



- ☐ 95. Hosting Tutorial: Linking Domain to the Web Hosting Server | Web Development Tutorials #95  
Free YouTube Video
- ☐ 96. Hosting Tutorial: Host Multiple Websites On One Single Hosting Server | Web Development Tutorials#96  
Free YouTube Video
- ☐ 97. Hosting Tutorial: Deploy NodeJs Apps in Production on Linux VPS | Web Development Tutorials#97  
Free YouTube Video
- ☐ 98. Installing MongoDB & Hosting

## Installing MongoDB & Hosting our Dance Website on Ubuntu VPS | Web Development Tutorials #98

### Installing MongoDB and Hosting our Dance Website on Ubuntu VPS

In this tutorial, we are going to see how to host the dance website that we created with the help of NodeJs app. Since we know how to transfer the files with the help of *Filezilla*, we will use the same concept here also.

We will start by making a new folder and copy all the files except node modules. We will not copy the node modules because we want to build them separately. We will now go to our terminal window and search our dance website folder. After searching, we will get the result as follows-

```
root@HarrysSite:~/home/dance
* Documentation: https://help.ubuntu.com
* Management:   https://landscape.canonical.com
* Support:       https://ubuntu.com/advantage

System information as of Mon Apr 13 14:31:18 UTC 2020

System load:  0.07          Processes:      98
Usage of /:   9.9% of 24.06GB Users logged in: 0
Memory usage: 29%          IP address for eth0: 139.59.89.220
Swap usage:   0%

* Canonical Livepatch is available for installation.
- Reduce system reboots and improve kernel security. Activate at:
https://ubuntu.com/livepatch

55 packages can be updated.
0 updates are security updates.

Last login: Thu Apr  9 15:50:45 2020 from 124.123.81.131
root@HarrysSite:~# ls
nohup.out
root@HarrysSite:~# cd ../
root@HarrysSite:~# ls
bin  dev  home  initrd.img.old  lib64  media  opt  root  sbin  srv  var  vmlinuz  vmlinuz.old
boot  etc  initrd.img  lib          lost+found  mnt  proc  run  snap  sys  usr  vmlinuz
root@HarrysSite:~# cd home/
root@HarrysSite:/home# ls
dance  nodeapp
root@HarrysSite:/home# cd dance/
root@HarrysSite:/home/dance# ls
app.js  package-lock.json  package.json  static  views
root@HarrysSite:/home/dance# vi
```

If by any condition the port 8000 does not open, then we can try the *ufw allow 800* command. Now we need to install MongoDB on the server. To install MongoDB, we need to write *sudo apt install -y mongodb*. And if we now go to our port 8000, then we will get the output as follows-

To view all the websites running on the server, we can write the command as *pm2 list*. After running the command, we will get the output as follows-

```
root@HarrysSite:/home/dance
app.js  node_modules  package-lock.json  package.json  static  views
root@HarrysSite:/home/dance# node app.js
body-parser deprecated extended: provide extended option app.js:24:17
(node:19192) DeprecationWarning: current Server Discovery and Monitoring engine is deprecated, and will be removed in a future version. To use the new Server Discover and Monitoring engine, pass option { useUnifiedTopology: true } to the MongoClient constructor.
The application started successfully on port 8000
^C
root@HarrysSite:/home/dance# ^C
root@HarrysSite:/home/dance# ^C
root@HarrysSite:/home/dance# ^C
root@HarrysSite:/home/dance# pm2 list

┌───┬───┬───┬───┬───┬───┬───┐
│ id │ name │ mode │  0  │ status │ cpu │ memory │
├───┴───┴───┴───┴───┴───┴───┤
│ 0   │ HarryBhai1 │ fork │ 0   │ online │ 0%  │ 45.9mb │
│ 1   │ HarryBhai2 │ fork │ 0   │ online │ 0.3% │ 52.2mb │
└───┴───┴───┴───┴───┴───┴───┘

root@HarrysSite:/home/dance# ls
app.js  node_modules  package-lock.json  package.json  static  views
root@HarrysSite:/home/dance# pm2 start app.js --name "Dance Website"
[PM2] Starting /home/dance/app.js in fork_mode (1 instance)
[PM2] Done.

┌───┬───┬───┬───┬───┬───┬───┐
│ id │ name │ mode │  0  │ status │ cpu │ memory │
├───┴───┴───┴───┴───┴───┴───┤
│ 2   │ Dance Website │ fork │ 0   │ online │ 0%  │ 18.2mb │
│ 0   │ HarryBhai1 │ fork │ 0   │ online │ 0%  │ 45.9mb │
│ 1   │ HarryBhai2 │ fork │ 0   │ online │ 0%  │ 50.3mb │
└───┴───┴───┴───┴───┴───┴───┘

[PM2][WARN] Current process list running is not in sync with saved list. Type 'pm2 save' to synchronize or 'pm2 save' to enable auto sync via 'pm2 set pm2:autodump true'
root@HarrysSite:/home/dance#
```

To save this list, we can write *pm2 save*. Now we will point our dance website with the domain *programmingwithharry.com*. We need to write the following commands as follows to do the same task-

```
root@HarrysSite:/home/dance
// PM2 SPECIFIC STUFF
app.set('view engine', 'pug') // Set the template engine as pug
app.set('views', path.join(__dirname, 'views')) // Set the views directory

// ENDPOINTS
app.get('/', (req, res)=>{
  const params = {}
  res.status(200).render('home.pug', params);
})

app.get('/contact', (req, res)=>{
  const params = {}
  res.status(200).render('contact.pug', params);
})

app.post('/contact', (req, res)=>{
  var myData = new Contact(req.body);
  myData.save().then(()=>{
    res.send("This item has been saved to the database")
  }).catch(()=>{
    res.status(400).send("Item was not saved to the database")
  });
  // res.status(200).render('contact.pug');
})

// START THE SERVER
app.listen(port, ()=>{
  console.log(`The application started successfully on port ${port}`);
});
```

And then change the port number from 3000 to 8000. And now if we reload the domain, we get our dance website on this domain.

I hope you must have understood how to serve any website on your server. You can now make different websites and point them to your server. Till then keep practicing.