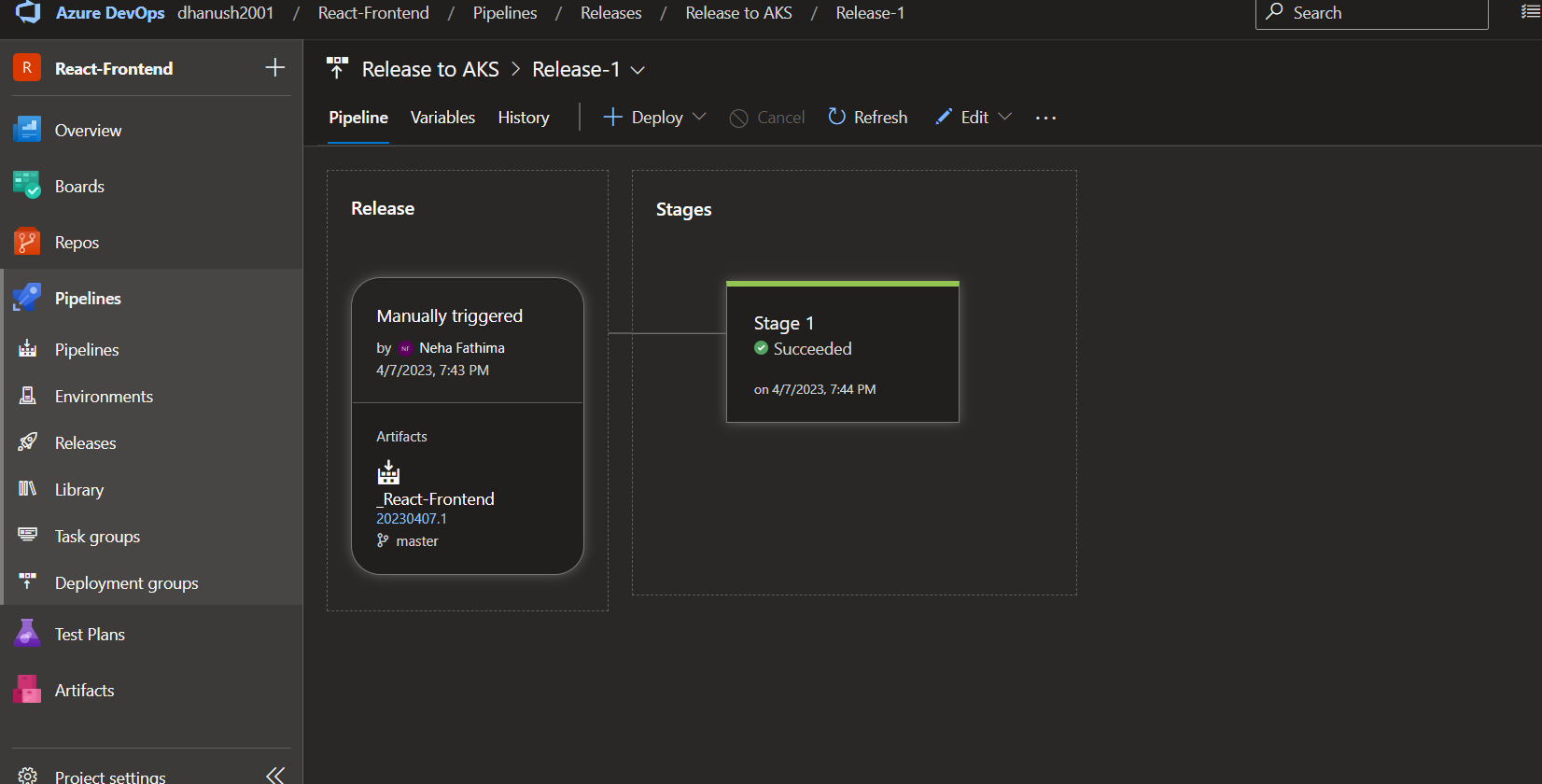
