



Teaching

Protected: CSC-394 HW-6 Tooling Setup EC2 Setup

Step 1 – Create an EC2 instance

What is EC2?

EC2 stands for `Elastic Compute Cloud`. It is a service provider of Cloud Computing in AWS's data centers. It allows you to provision, and deploy a Linux, Windows, macOS server in the cloud! Accessible from anywhere you like!

For this assignment we will use EC2 to provision a server to host the django application we created in the previous assignment.

In the search bar at the top of the `management console` type in EC2, and select the EC2 icon in the dropdown.

The screenshot shows the AWS Management Console search results for the term 'ec2'. The search bar at the top contains 'Search results for "ec2"'.

Services (8)

- EC2 ★ Virtual Servers in the Cloud
- EC2 Image Builder ★ A managed service to automate build, customize and deploy OS images
- AWS Compute Optimizer ★ Recommend optimal AWS Compute resources for your workloads
- AWS Firewall Manager ★ Central management of firewall rules

Features (See all 46 results ▾)

- Dashboard EC2 feature
- Limits EC2 feature
- AMIs EC2 feature
- Export snapshots to EC2 Lightsail feature

Blogs (See all 1,795 results ▾)

- New – Amazon EC2 M1 Mac Instances By: Sébastien Stormacq | Date: July 7, 2022

Welcome to AWS

- Getting started with AWS Learn the fundamentals and find valuable information to get the most out of AWS.
- Training and certification Learn from AWS experts and advance your skills and knowledge.
- What's new with AWS Discover new AWS services, features, and Regions.

Feedback Looking for language selection? Find it in the new Unified Settings © 2022, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

You should now see the EC2 Management Console.

It has A LOT of options and buttons. And can look quite intimidating. But for now we only need to focus on a few areas.

The screenshot shows the AWS EC2 Management Console dashboard. On the left, there's a sidebar with navigation links for New EC2 Experience, EC2 Dashboard, Events, Tags, Limits, Instances (with sub-links for Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances), Images (AMIs, AMI Catalog), Elastic Block Store (Volumes, Snapshots), and Network & Security (Security Groups, Elastic IPs, Placement Groups, Key Pairs, Network Interfaces). The main content area has a 'Resources' section showing counts for Instances (running), Dedicated Hosts, Elastic IPs, Instances, Key pairs, Load balancers, Placement groups, Security groups, and Snapshots. Below this is a callout for Microsoft SQL Server Always On availability groups. The 'Launch instance' section allows launching an Amazon EC2 instance. The 'Service health' section shows the service is operating normally. The 'Zones' section lists availability zones with their zone IDs: us-east-1a (use1-az2), us-east-1b (use1-az4), us-east-1c (use1-az6), us-east-1d (use1-az1), us-east-1e (use1-az3), and us-east-1f (use1-az5). The 'Explore AWS' section includes links for price performance, Graviton2 instances, and cost reduction tips.

AWS EC2 Management Console

Click the `Launch instance` button

The screenshot shows the AWS EC2 Management Dashboard. On the left, there's a sidebar with various navigation options like EC2 Dashboard, Instances, Images, Elastic Block Store, Network & Security, and more. The main area has a heading 'Resources' and a summary of current resources: Instances (running) 0, Dedicated Hosts 0, Elastic IPs 0, Instances 0, Key pairs 0, Load balancers 0, Placement groups 0, Security groups 1, Snapshots 0, and Volumes 0. Below this is a callout box with the text: 'Easily size, configure, and deploy Microsoft SQL Server Always On availability groups on AWS using the AWS Launch Wizard for SQL Server. Learn more'. The central part of the dashboard features a 'Launch instance' section with a sub-section titled 'To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.' It contains a large orange 'Launch instance' button and a smaller 'Migrate a server' button. To the right of this is a 'Service health' section showing 'US East (N. Virginia)' with a status of 'This service is operating normally'. Further down are sections for 'Scheduled events' (showing 'US East (N. Virginia)' with 'No scheduled events') and 'Migrate a server' (with a note about using AWS Application Migration Service). On the far right, there are sections for 'Account attributes' (listing supported platforms like VPC), 'Explore AWS' (with links to price performance, Graviton2, and cost reduction), and 'Additional information' (with links to getting started guide and documentation). The bottom of the page includes a feedback link and copyright information.

- Enter a name for your EC2 instance
- In the quick start section choose the Ubuntu option.

The screenshot shows the AWS EC2 'Launch an instance' wizard. In the 'Name and tags' section, the name 'csc-394-hw-2' is entered. Under 'Software Image (AMI)', the 'Canonical, Ubuntu, 22.04 LTS' AMI is selected. The 'Virtual server type (instance type)' is set to 't2.micro'. In the 'Storage (volumes)' section, it shows '1 volume(s) - 8 GiB'. A tooltip for the 'Free tier' indicates it covers 750 hours of t2.micro usage per month. The 'Launch Instance' button is at the bottom right.

This screenshot shows a more detailed view of the EC2 launch wizard. It includes sections for 'Architecture' (64-bit (x86)), 'Instance type' (t2.micro), 'Key pair (login)', 'Network settings', 'Configure storage', and 'Advanced details'. The 'Free tier' tooltip is visible again. The 'Launch Instance' button is present at the bottom.

The next set of steps are a little more complex.

First ensure that the Instance type is set to t2.micro .

The screenshot shows the 'Launch an instance | EC2 Man' page on the AWS console. The 'Summary' section indicates 1 instance is being launched. The 'Software Image (AMI)' is set to 'Ubuntu Server 22.04 LTS (HVM), SSD Volume Type'. The 'Instance type' is set to 't2.micro'. The 'Launch Instance' button is visible at the bottom right.

Summary

Number of instances: 1

Software Image (AMI): Canonical, Ubuntu, 22.04 LTS, amd64 jammy image build on 2022-06-09

Virtual server type (instance type): t2.micro

Firewall (security group): New security group

Storage (volumes): 1 volume(s) - 8 GiB

Instance type

Instance type: t2.micro (Free tier eligible)

Key pair (login)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

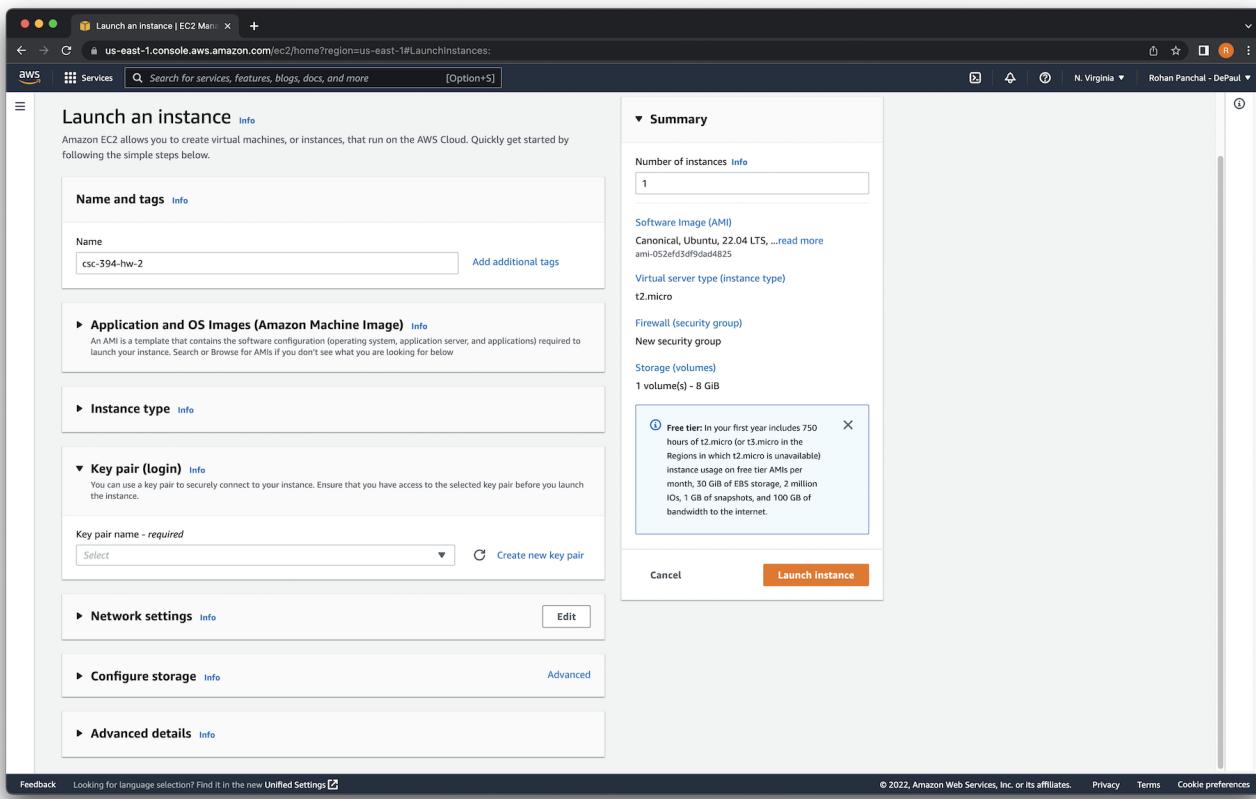
Network settings

Configure storage

Advanced details

Launch Instance

Then select the option to create a new key pair in the Key pair (login) section.



Enter a key pair name

Make sure the key pair type is RSA

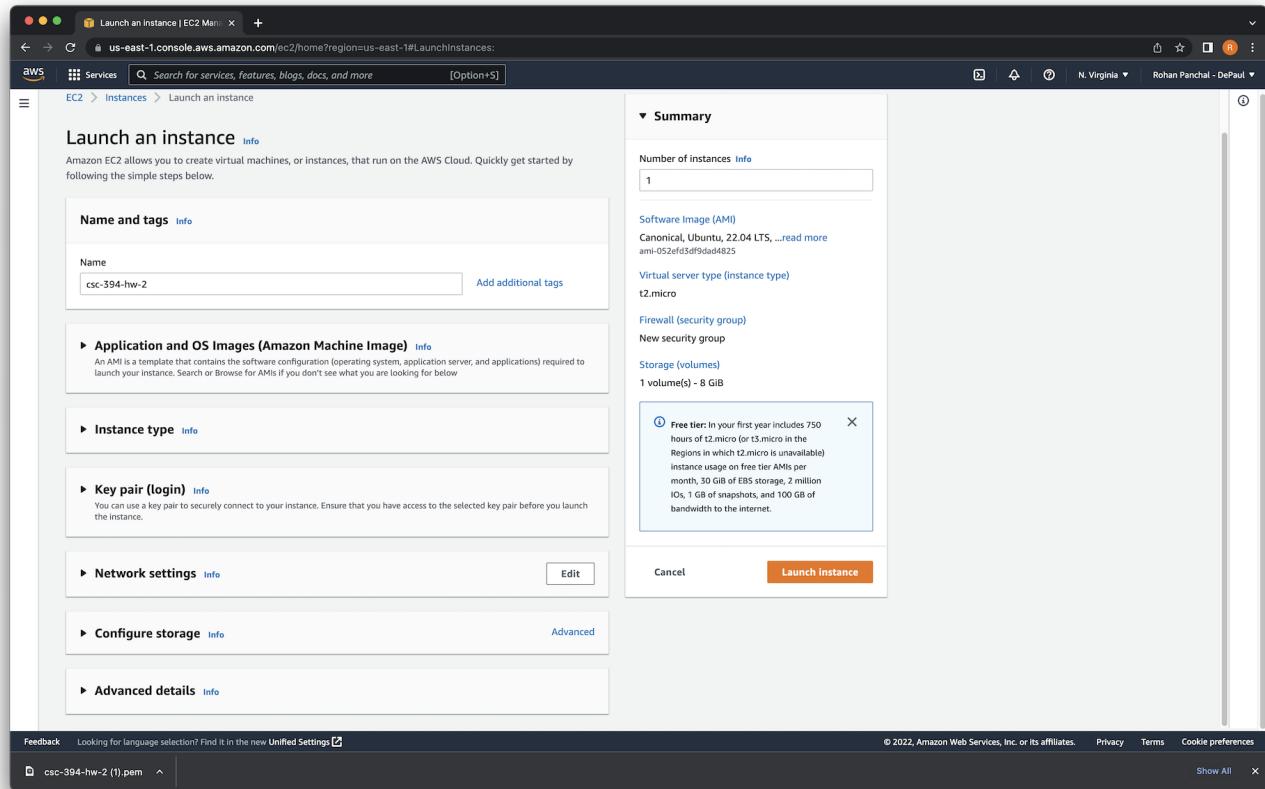
And the private key file format should either be .pem or .ppk if you are using PuTTY

Click Create key pair and it will download the .pem file for you.

