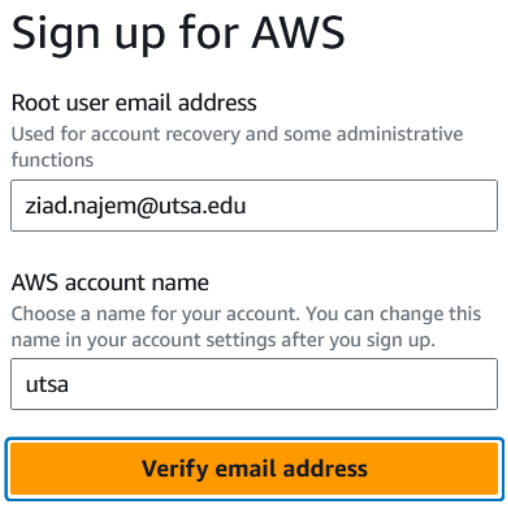
1. Creating an AWS (free) Account

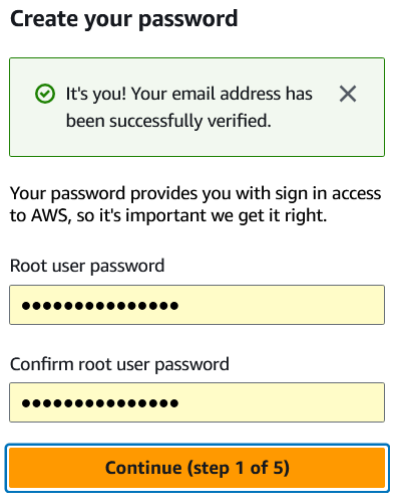
1. Go to https://aws.amazon.com/
2. Click on `[Create an AWS Account]` (top right of the page).
3. (if it takes you to a sign-in page, click on the `[Create an AWS Account]` button at the bottom of that page).

4. Fill in your email address. Use your `@utsa` address.
5. Name the account utsa.

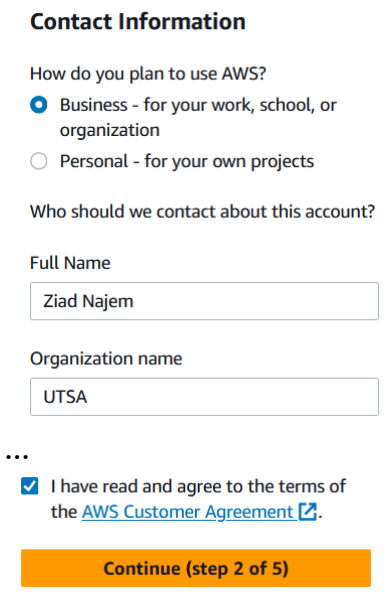


6. Click `[Verify email address]`
7. Check your email and copy the verification code

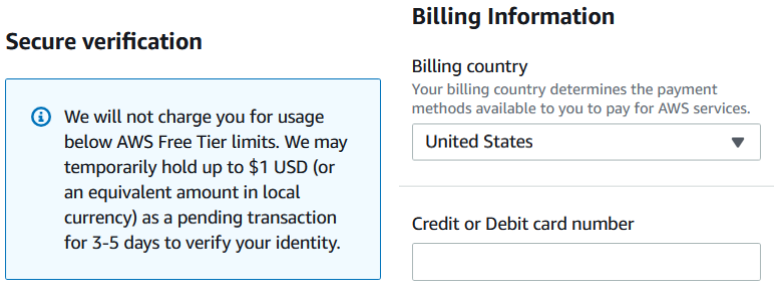
8. Create a strong password. Enter it twice. Save it somewhere. Never reuse passwords across services, especially critical ones (bank, LMS, SIS, etc.)



