Cloud DevOps MERN Stack demo

1. Create SSH key for EC2 connection

```
KEYPAIR_NAME=mern-demo-kp
aws ec2 create-key-pair --key-name=${KEYPAIR_NAME} --
query=KeyMaterial --output=text > ~/.ssh/${KEYPAIR_NAME}.pem
chmod 400 ~/.ssh/${KEYPAIR_NAME}.pem
```

2. Create an EC2 running Ubuntu (preferably, to align with existing instructions). The tested Ubuntu version is 20.04 LTS, but higher should work fine as well. An instance with 2 vCPU and 2 GB RAM is recommended. The user data below can be used to optimize the build:

```
#!/bin/bash -xe

# Update OS and install required packages
apt-get update
DEBIAN_FRONTEND=noninteractive apt-get upgrade -y

# Install required packages
DEBIAN_FRONTEND=noninteractive apt-get install -y \
  apt-transport-https \
  ca-certificates \
  curl \
  software-properties-common \
  git

# Add Docker's official GPG key
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | gpg --
dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o
/etc/apt/keyrings/docker.asc
```

```
chmod a+r /etc/apt/keyrings/docker.asc

# Add Docker repository to apt registry
echo \
  "deb [arch=$(dpkg --print-architecture) signed-
by=/etc/apt/keyrings/docker.asc]
https://download.docker.com/linux/ubuntu \
  $(. /etc/os-release && echo
"${UBUNTU_CODENAME:-$VERSION_CODENAME}") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

# Update apt package list again
apt-get update -y

# Install Docker
DEBIAN_FRONTEND=noninteractive apt-get install -y \
  docker-ce \
  docker-ce-cli \
  containerd.io \
  docker-buildx-plugin \
  docker-compose-plugin

# Start and enable Docker service
systemctl start docker
systemctl enable docker

# Install Docker Compose
curl -L
"https://github.com/docker/compose/releases/latest/download/docker-
compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
chmod +x /usr/local/bin/docker-compose

# Create application directory
mkdir -p /app
chown ubuntu:ubuntu /app
```

```
# Clone Git repositories
cd /app
sudo -u ubuntu git clone https://github.com/Acads-Ops-
GreatLearning/guided_project_e_comm_client.git react-app
sudo -u ubuntu git clone https://github.com/Acads-Ops-
GreatLearning/guided_project_e_comm_backend.git express-app
```

Setup Docker for non-root users

```
sudo groupadd docker
sudo usermod -aG docker ubuntu
newgrp docker
```

3. Connect to the EC2 and build the Docker images

```
ssh -i ~/.ssh/${KEYPAIR_NAME}.pem ubuntu@${EC2_PUBLIC_IP}

cd react-app
docker build -t react-app .

cd ../express-app

# Update the .env file within the express-app folder with the MongoDB
Atlas database URI
cat <<-EOF > .env
PORT=8080
DATABASE_URI=mongodb+srv://<db_username>:
<db_password>@<db_cluster>.mongodb.net/?
retryWrites=true&w=majority&appName=<db_name>
EOF
```

```
docker build -t express-app .
```

> Downloading SSH private key for SSH key created in CloudFormation

4. Whitelist the EC2 public IP in MongoDB Atlas

5. Run the Docker containers

```
# React app
docker run --name react-app \
  -d -p 3000:3000 \
  -e REACT_APP_BACKEND_URL_ENDPOINT=http://<EC2_PUBLIC_IP>:8080 \
  react-app

# Express app (run from location of .env file)
docker run --name express-app \
  -d -p 8080:8080 \
  -e "DATABASE_URI=mongodb+srv://<db_username>:
<db_password>@<db_cluster>.mongodb.net/?retryWrites=true&w=majority"
\
express-app
```

7. Access the apps via browser

```
React Frontend: http://<EC2_PUBLIC_IP>:3000
Express backend: http://<EC2_PUBLIC_IP>:8080
```

What are some possible improvements?

- Container hosting using ECS Fargate or EKS Fargate
- DocumentDB with MongoDB compatibility? Depends on the version (DocumentDB compatible up to MongoDB 4.0, but MongoDB at 8disolays

#great-learning #demo