**Linux Server Configuration**

This is the final project for Udacity's [Full Stack Web Developer Nanodegree](https://www.udacity.com/course/full-stack-web-developer-nanodegree--nd004).

This page explains how to secure and set up a Linux distribution on a virtual machine, install and configure a web and database server to host a web application.

* The Linux distribution is [Ubuntu](https://www.ubuntu.com/download/server) 16.04 LTS.
* The virtual private server is [Amazon](https://lightsail.aws.amazon.com/).
* The web application is my [Item Catalog project](https://github.com/SkBadulla/catalog) created earlier in this Nanodegree program.
* The database server is [PostgreSQL](https://www.postgresql.org/).

You can visit <http://54.161.86.157/> or <http://ec2-54-161-86-157.compute-1.amazonaws.com/> for the website deployed.

**Get a server**

**Step 1: Start a new Ubuntu Linux server instance on Amazon EC2**

* Login to [*aws.amazon.com*](https://console.aws.amazon.com/) and login to default user (ubuntu)
* Choose EC2 and Launch Instance with appropriate settings.
* Check for instance IPv4 public IP - 54.161.86.157
* we can download a .pem file and connect with following command
* ssh -i ItemCatalog\_19\_01\_2019.pem ubuntu@54.161.86.157
* 22 is Port by Default,Later we need to change it to 2200 as per the udacity-linux-server-configuration rubrics.

**Secure the server**

**Step 2: Update and upgrade installed packages**

sudo apt-get update

sudo apt-get upgrade

**Step 3: Change the SSH port from 22 to 2200**

* Edit the /etc/ssh/sshd\_config file: sudo vi /etc/ssh/sshd\_config.
* Change the port number on line 5 from 22 to 2200.
* Save and exit using esc and confirm with :wq.
* Restart SSH: sudo service ssh restart.
* Add one more rule in AWS EC2 Security group inbound rules --> Type : Custom TCP Rule as 2200 and HTTP Rule as 80 - -> Save Rule
* To check port 2200 weather working or not by using putty -> Take new putty : hostname(Ex: 3.81.228.22) - -> Port:2200

**Step 4: Configure the Uncomplicated Firewall (UFW)**

* Configure the default firewall for Ubuntu to only allow incoming connections for SSH (port 2200), HTTP (port 80), and NTP (port 123).
  + sudo ufw allow 2200/tcp
  + sudo ufw allow 80/tcp
  + sudo ufw allow 123/udp
  + sudo ufw enable
* sudo ufw status # The UFW should be inactive.
* sudo ufw default deny incoming # Deny any incoming traffic.
* sudo ufw default allow outgoing # Enable outgoing traffic.
* sudo ufw allow 2200/tcp # Allow incoming tcp packets on port 2200.
* sudo ufw allow www # Allow HTTP traffic in.
* sudo ufw allow 123/udp # Allow incoming udp packets on port 123.
* sudo ufw deny 22 # Deny tcp and udp packets on port 53.
* Turn UFW on: sudo ufw enable. The output should be like this:
* Command may disrupt existing ssh connections. Proceed with operation (y|n)? y
* Firewall is active and enabled on system startup
* Check the status of UFW to list current roles: sudo ufw status. The output should be like this:
* Status: active
* To Action From
* -- ------ ----
* 2200/tcp ALLOW Anywhere
* 80/tcp ALLOW Anywhere
* 123/udp ALLOW Anywhere
* 22 DENY Anywhere
* 2200/tcp (v6) ALLOW Anywhere (v6)
* 80/tcp (v6) ALLOW Anywhere (v6)
* 123/udp (v6) ALLOW Anywhere (v6)
* 22 (v6) DENY Anywhere (v6)

**Give grader access**

**Step 5: Create a new user account named grader**

* While logged in as ubuntu, add user: sudo adduser grader.
* Enter a password (twice) and fill out information for this new user.

**Step 6: Give grader the permission to sudo**

* Edits the sudoers file: sudo visudo.
* Search for the line that looks like this:
* root ALL=(ALL:ALL) ALL
* Below this line, add a new line to give sudo privileges to grader user.
* root ALL=(ALL:ALL) ALL
* grader ALL=(ALL:ALL) ALL
* Save and exit using CTRL+X and confirm with Y.
* Verify that grader has sudo permissions. Run su - grader, enter the password.

