Mern app deployment document on vps server

Step 1: Prerequisites

- VPS with a specific operating system (e.g., Ubuntu) installed.

Step 2: Download PUTTY

- PuTTY is a free and open-source terminal emulator that supports various network protocols, including SSH, Telnet, and others. Here are the steps to download and install PuTTY on a Windows system

- Go to the official PuTTY website: https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html
  You will find a list of download options. For most users, the 32-bit version is sufficient, but choose the appropriate version based on your system architecture.

- Download PuTTY Installer: On the download page, you'll see a section with different executables. Look for the installer version, which is usually named something like putty-<version>-installer.exe. Click on the link to download the installer.

- Run the Installer: Once the installer is downloaded, locate the file and double-click on it to run the installer.

Step3: Connect to VPS using PuTTY

- You will need Host address and password to connect to your server:

  puTTY configuration:
  Host Name (or IP address): [Your VPS IP]
  Port: [SSH Port, usually 22]

Step4: Update System and Install Dependencies:

- update the system packages

```
sudo apt update
sudo apt upgrade
```

- Install necessary dependencies such as Node.js, npm, and MongoDB.

Step5: Clone the MERN App Repository:

- Now clone the particular repository you want to work with in our case here is the github account link for Ecera System github account: https://github.com/orgs/Ecera-System/repositories

- Command to clone the repository:
  ```
  git clone [repository_url]
  ```

Step 6: Install Node.js Packages

- You have to run the following command inside your client directory
  ```
  npm install
  ```
- You have to setup environment variables needed for your MERN app (e.g., database connection strings, API keys) In dotenv file inside your client directory.

Step 7: Build and Start the Application:

```
npm run build
npm start
```

Step 8:Configure Reverse Proxy

- **Install Nginx:**
  ```
  sudo apt update
  sudo apt install nginx
  ```

- **Configure Nginx:**

  Navigate to the Nginx configuration directory and create a new configuration file for your application. For example:

  ```
  sudo nano /etc/nginx/sites-available/your_app
  ```

- **Configure Reverse Proxy:**
  Edit the configuration file to set up the reverse proxy. Here's a basic example assuming your Node.js app is running on localhost:3000:

```
server {
    listen 80;
    server_name your_domain.com www.your_domain.com;

    location / {
        proxy_pass http://localhost:3000;
        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;
    }

    # Additional configurations can be added here, e.g., SSL settings.
}
```

- **Enable the Site Configuration:**
  Create a symbolic link to enable the site configuration:

```
sudo ln -s /etc/nginx/sites-available/your_app /etc/nginx/sites-enabled
```

- **Test Nginx Configuration:**
  Ensure there are no syntax errors in your Nginx configuration:

```
sudo nginx -t
```

  If the test is successful, restart Nginx:

```
sudo systemctl restart nginx
```

Step 9: Set Up and Start Your Backend

```
npm install -g pm2
pm2 start your-server.js
```