**<project\_name> SERVER SETUP ON AMAZON**

1. **EC2 INSTANCE SETUP ON AMAZON CLOUD**
   1. Open the **EC2 Console** link and click on the button **Launch Instance**.
   2. Select **Ubuntu 16.04** as operating system and choose the instance type. For free tier, choose t2.micro instance.
   3. Click on **Next : Configure Instance Details** button and configure the instance details as required.
   4. Click on **Next : Add Storage** button and configure the storage as required.
   5. Click on **Next : Add Tags** button and configure it as required.
   6. Click on the **Next : Configure Security** button and you can see SSH type with 22 port option. If not, click on the **Add Rule** button and choose **SSH** as rule from the list and add into list. Also, add **HTTP**, **HTTPS** from the rule list.
   7. Click on **Review & Launch** button and review your configured settings.
   8. Click on **Launch** button and a modal overlay will open to **Create / Select Key Pair**. Select option create a **New Key Pair** and mention the name of the key pair. Now download the .pem file after clicking on **Download Key Pair** button.
   9. Click on the **Launch Instance** button and your instance will get created. This might take some time to create an new EC2 instance.
   10. Go to the **Instances** link on the left menu and have look at the new instance created by you.
   11. Click on the **Name** column of instance and write the name of the instance of your choosing.
   12. Choose this new instance and click on the button **Connect**. This will open an modal overlay and instruction to login into new server.
2. **LOGIN ON AMAZON SERVER**
   1. When using the instance **for the first time**, locate the downloaded **Key Pair .pem** file and run the following command using terminal.

|  |  |
| --- | --- |
| >>> | chmod 400 <project\_name>.pem |

Now login on amazon server using this command as mentioned in the **clause A.12**. on aws site. Just copy paste the line on terminal and run it.

|  |  |
| --- | --- |
| >>> | ssh -i "<project\_name>.pem" ubuntu@<server\_ip> |

* 1. Otherwise login with following command,

|  |  |
| --- | --- |
| >>> | ssh ubuntu@<test.<project\_name>.com> |

1. **PASSWORD LESS ACCESS ON AMAZON SERVER**
   1. Search for **Public RSA Key** in your ssh folder of your **local machine**

|  |  |
| --- | --- |
| >>>  >>> | cat ~/.ssh/id\_rsa.pub  <ctrl+c> |

* 1. Paste your local machine public RSA key to **Amazon Server’s Authorized Keys** file

|  |  |
| --- | --- |
| >>>  >>>  >>>  >>>  >>> | vi ~/.ssh/authorized\_keys  <ctrl+v>  <esc>  :wq  logout |

* 1. Login again on Amazon server without using any password or .pem file

|  |  |
| --- | --- |
| >>> | ssh ubuntu@<host\_name> |

1. **UPDATE UBUNTU SERVER AND INSTALL DEPENDENCIES**
   1. Update the server with all new **OS upgrades**

|  |  |
| --- | --- |
| >>>  >>>  >>>  >>> | sudo apt-get update  sudo apt-get upgrade  sudo apt-get dist-upgrade  sudo reboot |

* 1. Export **Language Packs**

|  |  |
| --- | --- |
| >>>  >>>  >>>  >>>  >>> | export LANGUAGE=en\_US.UTF-8  export LANG=en\_US.UTF-8  export LC\_ALL=en\_US.UTF-8  sudo locale-gen en\_US.UTF-8  sudo dpkg-reconfigure locales |
|  |  |

1. **PYTHON 3.6 SETUP**
   1. As we will using Python3.6 and it is not inherently available on Ubuntu OS, we will install it through external repository

|  |  |
| --- | --- |
| >>>  >>>  >>> | sudo add-apt-repository ppa:jonathonf/python-3.6  sudo apt-get update  sudo apt-get install python3.6 |

1. **INSTALL PROJECT DEPENDENCIES**
   1. Install all dependencies for the project

|  |  |
| --- | --- |
| >>>  >>> | sudo apt-get install libffi-dev libpq-dev libxml2-dev libxslt1-dev python-dev zlib1g-dev supervisor nginx git python3-pip  sudo apt-get build-dep python-imaging libjpeg8 libjpeg62-dev libfreetype6 libfreetype6-dev git |

