Rev. 1.0

Smart Grader

One Pace Plaza, New York , NY 10038

Developer Manual

Table of Contents

[Deployment 2](#_Toc103278795)

[Virtual Environment 2](#_Toc103278796)

[Cloning Repository 2](#_Toc103278797)

[Gunicorn Setup 2](#_Toc103278798)

[Nginx Setup 3](#_Toc103278799)

[Database Setup 3](#_Toc103278800)

## Deployment

Application can be deployed on all Operating Systems though Linux is preferred. Instructions for Ubuntu are listed below. If you’re using a different Linux operating system, please substitute yum commands with your preferred package manager. The main components are the following and can be installed using yum.

* Python yum install python
* Git yum install git
* Nginx yum install nginx

## Virtual Environment

To keep python packages separated from rest of your environment, a virtual environment must be created using virtualenv package.

* pip install virtualenv
* virtualenv my\_env

## Cloning Repository

Create a new directory in your virtual environment and clone the repository from GitHub.

* git clone <https://github.com/joseph-dougal/CS691>
* CD into CS691 to make it your active directory
* Activate your virtual environment using activate /home/ubuntu/my\_env/bin/activate  
  (use the path appropriate to your directory structure)
* Run start.cmd file which will install all packages from requirements.txt and start the application
* Stop the application once all packages are installed and flask gets recognized

## Gunicorn Setup

Gunicron will be our Python web server which acts as a WSGI HTTP server for Linux environments.

* Create a service file to run your application
* cat /etc/systemd/system/my\_app.service
* ex:  
   [Unit]  
   Description=Gunicorn instance to serve myproject  
   After=network.target  
   [Service]  
   User=ubuntu  
   Group=www-data  
   WorkingDirectory=/home/ubuntu/my\_app/  
   Environment="PATH=/home/ununtu/my\_env/bin" Environment="PYTHONPATH=/home/ubuntu/my\_env/python:/home/ubuntu/my\_app"

ExecStart=/home/ubuntu/my\_env/bin/gunicorn --workers 3 --bind unix:my\_app.sock -m 007 app:app --timeout 6000  
 Restart=on-abort  
 [Install]  
 WantedBy=multi-user.target

## Nginx Setup

Nginx will act as a reverse proxy for our application.

* Create nginx config file for "my\_app"
* cat /etc/nginx/sites-enabled/my\_app
* ex:  
   server {  
   listen [::]:80;  
   listen 80;  
   server\_name my\_app.com www.my\_app;  
   location / {  
   include proxy\_params;  
   proxy\_pass http://unix:/home/ubuntu/my\_app/my\_app.sock;  
   }  
   }

## Database Setup

Though any SQL database can be used, we prefer to use Postgres SQL. For each system Postgres SQL setup differs, though details instructions can be found on Digital Ocean:

<https://www.digitalocean.com/community/tutorials/how-to-install-postgresql-on-ubuntu-20-04-quickstart>

* All the database tables are defined as model
* Create\_db.py file connects to the Postgres db and creates the tables  
  Command: python /CS691/project/db\_utils/create\_db.py