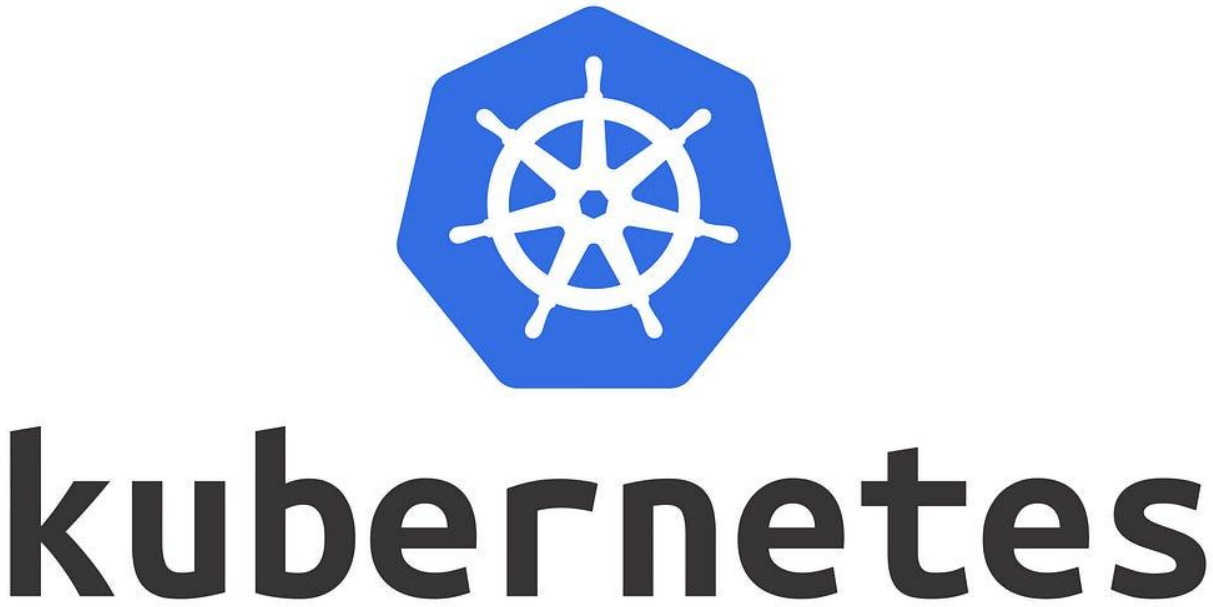
Build and Deploy MERN Stack Applications with Kubernetes

MERN stack, which stands for MongoDB, Express, React, and Node.js, is a popular web development technology stack for building web applications. Kubernetes, on the other hand, is an open-source container orchestration system that can help you manage and deploy your applications. In this blog, we will walk through the steps of deploying a MERN app on Kubernetes.