1. **INSTALL VIRTUAL ENVIRONMENT FOR PYTHON 3.6**
   1. To user virtual environment for Python3.6, install **virutalenv** using pip3

|  |  |
| --- | --- |
| >>> | sudo pip3 install virtualenv |

1. **POSTGRES DATABASE SETUP**
   1. The first step to install postgresql database and all the necessary dependencies

|  |  |
| --- | --- |
| >>> | sudo apt-get install -y postgresql postgresql-contrib |
|  |  |

* 1. Set password for user **Postgres**

|  |  |
| --- | --- |
| >>>  >>> | sudo passwd postgres  <postgres-database-password> |

* 1. Create a **new Database**

|  |  |
| --- | --- |
| >>>  >>>  >>>  >>>  >>>  >>>  >>>  >>>  >>> | su - postgres  <postgres\_database\_password>  psql  CREATE ROLE <project\_name>\_app;  ALTER ROLE <project\_name>\_app WITH PASSWORD '<project\_name>\_app';  ALTER ROLE <project\_name>\_app WITH LOGIN;  CREATE DATABASE <project\_name>\_app WITH OWNER <project\_name>\_app;  GRANT ALL PRIVILEGES ON DATABASE <project\_name>\_app TO <project\_name>\_app;  \q  logout |
|  |  |

1. **BITBUCKET PASSWORDLESS ACCESS THROUGH AMAZON SERVER**
   1. Create **RSA Public key** on Server

|  |  |
| --- | --- |
| >>>  >>>  >>> | cd ~  ssh-keygen -t rsa -b 4096 -C "ubuntu@<hostname>"  <no para phrase> |

* 1. Copy the RSA public key

|  |  |
| --- | --- |
| >>>  >>> | cat ~/.ssh/id\_rsa.pub  <ctrl+c> |

* 1. Now we need to use this key into Bitbucket Account and follow the mentioned steps.
     1. Login into **BitBucket Account**
     2. From Bitbucket page, choose **Bitbucket settings** options and you will land up in **Account Settings** page
     3. Select **SSH Keys** tab and you will find all the already mentioned SSH keys list
     4. Click on the button **Add Key** and paste the copied public SSH key
     5. Bitbucket adds the public key and send email for confirmation
  2. Now we are good to use for pulling the GIT repository from Bitbucket seamlessly without using any password every time.

1. **PROJECT SETUP**
   1. **Clone** the Git repository from Bitbucket

|  |  |
| --- | --- |
| >>>  >>>  >>>  >>>  >>> | mkdir ~/<project\_name>\_app  cd ~/<project\_name>\_app  git clone <bitbucket path>  cd <project\_name>\_app  git checkout master |

* 1. Setup **Virtual Environment** for this project

|  |  |
| --- | --- |
| >>>  >>>  >>> | cd ~/<project\_name>\_app  virtualenv --system-site-packages -p /usr/bin/python3.6 virt/<project\_name>\_app  source virt/<project\_name>\_app/bin/activate |

* 1. **Export** Django Settings Module

|  |  |
| --- | --- |
| >>>  >>> | cd ~/<project\_name>\_app/<project\_name>\_app  export DJANGO\_SETTINGS\_MODULE="<project\_name>\_app.settings.test" |
|  |  |

Note :- mind the quotes

* 1. Install **Requirements** for the project

|  |  |
| --- | --- |
| >>> | pip3 install -r <project\_name>\_app/config/requirements.txt |

* 1. Database **Migrations** and create Superuser

|  |  |
| --- | --- |
| >>>  >>>  >>>  >>>  >>> | ./manage.py migrate  ./manage.py createsuperuser  <username>  <email>  <password> |

