Profiles API application set up local via Vagrant, then lift and shift onto AWS EC2 via deploy bash scripts - GL

Repo:

- **deploy scripts:**

https://github.com/Repliika/profiles-API/tree/master/deploy

- **application source code:**

> https://github.com/Repliika/profiles-API

Problems when local: difficult to collaborate, different OS and packages need installing, conflicts with other applications, clogs system with dev tools

To make the development server we used Vagrant to make our VM with the requirements. We can then share this with people, create and destroy the server easily.

Application code used python and the Django framework as a web framework so we can easily create a user profile application using the rest framework that belong with Django.

Run virtually: SSH into the AWS instance and download +execute the deploy script from GitHub which will set up a Nginx webserver to host the application which. And finally

running the user profile application in AWS public cloud.

We have a script to fetch updates on the repo, this is done because the initial configure script has installed Git.

Technologies

Git

VirtualBox

Python, Django, rest framework

Vagrant

AWS EC2

We need to initialize first for our project

```
ginoo@DESKTOP-UKJ6M58 MINGW64 ~/Documents/ProfilesAPI
$ git init
Initialized empty Git repository in C:/Users/ginoo/Documents/ProfilesAPI/.git/
```

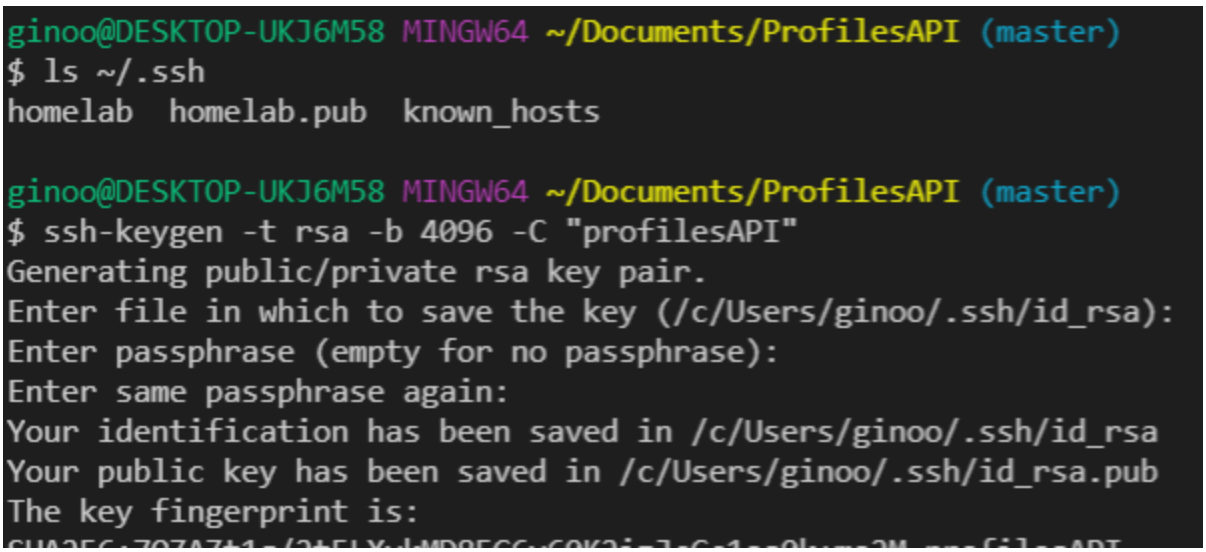
When creating a Repo we can make a .gitignore file containing a list of files to not include when pushing up your code.

We need to make our first commit to test

```
ginoo@DESKTOP-UKJ6M58 MINGW64 ~/Documents/ProfilesAPI (master)
$ git add .

ginoo@DESKTOP-UKJ6M58 MINGW64 ~/Documents/ProfilesAPI (master)
$ git commit -am "first commit"
[master (root-commit) 55fdd67] first commit
 2 files changed, 1 insertion(+)
 create mode 100644 .gitignore
 create mode 100644 README.md
```

Generating an SSH Key

We are going to create an encrypted keypair so we can access things in the project via SSH

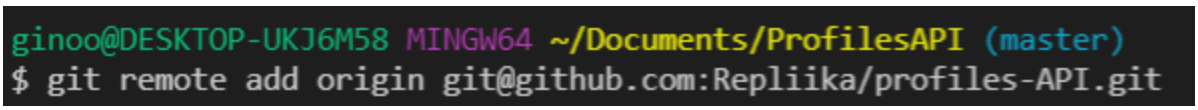


An SSH key is made of a asymmetric private and public key.