The screenshot shows the 'Launch an instance' wizard on the AWS Management Console. The current step is 'Create key pair'. A modal window titled 'Create key pair' is open, asking for a 'Key pair name' (set to 'csc-394-hw-2') and a 'Key pair type' (set to 'RSA'). Below the modal, there are sections for 'Network settings', 'Configure storage', and 'Advanced details'.

In the Network settings opt into the defaults as shown here:

The screenshot shows the 'Launch an instance' wizard on the AWS Management Console, specifically the 'Network settings' step. A modal window titled 'Free tier' is open, explaining that the first year includes 750 hours of t2.micro usage. It lists included benefits: Canonical, Ubuntu, 22.04 LTS AMI, t2.micro instance type, New security group, and 1 volume(s) - 8 GiB storage. The 'Launch instance' button is visible at the bottom of the modal.



Leave the Configure storage and Advanced details sections to their default options. And then click the Launch instance button on the right sidebar.

The screenshot shows the AWS EC2 "Launch an instance" page after a successful launch. The main message says "Successfully initiated launch of instance (i-03ba7655d4b96ba28)". Below it is a "Launch log" table:

Initializing requests	Succeeded
Creating security groups	Succeeded
Creating security group rules	Succeeded
Launch initiation	Succeeded

Next Steps

- Get notified of estimated charges**: Create billing alerts to get an email notification when estimated charges on your AWS bill exceed an amount you define (for example, if you exceed the free usage tier).
- How to connect to your instance**: Your instance is launching and it might be a few minutes until it is in the running state, when it will be ready for you to use. Click View Instances to monitor your instance's status. Once your instance is in the 'running' state, you can connect to it from the Instances screen. Find out how to connect to your instance.

[View more resources to get you started](#) [View all instances](#)

Click on the instance number to navigate to the EC2 Instance List

The screenshot shows the same AWS EC2 "Launch an instance" page after a successful launch. A red arrow points to the instance ID "i-03ba7655d4b96ba28" in the "Successfully initiated launch of instance" message.

Click the Instance ID to navigate to the EC2 Instance Dashboard

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with various navigation options like EC2 Dashboard, Events, Tags, Limits, Instances (with 'Instances New' selected), Images, Elastic Block Store, Network & Security, and more. The main area displays a table of instances. One instance, 'csc-394-hw-2' (Instance ID: i-03ba7655d4b968a28), is highlighted with an orange arrow pointing to its 'Status check' column which says 'Running'. Other columns include Name, Instance ID, Instance state, Instance type (t2.micro), Status check, Alarm status, Availability Zone (us-east-1a), Public IPv4 DNS (ec2-54-89-254-124.compute-1.amazonaws.com), and Public IPv4 IP (54.89.254.124). At the top right, there are buttons for 'Connect', 'Actions', and 'Launch instances'.

The screenshot shows the AWS EC2 Instance details page for the instance 'csc-394-hw-2'. The left sidebar is identical to the previous screenshot. The main content is divided into sections: 'Instance summary' (with a note 'Updated less than a minute ago'), 'Details' tab (selected), 'Security', 'Networking', 'Storage', 'Status checks', 'Monitoring', and 'Tags'. In the 'Details' tab, there are two main sections: 'Instance details' (with a 'Info' button) and 'Monitoring'. The 'Instance details' section contains fields like Instance ID (i-03ba7655d4b968a28), Public IPv4 address (54.89.254.124), Instance state (Running), and VPC ID (vpc-06e4fadd14c1e3ef5). The 'Monitoring' section shows that monitoring is disabled. Other tabs like 'Security' and 'Networking' also provide specific details about the instance's configuration.

Success! You have an EC2 instance setup!

Updating EC2 security

Back in your EC2 instance dashboard, click on the security tab, and click on the Security Group instance ID

The screenshot shows the AWS EC2 Instance Details page for an instance with ID i-03ba7655d4b968a28. The Security tab is selected. In the Inbound rules section, there is a table with one rule:

Security group rule ID	Port range	Protocol	Source	Security groups
sgr-0d65cd4bbaaf4d6c2	22	TCP	0.0.0.0/0	launch-wizard-1

A red arrow points to the 'Edit inbound rules' button in the Inbound rules section.

Click on the Edit inbound rules button

EC2 Management Console

Search for services, features, blogs, docs, and more [Option+S]

New EC2 Experience Tell us what you think

EC2 > Security Groups > sg-044b46f8b0bbe45eb - launch-wizard-1

sg-044b46f8b0bbe45eb - launch-wizard-1

Details

Security group name	Security group ID	Description	VPC ID
launch-wizard-1	sg-044b46f8b0bbe45eb	launch-wizard-1 created 2022-09-05T20:57:05.439Z	vpc-06e4fadd14c1e3ef5
Owner	Inbound rules count	Outbound rules count	
379100231278	1 Permission entry	1 Permission entry	

Inbound rules Outbound rules Tags

You can now check network connectivity with Reachability Analyzer Run Reachability Analyzer

Inbound rules (1/1)

Name	Security group rule...	IP version	Type	Protocol	Port range	Source
-	sgr-0d65cd4bbaaf4d6c2	IPv4	SSH	TCP	22	0.0.0.0/0

Manage tags Edit inbound rules

Feedback Looking for language selection? Find it in the new Unified Settings ?

© 2022, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

EC2 Management Console

Search for services, features, blogs, docs, and more [Option+S]

EC2 > Security Groups > sg-044b46f8b0bbe45eb - launch-wizard-1 > Edit inbound rules

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-0d65cd4bbaaf4d6c2	SSH	TCP	22	Custom	0.0.0.0/0

Add rule Cancel Preview changes Save rules

Feedback Looking for language selection? Find it in the new Unified Settings ?

© 2022, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Click Add Rule

In the new line item, click the Type dropdown and select HTTP

The screenshot shows the AWS EC2 Management Console with the URL us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#ModifyInboundSecurityGroupRules:securityGroupId=sg-044b46f8b0bbe45eb. The page title is "Edit inbound rules". The "Type" dropdown for the first rule (SSH) is open, showing "HTTP" as the selected option. Other options include "Custom TCP", "Custom UDP", "Custom ICMP - IPv4", "Custom Protocol", "All TCP", "All UDP", "All ICMP - IPv4", "All ICMP - IPv6", "All traffic", "SSH", "SMTP", "DNS (UDP)", "DNS (TCP)", "HTTP", "POP3", "IMAP", "LDAP", and "HTTPS". The "Protocol" dropdown is set to "TCP", and the "Port range" dropdown is set to "22". The "Source" dropdown is set to "Custom" and contains the entry "0.0.0.0/0". The "Description" field is empty. At the bottom right, there are "Cancel", "Preview changes", and "Save rules" buttons.

In the dropdown next to the Source column, select the 0.0.0.0/0 entry.

Click Save Rules

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules	Type	Protocol	Port range	Source	Description - optional
sgr-0d65cd4bbaf4d6c2	SSH	TCP	22	Custom	<input type="text"/> 0.0.0.0/0 <input type="button" value="Delete"/>
-	HTTP	TCP	80	Custom	<input type="text"/> 0.0.0.0/0 <input type="button" value="Delete"/>

Add rule

CIDR blocks: 0.0.0.0/0, 0.0.0.8, 0.0.0.16, 0.0.0.24, 0.0.0.32, ::/0, ::/16, ::/32, ::/48, ::/64

Security Groups: default | sg-04c35a9eb9010c62, launch-wizard-1 | sg-044b4f68b0bbe45eb

Prefix lists: com.amazonaws.us-eas... | pl-072cd2e6h

Cancel Preview changes Save rules

Navigate back to the EC2 instance dashboard and copy the Public IPv4 address

Instance summary for i-03ba7655d4b968a28 (csc-394-hw-2)

Public IPv4 address: 54.89.254.124 [open address]

Instance state: Running

Private IP DNS name (IPv4 only): ip-172-31-84-201.ec2.internal

Instance type: t2.micro

VPC ID: vpc-06e4fadd14c1e3ef5

Subnet ID: subnet-0e7420ead0405a506

AMI ID: ami-052efd9dad4825

AMI name: ubuntu/images/hvm-ssd/ubuntu-jammy-22.04-amd64-server-20220609

Launch time: Mon Sep 05 2022 16:10:42 GMT-0500 (Central Daylight Time) (about 1 hour)

Lifecycle: normal

Key pair name: csc-394-hw-2

Kernel ID:

Monitoring: disabled

Termination protection: Disabled

AMI location: amazon/ubuntu/images/hvm-ssd/ubuntu-jammy-22.04-amd64-server-20220609

Stop-hibernate behavior: disabled

State transition reason: -

State transition message: -

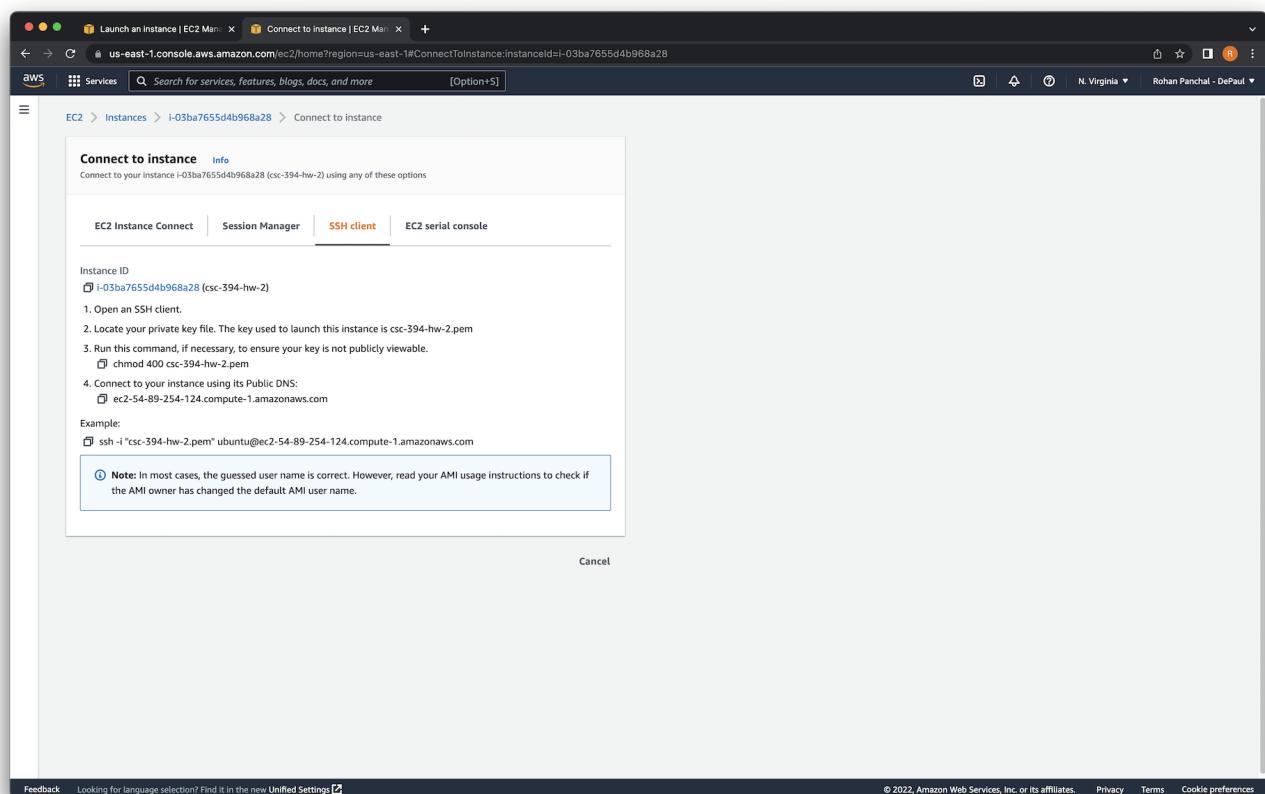
Step 2 – Deploy your site on an EC2 instance

Click the `Connect` button on the top right of the screen

You now see the `Connect to instance` selector

Click on the `SSH client` selector

And then copy the `example` at the bottom.