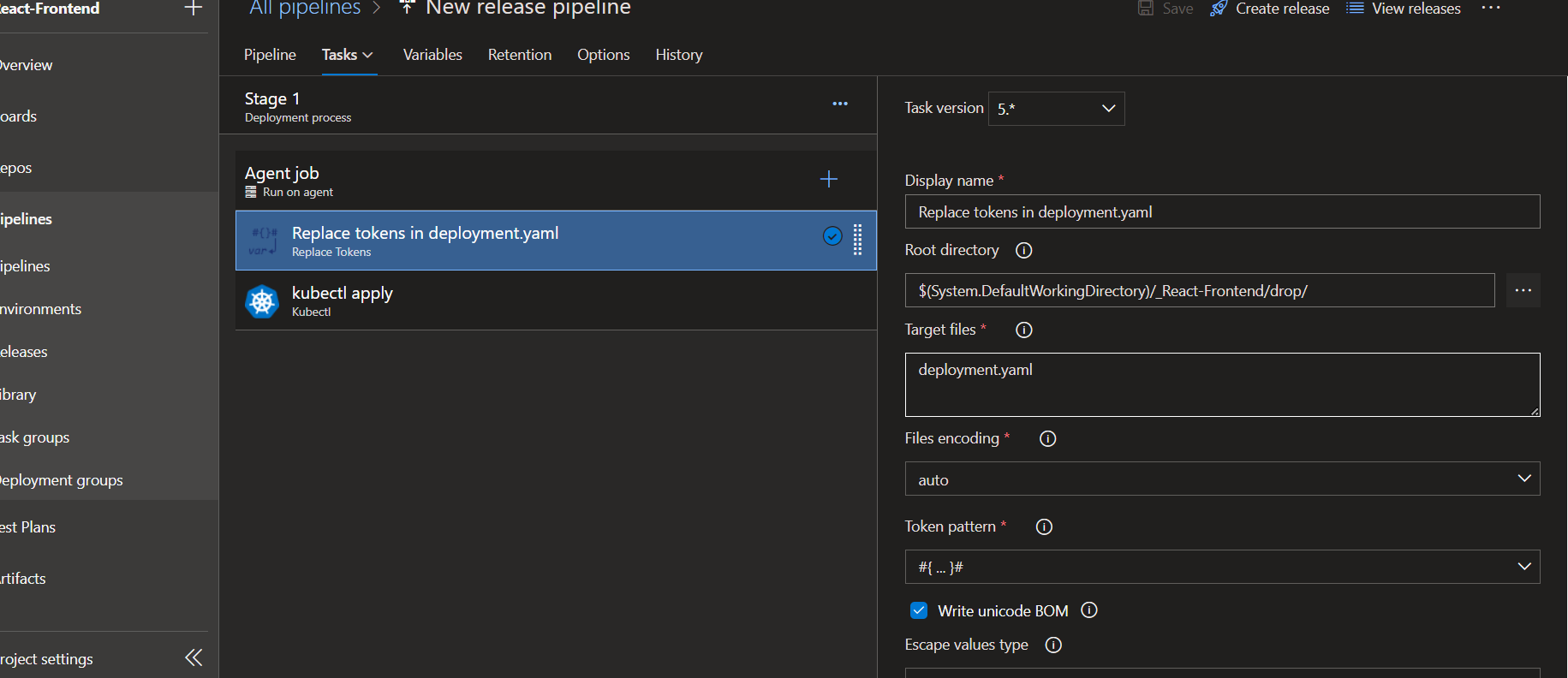