We are going to create a repo for our source code so we can push to it.

After keypair creation we are going to copy the public key into our GitHub account in order to push our project to gitHub via SSH

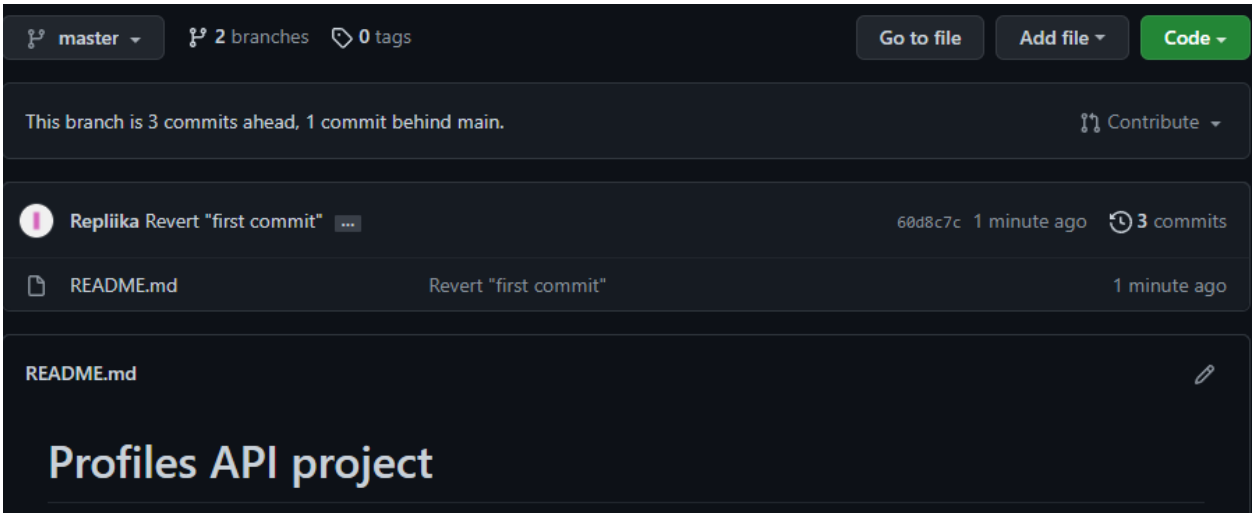
After the SSH key is in the gitHub account we can create an origin to that repo\



Check to see if we connected to the right repo and then push our first commit.

```
ginoo@DESKTOP-UKJ6M58 MINGW64 ~/Documents/ProfilesAPI (master)
$ git remote -v
origin  git@github.com:Repliika/profiles-API.git (fetch)
origin  git@github.com:Repliika/profiles-API.git (push)

ginoo@DESKTOP-UKJ6M58 MINGW64 ~/Documents/ProfilesAPI (master)
$ git push -u origin master
The authenticity of host 'github.com (140.82.112.4)' can't be established.
```

We should see the repo populated.



Now that we have a working repo we can go ahead and create the dev server via Vagrant

We need to initialize a vagrant file in the folder or you can paste one

```
ginoo@DESKTOP-UKJ6M58 MINGW64 ~/Documents/ProfilesAPI (master)
$ vagrant init
A `Vagrantfile` has been placed in this directory. You are now
ready to `vagrant up` your first virtual environment! Please read
the comments in the Vagrantfile as well as documentation on
`vagrantup.com` for more information on using Vagrant.
```

We are going to update the box and install python

We are running a ubuntu machine and mapping port 8000 on the server to 8000 on the host machine where the box is running on top of.