My ssh connection command was:

```
ssh -i "csc-394-hw-project.pem" ubuntu@ec2-54-89-254-124.compute-1.amazonaws.com
```

Yours will have a different ip address and the location of your `.pem` file may be at a different directory (mine was in my `Downloads` directory). But with this command you

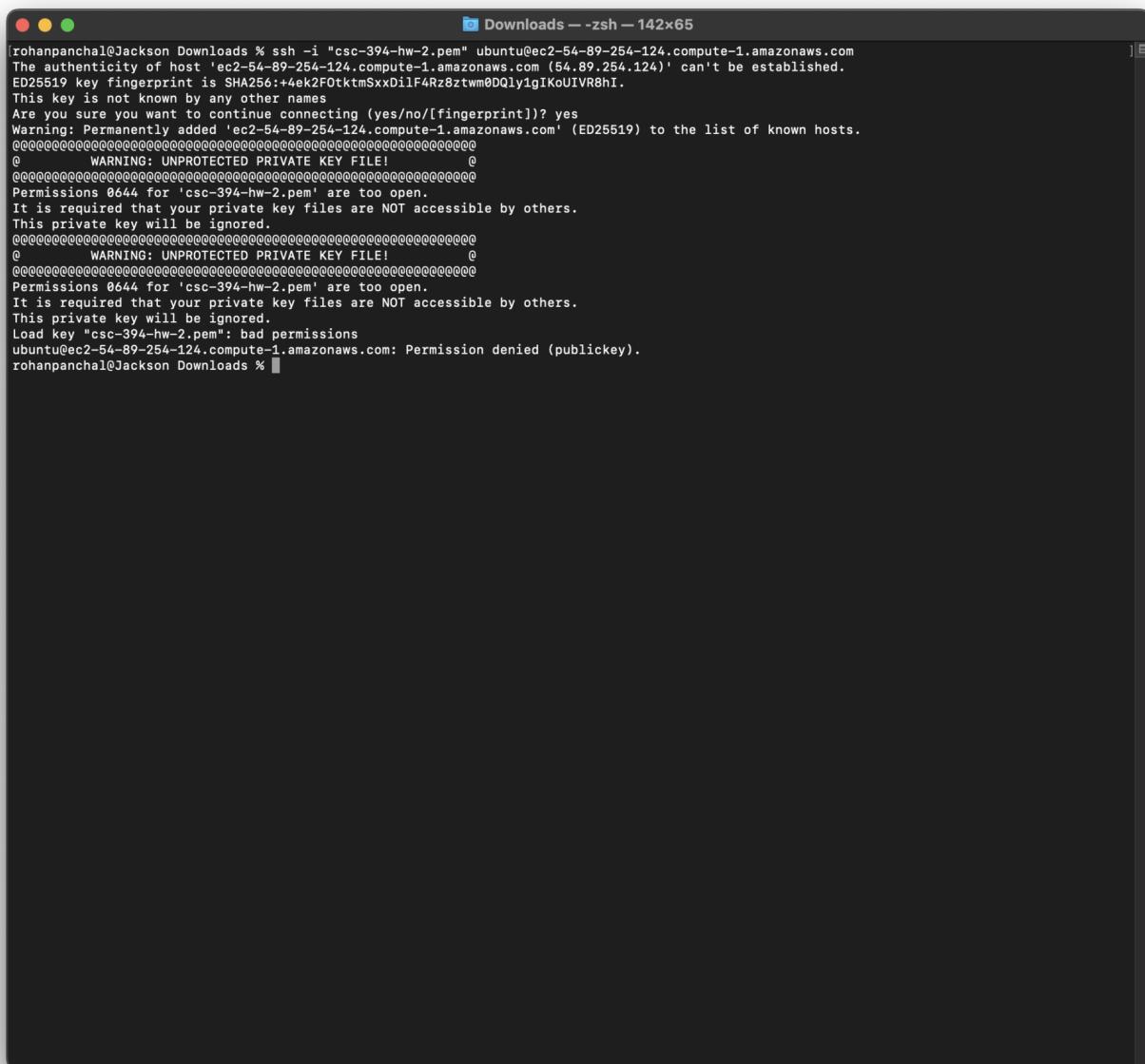
can launch a terminal or command prompt and connect to your linux instance!

Connect to your instance

In your terminal paste the command to connect via SSH to your EC2 instance.

Note:

If you run into a WARNING as follows:



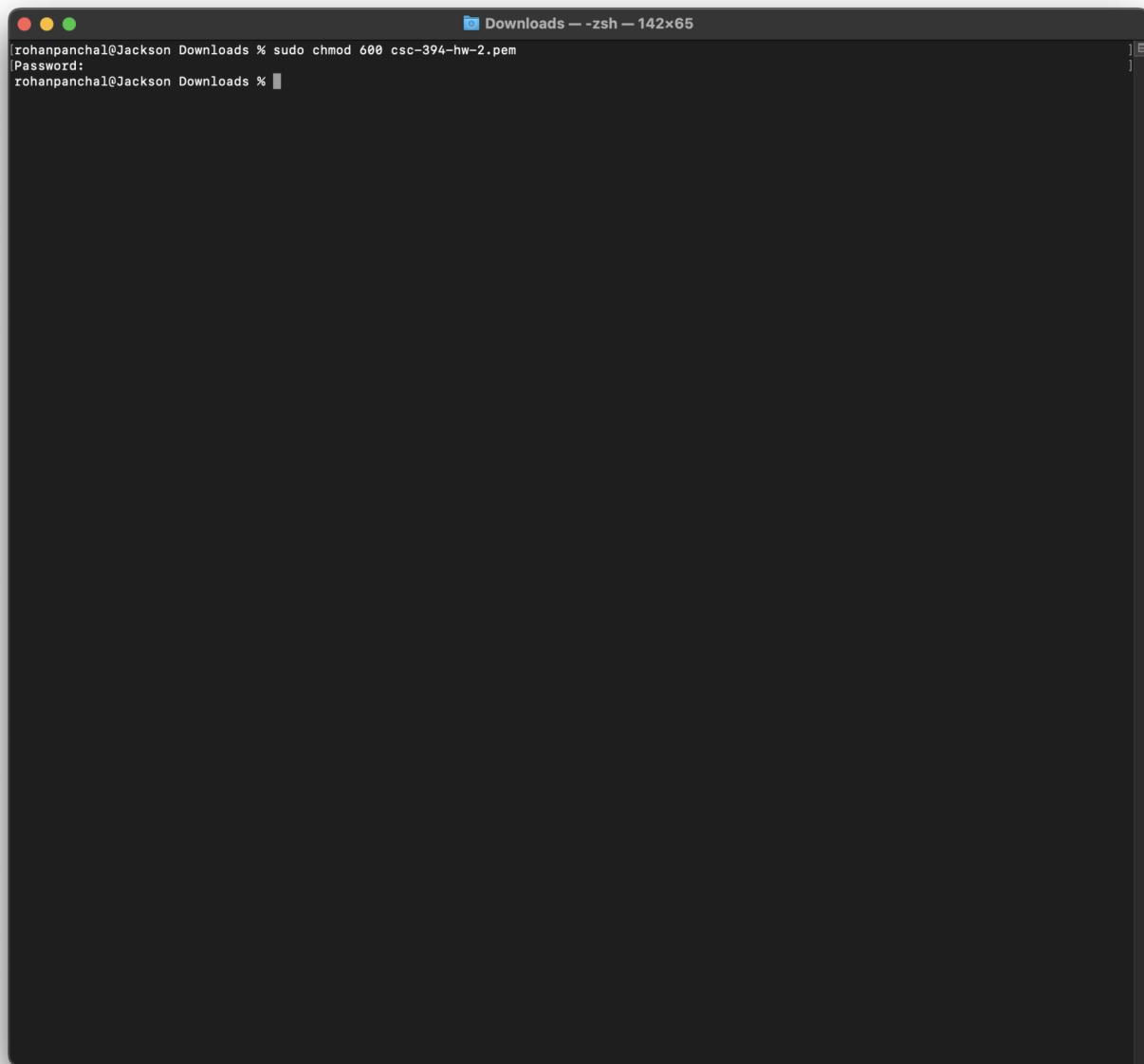
The screenshot shows a terminal window titled "Downloads -- zsh -- 142x65". The window contains the following text output from an SSH session:

```
rohanpanchal@Jackson Downloads % ssh -i "csc-394-hw-2.pem" ubuntu@ec2-54-89-254-124.compute-1.amazonaws.com
The authenticity of host 'ec2-54-89-254-124.compute-1.amazonaws.com (54.89.254.124)' can't be established.
ED25519 key fingerprint is SHA256:+4ek2F0tktmSxxDilF4Rz8ztwm0DQly1gIKoUVR8hI.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-89-254-124.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
@@@@@@@WARNING: UNPROTECTED PRIVATE KEY FILE!@@@@@@@
Permissions 0644 for 'csc-394-hw-2.pem' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
@@@@@@@WARNING: UNPROTECTED PRIVATE KEY FILE!@@@@@@@
Permissions 0644 for 'csc-394-hw-2.pem' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
Load key "csc-394-hw-2.pem": bad permissions
ubuntu@ec2-54-89-254-124.compute-1.amazonaws.com: Permission denied (publickey).
rohanpanchal@Jackson Downloads %
```

You need to modify the permissions of your .pem file.