**Step 7: Create an SSH key pair for grader**

-Configure key-based authentication for grader user

* create .ssh folder by mkdir /home/grader/.ssh
* Run this command: sudo cp /home/ubuntu/.ssh/authorized\_keys /home/grader/.ssh/authorized\_keys
* change ownership chown grader.grader /home/grader/.ssh
* add 'grader' to sudo group:  sudo usermod -aG sudo grader
* change permissions for .ssh folder:  chmod 0700 /home/grader/.ssh/, for authorized\_keys:
* Change directory to authorized\_keys : cd /home/grader/.ssh/
* sudo chmod 644 authorized\_keys
* Check in sudo vi /etc/ssh/sshd\_config file if PermitRootLogin is set to no
* Restart SSH: sudo service ssh restart
* On the local machine, checking if the grader account working or not by using putty -> Take new putty : hostname(Ex: grader@3.81.228.22) - -> Port:2200

**Prepare to deploy the project**

**Step 8: Configure the local timezone to UTC**

* While logged in as grader, configure the time zone: sudo dpkg-reconfigure tzdata. Choose time zone UTC.

**Step 9: Install and configure Apache to serve a Python mod\_wsgi application**

* While logged in as grader, install Apache: sudo apt-get install apache2.
* Enter public IP of the Amazon EC2 instance into browser. Check Apache is working or not by executing public IP.
* My project is built with Python 3. So, I need to install the Python 3 mod\_wsgi package:  
  sudo apt-get install libapache2-mod-wsgi-py3.
* Enable mod\_wsgi using: sudo a2enmod wsgi.

**Step 10: Install and configure PostgreSQL**

* sudo apt-get install libpq-dev python-dev
* sudo apt-get install postgresql postgresql-contrib
* sudo su - postgres
* psql
* CREATE USER catalog WITH PASSWORD 'catalog';
* ALTER USER catalog CREATEDB;
* CREATE DATABASE catalog WITH OWNER catalog;
* \c catalog
* REVOKE ALL ON SCHEMA public FROM public;
* GRANT ALL ON SCHEMA public TO catalog;
* \q
* exit
* Switch back to the grader user: exit.

**Step 11: Install git**

* While logged in as grader :- su - grader
* Install git: sudo apt-get install git.

**Deploy the Item Catalog project**

**Step 12.1: Clone and setup the Item Catalog project from the GitHub repository**

* While logged in as grader,
* From the cd /var/www directory, Clone the catalog project:  
  sudo git clone https://github.com/SkBadulla/catalog.git.
* Change the ownership of the catalog directory to grader using: sudo chown -R grader:grader catalog/.
* Change to the cd /var/www/catalog/ directory(Git Directory: /var/www/catalog/).
* ls
* Rename the mainpage.py file to \_\_init\_\_.py using: mv mainpage.py \_\_init\_\_.py.
* Check the file saved or not : ls
* We need to change sqlite to postgresql create\_engine in \_\_init\_\_.py,db\_setup.py and db\_init.py,
* Edit all file in nano editing tool : nano \_\_init\_\_.py, nano database.py, database\_init.py(All files in git folder) and change database path sqlite to postgresql and save the files - -> Ctrl+x --> Y - -> ENTER
* ####Original code:::: engine = create\_engine("sqlite:///catalog.db")
* Replace This code ::::: engine = create\_engine('postgresql://catalog:catalog@localhost/catalog')

**Step 12.2: Authenticate login through Google**

* Go to [Google Cloud Plateform](https://console.cloud.google.com/).
* Click APIs & services on left menu.
* Click Credentials.
* Create Project: project name & location - -> Click on Create
* Create new Credentials
* Choose OAuth Client ID
* Configure consent screen
* Use Type: Internal (or) External - -> edit name, logo, support email, ADD All scope files - -> domain name: [www.google.com](http://www.google.com) - -> Application Home page link, App Privacy, App terms link as -- <http://www.google.com>
* Click on save
* Back to credentials - -> create credentials - -> OAuth Client ID - -> Application Type : Web Application - -> edit name, Authorized java script & Authorized Redirect URL(<https://www.google.com>)
* Click on create and after created (your client id & your client secret) - -> clickon ok
* Click on your project name(in credentials tab- -> OAuth Client IDs - -> name(click on that name(ex: webclient1)))
* Add [http://54.161.86.157.xip.io](http://54.161.86.157.xip.io/) and <http://ec2-54-161-86-157.compute-1.amazonaws.com/> as authorized JavaScript origins.
* Add [http://54.161.86.157.xip.io/login,http://54.161.86.157.xip.io/gconnect,http://54.161.86.157.xip.io/callback](http://54.161.86.157.xip.io/login,http:/54.161.86.157.xip.io/gconnect,http:/54.161.86.157.xip.io/callback) as authorized redirect URI.
* Download the corresponding JSON file, open it and copy the contents.
* In grader Open nano /var/www/catalog/client\_secrets.json and paste the previous contents into the this file.
* Replace the OAuth client ID templates/login.html file in the project directory.
* nano /var/www/catalog/templates/login.html
* change data-clientid(is in google cloud platform 🡪 credentials 🡪 OAuth 2.0 client ids)
* save login.html file