Disable auto update because it will interfere with the manual update we are specifying as inline shell arguments. Going to install python venv and zip package

Also as good practice changing python alias to python3 that way we do not have to do python3 every time we run something because python referred to the old v2 and new current python is v3

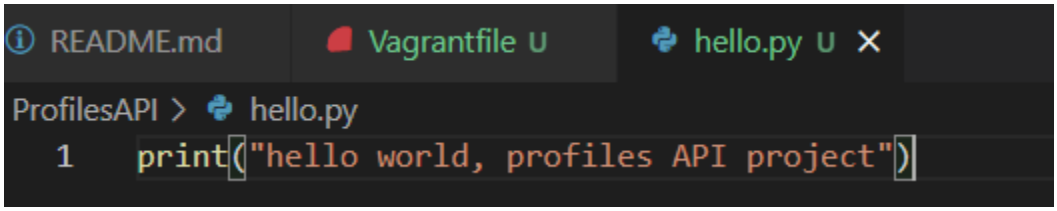
Do Vagrant up and we should be in the box

```
ginoo@DESKTOP-UKJ6M58 MINGW64 ~/Documents/ProfilesAPI (master)
$ vagrant ssh
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 4.15.0-163-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Thu Dec 30 17:10:34 UTC 2021

  System load:  0.65              Processes:             104
  Usage of /:   3.2% of 38.71GB   Users logged in:       0
  Memory usage: 14%               IP address for enp0s3: 10.0.2.15
  Swap usage:   0%


0 updates can be applied immediately.

New release '20.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


vagrant@ubuntu-bionic:~$
```

Vagrant gives us a synced folder called Vagrant on the system and your working directory on host

```
vagrant@ubuntu-bionic:/$ cd /vagrant
vagrant@ubuntu-bionic:/vagrant$ ls
README.md  Vagrantfile  ubuntu-bionic-18.04-cloudimg-console.log
vagrant@ubuntu-bionic:/vagrant$
```

To test the server we can create a python file to execute in our home folder which should then be seen inside the box




```
vagrant@ubuntu-bionic:/vagrant$ ls
README.md  Vagrantfile  hello.py  ubuntu-bionic-18.04-cloudimg-console.log
vagrant@ubuntu-bionic:/vagrant$ python hello.py
hello world, profiles API project
vagrant@ubuntu-bionic:/vagrant$
```

We can now push everything to the repo as our server is complete. Exit out, destroy the instance if need be. Finally git add, commit and push

```
vagrant@ubuntu-bionic:/vagrant$ exit
logout
Connection to 127.0.0.1 closed.

ginoo@DESKTOP-UKJ6M58 MINGW64 ~/Documents/ProfilesAPI (master)
$ Vagrant destroy
    default: Are you sure you want to destroy the 'default' VM? [y
==> default: Forcing shutdown of VM...
==> default: Destroying VM and associated drives...

ginoo@DESKTOP-UKJ6M58 MINGW64 ~/Documents/ProfilesAPI (master)
$ git add .

ginoo@DESKTOP-UKJ6M58 MINGW64 ~/Documents/ProfilesAPI (master)
$ git commit -am "Vagrant dev server with python"
[master 6b36df6] Vagrant dev server with python
 4 files changed, 173 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 .vscode/settings.json
 create mode 100644 Vagrantfile
 create mode 100644 hello.py

ginoo@DESKTOP-UKJ6M58 MINGW64 ~/Documents/ProfilesAPI (master)
$ git push origin
Enumerating objects: 8, done.
```

Because we are using a django project utilizing the rest API we need to install some requirements

create a requirements.txt containing the packages for the application and setup a virtual environment to install it in, this way we have a container for the dependencies and it does not interfere with the machine

```
vagrant@ubuntu-bionic:~$  cd /vagrant
vagrant@ubuntu-bionic:/vagrant$ python -m venv ~/env
vagrant@ubuntu-bionic:/vagrant$ source ~/env/bin/activate
(env) vagrant@ubuntu-bionic:/vagrant$ pip install -r requirements.txt
```

Once everything is installed we can create a django project to manage applications. make sure to create it in the folder that is synced.

