

[NextJS / Deployment /](#)

User guide for deploying the Next.js app in production using PM2 and Nginx?

Published on 10-17-2022



To make our Nextjs webapp accessible publicly on the internet, we need to push the code to a Linux server. But the Node.js server can't handle the client request. Hence we use Nginx to manage the client request and pm2 for running Nextjs in the background in a production environment.

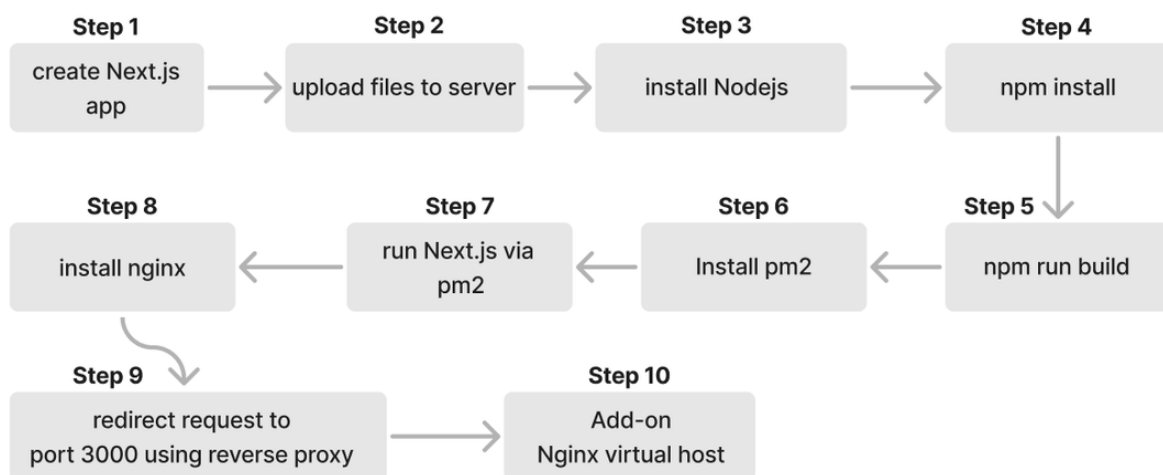
Introduction :

While developing your web app or website local on your machine, it is very easy to see the result appear on the browser, just by typing this command `npm run dev`. Under the hood runs a local server to provide features like hot-code reloading, error reporting and more.

But the real problem occurs when you think to deploy your application on a cloud server (ubuntu) for production purposes. On the Next.js official [documentation](#), they have done a good job of explaining the commands and use cases. They left out some important things like which tool should I use to manage my application running in the background, **which web server should I choose, how to configure a reverse proxy, Virtual host using the domain name** and much more.

So don't worry in this tutorial we are going to see step by step process, of how to deploy the Next.js application or website on a remote server.

Flow Diagram :



Prerequisites

- Github account ([Learn about Github?](#))
- Ubuntu server ([How to create an ubuntu server in AWS?](#))
- Next.js web application or website.

Steps :

Step 1: Create a Next.js application or website

In Next.js you have to follow certain rules, such as **using the Image Tag provided by the Next.js**. Likewise, there are some rules to follow while developing your Next.js web application or website.

Step 2: Upload files to a remote server

If you don't know about git or Github [read this article](#)

On the local machine, Run these commands:

```
1  git add .
2  git commit -m "npm run build done"
3  git push
```

