PEAN Stack Deployment on AWS EC2 (Step-by-Step Guide)

This guide provides a step-by-step process for deploying a PEAN (PostgreSQL, Express, Angular, Node.js) stack
application on an AWS EC2 instance running Ubuntu. Each step is explained in simple terms, making it accessible
for those new to deployment. Follow these instructions to set up and deploy your application.

Step 1: Set Up Your EC2 Instance

1. Log in to AWS and go to the EC2 Dashboard.
2. Launch a new instance:
   - Select Ubuntu Server 20.04 LTS as the operating system.
   - Choose an instance type, like t2.micro.
   - Configure a security group with rules to open HTTP (port 80), HTTPS (port 443), and a custom port (5000) for Node.js if needed.
3. Launch the instance and wait for it to initialize.

Step 2: Connect to EC2 and Install Essential Software

1. Connect to the EC2 instance using the EC2 Terminal in the AWS Console.
2. Update the system:
   ```
   sudo apt update && sudo apt upgrade -y
   ```
3. Install Git, Node.js, npm, and PostgreSQL:

```

sudo apt install git nodejs npm postgresql postgresql-contrib -y

```

Step 3: Set Up PostgreSQL Database

1. Start PostgreSQL and enable it to start automatically on boot:

```

sudo systemctl start postgresql

sudo systemctl enable postgresql

```

2. Create a database and user:

```

sudo -i -u postgres

psql

CREATE DATABASE pean_app_db;

CREATE USER pean_user WITH PASSWORD 'yourpassword';

GRANT ALL PRIVILEGES ON DATABASE pean_app_db TO pean_user;

\q

exit

```

Step 4: Deploy Node.js Backend

1. Clone the repository and navigate to the backend directory:

```
```

```
git clone <your_repository_url>
cd <your_repository_folder>/backend
```

2. Install backend dependencies:

```
npm install
```

3. Create and edit a .env file with environment variables:

```
touch .env
nano .env
```

Example .env:

```
DATABASE_URL=postgresql://pean_user:yourpassword@localhost:5432/pean_app_db
PORT=5000
```

4. Start the backend server:

```
npm start
```

Step 5: Deploy Angular Frontend

1. Navigate to the frontend directory:

```
cd ../frontend
```

```

2. Install frontend dependencies:

```

npm install

```

3. Configure API endpoint in environment.ts in src/environments:

```typescript

export const environment = {

  production: false,

  apiUrl: 'http://localhost:5000'

};

```

4. Build for production:

```

ng build --prod

```

Step 6: Set Up NGINX to Serve Angular and Backend

1. Install NGINX:

```

sudo apt install nginx -y

```

2. Configure NGINX for Angular and API:

```

sudo nano /etc/nginx/sites-available/pean_app

```
```

Sample configuration:

```nginx
server {
  listen 80;
  server_name <your_domain_or_IP>;
  location / {
    root /path/to/angular/dist;
    try_files $uri /index.html;
  }
  location /api/ {
    proxy_pass http://localhost:5000/;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection 'upgrade';
    proxy_set_header Host $host;
    proxy_cache_bypass $http_upgrade;
  }
}
```

3. Enable and restart NGINX:

```
sudo ln -s /etc/nginx/sites-available/pean_app /etc/nginx/sites-enabled/
sudo systemctl restart nginx
```

Step 7: Automate Node Server Startup (Optional)

1. Install PM2 for managing the Node.js process:

   ```

   sudo npm install -g pm2

   ```

2. Start backend with PM2:

   ```

   pm2 start server.js

   pm2 save

   pm2 startup

   ```

Step 8: Test the Application

- Open a web browser and go to http://<EC2_Public_IP> to access the frontend.
- Check http://<EC2_Public_IP>/api to verify backend API responses.