Ganpat University
॥ विद्यया समाजोत्कर्षः ॥
Institute of Computer Technology

Name: Tushar Panchal

En.No: 21162101014

Sub: MICROSEVICES

Branch: CBA

Batch:51

----------PRACTICAL 14----------

❖ Question (TASK) :

To understand the working of a Microservice that builds using Node JS and Docker mechanisms.

You are asked to create a website that offers users information about sharks. The application will have a main entry point, app.js, and a views directory that will include the project's static assets. The landing page, index.html, will offer users some preliminary information and a link to a page with more detailed shark information, sharks.html. In the views directory, create both the landing page and sharks.html.

Tasks:

1. Create a directory for your project in your non-root user's home directory.

2. Create a package.json file with your project's dependencies and other identifying information and install them.

3. Open app.js in the main project directory to define the project's routes.

4. Add some static content to the application, and create the custom CSS style sheet that you've linked to in index.html and sharks.html by first creating a css folder in the views directory.

5. Start the application.

6. Create the Dockerfile and the .dockerignore file.

7. Build the image and run the container that has this image.

8. Push the image to Docker Hub.

🔗 **Github Link :**
https://github.com/Tushar007079/MICROSERVICES_PRACTICALS/tree/main/14

📌1. Create a directory for your project in your non-root user's home directory :

» create a basic Node.js web application using a package like Express.js :

```
mkdir ~/14
cd ~/14
```

📌2. Create a package.json file and install dependencies :

» You can use **npm init** to create a **package.json** file, and then install necessary dependencies :

```
npm init
npm install express ejs
```

» In the command **npm install express ejs**, **ejs** is a package that stands for "Embedded JavaScript." EJS is a popular template engine for JavaScript that allows you to embed JavaScript code within your HTML templates to create dynamic web pages.

📌3. Define project's routes in app.js :

✓ **app.js :-**

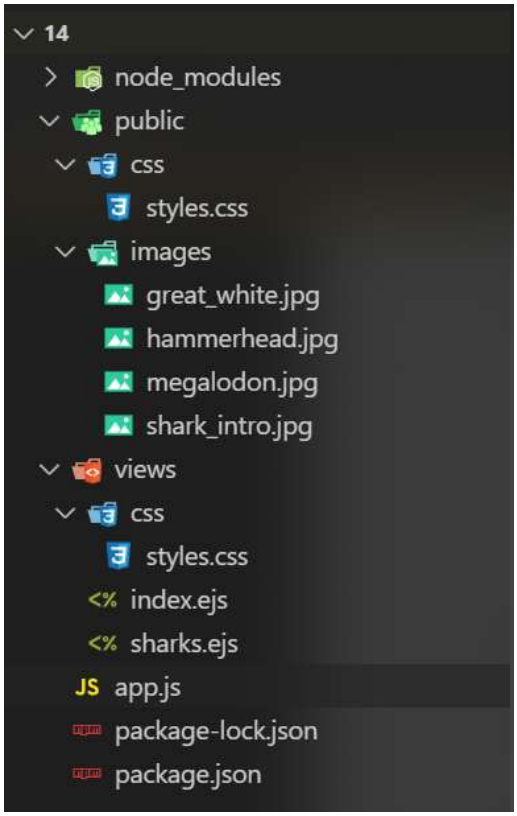
```
const express = require('express');
const app = express();
const path = require('path');

app.set('view engine', 'ejs');
```

```
app.set('views', path.join(__dirname, 'views'));

// Serve static assets from the public directory
app.use(express.static(path.join(__dirname, 'public')));

app.set('views', path.join(__dirname, 'views'));
app.get('/', (req, res) => {
  res.render('index.ejs'); // Use '.ejs' extension
});

app.get('/sharks', (req, res) => {
  res.render('sharks.ejs'); // Use '.ejs' extension
});

app.listen(3000, () => {
  console.log('Server is running on  http://localhost:3000');
});
```

4. Add static content and create CSS :

Create a **views** directory for your HTML templates and a **public** directory for your static assets (CSS, images, etc.). Create a CSS file in a **css** directory in the **views** folder and link it in your HTML files.