copy

On the remote server

- If you already cloned the repo, execute the below command

```
1  git pull
```

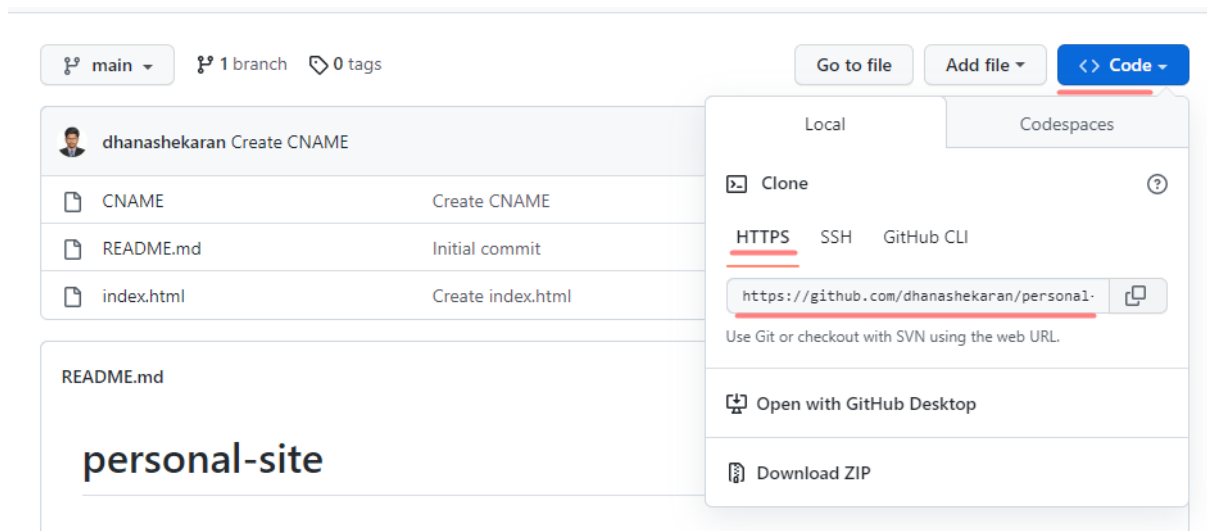
copy

- If you have not cloned the repo, execute the below command

```
1  git clone github-repo-url
```

copy

How to get Github repo URL for cloning. As given in the screenshot below:



Step 3: Install Node.Js

Always use the LTS version for long-term support from the developers. The Node.js version used in this tutorial is **16 LTS**. Execute the below commands:

```
1  curl -fsSL https://deb.nodesource.com/setup_lts.x | sudo -E bash - copy
2  sudo apt-get install -y nodejs
```

Step 4: npm install

Navigate to the project directory/folder and execute the command `npm install`, to install dependencies required to run the Next.js web application or website.

```
1  cd /project-root-folder copy
2  npm install
```

Step 5: npm run build

To make the application ready for the production stage, we need to bundle the javascript files. This process is done by Next.js by executing the following command:

Note*: your server must meet the minimum requirement to process the files.

RAM: 1GB and above.

Processor core: 1 and above.

```
1  cd /project-root-folder copy
2  npm run build
```

The output screenshot is given for your reference.

Page	Size	First Load JS
o /	4.91 kB	134 kB
o /_app	0 B	95.8 kB
o /404	193 B	96 kB
o /aboutme	3.48 kB	99.3 kB
o /contact	200 B	129 kB
o /solutions	291 B	96.1 kB
o /solutions/automation	1.88 kB	97.7 kB
o /solutions/aws	2.22 kB	98 kB
o /solutions/django	2.5 kB	98.3 kB
o /solutions/elk	3.05 kB	98.8 kB
o /solutions/search	2.42 kB	98.2 kB
o /solutions/serverless	2.04 kB	97.8 kB
o /solutions/webapp	2.34 kB	98.1 kB
+ First Load JS shared by all	95.8 kB	
chunks/framework-4556c45dd113b893.js	45.2 kB	
chunks/main-35f06ba0f7da9325.js	28.7 kB	
chunks/pages/_app-c69db95b021b5798.js	21 kB	
chunks/webpack-24780b5468e42e63.js	881 B	
css/e67ceffb57cb9b5a.css	9.07 kB	
o (Static) automatically rendered as static HTML (uses no initial props)		

Step 6: Install PM2

We need a tool to manage the Next.js process running in the background after the terminal is closed. so PM2 is going to do the work.

- Install PM2 using the below command:

```
1 sudo npm install pm2 -g copy
```

- To verify PM2 installation, execute the following command `pm2`, you will get a response similar to the screenshot given below.

```
ubuntu@ip-172-31-21-153:~$ pm2
usage: pm2 [options] <command>

pm2 -h, --help          all available commands and options
pm2 examples            display pm2 usage examples
pm2 <command> -h        help on a specific command

Access pm2 files in ~/.pm2
```

Step 7: Run Next.js via PM2 in the background

We need to run, stop and restart the Next.js application even after the terminal is closed. This can be achieved using the PM2 tool.

- Execute the below code to run Next.js with PM2: (The explanation for the below command is given on the image.)

```
1 pm2 start npm --name app1 -- run start -- -p 3000
```

copied!