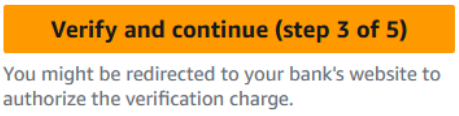
9. Fill in the contact information, (The form is longer than what's shown below.) Once completed, tick to accept the agreement, then click `[continue]`



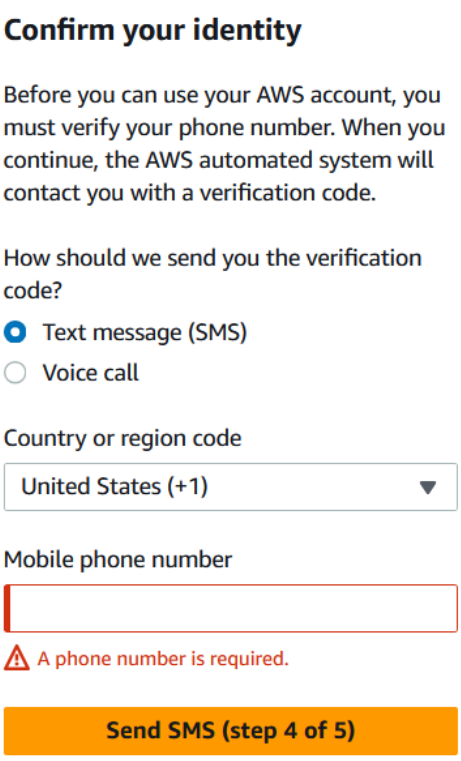
10. Fill in the Billing information (again longer than what's shown here)

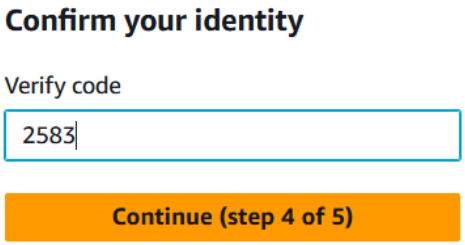


11. Click `[verify and continue]`

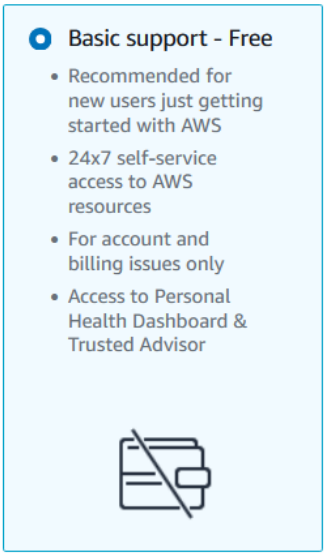


12. Now verify your phone number. Follow the steps below.





13. Finally, make sure you choose the FREE support.

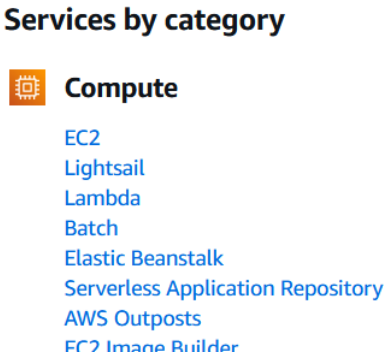


14. Congratulations. You're done. You may now go to the console screen.

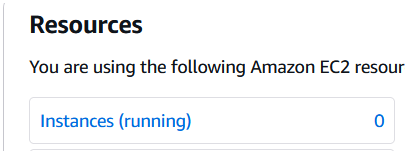
2. Creating your server (EC2 instance)

1. Once on the management console page, click on **[view all services]**. You can find it under the burger menu on the top-left corner of the console page.