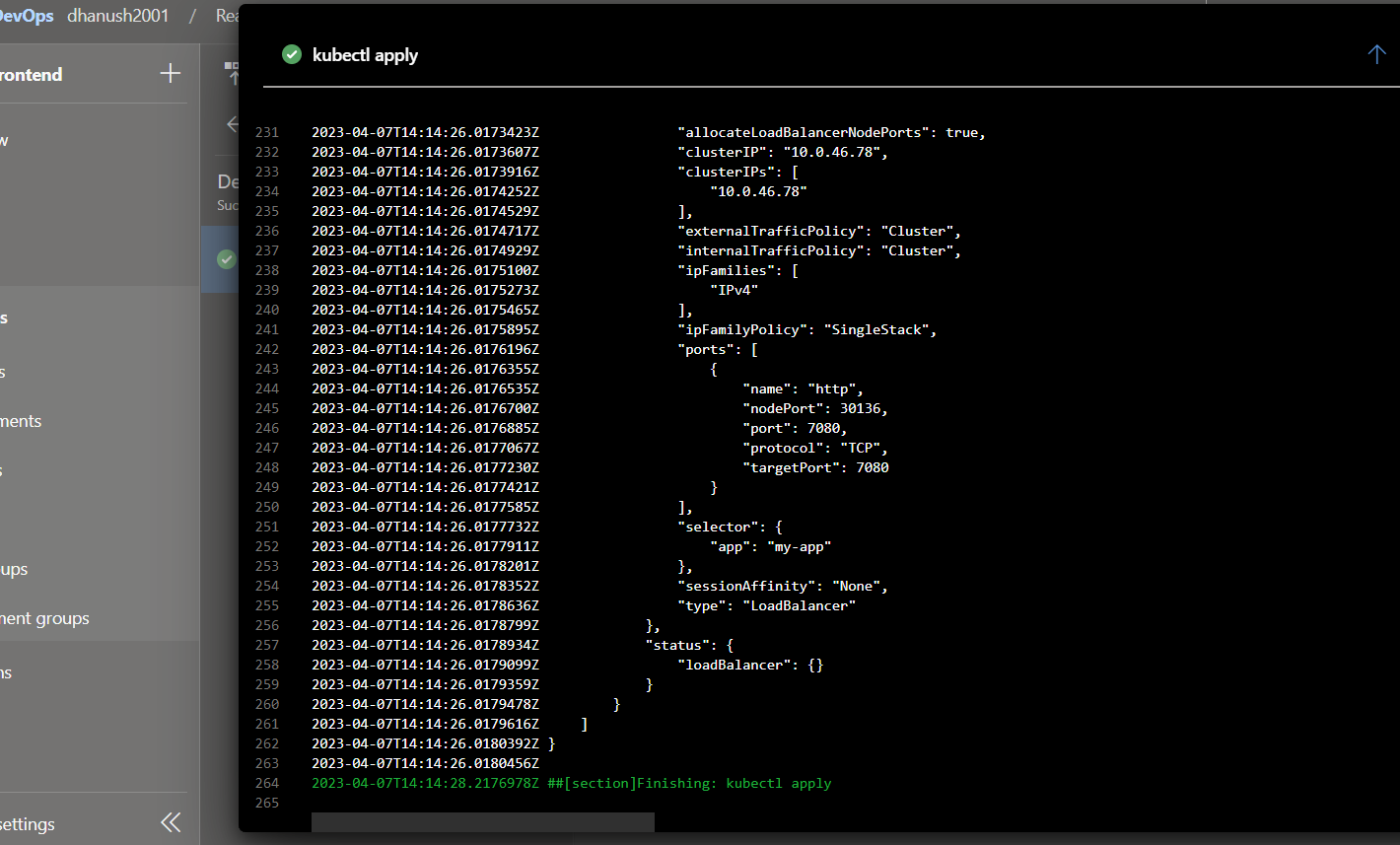
**After CI reactjs docker image with be stored in ACR with a latest tag**