- Check the **status** of **app1**.

```
1 pm2 list app1
```

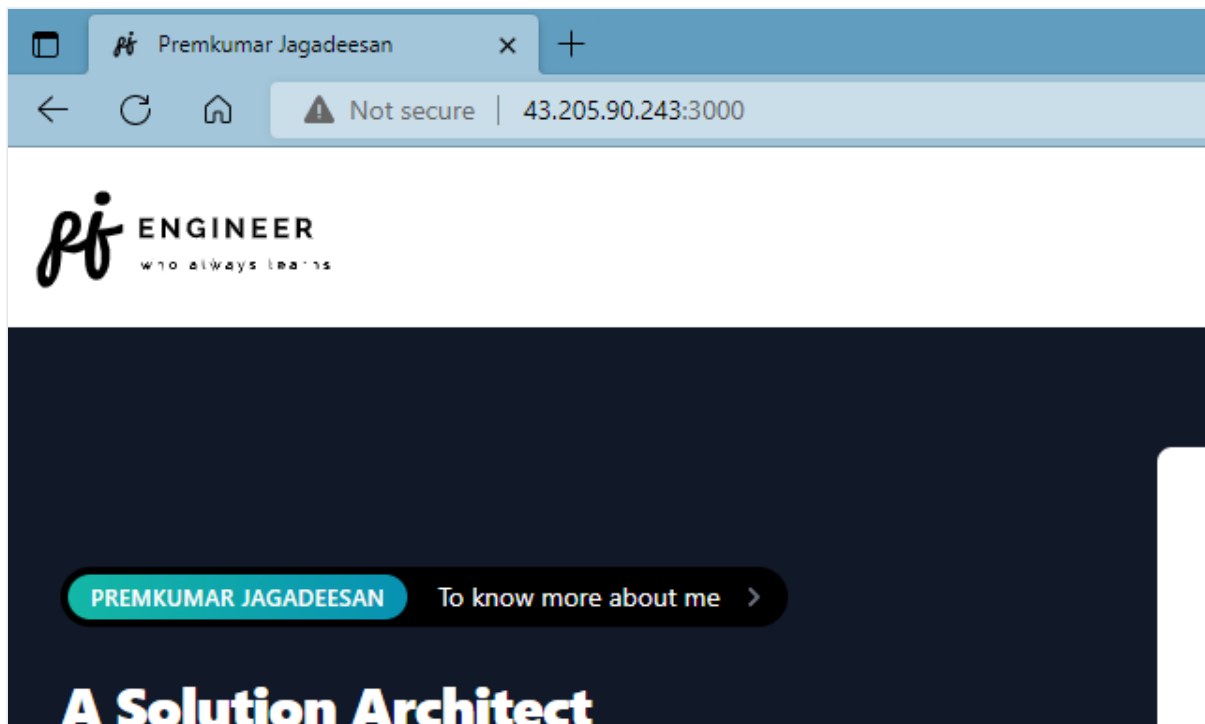
copy

Episyche

id	name	namespace	version	mode	pid	uptime	o	status	cpu	mem	user	watching
0	app1	default	N/A	fork	4581	92s	0	online	0%	57.2mb	ubuntu	disabled

ubuntu@ip-172-31-21-153:~/prem jagadeesan\$

- To check whether the Next.js application is actually working or not, Enter the public IP of the server along with a port number (3000) in the browser. **eg: 0.0.0.0:3000**



- To **stop** the app1 process

```
1 pm2 stop app1
```

copy

```
ubuntu@ip-172-31-21-153:~/prem jagadeesan$ pm2 stop app1
[PM2] Applying action stopProcessId on app [app1](ids: [ 0 ])
[PM2] [app1](0) ✓
```

id	name	namespace	version	mode	pid	uptime	u	status	cpu	mem	user	watching
0	app1	default	N/A	fork	0	0	0	stopped	0%	0b	ubuntu	disabled

```
ubuntu@ip-172-31-21-153:~/prem jagadeesan$
```