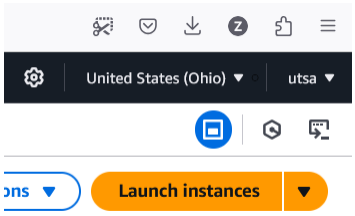
2. Now select **[EC2]** from **[compute]**



3. Click on the **[instances (running)]** link.

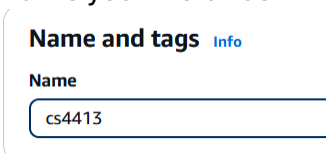


4. Make sure you're on the **Ohio** cloud (top-right corner of the page).

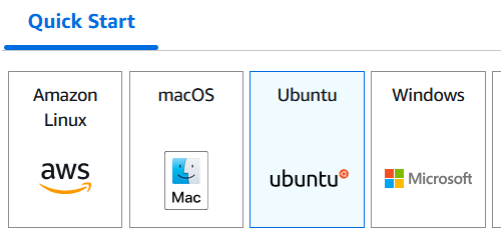


5. Now click the orange button **[Launch instances]**

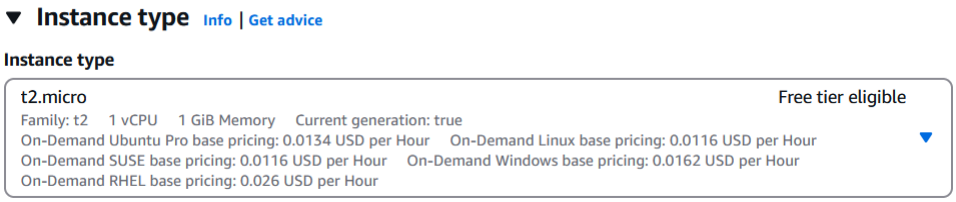
   a. Name your instance



   b. Choose **Ubuntu** and don't change anything in the configuration.

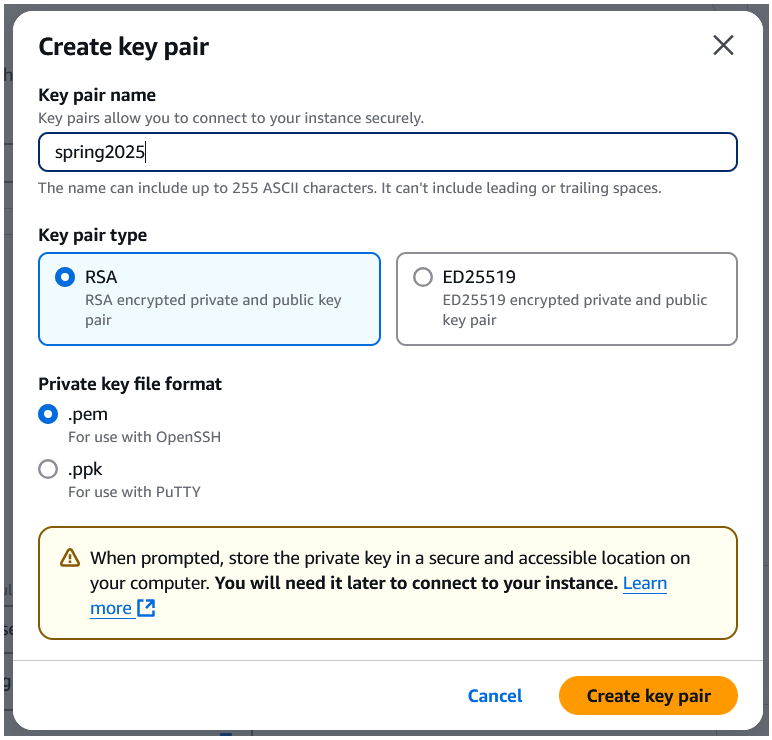


c. Double check that Instance type is **[t2.micro]** which is the default. That's the free instance.