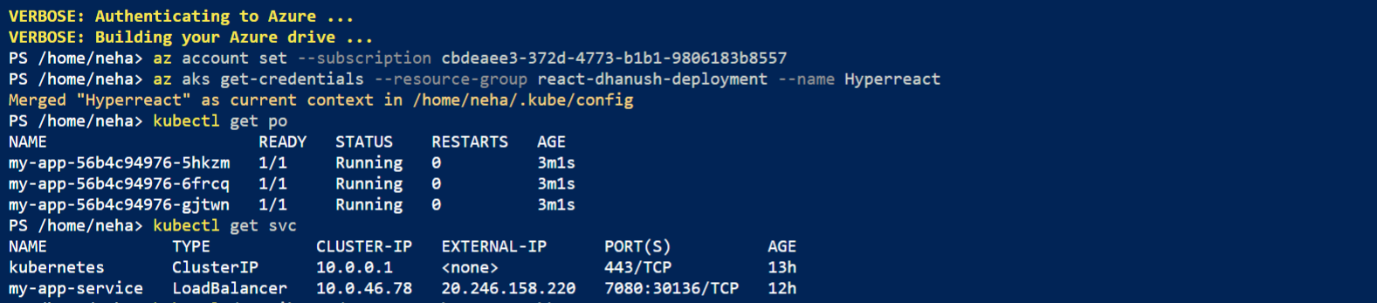
****

**Release Pipeline**

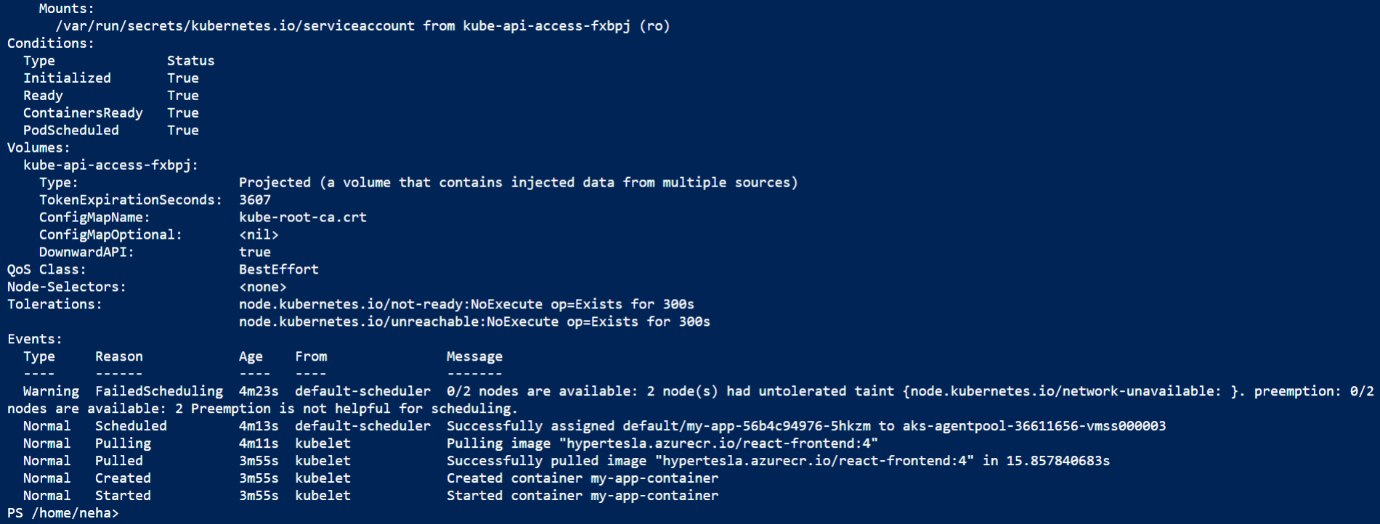
****