- To **start** the app1 process

```
1 pm2 start app1
```

copy

- To **restart** the app1 process

```
1 pm2 restart app1
```

copy

- To **Delete** the app1 process

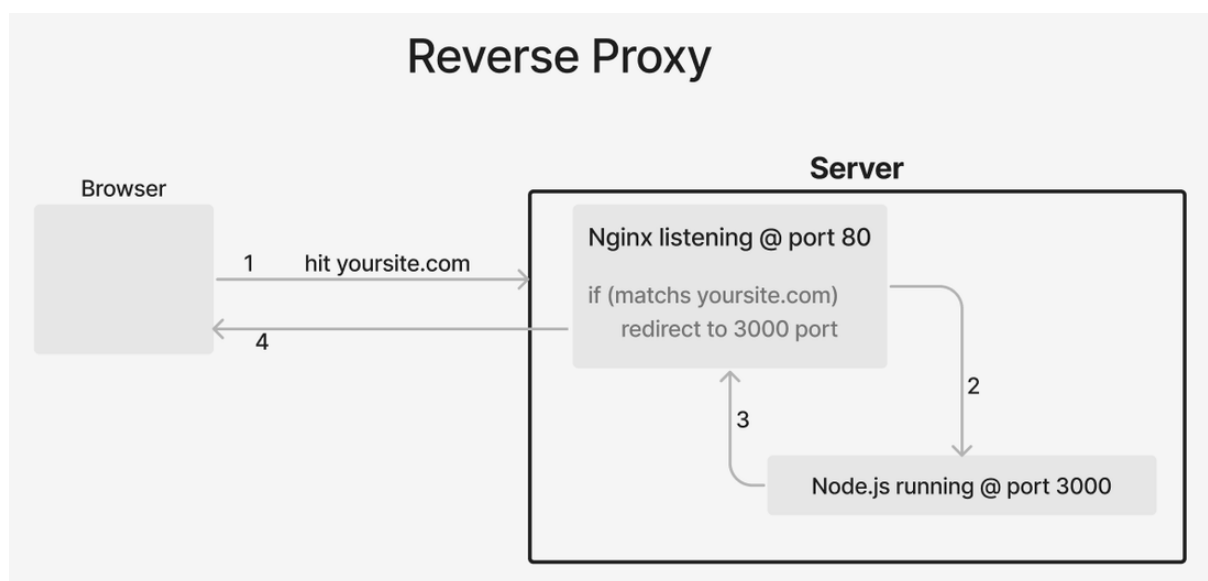
```
1 pm2 delete app1
```

copy

Step 8: Install the Nginx web server

In order to directly access the application without mentioning the **port number** on the browser. We need to set up a reverse proxy with a web server. The web server we're going to use is Nginx to read more about Nginx [click here](#).

The **reverse proxy concept** is explained below in the image:



Note*: yousite.com will be replaced with the actual domain name.

On the **ubuntu server**, execute the following command:

- Update apt packages information

```
1 sudo apt update copy
```

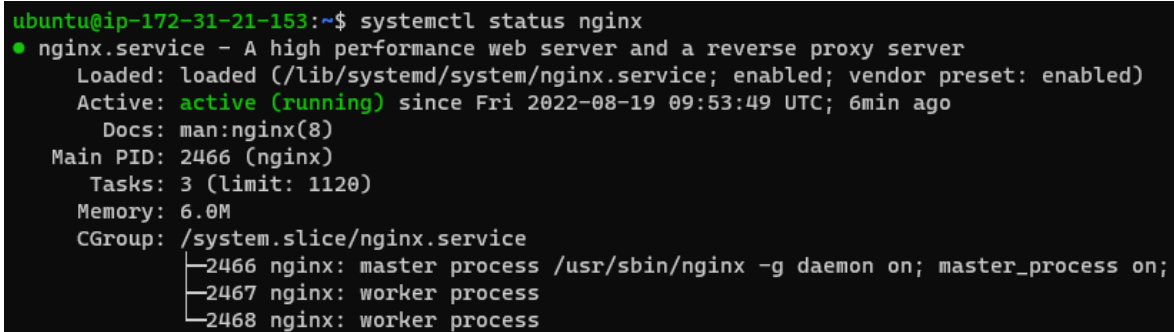
- Install Nginx

```
1 sudo apt install nginx -y copy
```

- Check the status of the Nginx web server

```
1 systemctl status nginx copy
```

Example output screenshot :



```
ubuntu@ip-172-31-21-153:~$ systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2022-08-19 09:53:49 UTC; 6min ago
     Docs: man:nginx(8)
  Main PID: 2466 (nginx)
    Tasks: 3 (limit: 1120)
   Memory: 6.0M
   CGroup: /system.slice/nginx.service
           └─2466 nginx: master process /usr/sbin/nginx -g daemon on; master_process on;
              └─2467 nginx: worker process
                 └─2468 nginx: worker process
```

Step 9: Redirect Nginx request to Next.js

When the user hits the domain or public address of your server, they will be presented with the Nginx sample page. In order to redirect the request to the next.js app running at port 3000. We need to implement a technique called **reverse proxy** in **Nginx**.

```
1 cd /etc/nginx/sites-available copy
2 sudo vim default
```

- **Delete** the highlight block, displayed below in the screenshot :


```
server_name _;

location / {
    # First attempt to serve request as file, then
    # as directory, then fall back to displaying a 404.
    try_files $uri $uri/ =404;
}

# pass PHP scripts to FastCGI server
#
#location ~ \.php$ {
```

