mkdir hello-world-app

cd hello-world-app

npm init -y

npm install express

**Create styles.css**

*body {*

*font-family: 'Arial', sans-serif;*

*background-color: #f4f4f4;*

*display: flex;*

*align-items: center;*

*justify-content: center;*

*height: 100vh;*

*margin: 0;*

*}*

*h1 {*

*color: #333;*

*text-align: center;*

*padding: 20px;*

*background-color: #fff;*

*border-radius: 8px;*

*box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);*

*}*

*.container {*

*text-align: center;*

*margin-top: 50px;*

*}*

*img {*

*border-radius: 8px;*

*margin-bottom: 20px;*

*}*

**HTML UI**

*<body>*

*<div class="container">*

*<img src="hello-world-image.jpg" alt="Hello World Image" width="200">*

*<h1>Hello, World!</h1>*

*<h2 id="datetime"></h2>*

*</div>*

*<script>*

*// JavaScript code to update the datetime element*

*setInterval(() => {*

*const datetimeElement = document.getElementById('datetime');*

*const now = new Date();*

*const formattedDateTime = now.toLocaleString();*

*datetimeElement.textContent = formattedDateTime;*

*}, 1000);*

*</script>*

*</body>*

**Create App.js**

const express = require('express');

const app = express();

const port = 3000;

app.use(express.static('public'));

app.listen(port, () => {

console.log(`Server listening at http://localhost:${port}`);

});

**Run the App**

node app.js

**Create Docker File**

# Use the official Node.js image as base image

FROM node: 20.10.0

# Set the working directory inside the container

WORKDIR /usr/src/app/ helloworld

# Copy package.json and package-lock.json to the container

COPY package\*.json ./

# Install dependencies

RUN npm install

# Copy the rest of the application files to the container

COPY . .

# Expose port 3000 for the application

EXPOSE 3000

# Command to run the application

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

**Build the Docker Image**

docker build -t hello-world-opb .

**Run the Docker Container**

docker run -p 3000:3000 hello-world-opb

browser url http://localhost:3000

**Deploying Image to Azure Container Registry**

Az login

You can create a acr manually by going to portal or run the following command

az acr create --resource-group <resource\_group> --name <registry\_name> --sku Basic --admin-enabled true

**Login to registry**

az acr login --name <registry\_name>

**Tag the local Image to ACR**

docker tag <<LocalImageName>> helloworld02jan20242.azurecr.io/hello-world-app:v1

**Push the image to ACR**

docker push <registry\_name>.azurecr.io/hello-world-app:<image\_tag>

<image\_tag> in this case is v1

**(Optional)** : **Verify Image**

az acr repository list --name <registry\_name> --output table

az acr update -n <registry\_name> --admin-enabled true

**Create a Namespace in AKS**

kubectl create namespace <namespace-name>

**Authenticate with ACR**

kubectl create secret docker-registry acr-auth \

--namespace=<namespace-name> \

--docker-server=<acr-name>.azurecr.io \

--docker-username=<acr-service-principal-id> \

--docker-password=<acr-service-principal-password>

Deploy to AKS container using deployment.yaml

Create a Deployment.yaml file with below content

apiVersion: apps/v1

kind: Deployment

metadata:

name: nodejs-app

namespace: <namespace-name>

spec:

replicas: 1

selector:

matchLabels:

app: nodejs-app

template:

metadata:

labels:

app: nodejs-app

spec:

containers:

- name: nodejs-app

image: <acr-name>.azurecr.io/nodejs-app:latest

ports:

- containerPort: 3000

imagePullSecrets:

- name: acr-auth

kubectl apply -f deployment.yaml

**Expose the Deployment via a Service**

apiVersion: v1

kind: Service

metadata:

name: nodejs-app-service

namespace: <namespace-name>

spec:

type: LoadBalancer

selector:

app: nodejs-app

ports:

- protocol: TCP

port: 80

targetPort: 3000

kubectl apply -f service.yaml

**Accessing the Node.js App**

kubectl get svc nodejs-app-service --namespace=<namespace-name>