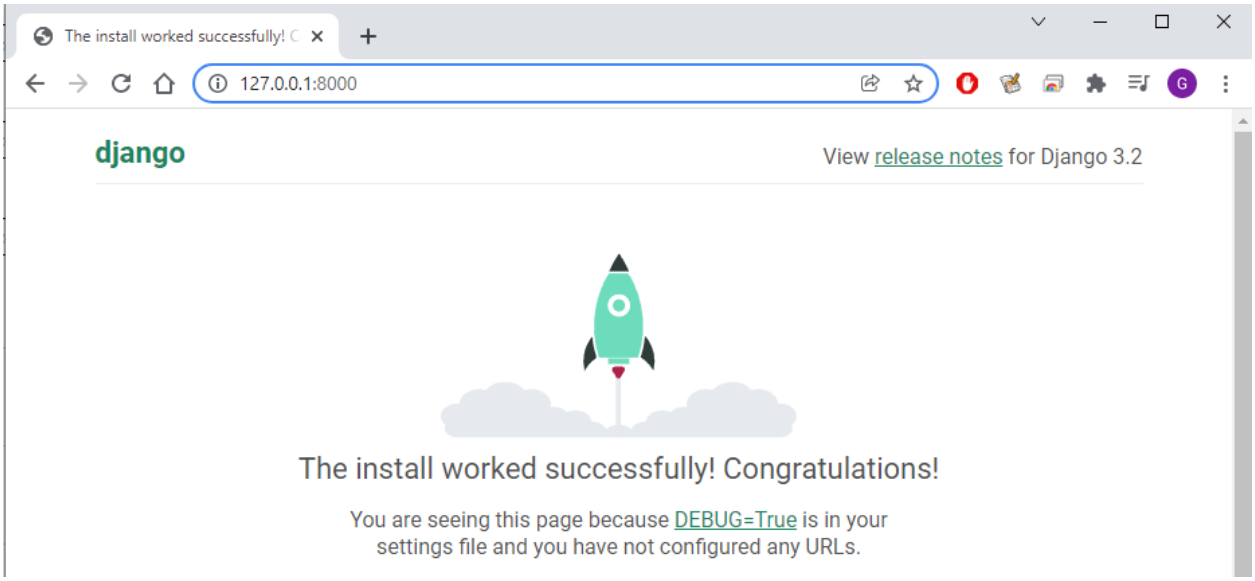
```
(env) vagrant@ubuntu-bionic:/vagrant$ django-admin.py startproject profiles_project .
(env) vagrant@ubuntu-bionic:/vagrant$ python manage.py startapp profilesAPI
(env) vagrant@ubuntu-bionic:/vagrant$
```

Make sure to add the application to the settings.py

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'rest_framework',
    'rest_framework.authtoken',
    'profilesAPI',
]
```

We can now start our django application by running it on port 8000 which was mapped in the vagrant file to 8000 on our host. Go to 127.0.0.1:8000 and you should be able to see the app

```
(env) vagrant@ubuntu-bionic:/vagrant$ python manage.py runserver 0.0.0.0:8000
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 21 unapplied migration(s). Your project may not work properly until you apply the migrat
, authtoken, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
December 31, 2021 - 13:12:20
Django version 3.2.10, using settings 'profiles_project.settings'
Starting development server at http://0.0.0.0:8000/
Quit the server with CONTROL-C.
[31/Dec/2021 13:13:36] "GET / HTTP/1.1" 200 10697
[31/Dec/2021 13:13:36] "GET /static/admin/css/fonts.css HTTP/1.1" 200 423
[31/Dec/2021 13:13:36] "GET /static/admin/fonts/Roboto-Light-webfont.woff HTTP/1.1" 200 85692
[31/Dec/2021 13:13:36] "GET /static/admin/fonts/Roboto-Regular-webfont.woff HTTP/1.1" 200 85876
[31/Dec/2021 13:13:36] "GET /static/admin/fonts/Roboto-Bold-webfont.woff HTTP/1.1" 200 86184
Not Found: /favicon.ico
[31/Dec/2021 13:13:36] "GET /favicon.ico HTTP/1.1" 404 2120
```



Now we are going to migrate our application into the DB

```
(env) vagrant@ubuntu-bionic:/vagrant$ python manage.py makemigrations profilesAPI
Migrations for 'profilesAPI':
  profilesAPI/migrations/0001_initial.py
    - Create model UserProfile
(env) vagrant@ubuntu-bionic:/vagrant$
```

Now we can create the DB