- **Append** the new code block, given below:

```
1      location / {
2          proxy_pass          http://127.0.0.1:3000;
3          proxy_read_timeout  60;
4          proxy_connect_timeout 60;
5          proxy_redirect      off;
6
7          # Allow the use of websockets
8          proxy_http_version 1.1;
9          proxy_set_header Upgrade $http_upgrade;
10         proxy_set_header Connection 'upgrade';
11         proxy_set_header Host $host;
12         proxy_cache_bypass $http_upgrade;
13     }
```

copy

Nginx config file with the changes mentioned is attached below for your reference:

- Run the Next.js app using the PM2 command.

```
1 pm2 start npm --name app1 -- run start -- -p 3000
```

copy

- Restart Nginx to see the result

```
1 sudo systemctl restart nginx
```

copy

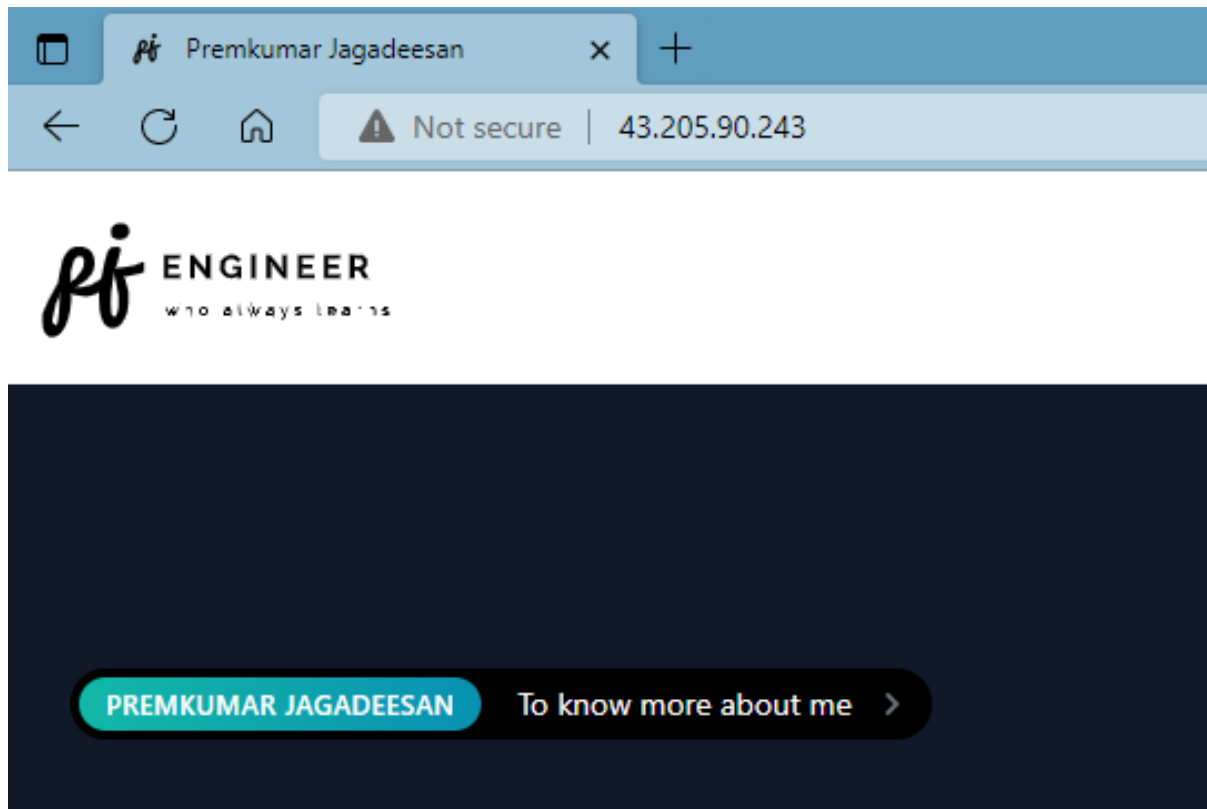
Step 10: Virtual Host

If you want to access your Next.js web application/website using the **domain name** and not with an IP address. Check out the tutorial on [how to set up a virtual host in Nginx](#).

Result:

If the following steps are done correctly, you can access your web application/website with the public IP address on the browser

Example output screenshot :



Comments

enter your comment he

Post

Subscribe to our Newsletter

The latest news, blogs, and fun memes from the community!

Subscribe

[Services](#) [Products](#) [Blog](#) [Contact](#)

Copyright @2022 Episyche Technologies Pvt Ltd