Experiment-5.3

Deploy Full Stack App on AWS with Load Balancing

Code-

1. Backend: Node.js (index.js)

This file includes the API routes and a crucial '/api/health' endpoint for the ALB health check.

```
const express = require('express');
const cors = require('cors'); // You may not need CORS with the ALB setup
const app = express();
const PORT = 5000;

// ALB Health Check Endpoint
app.get('/api/health', (req, res) => {
  res.status(200).send({ status: 'healthy' });
});

// Your actual API route
app.get('/api/data', (req, res) => {
  res.json({ message: 'Hello from the backend!', instance: process.env.HOSTNAME
    });
});

app.listen(PORT, () => {
  console.log(`Backend server running on port ${PORT}`);
});
```

Listing 1: backend/index.js (Port 5000)

2. Frontend: React (App.js)

The frontend makes a relative API call to '/api/data'. The ALB routes this request to the backend.

```
import React, { useState, useEffect } from 'react';

function App() {
  const [message, setMessage] = useState('Loading...');

  useEffect(() => {
    // Note: No 'http://localhost:5000'. Just the relative path!
    // The browser will request 'http://<your-alb-domain>/api/data'
    fetch('/api/data')
      .then(res => res.json())
      .then(data => setMessage(`[${data.message}] from instance: ${data.instance
    }`))
      .catch(err => setMessage('Error fetching data'));
  }, []);

  return (
    <div className="App">
      <h1>Full Stack AWS Deployment</h1>
      <p>Data from backend: <strong>{message}</strong></p>
    </div>
  );
}
export default App;
```

Listing 2: frontend/src/App.js

3. EC2 User Data: Backend Server

This script auto-configures the backend instances on boot. It installs Node.js, Git, and PM2, then clones the repo and starts the app.

```
#!/bin/bash
sudo yum update -y
# Install Node.js
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.7/install.sh | bash
export NVM_DIR="$HOME/.nvm"
[ -s "$NVM_DIR/nvm.sh" ] && \. "$NVM_DIR/nvm.sh"
nvm install 18
# Install Git and PM2 (process manager)
sudo yum install git -y
npm install -g pm2
# Clone and run the app
git clone <YOUR_REPO_URL> /home/ec2-user/app
cd /home/ec2-user/app/backend
npm install
pm2 start index.js --name backend-api
```

Listing 3: User Data for 'backend-server' Instances

4. EC2 User Data: Frontend Server

This script auto-configures the frontend instance. It installs Nginx, Node.js (to build the app), and Git, then serves the built React app via Nginx.

```
#!/bin/bash
sudo yum update -y
# Install Nginx
sudo amazon-linux-extras install nginx1 -y
# Install Node.js (to build the React app)
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.7/install.sh | bash
export NVM_DIR="$HOME/.nvm"
[ -s "$NVM_DIR/nvm.sh" ] && \. "$NVM_DIR/nvm.sh"
nvm install 18
# Install Git
sudo yum install git -y
# Clone, build, and deploy the React app
git clone <YOUR_REPO_URL> /home/ec2-user/app
cd /home/ec2-user/app/frontend
npm install
npm run build
# Copy the built app to Nginx's web directory
sudo rm -rf /usr/share/nginx/html/*
sudo cp -r build/* /usr/share/nginx/html/
# Start Nginx
sudo systemctl start nginx
sudo systemctl enable nginx
```

Listing 4: User Data for 'frontend-server' Instance