* 1. Populate all **Countries Details** from Django-Country Module

|  |  |
| --- | --- |
| >>> | ./manage.py update\_countries\_plus |

* 1. Collect all **static files**

|  |  |
| --- | --- |
| >>> | ./manage.py collectstatic |

1. **MAKE PROJECT LIVE ON THE SITE USING SUPERVISOR**
   1. Create **Log folder** for Supervisor

|  |  |
| --- | --- |
| >>>  >>>  >>> | mkdir ~/<project\_name>\_app/log  touch ~/<project\_name>\_app/log/infolog.log  touch ~/<project\_name>\_app/log/errorlog.log |

* 1. Copy **c10\_site\_supervisor.conf** file and paste into server’s supervisor folder

|  |  |
| --- | --- |
| >>> | sudo cp ~/<project\_name>\_app/<project\_name>\_app/<project\_name>\_app/server\_config/<project\_name>\_app\_supervisor.conf /etc/supervisor/conf.d/<project\_name>\_app\_supervisor.conf |

* 1. Run **Supervisor** commands and start the service

|  |  |
| --- | --- |
| >>>  >>>>>> | sudo supervisorctl reread  sudo supervisorctl update  sudo supervisorctl restart all |
|  |  |

1. **SSL SETUP WITH LET’S ENCRYPT**
   1. Stop **Nginx** as standalone method would be used to install certificates

|  |  |
| --- | --- |
| >>> | sudo service nginx stop |

* 1. Install **Certbot**

|  |  |
| --- | --- |
| >>>  >>>  >>> | sudo add-apt-repository ppa:certbot/certbot  sudo apt-get update  sudo apt-get install certbot |

* 1. Install **Certificates** : you canalso use **--staging** option while testing out the command

|  |  |
| --- | --- |
| >>> | sudo certbot certonly --standalone -d beta.forthe<project\_name>.com --preferred-challenges http |

* 1. Check the **IMPORTANT NOTES** section after the above command and check the following line **“Congratulations! Your certificate and chain have been saved at:”**. Now check the path of **fullchain.pem** and **privkey.pem**
  2. Open the **<project\_name>\_app\_nginx.conf** from server\_config folder of **fullchain.pem** and **privkey.pem** app and check paths of **fullchain.pem** and **privkey.pem** are written properly as mentioned in above step M.4.

1. **NGINX SETUP**
   1. Copy **<project\_name>\_app\_nginx.conf** file and paste into server’s nginx folder

|  |  |
| --- | --- |
| >>>  >>> | cd  sudo cp <project\_name>\_app/<project\_name>-web/<project\_name>\_app/server\_config/<project\_name>\_app\_nginx.conf /etc/nginx/sites-available/<project\_name>\_app\_nginx |

* 1. Create **Symlink** in sites-enabled folder of nginx

|  |  |
| --- | --- |
| >>>  >>>  >>> | sudo ln -sf /etc/nginx/sites-available/<project\_name>\_app\_nginx /etc/nginx/sites-enabled/<project\_name>\_app\_nginx  cd /etc/nginx/sites-enabled  sudo rm default |
|  |  |

* 1. Run **Nginx** commands to start service

|  |  |
| --- | --- |
| >>>  >>> | sudo service nginx stop  sudo service nginx start |
|  |  |

1. **OPEN URL FOR MAINSERVER**
   1. Open URL [**https://test.<project\_name>.com/admin**](https://test.peeple.com/admin) and check the django admin page.

**ANNEXURE A**

1. **FOR CERBOTS AND HTTPS CERTIFICATES**

[**https://github.com/certbot/certbot**](https://github.com/certbot/certbot)

[**https://certbot.eff.org/docs/using.html#webroot**](https://certbot.eff.org/docs/using.html#webroot)

[**https://certbot.eff.org/#ubuntuxenial-nginx**](https://certbot.eff.org/#ubuntuxenial-nginx)

[**https://www.digitalocean.com/community/tutorials/how-to-use-certbot-standalone-mode-to-retrieve-let-s-encrypt-ssl-certificates**](https://www.digitalocean.com/community/tutorials/how-to-use-certbot-standalone-mode-to-retrieve-let-s-encrypt-ssl-certificates)

/etc/letsencrypt/live/beta.forthe<project\_name>.com/fullchain.pem

Your key file has been saved at:

/etc/letsencrypt/live/beta.forthe<project\_name>.com/privkey.pem