Execute the following command

```
sudo chmod 400 csc-394-hw-2.pem
```



```
[rohanpanchal@Jackson Downloads % sudo chmod 600 csc-394-hw-2.pem
>Password:
rohanpanchal@Jackson Downloads % ]
```

Reattempt connecting to your ssh instance and you should see the following:

The screenshot shows a terminal window with the following content:

```
[rohanpanchal@Jackson Downloads % ssh -i "csc-394-hw.pem" ubuntu@ec2-100-26-246-201.compute-1.amazonaws.com - 137...]
[rohanpanchal@Jackson Downloads % ssh -i "csc-394-hw.pem" ubuntu@ec2-100-26-246-201.compute-1.amazonaws.com
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-1026-aws x86_64)

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

System information as of Thu Jan 12 01:16:49 UTC 2023

System load: 0.0      Processes:         98
Usage of /: 28.4% of 7.57GB   Users logged in:     0
Memory usage: 26%          IPv4 address for eth0: 172.31.59.179
Swap usage:  0%

* Ubuntu Pro delivers the most comprehensive open source security and
  compliance features.

  https://ubuntu.com/aws/pro

15 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

*** System restart required ***
Last login: Tue Jan 10 22:04:45 2023 from 24.14.102.181
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-59-179:~$ ]
```

Update your EC2 instance's tooling

Execute the following command:

```
sudo apt-get update && sudo apt-get upgrade
```

```
[ubuntu@ip-172-31-84-201: ~ -- ssh -i csc-394-hw-2.pem ubuntu@ec2-54-89-254-124.compute-1.amazonaws.com -- 142x65
[ubuntu@ip-172-31-84-201: ~ -- ssh -i csc-394-hw-2.pem ubuntu@ec2-54-89-254-124.compute-1.amazonaws.com -- 142x65
ubuntu@ip-172-31-84-201:~$ sudo apt-get update && sudo apt-get upgrade
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease [114 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease [99.8 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 Packages [14.1 MB]
Get:5 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe Translation-en [5652 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 c-n-f Metadata [286 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 Packages [217 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse Translation-en [112 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 c-n-f Metadata [8372 B]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [542 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [128 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 c-n-f Metadata [8108 B]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [306 kB]
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted Translation-en [47.5 kB]
Get:16 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 c-n-f Metadata [524 B]
Get:17 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [254 kB]
Get:18 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe Translation-en [83.1 kB]
Get:19 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 c-n-f Metadata [4404 B]
Get:20 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 Packages [7000 B]
Get:21 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse Translation-en [2264 B]
Get:22 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 c-n-f Metadata [420 B]
Get:23 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/main amd64 Packages [3008 B]
Get:24 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/main Translation-en [1432 B]
Get:25 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/main amd64 c-n-f Metadata [272 B]
Get:26 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/restricted amd64 c-n-f Metadata [1116 B]
Get:27 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/universe amd64 Packages [6724 B]
Get:28 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/universe Translation-en [9216 B]
Get:29 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/universe amd64 c-n-f Metadata [352 B]
Get:30 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/multiverse amd64 c-n-f Metadata [116 B]
Get:31 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [295 kB]
Get:32 http://security.ubuntu.com/ubuntu jammy-security/main Translation-en [68.9 kB]
Get:33 http://security.ubuntu.com/ubuntu jammy-security/main amd64 c-n-f Metadata [3924 B]
Get:34 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 Packages [274 kB]
Get:35 http://security.ubuntu.com/ubuntu jammy-security/restricted Translation-en [42.3 kB]
Get:36 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 c-n-f Metadata [524 B]
Get:37 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 Packages [121 kB]
Get:38 http://security.ubuntu.com/ubuntu jammy-security/universe Translation-en [41.5 kB]
Get:39 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 c-n-f Metadata [2408 B]
Get:40 http://security.ubuntu.com/ubuntu jammy-security/multiverse amd64 Packages [4192 B]
Get:41 http://security.ubuntu.com/ubuntu jammy-security/multiverse Translation-en [900 B]
Get:42 http://security.ubuntu.com/ubuntu jammy-security/multiverse amd64 c-n-f Metadata [228 B]
Fetched 22.9 MB in 4s (6204 kB/s)
Reading package lists... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following packages have been kept back:
  linux-aws linux-headers-aws linux-image-aws
The following packages will be upgraded:
  apparmor apt base-files cloud-init cryptsetup cryptsetup-bin cryptsetup-initramfs curl dirmngr git git-man gnupg gnupg-l10n
  gnupg-utils gpg gpg-agent gpg-wks-client gpg-wks-server gpgconf gpgsm gpgv intel-microcode isc-dhcp-client isc-dhcp-common libapparmor1
  libapt-pkg6.0 libc-bin libc6 libcryptsetup12 libcurl3-gnutls libcurl4 libfreetype6 libgnutls30 libgstreamer1.0-0 libldap-2.5-0
  libldap-common libnetplan0 libnftables1 libnss-systemd libnss3 libpam-systemd libpython3.10 libpython3.10-minimal libpython3.10-stdlib
  libssl3 libsystemd0 libtirpc-common libtirpc3 libudevi1 libxlst1.1 locales motd-news-config netplan.io nftables open-vm-tools openssl
  python-apt-common python3-apt python3-distupgrade python3-gi python3-jwt python3-software-properties python3-twisted python3.10
  python3.10-minimal snapd software-properties-common systemctl systemd-sysv tzdata ubuntu-advantage-tools ubuntu-release-upgrader-core udev
74 upgraded, 0 newly installed, 0 to remove and 3 not upgraded.
34 standard security updates
Need to get 72.4 MB of archives.
After this operation, 5456 kB of additional disk space will be used.
Do you want to continue? [Y/n] ■
```

This may take a few seconds. At some point you will see a prompt to continue. Type Y and hit enter.

```

Downloads — ubuntu@ip-172-31-84-201: ~ — ssh -i csc-394-hw-2.pem ubuntu@ec2-54-89-254-124.compute-1.amazonaws.com — 142x65
Setting up gpgsm (2.2.27-3ubuntu2.1) ...
Setting up libcurl3-gnutls:amd64 (7.81.0-1ubuntu1.4) ...
Setting up systemd (249.11-0ubuntu3.4) ...
Setting up python3-distupgrade (1:22.04.13) ...
Setting up python3.10-minimal (3.10.4-3ubuntu0.1) ...
Setting up netplan.io (0.104-0ubuntu2.1) ...
Setting up libpython3.10-stdlib:amd64 (3.10.4-3ubuntu0.1) ...
Setting up dirmngr (2.2.27-3ubuntu2.1) ...
Setting up python3-software-properties (0.99.22.3) ...
Setting up ubuntu-release-upgrader-core (1:22.04.13) ...
Setting up git (1:2.34.1-1ubuntu1.4) ...
Setting up gpg-wks-server (2.2.27-3ubuntu2.1) ...
Setting up ubuntu-advantage-tools (27.10.1-22.04.1) ...
Installing new version of config file /etc/ubuntu-advantage/uaclient.conf ...
Created symlink /etc/systemd/system/multi-user.target.wants/ubuntu-advantage.service → /lib/systemd/system/ubuntu-advantage.service.
Setting up cryptsetup-initramfs (2:2.4.3-1ubuntu1.1) ...
update-initramfs: deferring update (trigger activated)
Setting up snapd (2.56.2+22.04ubuntu1) ...
snapd.failure.service is a disabled or a static unit not running, not starting it.
snapd.snap-repair.service is a disabled or a static unit not running, not starting it.
Setting up libpython3.10:amd64 (3.10.4-3ubuntu0.1) ...
Setting up systemd-sysv (249.11-0ubuntu3.4) ...
Setting up cloud-init (22.2-0ubuntu1~22.04.3) ...
Setting up python3.10 (3.10.4-3ubuntu0.1) ...
Setting up gpg-wks-client (2.2.27-3ubuntu2.1) ...
Setting up libnss-systemd:amd64 (249.11-0ubuntu3.4) ...
Setting up software-properties-common (0.99.22.3) ...
Setting up gnupg (2.2.27-3ubuntu2.1) ...
Setting up libpam-systemd:amd64 (249.11-0ubuntu3.4) ...
Processing triggers for initramfs-tools (0.140ubuntu13) ...
update-initramfs: Generating /boot/initrd.img-5.15.0-1011-aws
Processing triggers for libc-bin (2.35-0ubuntu3.1) ...
Processing triggers for rsyslog (8.21.12.0-2ubuntu2.2) ...
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for plymouth-theme-ubuntu-text (0.9.5+git20211018-1ubuntu3) ...
update-initramfs: deferring update (trigger activated)
Processing triggers for microcode-initrd (2build1) ...
Processing triggers for dbus (1.12.20-2ubuntu4) ...
Processing triggers for install-info (6.8-4build1) ...
Processing triggers for initramfs-tools (0.140ubuntu13) ...
update-initramfs: Generating /boot/initrd.img-5.15.0-1011-aws
Scanning processes...
Scanning candidates...
Scanning linux images...

Running kernel seems to be up-to-date.

Restarting services...
systemctl restart acpid.service chrony.service cron.service multipathd.service packagekit.service polkit.service serial-getty@ttyS0.service s
sh.service udisks2.service
Service restarts being deferred:
systemctl restart ModemManager.service
/etc/needrestart/restart.d/dbus.service
systemctl restart getty@tty1.service
systemctl restart networkd-dispatcher.service
systemctl restart systemd-logind.service
systemctl restart unattended-upgrades.service
systemctl restart user@1000.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-84-201:~$ 
```

EC2 box updated

Install development packages

Execute the following command:

```
sudo apt-get install build-essential python3-dev libpq-dev
```

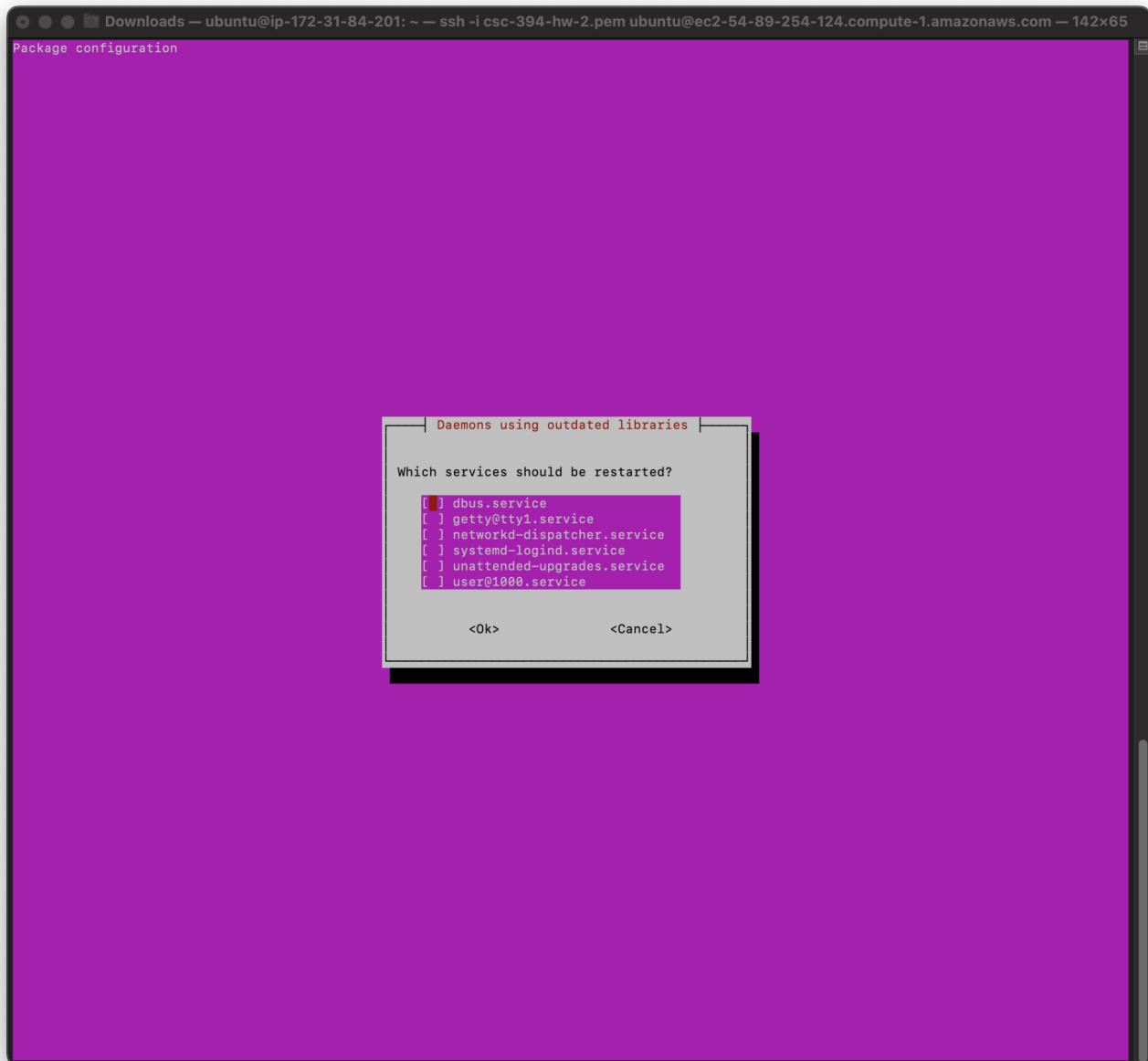
Install NGINX

Execute the following command:

```
sudo apt-get install nginx
```

This may take a few seconds. At some point you will see a prompt to continue. Type Y and hit enter.