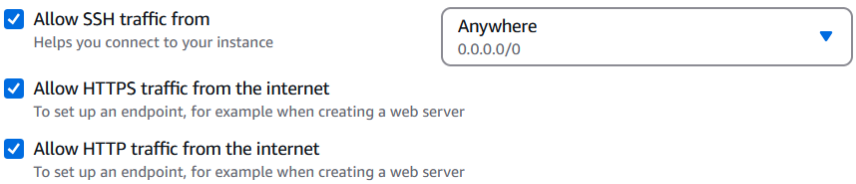
d. Create a Key Pair for secured login. Click `[Create key pair]`.

e. Give the key a name and keep everything else as the default is. Double check with the figure below:



f. Make sure you save and never lose the **.pem** file that gets downloaded. If you lose it, you'll need to login using the traditional user/password and then regenerate/reconfigure a new pair.

g. Under network settings, make sure you enable ssh from anywhere, https, and http.

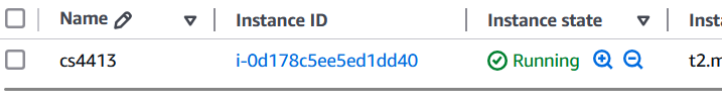


h. Under [configure storage], claim the whole 30 GiB for the root volume,

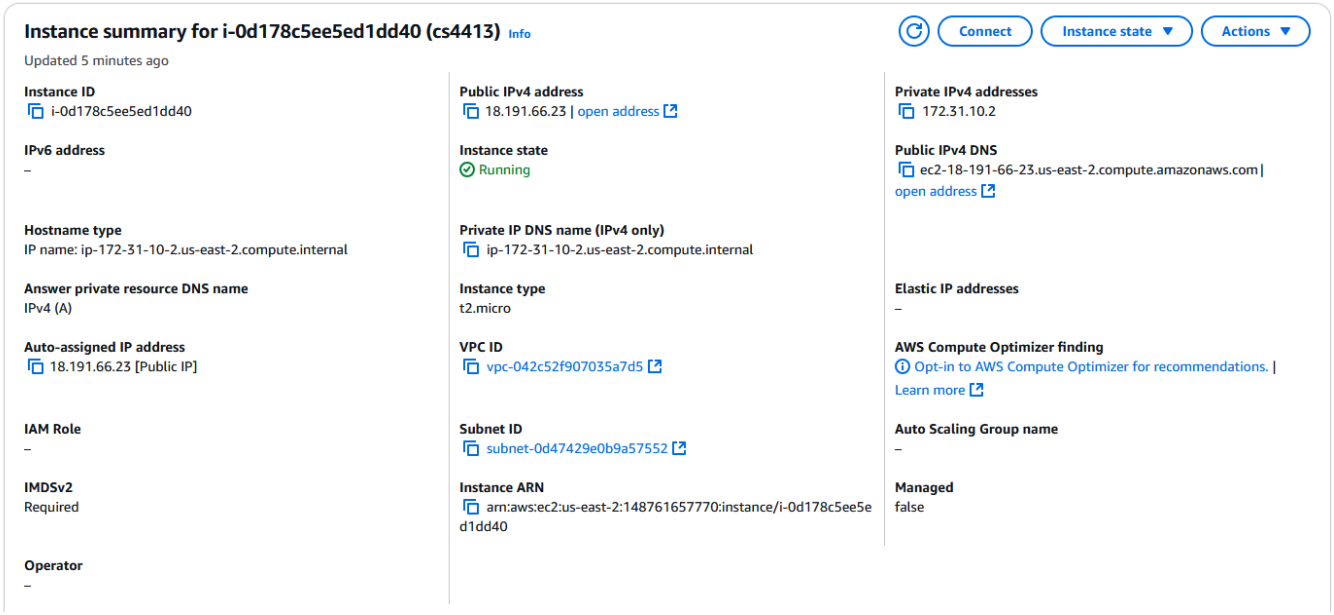
i. Finally, click  on the right side of the page.

