Online Resume

Using Django, Vue.js and MongoDB

July 7, 2020

Prepared by: Muhammad Bilal Khan

Hosted at: Amazon Web Service
Source Code: Git Hub

1 Introduction

This report was establish for the purpose of deployment of an online website implemented using Django as the backend engine and Vue.js as the frontend for manipulating the DOM plus **MongoDB** for the database. The report discusses the layout of the project folder and describes the content held within. Overviewing and bit about which file implements what and how. The **links** stated at the start of the documents links to some of the important aspects of the task. *Amazon Web Service* holds the code made live running on a virtual machine. Please visit my *Git Hub* for viewing and inspecting the complete code. *Please consult this report inplace for the read.me file; writing this in markdown is a task for another day.*

2 File Structure

The repository contains 2 folders mainly

1. website-project
2. Documentation

In the order of alphabetic listing each folder has been described. **Blog** folder is an app created within the Django app, it holds the model implementation for Django ORM for a blogs page which has been left for future implementation. Like wise the **homepage** foler is app and model implementation for the landing page and deals with communicating with the **MongoDB** port running at the back. You'll find the database model here as well. It also holds the *index.html* file within the **templates** folder which served for the purpose of displaying the home or the landing page of the single page application. **Media** is the folder for the Django ORM to store files and database structure for accessability, currently it only holds images, hence just a sinel **images** folder within. **Static** folder the location for Django to serve static files from. CSS, JavaScript, images and PDf docuemnts have been collected using the *collectstatic* functionality of Django. **Website** is the project folder, and houses the setting, url and views file; although they would be found empty, since the website has been redirected to the homepage app of the project. The *settings file* will provide information on the database that has been set up and some other dependencies. An important feature of the project is that it has been *dockerized*, so just prepare *docker engine* and *docker compose* and fire up the **Dockerfile** to run the website regardless of the requirements. Speaking of requirements, **requirements.txt** is the file created from the python virtual environment and holds the python required to run the django server.

Documentation folder houses this report and couple other LaTeX dependencies.

3 Deployment

For the purpose of hosting this single page application I've utilized my **Amazon Web Service**, reason because this are familar to me and already a couple other applications are up and running on the platform. The project has been dockerized and contained using docker and uploaded to my git hub. From there an *EC2* instance was deployed running a Ubuntu system. Docker was then installed on the virtual machine and the project's git repository cloned. After that it was just *docker-compose build and up.*

An effort was also made to host the application on **Heroku**, but unlike before I was having a static file problem which I'm currently debugging.