University of New South Wales

COMP3900

Project Report: LocalHost

Group Name: *Undergrads*

Edward Dai	z3415169@unsw.edu.au	z3415169	Developer
Jasper Lowell	z5180332@unsw.edu.au	z5180332	Scrum Master, Developer
Steven Tudo	z3466724@unsw.edu.au	z3466724	Developer
Thomas Lam	z5136612@unsw.edu.au	z5136612	Developer

October 6, 2018

Contents

1 Deployment

LocalHost is deployed using *Amazon Web Services* on an *Amazon Elastic Compute Cloud* instance with the following specifications.

```
OS: Ubuntu 18.04.1 LTS x86_64
Host: HVM domU 4.2.amazon
Kernel: 4.15.0-1023-aws
CPU: Intel Xeon E5-2676 v3 (1) @ 2.4
GPU: Cirrus Logic GD 5446
Memory: 983MiB
```

1.1 Amazon Web Services Configuration

After creating an instance, the only port on the server accessible from the internet is `22 (SSH)`. For the server to accept web requests, ports `80 (HTTP)` and `443 (SSL)` must be opened. This can be done by navigating to the dashboard, selecting the running *Amazon Elastic Compute Cloud* instance, and selecting `launch-wizard`. A tab will then appear at the bottom of the page and will contain a form that can open ports.

1.2 Software

The default user account provided on an *Amazon Elastic Compute Cloud* instance running *Ubuntu* is `ubuntu` and the following commands rely on this assumption. Commands requiring sudo privileges are prepended with `#` and commmands executed under user permissions are prepended with `$`.

1.2.1 Repository

To ease updating, an `SSH` key will be generated to authenticate the server when accessing the repository.

```
$ ssh-keygen -t rsa -b 4096
Enter file in which to save the key (/home/ubuntu/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in id_rsa.
Your public key has been saved in id_rsa.pub.
```

After adding `/home/ubuntu/.ssh/id_rsa.pub` to *BitBucket* the repository can then be cloned.

```
$ git clone git@bitbucket.org:jtalowell/localhost.git
    /home/ubuntu/localhost
```

1.2.2 Django & Immediate Dependencies

Django and its immediate dependencies must be installed.

```
$ cd /home/ubuntu/localhost
$ python -m venv venv
$ source venv/bin/activate
$ (venv) pip install -r requirements.txt
```

1.2.3 PostgreSQL

PostgreSQL must be installed and it's accompaning system service enabled.

```
# apt install postgresql postgresql-contrib
# systemctl enable postgresql.service
# systemctl start postgresql.service
```

The database cluster must be created.

```
$ sudo -u postgres -i
[postgres]$ /usr/lib/postgresql/10/bin/initdb -D
    '/var/lib/postgresql/data'
```

A user to administer the database clusted must be created.

```
[postgres]$ createuser --interactive
Enter name of role to add: ubuntu
Shall the new role be a superuser? (y/n) y
[postgres]$ exit
```

The database to use for the project can now be created.

```
$ createdb localhost_db
$ sudo -u postgres -i
[postgres]$ psql
postgres=# \password ubuntu
Enter new password: [DB_PW]
Enter it again: [DB_PW]
postgres=# GRANT ALL PRIVILEGES ON DATABASE localhost_db TO ubuntu;
postgres=# ALTER ROLE ubuntu SET client_encoding TO ''utf8'';
postgres=# ALTER ROLE ubuntu SET default_transaction_isolation TO
    ''read committed'';
```

```
postgres=# \q
[postgres]$ exit
```

1.2.4 Redis

Ubuntu's package for *Redis* works without any extended configuration and its services are automatically started and enabled.

```
# apt install redis-server
```

1.2.5 Daphne

Daphne is installed as one of *Django*'s immediate dependencies. The only thing left is to register a service script with *Systemd*.

```
# ln -sf /home/ubuntu/localhost/deploy/systemd/daphne.service
   /etc/systemd/system
# nano /etc/systemd/system/daphne.service
```

The environment variables for `SECRET_KEY`, `DB_USER`, and `DB_PW` must be set prior to enabling. Below `Environment="DJANGO_SETTINGS_MODULE=localhost.settings_productio` add:

```
Environment="SECRET_KEY=YOUR KEY HERE"
Environment="DB_USER=ubuntu"
Environment="DB_PW=YOUR DB PASSWORD HERE"
```

1.2.6 Gunicorn

1.2.7 NGINX

1.2.8 Maintenance