6. Choose `[dashboard]` from the burger menu. Notice that you have one instance running .

7. Click on the `[running instances]` link and then verify that your instance is indeed running.



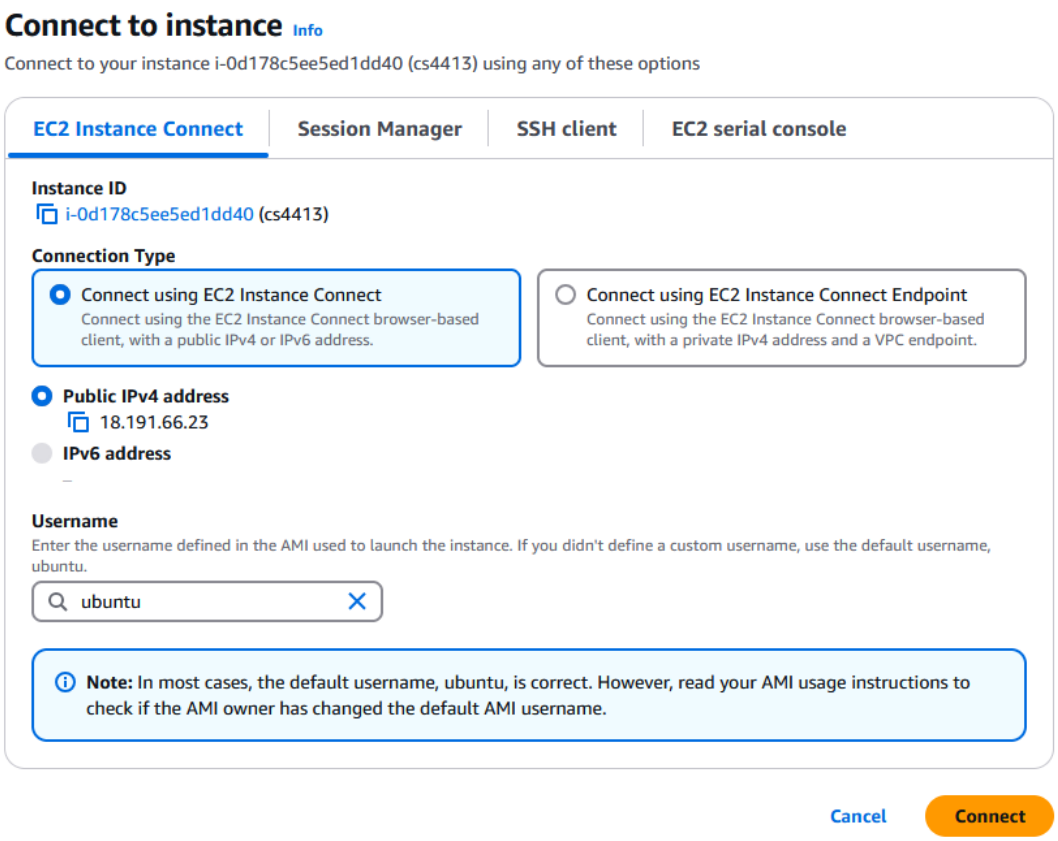
8. Now click on the instance ID of your server. This should give you interesting details about your server and how to access it.



Never stop the instance!

You can reboot it, but never stop it. If stopped, you'll lose the IP address given to you, which will require you redo many steps we'll be taking to configure the webserver based on the current IP.

9. Now click the `[connect]` button on the top-right of the page. This will take you to a screen with four methods of connecting to the server.