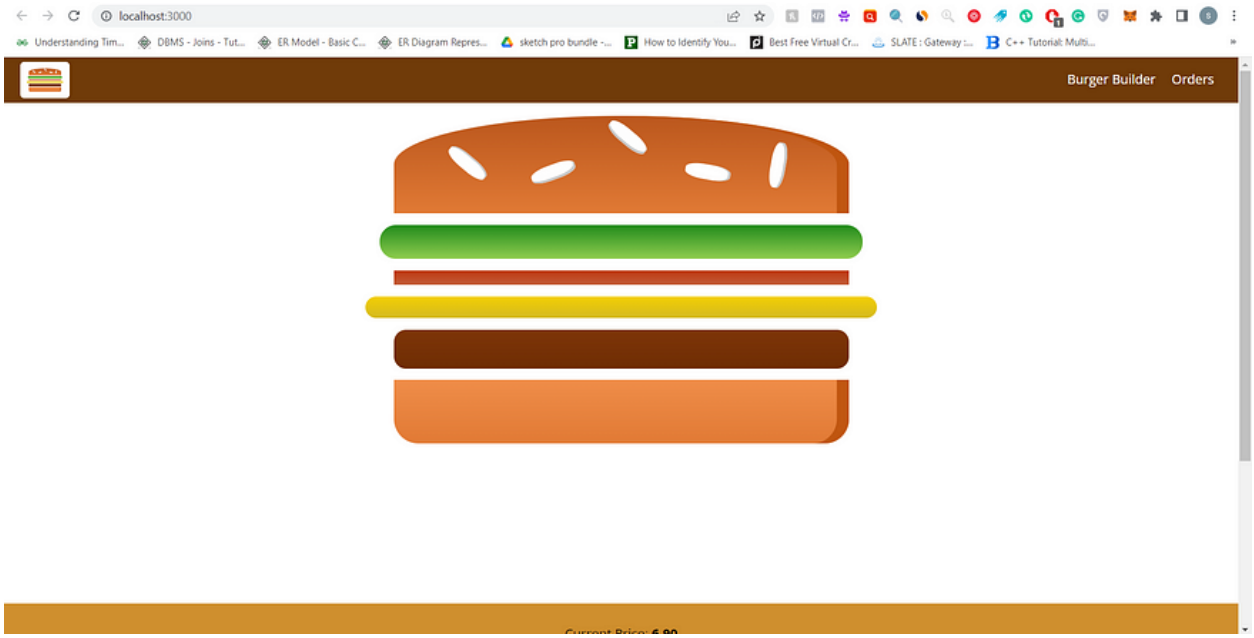


Prerequisites

- A MERN app that is ready to be deployed.
- A Kubernetes cluster that is up and running.
- A container registry account. We will be using Docker Hub for this tutorial.

If you don't have a MERN app, you can use the sample MERN application available at

https://github.com/SaadBajwa24/Mern-Podman/tree/Feature-Branch



Navigate to your MREN directory and build the application using the following commands:

cd react-burger-app
npm install
npm run build

Step 1: Dockerize Your MERN App

The first step in deploying your MERN app on Kubernetes is to create a Docker image of your application. You can create a Dockerfile in the root directory of your application with the following content:

```
# 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 dependencies
RUN npm install

# Copy the remaining application files to the working directory
```

```
COPY . .

# Build the React app
RUN npm run build

# Expose port 3000
EXPOSE 3000

# Start the server
CMD ["npm", "start"]
```

This Dockerfile sets up a Node.js environment, installs dependencies, builds the React app, exposes port 3000, and starts the server. Once you have created the Dockerfile, you can build the Docker image by running the following command:

```
docker build -t mern-stack .
```

Once the image is built, you can run it with the following command:

```
docker run -p 3000:3000 mern-stack
```

Step 2: Push Your Docker Image to a Container Registry

The next step is to push your Docker image to a container registry so that it can be accessed by Kubernetes. In this tutorial, we will use Docker Hub as our container registry. You can push your Docker image by running the following command

```
docker push mern-stack
```

Step 3: Create a Kubernetes Deployment File

The deployment file is used to define the specifications of the Kubernetes deployment. Create a deployment.yaml file in the root directory of your application with the following content:

```
apiVersion: apps/v1
kind: Deployment
```

```
metadata:
 name: mern-stack
spec:
 replicas: 3
 selector:
   matchLabels:
     app: mern-stack
 template:
   metadata:
     labels:
       app: mern-stack
   spec:
     containers:
     - name: mern-stack
       image: mern-stack:latest
       ports:
       - containerPort: 3000
```

This deployment file specifies the deployment's name, the number of replicas, the container image to be used, and the port to be exposed.

Step 4: Create a Kubernetes Service File

The service file is used to define the specifications of the Kubernetes service. Create a service.yaml file in the root directory of your application with the following content:

```
apiVersion: v1
kind: Service
metadata:
 name: mern-stack-service
spec:
 selector:
   app: mern-stack
 ports:
   - protocol: TCP
     port: 80
     targetPort: 3000
 type: LoadBalancer
```

This service file specifies the name of the service, the selector that matches the deployment labels, the port to be exposed, and the type of the service.

Step 5: Deploy Your MERN App on Kubernetes

Now that you have created the deployment and service files, you can deploy your MERN app on Kubernetes by running the following commands:

```
kubectl apply -f deployment.yaml
kubectl apply -f service.yaml
```

The first command applies the deployment file to create a deployment with the specified specifications. The second command applies the service file to create a service with the specified specifications.

Step 6: Access Your MERN App

Once your MERN app is deployed on Kubernetes, you can access it by running the following command:

```
kubectl get service mern-stack-service
```

This command will return the external IP address of the service. You can access your MERN app by visiting this IP address in your web browser.

Conclusion

Congratulations! You have successfully deployed your MERN app on Kubernetes. By following the steps outlined in this tutorial, you can easily deploy your MERN app on Kubernetes and take advantage of its container orchestration capabilities to manage and scale your application.