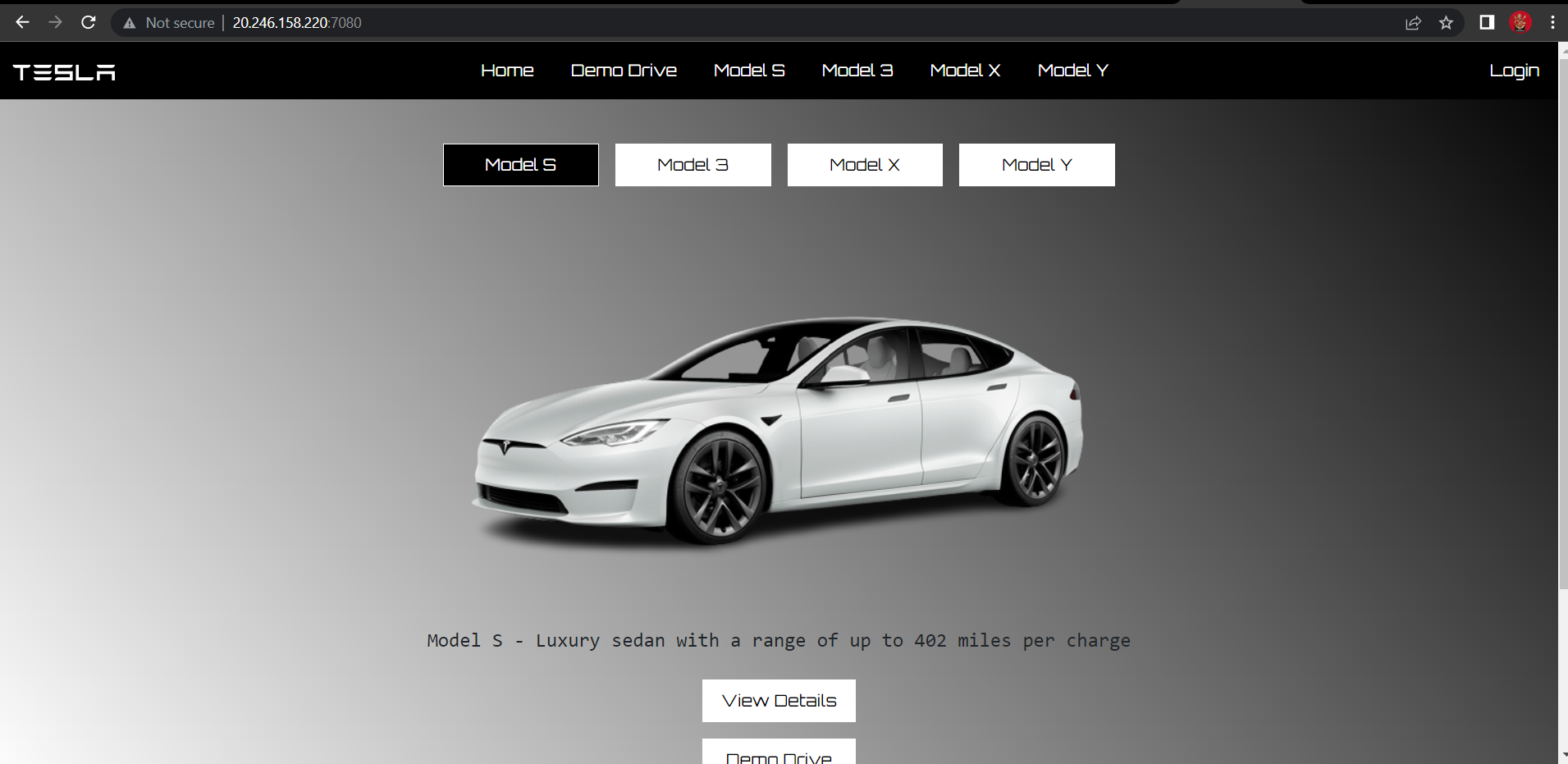
**Replacing build ID with the tag of the image **

****

****

**Description of the pod my-app**

****

****