File Structure like these :



5. Start the application :

» You can start your application using **node app.js** :

`node app.js`

Your Node.js application should now be accessible at **http://localhost:3000**.

6. Create a Dockerfile and .dockerignore file :

» **Dockerfile :**

```
# Use an official Node.js runtime as a parent image
FROM node:14

# Set the working directory to /app
WORKDIR /app

# Copy package.json and package-lock.json to the working directory
COPY package*.json ./

# Install app dependencies
RUN npm install

# Copy the rest of the application code to the working directory
COPY . .

# Expose the port the app will run on
EXPOSE 3000

# Define the command to run your app
CMD [ "node", "app.js" ]
```

» **.dockerignore :**

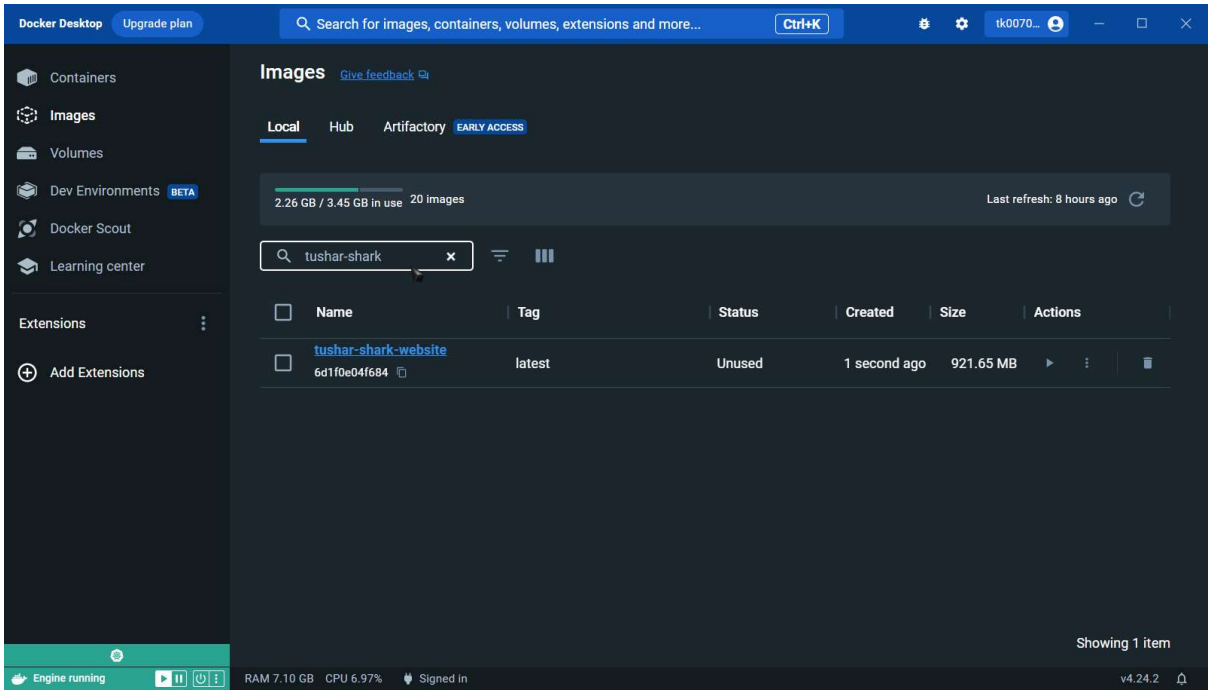
```
node_modules
npm-debug.log
```

7. Build the Docker image and run a container :

» Build the Docker image using the **docker build** command :

`docker build -t tushar-shark-website .`

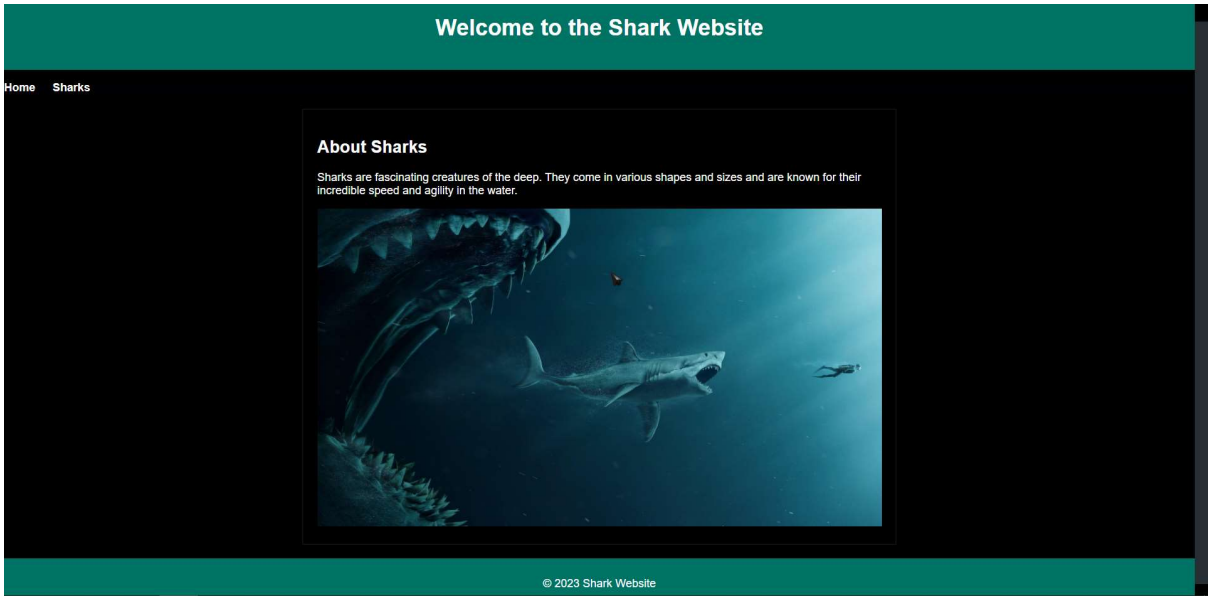
```
pwsh 14 65ms                                    20.8.1 minikube :: default 79% 20,19:51
>> docker build -t tushar-shark-website .
[+] Building 17.4s (11/11) FINISHED                                          docker:default
 => [internal] load .dockerignore                                                      0.3s
 => => transferring context: 67B                                                       0.0s
 => [internal] load build definition from Dockerfile                                   0.3s
 => => transferring dockerfile: 495B                                                   0.1s
 => [internal] load metadata for docker.io/library/node:14                             2.1s
 => [auth] library/node:pull token for registry-1.docker.io                            0.0s
 => [1/5] FROM docker.io/library/node:14@sha256:a158d3b9b4e3fa813fa6c8c590b8f0a860e015ad4e59bbce5744d2f6fd8461  0.0s
 => [internal] load build context                                                      0.5s
 => => transferring context: 3.03MB                                                    0.3s
 => CACHED [2/5] WORKDIR /app                                                          0.0s
 => [3/5] COPY package*.json ./                                                        0.2s
 => [4/5] RUN npm install                                                             13.1s
 => [5/5] COPY . .                                                                     0.2s
 => exporting to image                                                                 0.5s
 => => exporting layers                                                                0.5s
 => => writing image sha256:6d1f0e04f68460ebd4ec554d34a8c4b4d2e071ff0bfdfda10294f57fc8671d17  0.0s
 => => naming to docker.io/library/tushar-shark-website                                0.0s

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview
```

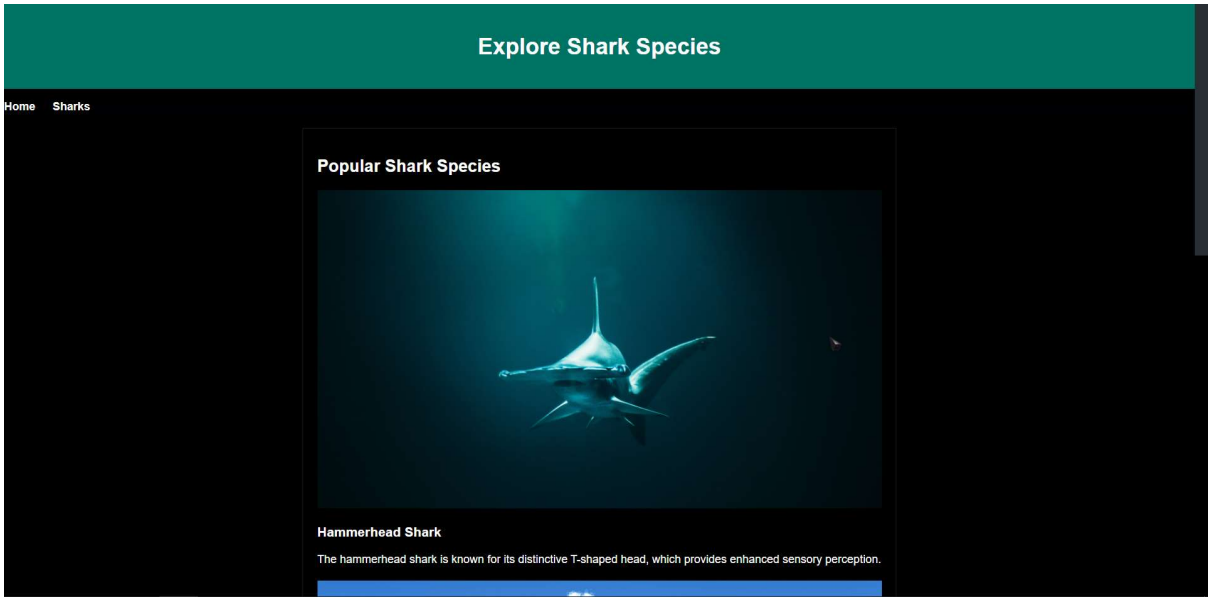


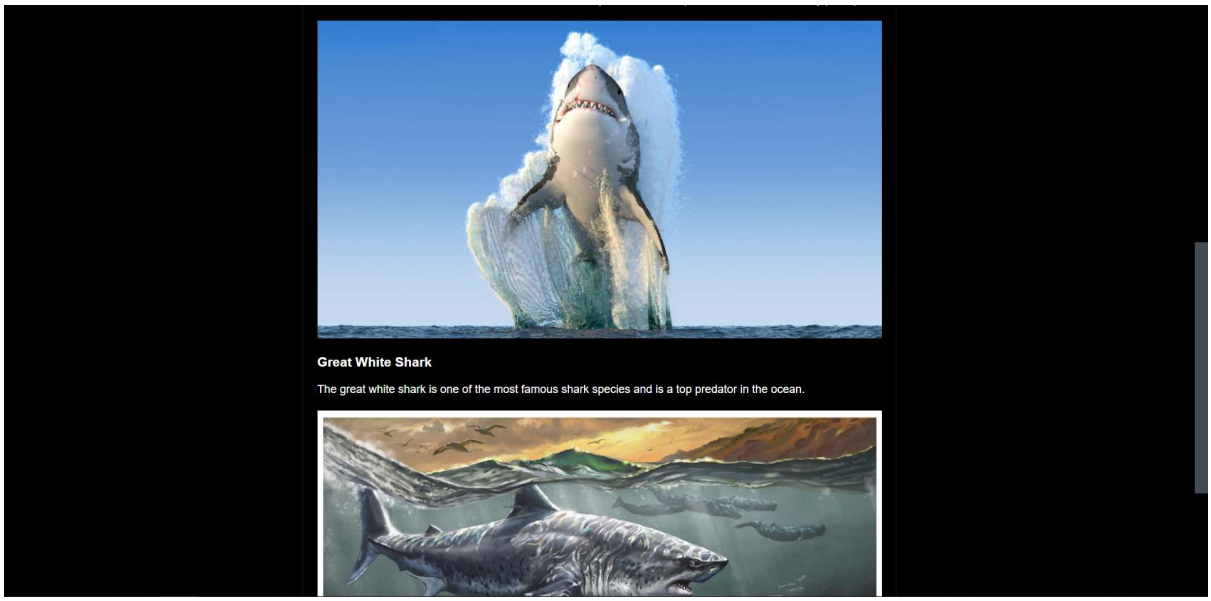
Run a Docker container from the image :

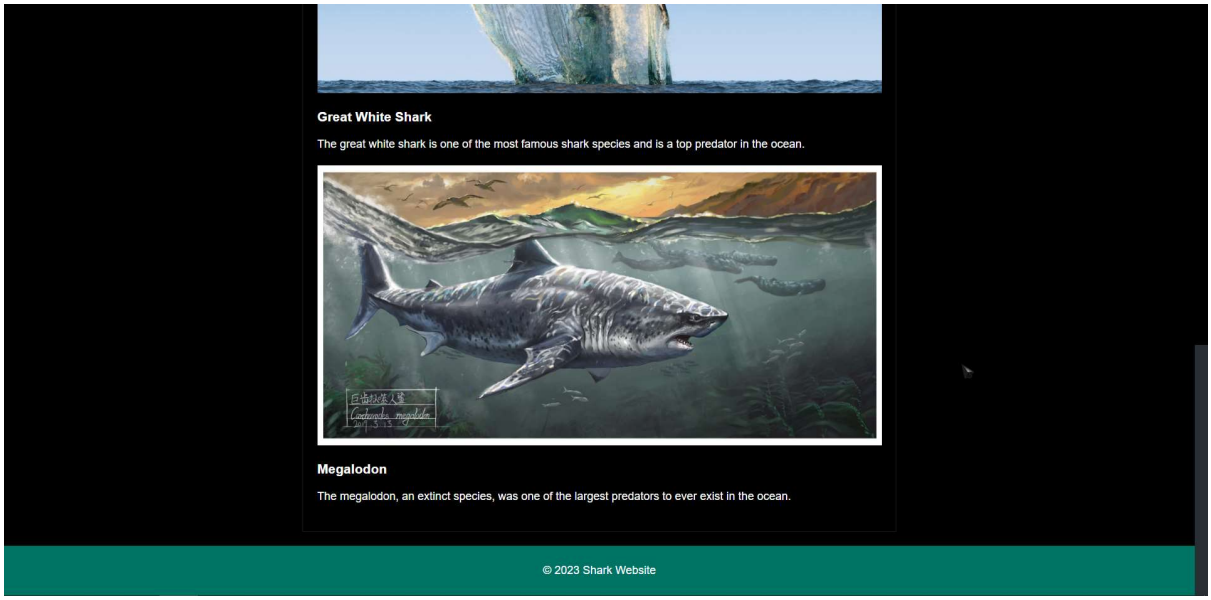
```
docker run -p 3000:3000 tushar-shark-website
```

Your application should now be accessible inside a Docker container at http://localhost:3000.


Welcome to the Shark Website
Home
Sharks
About Sharks
Sharks are fascinating creatures of the deep. They come in various shapes and sizes and are known for their incredible speed and agility in the water.
© 2023 Shark Website


Explore Shark Species
Home
Sharks
Popular Shark Species
Hammerhead Shark
The hammerhead shark is known for its distinctive T-shaped head, which provides enhanced sensory perception.


Great White Shark
The great white shark is one of the most famous shark species and is a top predator in the ocean.



9. Push the image to Docker Hub :

To push your Docker image to Docker Hub, you need to tag it appropriately and authenticate with your Docker Hub account. Replace **your-docker-hub-username** and **node-docker-app** with your Docker Hub username and image name :

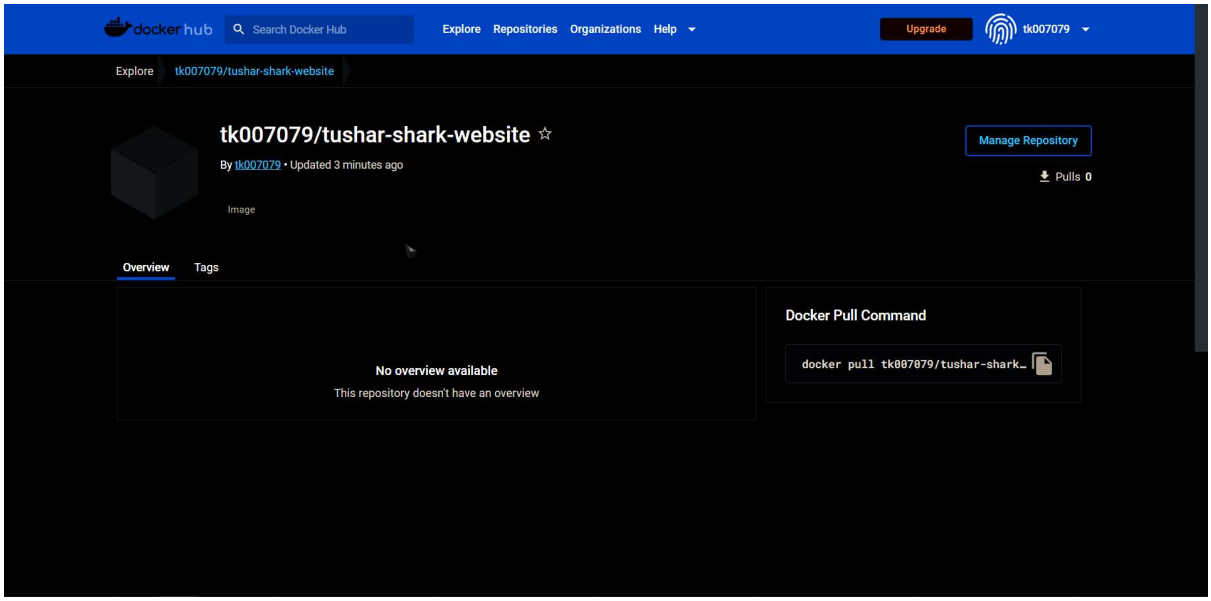
```
# Log in to Docker Hub
docker login

# Tag the image
docker tag shark-website yourusername/shark-website

# Push the image to Docker Hub
docker push yourusername/shark-website
```

This is my command to push my docker image :

```
# Log in to Docker Hub
docker login

# Tag the image
docker tag tushar-shark-website tk007079/tushar-shark-website

# Push the image to Docker Hub
docker push tk007079/tushar-shark-website
```

```
pwsh 14 1ms 20.8.1 minikube :: default 79%
>> docker login
Authenticating with existing credentials...
Login Succeeded
pwsh 14 3s 58ms 20.8.1 minikube :: default 79%
>> docker tag tushar-shark-website tk007079/tushar-shark-website
pwsh 14 147ms 20.8.1 minikube :: default 79%
>> docker push tk007079/tushar-shark-website
Using default tag: latest
The push refers to repository [docker.io/tk007079/tushar-shark-website]
1a03faad1653: Pushed
1eff567df691: Pushed
6fce5c170c26: Pushed
7d34f051778b: Mounted from tk007079/tushar-docker-app
0d5f5a015e5d: Mounted from tk007079/tushar-docker-app
3c777d951de2: Mounted from tk007079/tushar-docker-app
f8a91dd5fc84: Mounted from tk007079/tushar-docker-app
cb81227abde5: Mounted from tk007079/tushar-docker-app
e01a454893a9: Mounted from tk007079/tushar-docker-app
c45660adde37: Mounted from tk007079/tushar-docker-app
fe0fb3ab4a0f: Mounted from tk007079/tushar-docker-app
f1186e5061f2: Mounted from tk007079/tushar-docker-app
b2dba7477754: Mounted from tk007079/tushar-docker-app
latest: digest: sha256:e349884674c2d841a3e804a14c22becde0e15c34cd6a971cff5c064687badf56 size: 3051
```

Your Docker image will now be available on Docker Hub under your username :



Now, your microservice for the shark website is containerized using Docker and can be deployed to various hosting platforms.