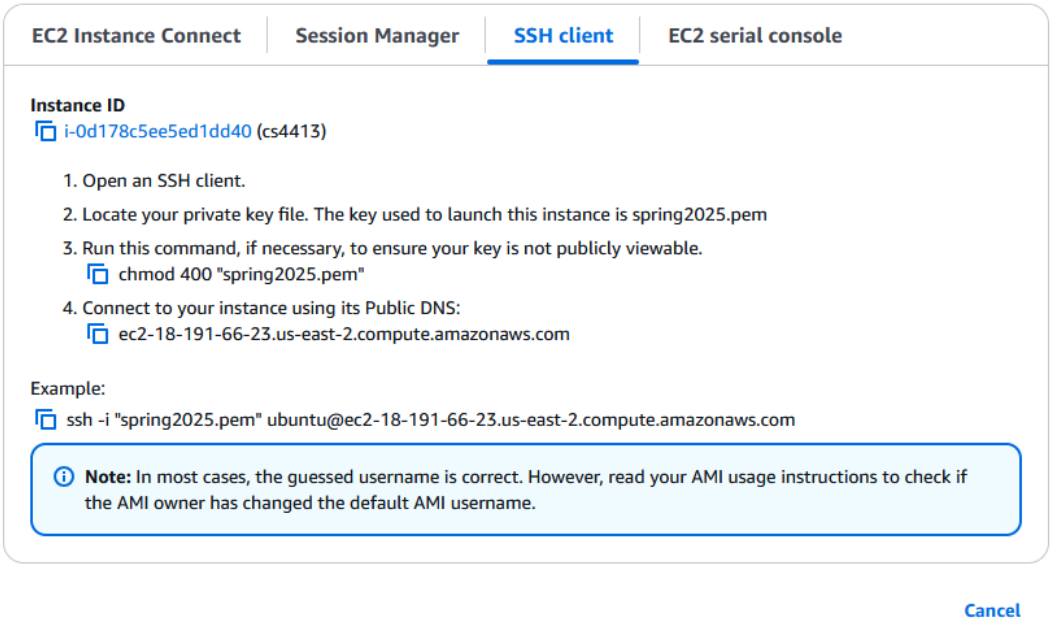


10. The `[EC2 Instance Connect]` is your safe way to connect to the server through the browser. Understandably it will be slow, but it could be useful as a last resort. Just make sure you can get to your aws dashboard.

3. Connecting to your server

In normal operation, you'll connect to your AWS server using using SSH. Follow the instructions in the previous section to go to the connections page, then continue with the following steps:

1. Click on the **`[SSH client]`** tab.



2. Copy the ssh command that's shown in the Example. Notice that the command already includes the DNS name for your AWS server, and the file name for your .pem key.
3. Open the command-line (CLI) on your local machine (Terminal, PowerShell, etc.) You can also use any of the dedicated SSH apps.
4. Paste the ssh command you copied from the browser (step 2) into the CLI.
5. It is normal for ssh to ask you for confirmation the first time you try to connect to a new machine (something about fingerprint). Answer [yes].
6. The command will fail complaining that your .pem file is not found.
7. Copy the .pem (from wherever it got downloaded) into whichever local directory your CLI is in. Normally that's your home directory
8. Now redo step 4.
9. Are you on your server? If so, go to the next section. If not, call for help!

4. Updating / Upgrading your Ubuntu server

It's a good practice to always check if you need to update/upgrade your server... especially upgrades related to security. Updating packages, however, is a critical process since packages are usually inter dependent on each other, and updating one might break some dependency for the another.

> **Bottom line!**
>
> **we'll only update our system once right after creating the instance, but never again during this semester. But do keep in mind that this is not the right approach if you're a system admin.**

To update/upgrade your Ubuntu, issue the following two commands: (don't copy the $ sign. That's just an indication that the line is a UNIX shell command)

```
$ sudo apt update
```

```
$ sudo apt upgrade
```

5. Creating a self-signed certificate

Normally, if you're planning to offer https services through your webserver, you'd get a certificate from a 3rd party (an independent entity that can verify the authenticity of your webserver to the user.) But for this course, we'll issue our own.