**Step 13.1: Install the virtual environment and dependencies**

* While logged in as grader, install pip: sudo apt-get install python3-pip.
* Install the virtual environment: sudo apt-get install python-virtualenv
* Change to the cd /var/www/catalog/ directory.
* Create the virtual environment: sudo virtualenv -p python3 venv3.
* Change the ownership to grader with: sudo chown -R grader:grader venv3/.
* Activate the new environment: . venv3/bin/activate.
* Install the following dependencies:
* pip install httplib2
* pip install requests
* pip install --upgrade oauth2client
* pip install sqlalchemy
* pip install flask
* sudo apt-get install libpq-dev
* pip install psycopg2-binary

**Step 13.2: Set up and enable a virtual host**

Configure and enable a new virtual host

* Run this: sudo vi /etc/apache2/sites-available/catalog.conf
* Paste this code:

<VirtualHost \*:80>

ServerName 3.81.228.22.xip.io

ServerAlias ec2-3-81-228-22.compute-1.amazonaws.com

ServerAdmin ubuntu@3.81.228.22

WSGIDaemonProcess catalog python-path=/var/www/catalog:/var/www/catalog/venv3/lib/python3.6/site-packages

WSGIProcessGroup catalog

WSGIScriptAlias / /var/www/catalog/catalog.wsgi

<Directory /var/www/catalog/>

Order allow,deny

Allow from all

</Directory>

Alias /static /var/www/catalog/static

<Directory /var/www/catalog/static/>

Order allow,deny

Allow from all

</Directory>

ErrorLog ${APACHE\_LOG\_DIR}/error.log

LogLevel warn

CustomLog ${APACHE\_LOG\_DIR}/access.log combined

</VirtualHost>

* Enable the virtual host sudo a2ensite catalog

Enabling site catalog. To activate the new configuration, you need to run: service apache2 reload

* Reload Apache: sudo service apache2 reload.

**Step 13.3: Set up the Flask application**

* Create nano /var/www/catalog/catalog.wsgi file add the following lines:
* import sys
* import logging
* logging.basicConfig(stream=sys.stderr)
* sys.path.insert(0, "/var/www/catalog/")
* from catalog import app as application
* application.secret\_key = 'supersecretkey'
* Restart Apache: sudo service apache2 restart.
* From the /var/www/catalog/catalog/ directory, activate the virtual environment: . venv3/bin/activate.
* Run: python database\_setup.py.
* Deactivate the virtual environment: deactivate.

**Step 13.4: Disable the default Apache site**

* Disable the default Apache site: sudo a2dissite 000-default.conf. The following prompt will be returned:
* Site 000-default disabled.
* To activate the new configuration, you need to run:
* service apache2 reload
* Reload Apache: sudo service apache2 reload.

**Step 13.5: Launch the Web Application**

* Restart Apache again: sudo service apache2 restart.
* Open your browser to [http://54.161.86.157](http://54.161.86.157/) or [http://ec2-54-161-86-157.compute-1.amazonaws.com](http://ec2-54-161-86-157.compute-1.amazonaws.com/).

**Step 14: Automatically install updates(optional)**

The unattended-upgrades package can be used to automatically install important system updates.

* Enable automatic (security) updates: sudo apt-get install unattended-upgrades.
* Edit /etc/apt/apt.conf.d/50unattended-upgrades, uncomment the line ${distro\_id}:${distro\_codename}-updates and save it.
* Modify /etc/apt/apt.conf.d/20auto-upgrades file so that the upgrades are downloaded and installed every day:
* APT::Periodic::Update-Package-Lists "1";
* APT::Periodic::Download-Upgradeable-Packages "1";
* APT::Periodic::AutocleanInterval "7";
* APT::Periodic::Unattended-Upgrade "1";
* Enable it: sudo dpkg-reconfigure --priority=low unattended-upgrades.
* Restart Apache: sudo service apache2 restart.

**Useful commands**

* To get log messages from Apache server: sudo tail /var/log/apache2/error.log.
* To restart Apache: sudo service apache2 restart.

**Other Helpful Resources**

* DigitalOcean [How To Deploy a Flask Application on an Ubuntu VPS](https://www.digitalocean.com/community/tutorials/how-to-deploy-a-flask-application-on-an-ubuntu-vps)
* GitHub Repositories
  + [adityamehra/udacity-linux-server-configuration](https://github.com/adityamehra/udacity-linux-server-configuration)
  + [anumsh/Linux-Server-Configuration](https://github.com/anumsh/Linux-Server-Configuration)
  + [bencam/linux-server-configuration](https://github.com/bencam/linux-server-configuration)
  + [iliketomatoes/linux\_server\_configuration](https://github.com/iliketomatoes/linux_server_configuration)

*Special Thanks to*[*Alain Boisvert*](https://github.com/boisalai)*for a very helpful README in Linux Server Configuration Project-Udacity*\*