You may see a large purple screen asking you to restart services on your EC2 box. Just hit Enter .



```

Downloads — ubuntu@ip-172-31-84-201: ~ - ssh -i csc-394-hw-2.pem ubuntu@ec2-54-89-254-124.compute-1.amazonaws.com — 142x65
Unpacking libnginx-mod-http-image-filter (1.18.0-6ubuntu14.1) ...
Selecting previously unselected package libnginx-mod-http-xslt-filter.
Preparing to unpack .../14-libnginx-mod-http-xslt-filter_1.18.0-6ubuntu14.1_amd64.deb ...
Unpacking libnginx-mod-http-xslt-filter (1.18.0-6ubuntu14.1) ...
Selecting previously unselected package libnginx-mod-mail.
Preparing to unpack .../15-libnginx-mod-mail_1.18.0-6ubuntu14.1_amd64.deb ...
Unpacking libnginx-mod-mail (1.18.0-6ubuntu14.1) ...
Selecting previously unselected package libnginx-mod-stream.
Preparing to unpack .../16-libnginx-mod-stream_1.18.0-6ubuntu14.1_amd64.deb ...
Unpacking libnginx-mod-stream (1.18.0-6ubuntu14.1) ...
Selecting previously unselected package libnginx-mod-stream-geoip2.
Preparing to unpack .../17-libnginx-mod-stream-geoip2_1.18.0-6ubuntu14.1_amd64.deb ...
Unpacking libnginx-mod-stream-geoip2 (1.18.0-6ubuntu14.1) ...
Selecting previously unselected package nginx-core.
Preparing to unpack .../18-nginx-core_1.18.0-6ubuntu14.1_amd64.deb ...
Unpacking nginx-core (1.18.0-6ubuntu14.1) ...
Selecting previously unselected package nginx.
Preparing to unpack .../19-nginx_1.18.0-6ubuntu14.1_amd64.deb ...
Unpacking nginx (1.18.0-6ubuntu14.1) ...
Setting up libxml2:amd64 (1:3.5.12-1build2) ...
Setting up libdeflate0:amd64 (1.10-2) ...
Setting up nginx-common (1.18.0-6ubuntu14.1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/nginx.service → /lib/systemd/system/nginx.service.
Setting up libjbig0:amd64 (2.1-3.1build3) ...
Setting up libjbig0:amd64 (1.18.0-6ubuntu14.1) ...
Setting up fonts-dejavu-core (2.37-2build1) ...
Setting up libjpeg-turbo0:amd64 (2.1.2-0ubuntu1) ...
Setting up libwebp7:amd64 (1.2.2-2) ...
Setting up libjpeg8:amd64 (8c-2ubuntu10) ...
Setting up libnginx-mod-mail (1.18.0-6ubuntu14.1) ...
Setting up fontconfig-config (2.13.1-4.2ubuntu5) ...
Setting up libnginx-mod-stream (1.18.0-6ubuntu14.1) ...
Setting up libtiff5:amd64 (4.3.0-6) ...
Setting up libfontconfig1:amd64 (2.13.1-4.2ubuntu5) ...
Setting up libnginx-mod-stream-geoip2 (1.18.0-6ubuntu14.1) ...
Setting up libgd3:amd64 (2.3.0-2ubuntu2) ...
Setting up libnginx-mod-http-image-filter (1.18.0-6ubuntu14.1) ...
Setting up nginx-core (1.18.0-6ubuntu14.1) ...
* Upgrading binary nginx
Setting up nginx (1.18.0-6ubuntu14.1) ...
Processing triggers for ufw (0.36.1-4build1) ...
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for libc-bin (2.35-0ubuntu3.1) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Running kernel seems to be up-to-date.

Restarting services...
Service restarts being deferred:
/etc/needrestart/restart.d/dbus.service
systemctl restart getty@tty1.service
systemctl restart networkd-dispatcher.service
systemctl restart systemd-logind.service
systemctl restart unattended-upgrades.service
systemctl restart user@1000.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-84-201: $ [OK]

```

NGINX installed!

Install PostgreSQL on your EC2 instance.

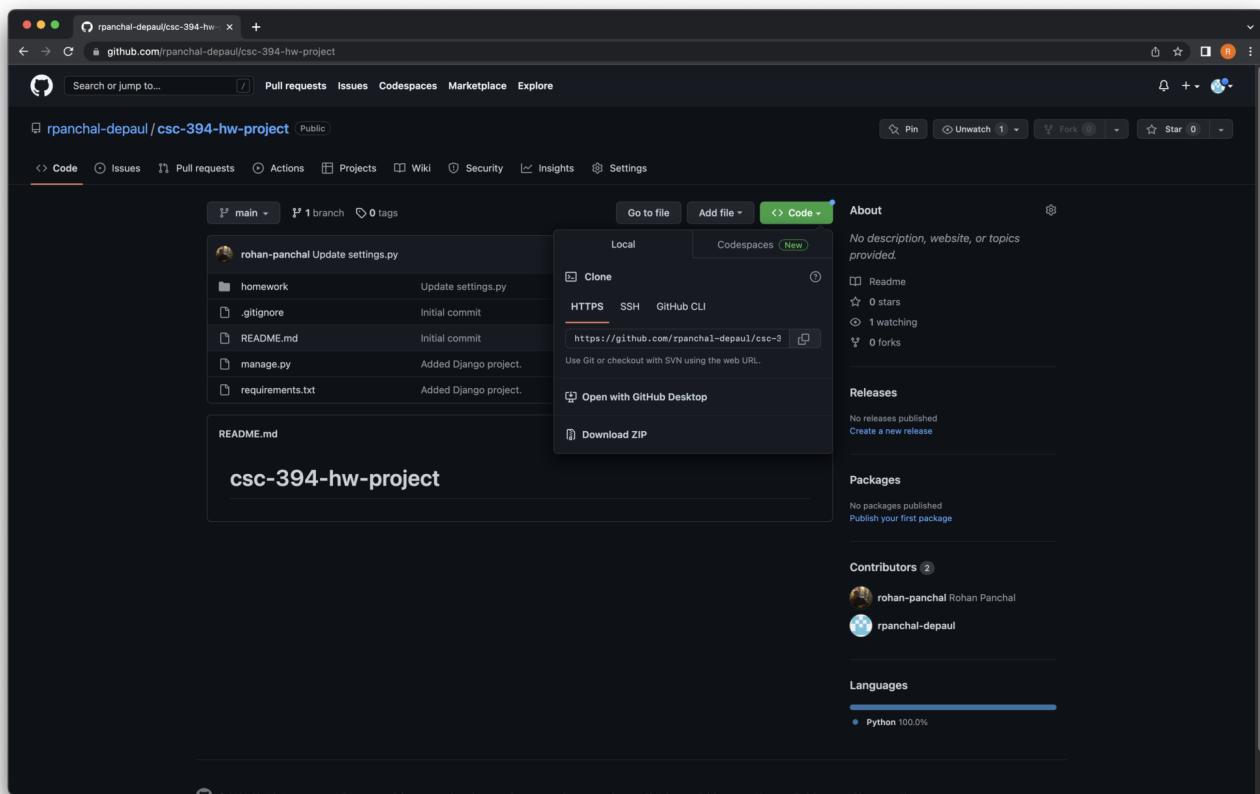
Execute the command:

```
sudo apt install postgresql postgresql-contrib libpq-dev
```

Clone your GitHub repo

Navigate to your GitHub repository you created for your django app.

Copy the url in the `Code` dropdown.



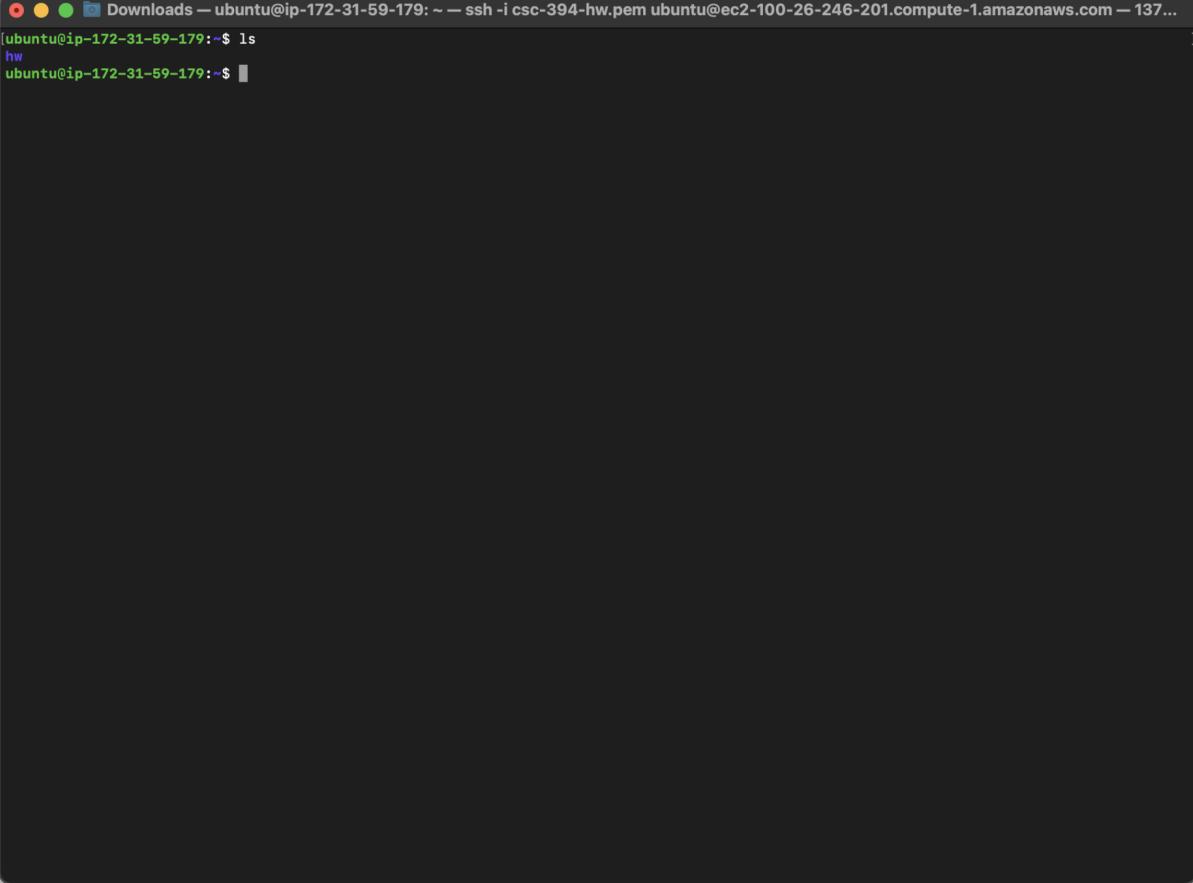
In your terminal window, execute the following command:

```
git clone {YOUR GITHUB REPO URL HERE} hw
```