```
(env) vagrant@ubuntu-bionic:/vagrant$ python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, authtoken, contenttypes, profilesAPI, sessions
Running migrations:
```

Created a superuser with the django manage cli

```
(env) vagrant@ubuntu-bionic:/vagrant$ python manage.py createsuperuser
Email: ginoo@superuser.com
Name: Gino
Password:
Password (again):
This password is too common.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.
(env) vagrant@ubuntu-bionic:/vagrant$
```

Now that we have successfully ran out application locally and automating the web server with Vagrant, we can now migrate our application to the cloud, specifically AWS.

We are going to put all the code into a repo, and we are going to download the installation script which will copy necessary files, configure nginx webserver, update and install required packages, create python virtual env, install supervisor which will allow us to create a process of the django application in order for us to run on the webserver and this will be connected via uwsgi interface.

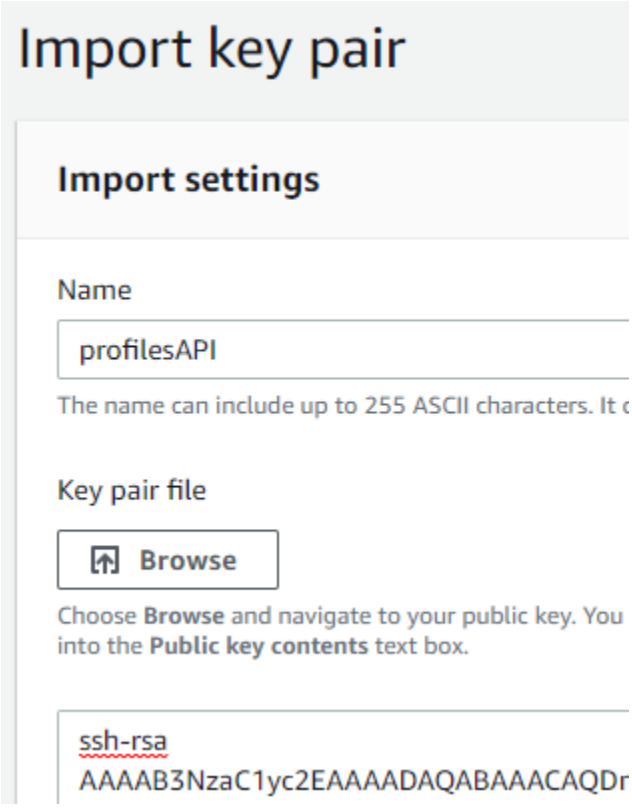
We will also create a script to create an admin for us to navigate the admin page

Find the same image you used for vagrant in the AWS AMI marketplace in our case ‘Ubuntu Server 18.04’

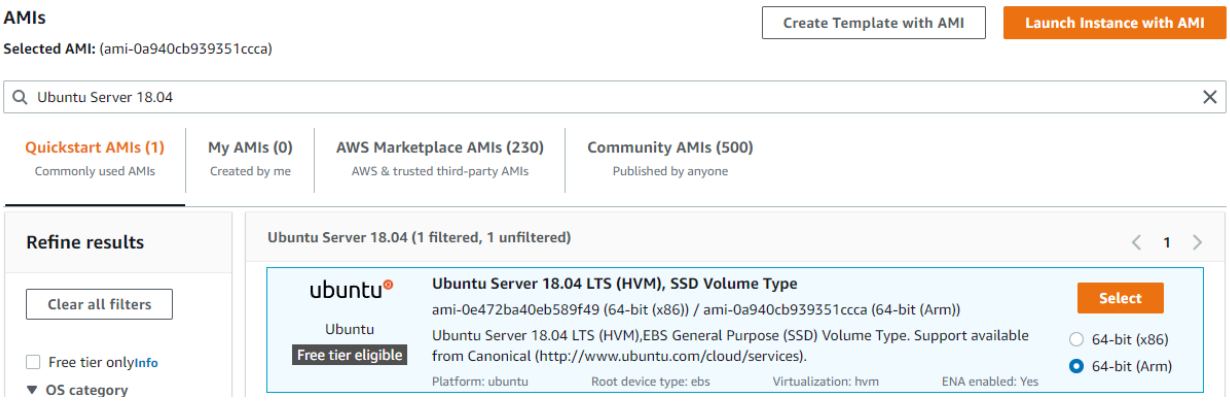
With this image, we are going to create our instance, for the keypairs we are going to use the same pair we made initially for Vagrant to save time.

Grab the public key located locally to paste into AWS

