MINARMARKET DEPLOYEMENT

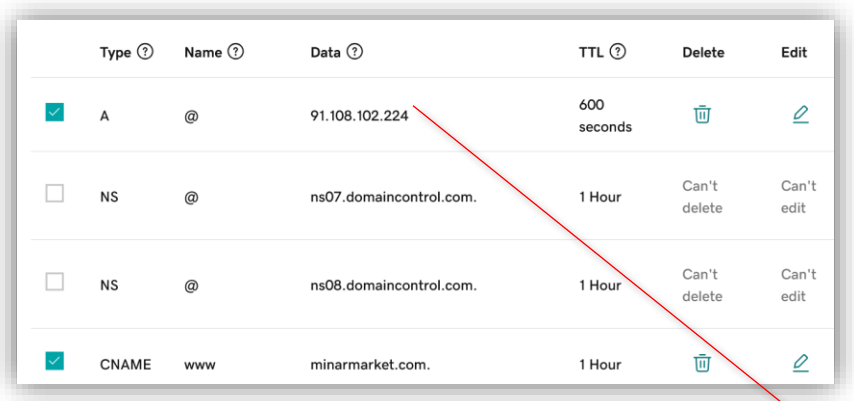
Domain: GoDaddy
Deployment Platform: Hostinger VPS (KVM-2)

Domain Configuration:

The domain is pointed towards VPS using the DNS record. Here is the tutorial:
https://www.youtube.com/watch?v=NM2NkMfQLWI&t=108s&ab_channel=HostingerAcademy

GoDaddy:

Login -> DNS -> Create/Edit A record -> Record type: A -> Name: @ -> Paste the IP address of VPS -> Do not change the TTL -> Add record



Type	Name	Data	TTL	Delete	Edit
A	@	91.108.102.224	600 seconds		
NS	@	ns07.domaincontrol.com.	1 Hour	Can't delete	Can't edit
NS	@	ns08.domaincontrol.com.	1 Hour	Can't delete	Can't edit
CNAME	www	minarmarket.com.	1 Hour		

Figure 1: Minar Market DNS page -GoDaddy

Hostinger VPS Panel:

Login -> VPS -> VPS information -> Copy IP address

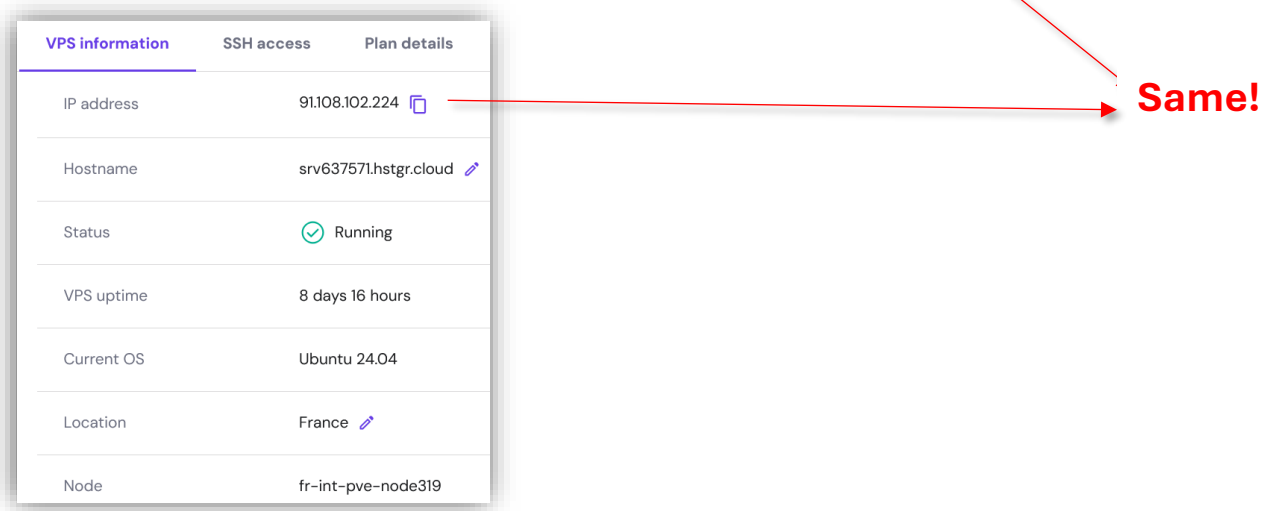


Figure 2: VPS Information - Hostinger

Deploy Full Stack MERN Project on Hostinger VPS

Here is the complete tutorial link:

https://github.com/GreatStackDev/notes/blob/main/Deploy_MERN_on_VPS.md#1-preparing-the-vps-environment

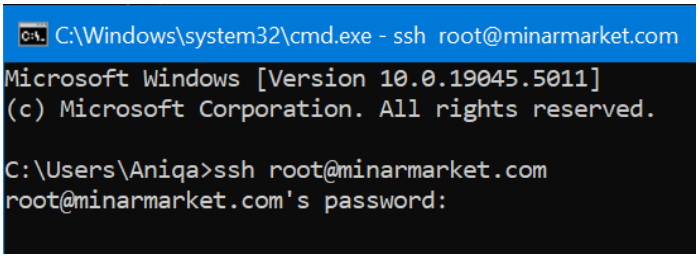
The deployment comprises of the following steps:

1. **Preparing the VPS Environment:**

   Login to Hostinger -> VPS terminal -> Copy IP address.

   Since we have configured the domain i-e the IP address is pointing towards the domain so we can use either.

   - ssh root@91.108.102.224
   - Enter the root password



2. **Setting Up the MongoDB Database (Optional)**
   If you want to setup MongoDB on VPS Follow this Guide: click here

3. **Deploying the Express and Node.js Backend**
   Clone the git repository:
   - git clone https://github.com/umerjamil28/MinarMarket_Development.git
   - cd your-repo/backend

4. **Deploying the React Frontends**
   Creating Build of React Applications. Follow the tutorial

5. **Configuring Nginx as a Reverse Proxy**
   We are using **Nginx** as compared to Apache because it provides reverse proxy. The purpose of using a reverse proxy like Nginx in a MERN stack application is to cache frequently accessed data and reduce database load, thereby improving the application's

performance and responsiveness. It provides an additional level of abstraction and control to ensure the smooth flow of network traffic between clients and servers. There are 2 methods to do this:

- By creating a symbolic link as mentioned in the tutorial. It didn't work in our case. The reason is unknown
- So, we setup reverse proxy by using rsync. Here is the tutorial: rsync-nginx

6. **Setting Up SSL Certificates**

Final Notes:

The deployment setup that we have created until now is permanent setup. Each time we have to run 5 6 commands to deploy.

For Backend:

- cd backend
- npm i
- service restart

For frontend:

- cd frontend
- npm i
- Check sync:
  - If sync -> proceed
  - If not -> run script ./sync.sh
- npm run build
- restart nginx