Example:

```
git clone https://rpanchal-depaul/hw.git hw
```

If you execute the `ls` command you should now see the following:



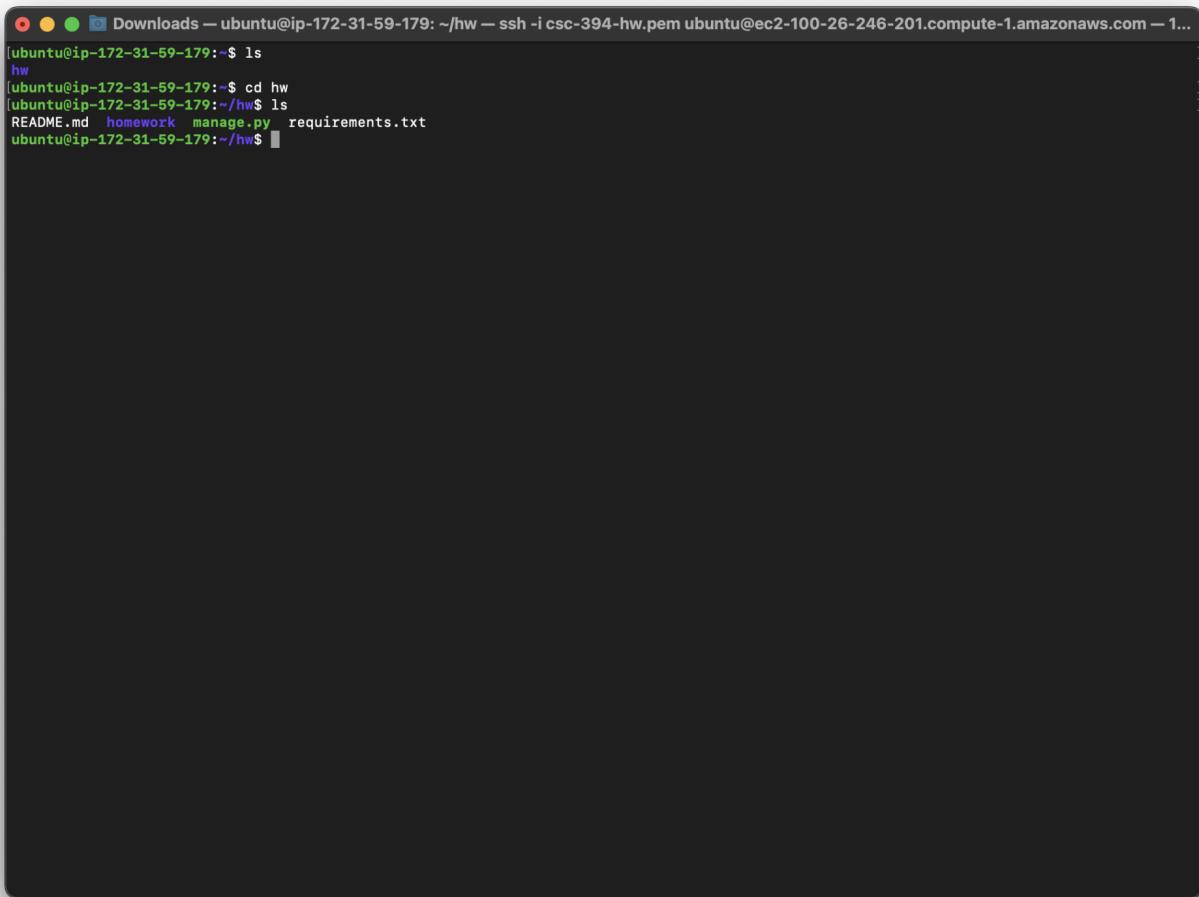
A screenshot of a terminal window titled "Downloads" on an Ubuntu system. The window shows the command "ls" being run, which returns a single file named "hw". The terminal has a dark background with light-colored text.

```
Downloads — ubuntu@ip-172-31-59-179: ~ — ssh -i csc-394-hw.pem ubuntu@ec2-100-26-246-201.compute-1.amazonaws.com — 137...
[ubuntu@ip-172-31-59-179:~]$ ls
hw
[ubuntu@ip-172-31-59-179:~]$ ]
```

If we execute the `cd` command (change directory) as follows:

```
cd hw
```

And execute the command `ls` :



A screenshot of a terminal window titled "Downloads" on an Ubuntu system. The terminal shows the command "ls" being run in the directory "/hw". The output lists several files: "hw", "homework", "manage.py", and "requirements.txt". The "hw" directory is also listed. The terminal prompt is "ubuntu@ip-172-31-59-179:~/hw\$".

We will see the contents of our project here now!

Step 3 – Setup Django on your EC2 Server.

Let's install virtualenv on our EC2 server.

Execute the following command:

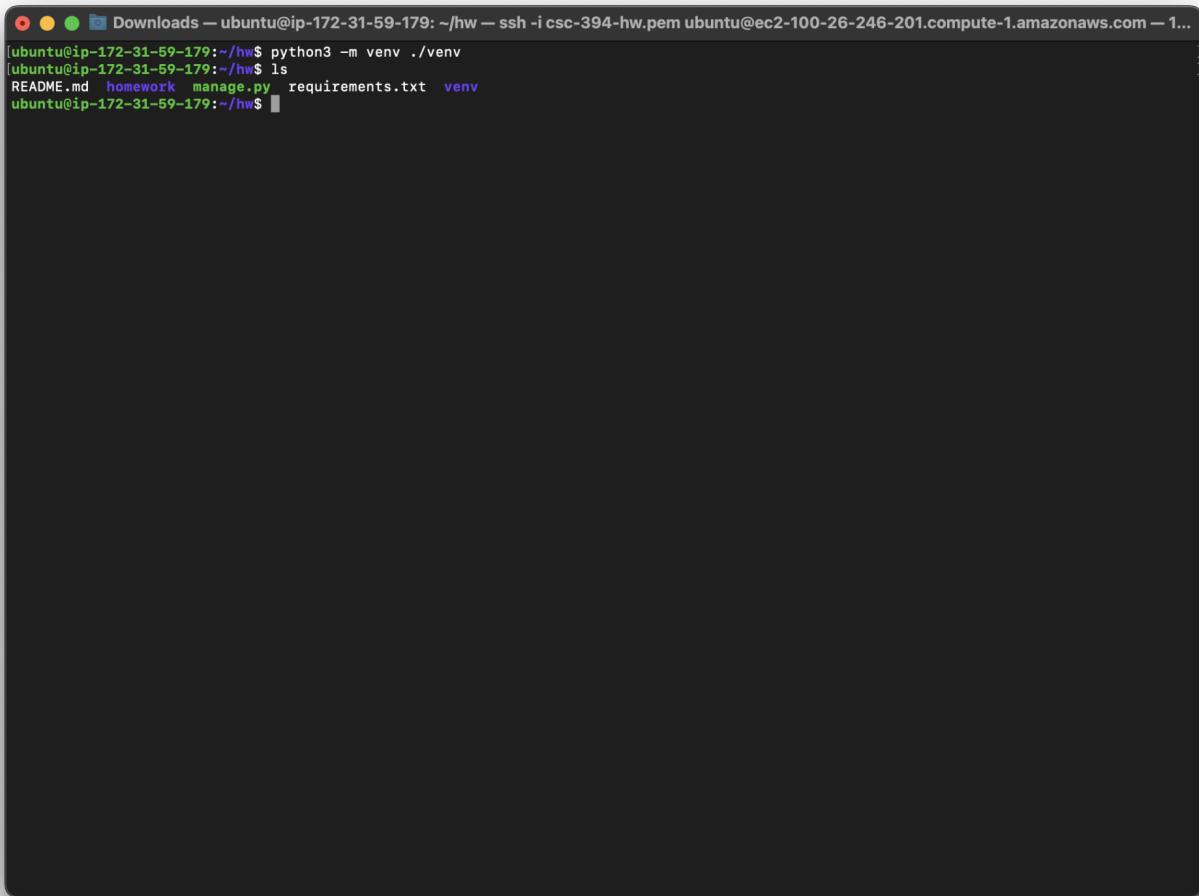
```
sudo apt install python3.10-venv
```

And then let's create a virtualenv .

Execute the following command:

```
python3 -m venv ./venv
```

We should now see the following directories:



A screenshot of a terminal window titled "Downloads" on an Ubuntu system. The terminal shows the command `python3 -m venv ./venv` being run, followed by an `ls` command which lists files: `README.md`, `homework`, `manage.py`, `requirements.txt`, and `venv`. The terminal prompt is `ubuntu@ip-172-31-59-179:~/hw$`.

Activate the `virtualenv` using the command:

```
source ./venv/bin/activate
```

And then install the requirements using the command:

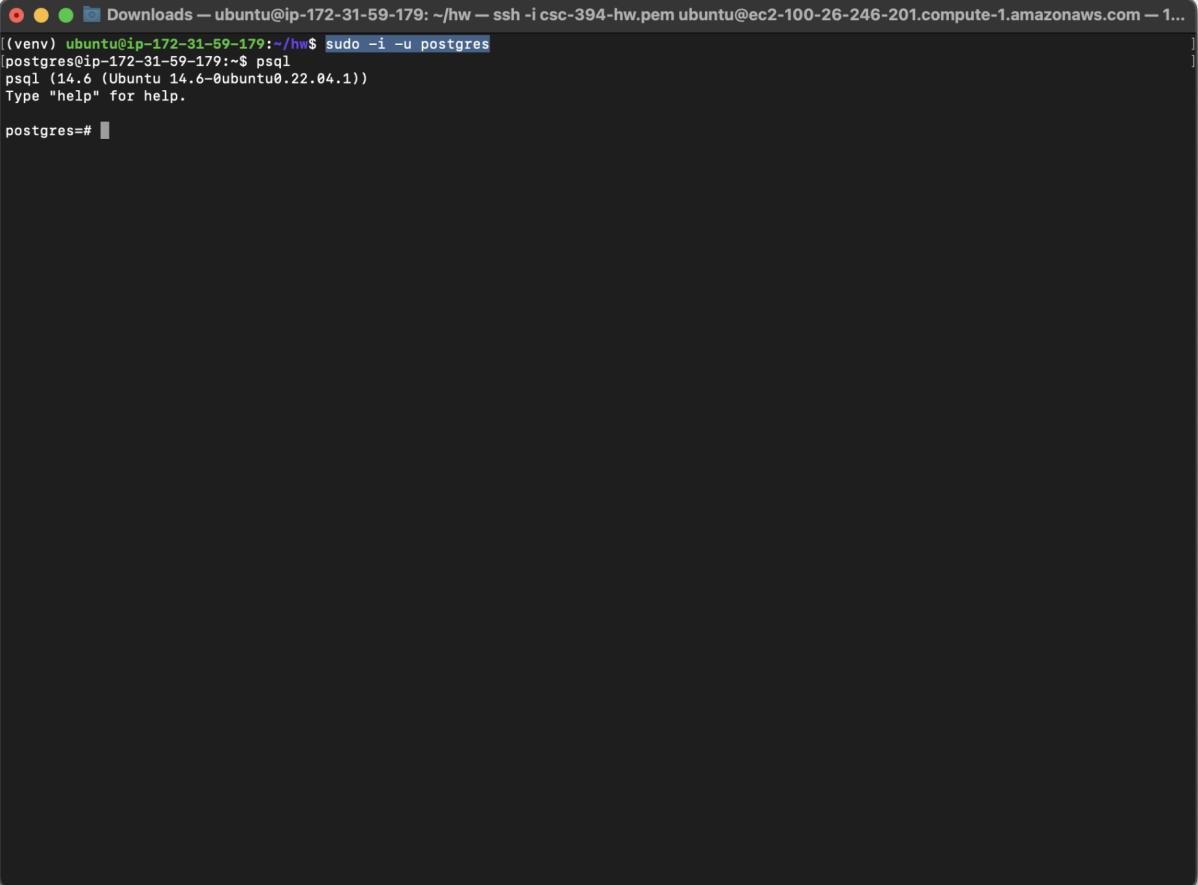
```
pip install -r requirements.txt
```

Setup postgres database on the server

Sign in to the postgres user on Ubuntu:

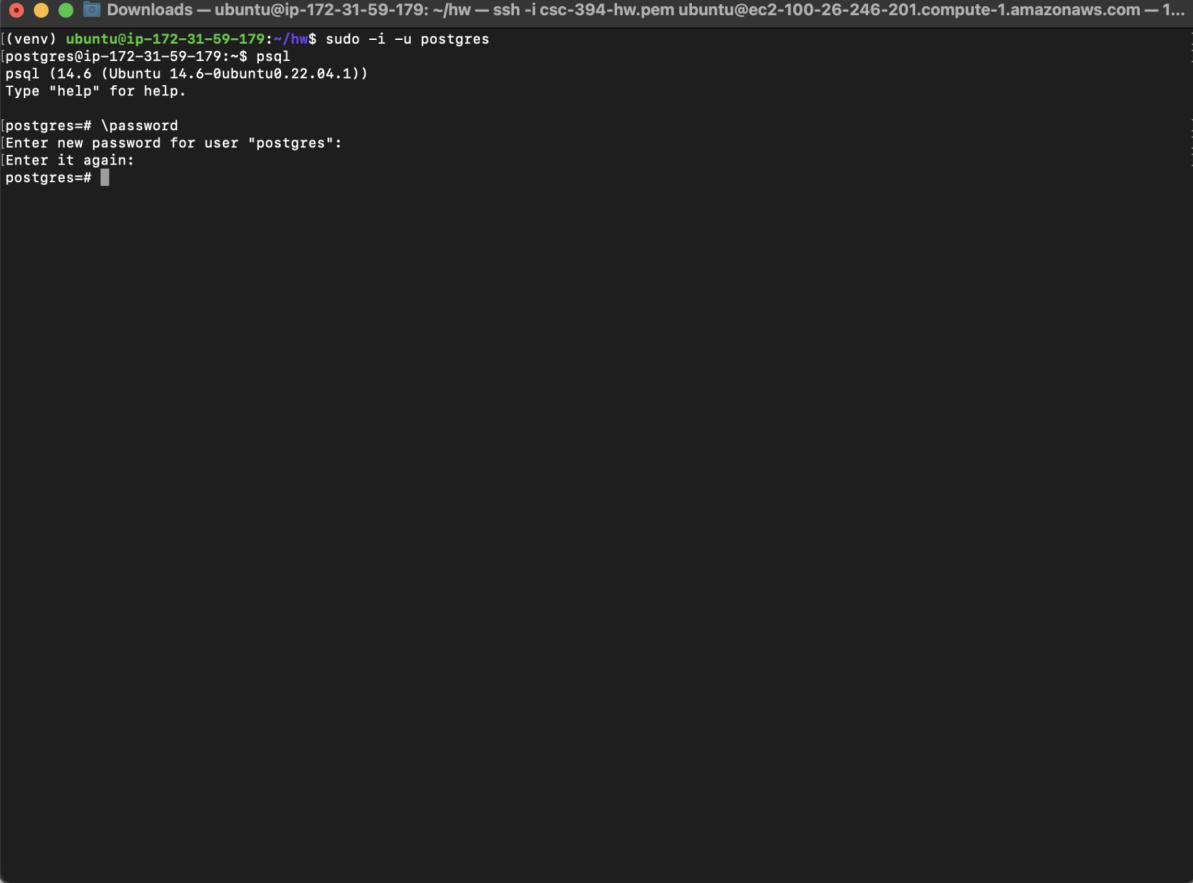
```
sudo -i -u postgres
```

Followed by executing the command: psql .



A screenshot of a terminal window titled "Downloads — ubuntu@ip-172-31-59-179: ~/hw — ssh -i csc-394-hw.pem ubuntu@ec2-100-26-246-201.compute-1.amazonaws.com — 1...". The window shows the command "sudo -i -u postgres" being run, followed by the PostgreSQL prompt "psql (14.6 (Ubuntu 14.6-0ubuntu0.22.04.1))". The prompt "Type "help" for help." is visible, and the command "postgres=#" is entered at the bottom.

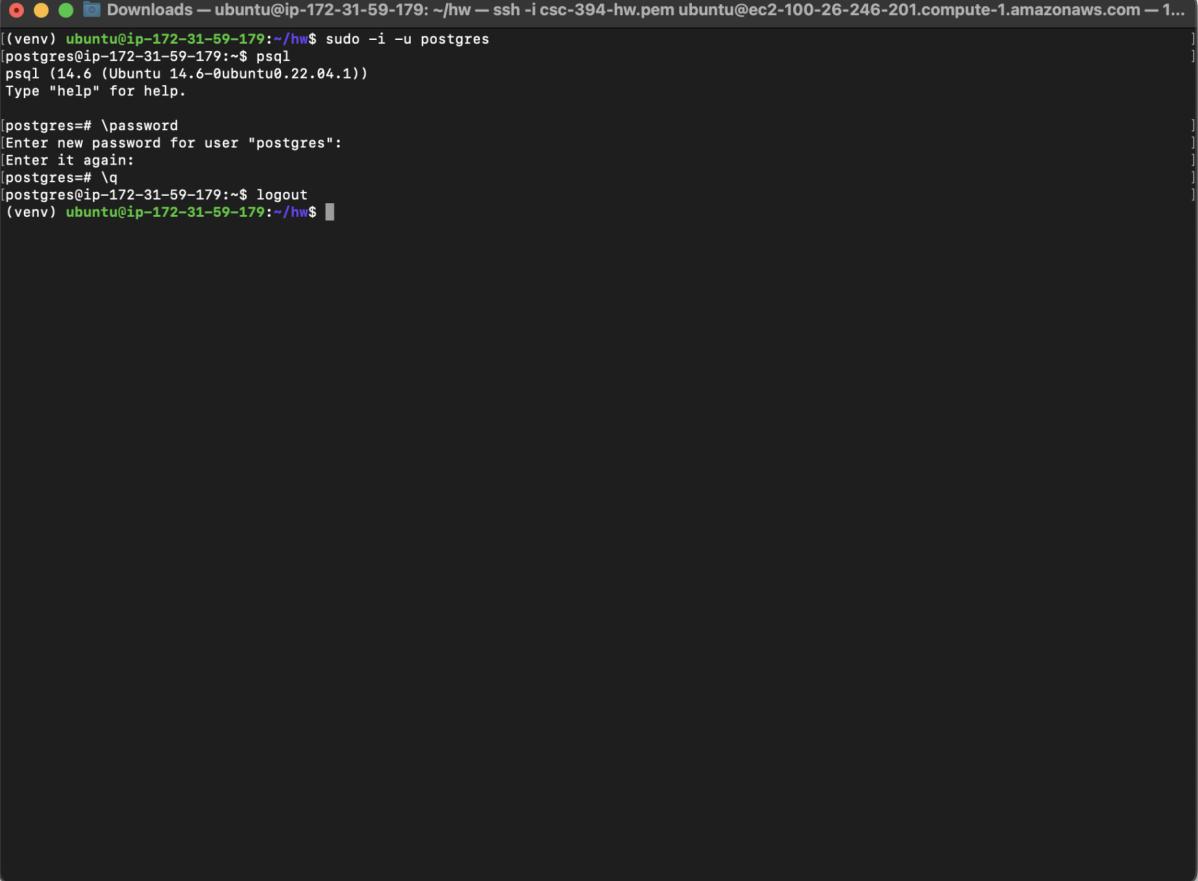
Execute the command \password and fill in values for new passwords.

A screenshot of a terminal window titled "Downloads — ubuntu@ip-172-31-59-179: ~/hw". The window shows a PostgreSQL prompt: "(venv) ubuntu@ip-172-31-59-179:~/hw\$ sudo -i -u postgres". It then enters the psql command: "psql (14.6 (Ubuntu 14.6-0ubuntu0.22.04.1))". A message says "Type "help" for help." followed by a blank line. The next line shows the command "\password" being entered. A password prompt follows: "(Enter new password for user "postgres":)". The user is prompted to "Enter it again:" and then the command "postgres=#" is shown again.

```
(venv) ubuntu@ip-172-31-59-179:~/hw$ sudo -i -u postgres
[postgres@ip-172-31-59-179:~$ psql
psql (14.6 (Ubuntu 14.6-0ubuntu0.22.04.1))
Type "help" for help.

[postgres=# \password
[Enter new password for user "postgres":
[Enter it again:
postgres=# ]]
```