1. Issue the following command (this should all be copied as one line.)

```
$ sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout
/etc/ssl/private/nginx-selfsigned.key -out /etc/ssl/certs/nginx-
selfsigned.crt
```

2. Enter the requested information as seen below. Make sure you enter your own email utsa email address and the FQDN of your server (found on the dashboard, see the screenshot that follows)

```
Country Name (2 letter code) [AU]:US
State or Province Name (full name) [Some-State]:Texas
Locality Name (eg, city) []:San Antonio
Organization Name (eg, company) [Internet Widgits Pty Ltd]:UTSA
Organizational Unit Name (eg, section) []:Computer Science
Common Name (e.g. server FQDN or YOUR name) []:ec2-….amazonaws.com
Email Address []:your.email@utsa.edu
```

3. We'll use a dhparam file to strengthen the security of TLS key exchanges, making it harder for attackers to decrypt our encrypted communications. Issue the following command:

```
$ sudo openssl dhparam -out /etc/ssl/certs/dhparam.pem 2048
```

6. Installing and configuring your very own nginx HTTP Server

1. Let's install the NGINX (engine x) http/https server:

```
$ sudo apt install nginx
```

2. Configure the self-signed certificate:

```
$ echo
'c3NsX2NlcnRpZmljYXRlIC9ldGMvc3NsL2NlcnRzL25naW54LXNlbGZzaWduZWQuY3J0Owpzc2xfY2VydGlm
aWNhdGVfa2V5IC9ldGMvc3NsL3ByaXZhdGUvbmdpbngtc2VsZnNpZ25lZC5rZXk7Cg==' | base64 -d - |
sudo tee /etc/nginx/snippets/self-signed.conf > /dev/null
```

3. Configure the SSL parameters:

```
$ echo
'c3NsX3Byb3RvY29scyBUTFN2MS4yOwpzc2xfcHJlZmVyX3NlcnZlcl9jaXBoZXJzIG9uOwpzc2xfY2lwaGVy
cyAiRUVDREgrQUVTR0NNOkVESCtBRVNHQ006QUVTMjU2K0VFQ0RIOkFFUzI1NitFREgiOwpzc2xfZWNkaF9jd
XJ2ZSBzZWNwMzg0cjE7CnNzbF9zZXNzaW9uX2NhY2hlIHNoYXJlZDpTU0w6MTBtOwpzc2xfc2Vzc2lvbl90aW
NrZXRzIG9mZjsKc3NsX3N0YXBsaW5nIG9uOwpzc2xfc3RhcGxpbmdfdmVyaWZ5IG9uOwpyZXNvbHZlciA4Ljg
uOC44IDguOC40LjQgdmFsaWQ9MzAwczsKcmVzb2x2ZXJfdGltZW91dCA1czsKYWRkX2hlYWRlciBTdHJpY3Qt
VHJhbnNwb3J0LVNlY3VyaXR5ICJtYXgtYWdlPTYzMDcyMDAwOyBpbmNsdWRlU3ViZG9tYWlucyI7CmFkZF9oZ
WFkZXIgWC1GcmFtZS1PcHRpb25zIERFTlk7CmFkZF9oZWFkZXIgWC1Db250ZW50LVR5cGUtT3B0aW9ucyBub3
NuaWZmOwpzc2xfZGhwYXJhbSAvZXRjL3NzbC9jZXJ0cy9kaHBhcmFtLnBlbTsK' | base64 -d - | sudo
tee /etc/nginx/snippets/ssl-params.conf > /dev/null
```

4. Make a backup of the main configuration file:

```
$ sudo cp /etc/nginx/sites-available/default /etc/nginx/sites-available/distrib
```