```
ginoo@DESKTOP-UKJ6M58 MINGW64 ~/Documents/awsCredentials
$ cat ~/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQDmvTWHmw6OzbLBCadU
```



For the instance itself we are going to grab a similar ubuntu 18.04 OS from the AMI marketplace



For the security group, for now only using my IP to see if we can connect and see if everything works before allowing public use.

AMI Details

ubuntu/images/hvm-ssd/ubuntu-bionic-18.04-arm64-server-20211129 - ami-0a940cb939351ccca

Canonical, Ubuntu, 18.04 LTS, arm64 bionic image build on 2021-11-29

Root Device Type: ebs Virtualization type: hvm

Instance Type

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)
t4g.micro	-	2	1	EBS only

Security Groups

Security group name launch-wizard-2
Description launch-wizard-2 created 2022-01-02T05:22:47.555-05:00

Type	Protocol	Port Range
SSH	TCP	22
HTTP	TCP	80

Instance Details

Now we can connect via ssh and start building the environment

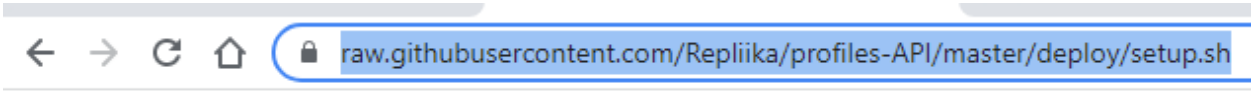
```
ginoo@DESKTOP-UKJ6M58 MINGW64 ~/Documents/awsCredentials
$ ssh -i "profilesAPI.pem" ubuntu@ec2-54-84-96-77.compute
Warning: Identity file profilesAPI.pem not accessible: No
The authenticity of host 'ec2-54-84-96-77.compute-1.amazo
ED25519 key fingerprint is SHA256:IqNkHa3qumlxZXtilPqRbd1
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fin
Warning: Permanently added 'ec2-54-84-96-77.compute-1.ama
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-1060-aws a
```

Deploy scripts

https://github.com/Repliika/profiles-API/tree/master/deploy

Now we can curl to fetch our setup.sh file and execute it. Another way to do this is to paste the script into user data when the instance starts up.

Make sure to grab the raw file from github.




```
#!/usr/bin/env bash

set -e

# TODO: Set to URL of git repo.
PROJECT_GIT_URL='https://github.com/Repliika/profiles-API.git'

PROJECT_BASE_PATH='/usr/local/apps/profilesAPI'
```

HOWEVER, if you just curl it'll spit out the file instead of executing.

```
ubuntu@ip-172-31-93-113:~$ curl -sL https://raw.githubusercontent.com/Repliika/profiles-API/master/deploy/setup.sh
#!/usr/bin/env bash

set -e

# TODO: Set to URL of git repo.
PROJECT_GIT_URL='https://github.com/Repliika/profiles-API.git'
```

Now that we have an instance running we can add the ipv4 public name to the settings.py file

Add the local host IP if you'd like it ran on there too or any other machine. and push to git

```
ALLOWED_HOSTS = [
    'ec2-54-84-96-77.compute-1.amazonaws.com',
```

curl -sL https://raw.githubusercontent.com/Repliika/profiles-API/master/deploy/setup.sh | sudo bash -

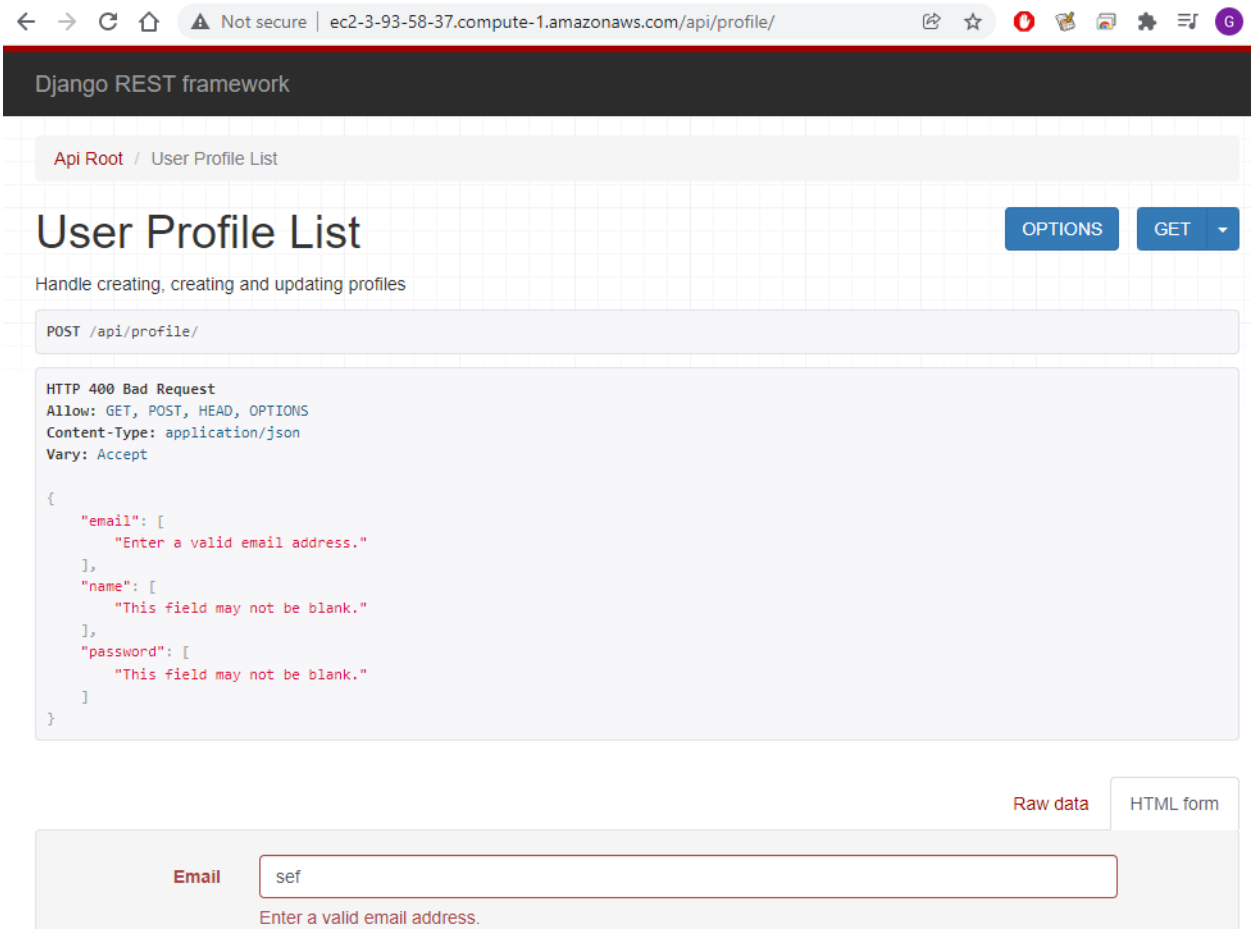
do curl -sL url | sudo bash - ← pipe the url into 'bash -' the '-' means to run on bash, not an input option.

```
ubuntu@ip-172-31-93-113:~$ curl -sL https://raw.githubusercontent.com/Repliika/profiles-API/master/deploy/setup.sh | sudo bash -
Installing dependencies...
Hit:1 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports bionic InRelease
Hit:2 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports bionic-updates InRelease
Hit:3 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports bionic-backports InRelease
Hit:4 http://ports.ubuntu.com/ubuntu-ports bionic-security InRelease
```

Should see this at the bottom of the output and we can now view our site on AWS

```
161 static files copied to '/usr/local/apps/profilesAPI/static'.
profilesAPI: available
profilesAPI: added process group
profilesAPI: stopped
profilesAPI: started
nginx configured!
```

Just note below the EC2 address is different as I terminated the previous one and spun up a new one.



Done!

Anytime you update the repo you can run the update script in the deploy folder to update*