Quit by executing the command: `\q` followed by `logout`

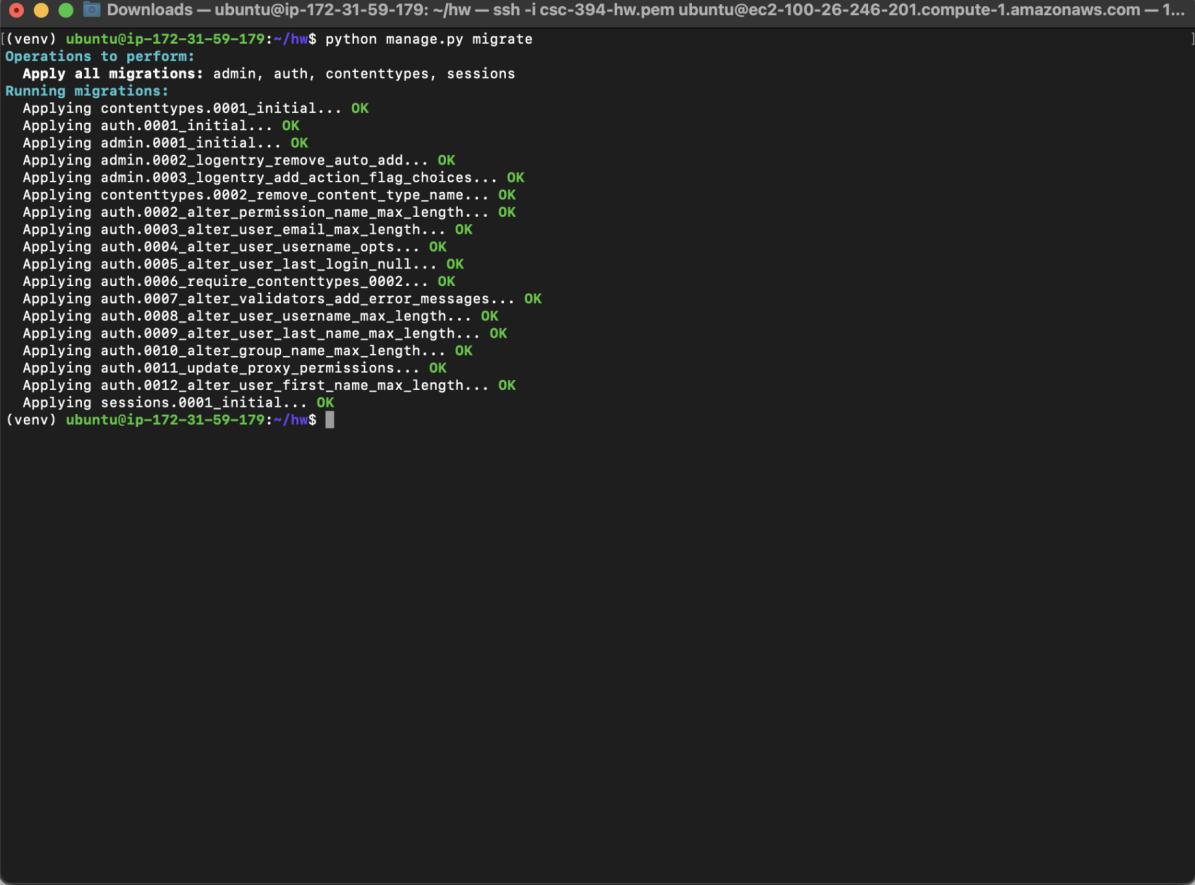


```
(venv) ubuntu@ip-172-31-59-179:~/hw$ sudo -i -u postgres
[postgres@ip-172-31-59-179:~$ psql
psql (14.6 (Ubuntu 14.6-0ubuntu0.22.04.1))
Type "help" for help.

[postgres=# \password
[Enter new password for user "postgres":
[Enter it again:
[postgres=# \q
[postgres@ip-172-31-59-179:~$ logout
(venv) ubuntu@ip-172-31-59-179:~/hw$ ]]
```

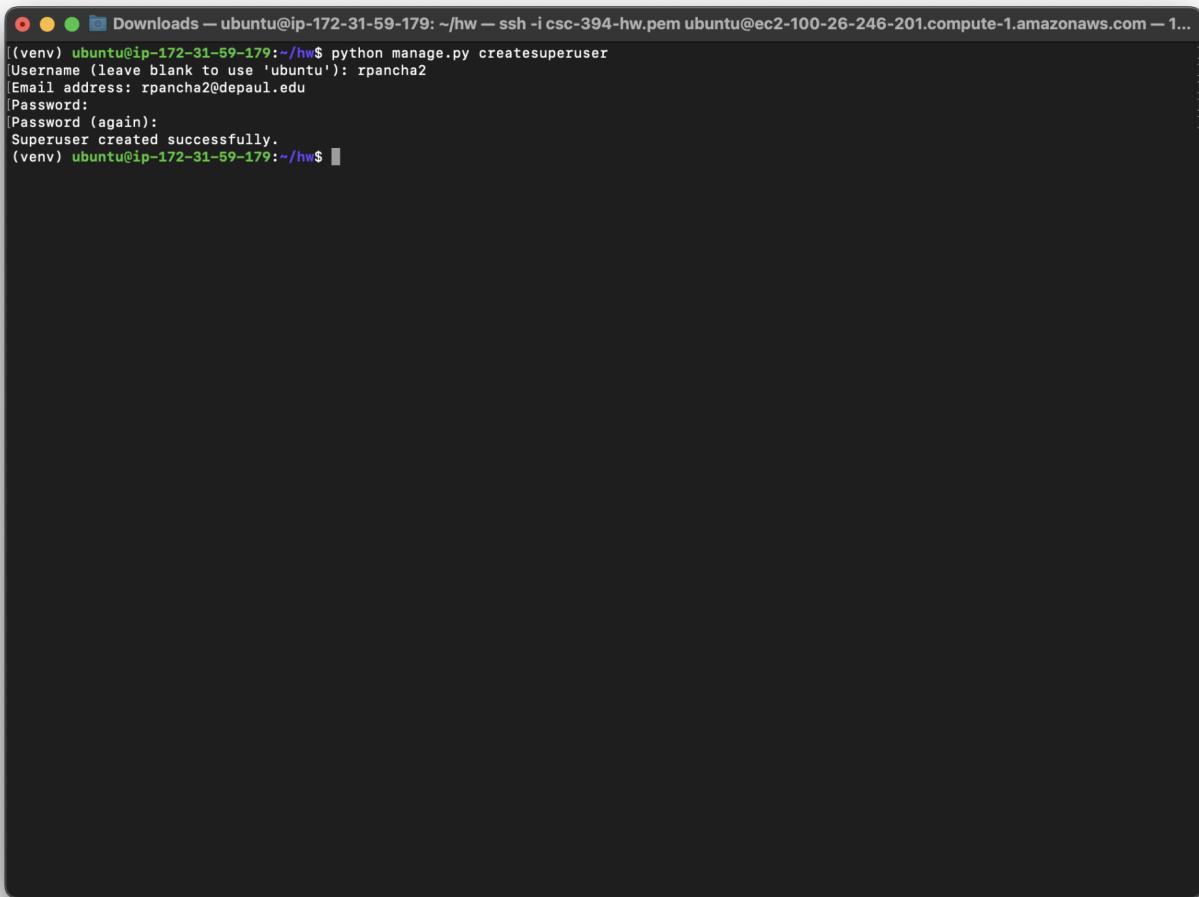
Run django migrations

Execute the django migration command: `python manage.py migrate .`



```
[venv] ubuntu@ip-172-31-59-179:~/hw$ python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying auth.0012_alter_user_first_name_max_length... OK
  Applying sessions.0001_initial... OK
(venv) ubuntu@ip-172-31-59-179:~/hw$
```

Followed by creating a super user: `python manage.py createsuperuser`



```
[venv] ubuntu@ip-172-31-59-179:~/hw$ python manage.py createsuperuser
[Username (leave blank to use 'ubuntu'): rpancha2
[Email address: rpancha2@depaul.edu
[Password:
[Password (again):
Superuser created successfully.
(venv) ubuntu@ip-172-31-59-179:~/hw$ ]]
```

Configure Nginx

We have the database setup, let's point our nginx server to our django app.

First let's clear out the existing configuration for our Nginx install.

```
sudo rm /etc/nginx/sites-enabled/default.conf
sudo rm /etc/nginx/sites-available/default.conf
```

```
python manage.py
```

Now let's create a new configuration file.

```
sudo vi /etc/nginx/sites-available/hw.conf
```

And let's paste the following:

```
upstream hw {  
    # Change the port based on what port your app is using.  
    server localhost:8000;  
}  
  
server {  
    listen 80;  
  
    # Change the following IP Address to your machine's public IP  
    # address.  
    server_name 0.0.0.0;  
  
    location / {  
        proxy_pass http://hw/;  
  
        proxy_set_header X-Real-IP $remote_addr;  
  
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;  
        proxy_set_header Host $http_host;  
        proxy_set_header X-NginX-Proxy true;  
    }  
}
```

To find the appropriate IP address, for the `server_name` , look at the EC2 instance on AWS.

The screenshot shows the AWS EC2 Instances page for an instance named 'i-02ec6e84165f74aa5 (csc-394-hw)'. The instance is running and has a public IP of 100.26.246.201. It is associated with a VPC ID (vpc-06e4fadd14c1e5ef5) and a subnet ID (subnet-035552462ac9b13b6). The instance type is t2.micro. The instance was launched on Tue Jan 10 2023 at 16:00:21 GMT-0600. The AMI ID is ami-06878d26597831ca, and the AMI name is ubuntu/images/hvm-ssd/ubuntu-jammy-22.04-amd64-server-20221206. The instance is currently running and has no termination protection.

Save and quit by hitting the escape key, followed by hitting : , followed by typing wq and wacking enter .

Next, execute the following command:

```
sudo ln -v -s /etc/nginx/sites-available/hw.conf /etc/nginx/sites-enabled/hw.net
```

Wonderful, the appropriate configuration files have been created.

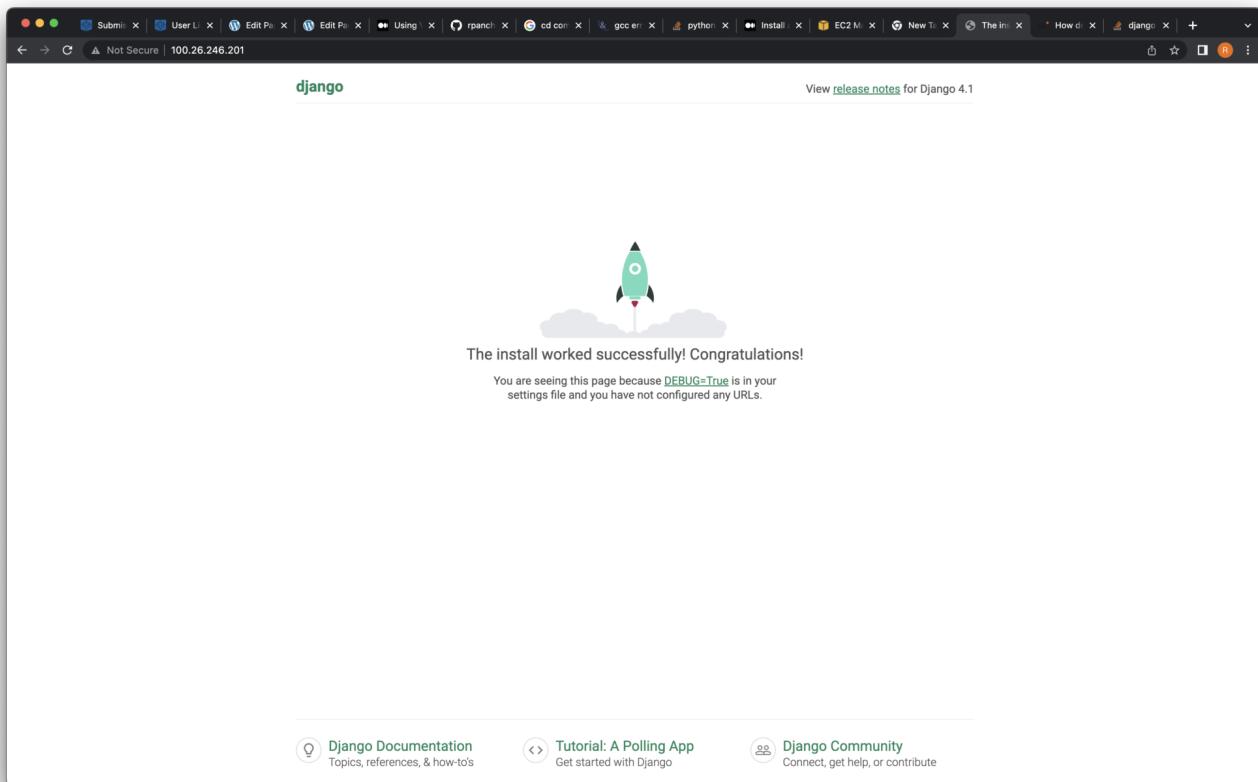
Let's restart the NginX server, execute the following command:

```
sudo service nginx restart
```

Back in the hw directory, start your django app!

```
python manage.py runserver
```

And now navigate to the ip address from earlier on your web browser, and see what happens!



Submission

Submit the following to D2L:

- A link to your IP Address
- A screenshot of your website running, showing the ip address in your url bar.

Theme: Noto Simple