5. Now we can do our changes to the main configuration file.

```
$ echo
'c2VydmVyIHsKCSMgU1NMIGNvbmZpZ3VyYXRpb24KCWxpc3RlbiA0NDMgc3NsIGh0dHAyIGRlZmF1bHRfc2Vy
dmVyOwoJbGlzdGVuIFs6Ol06NDQzIHNzbCBodHRwMiBkZWZhdWx0X3NlcnZlcjsKCWluY2x1ZGUgc25pcHBld
HMvc2VsZi1zaWduZWQuY29uZjsKCWluY2x1ZGUgc25pcHBldHMvc3NsLXBhcmFtcy5jb25mOwoKCXJvb3QgL3
Zhci93d3cvaHRtbDsKCgkjIEFkZCBpbmRleC5waHAgdG8gdGhlIGxpc3QgaWYgeW91IGFyZSB1c2luZyBQSFA
KCWluZGV4IGluZGV4Lmh0bWwgaW5kZXgucGhwOwoKCXNlcnZlcl9uYW1lIF87CgoJbG9jYXRpb24gLyB7CgkJ
IyBGaXJzdCBhdHRlbXB0IHRvIHNlcnZlIHJlcXVlc3QgYXMgZmlsZSwgdGhlbgoJCSMgYXMgZGlyZWN0b3J5L
CB0aGVuIGZhbGwgYmFjayB0byBkaXNwbGF5aW5nIGEgNDA0LgoJCXRyeV9maWxlcyAkdXJpICR1cmkvID00MD
Q7Cgl9CgojCWxvY2F0aW9uIH4gXC5waHAkIHsKIwkgIHJvb3QgL3Zhci93d3cvaHRtbDsKIwkgIHRyeV9maWx
lcyAkdXJpID00MDQ7CiMJICBmYXN0Y2dpX3NwbGl0X3BhdGhfaW5mbyBeKC4rXC5waHApKC8uKykkOwojCSAg
ZmFzdGNnaV9wYXNzIHVuaXg6L3Zhci9ydW4vcGhwL3BocC1mcG0uc29jazsKIwkgIGZhc3RjZ2lfaW5kZXggga
W5kZXgucGhwOwojCSAgZmFzdGNnaV9wYXJhbSBTQ1JJUFRfRklMRU5BTUUgJGRvY3VtZW50X3Jvb3QkZmFzdG
NnaV9zY3JpcHRfbmFtZTsKIwkgIGluY2x1ZGUgZmFzdGNnaV9wYXJhbXM7CiMJICBmYXN0Y2dpX3JlYWRfdGl
tZW91dCAzMDA7CiMJfQoKfQoKc2VydmVyIHsKCWxpc3RlbiA4MCBkZWZhdWx0X3NlcnZlcjsKCWxpc3RlbiBb
OjpdOjgwIGRlZmF1bHRfc2VydmVyOwoJIyBjaGFuZ2UgdGhlIG5leHQgbGluZSB0byB5b3VyIHNlcnZlcidzI
GZxZG4KCXNlcnZlcl9uYW1lIGVjMi5jb21wdXRlLmFtYXpvbmF3cy5jb20gOwoJcmV0dXJuIDMwMiBodHRwcz
ovLyRzZXJ2ZXJfbmFtZSRyZXF1ZXN0X3VyaTsKfQo=' | base64 -d - | sudo tee
/etc/nginx/sites-available/default > /dev/null
```

6. Let's make sure we got all three config files correct:

```
$ echo ce06df2c2a53f169cc48ad3e8f84d89e5c295c1c00886d31dee3606fdb58b92d
/etc/nginx/sites-available/default | sha256sum -c
```

```
$ echo 2ebe327e944269f402267f0a1e06eff60723d3847329c7d233aa20012bb574f3
/etc/nginx/snippets/self-signed.conf | sha256sum -c
```

```
echo cad3cb9841d0d8471f98470ae11b5e9501f5ebd563bfd582d6843f5b6d0fd354
/etc/nginx/snippets/ssl-params.conf | sha256sum -c
```

7. Manually edit **`/etc/nginx/sites-available/default`** so you can configure it with your server's specific FQDN. (look for the **`server_name`** directive in the 2nd stanza)

```
$ sudo vi /etc/nginx/sites-available/default
```

8. Test that the configuration is correct.

```
$ sudo nginx -t
```

9. Reload (or restart) the server for the modified configuration to take effect.

```
$ sudo service nginx reload
```

10. Let's get a home page quickly.

```
$ sudo cp /var/www/html/index.nginx-debian.html /var/www/html/index.html
```

11. Point your browser to your server. Try https: first, then see what happens when you try http: