National College of Ireland

Configuration Manual

MSc Research Project
Cloud Computing

Shryni Kedambadi Shreekar
Student ID: X18199011

School of Computing
National College of Ireland

Supervisor: Sean Heeney

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Shryni Kedambadi Shreekar
Student ID:	X18199011
Programme:	Cloud Computing
Year:	2020
Module:	MSc Research Project
Supervisor:	Sean Heeney
Submission Due Date:	17/08/2020
Project Title:	Configuration Manual
Word Count:	832
Page Count:	2

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

I agree to an electronic copy of my thesis being made publicly available on TRAP the National College of Ireland's Institutional Repository for consultation.

Signature:	Shryni Kedambadi Shreekar
Date:	16th August 2020

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Shryni Kedambadi Shreekar
X18199011

1 Pre-Requisites

For implementing the automated dockerization of python based web apps we have followed the following steps. Our approach was to make use of combination of ability to python programs to call shell commands and corresponding measure stacks around it. We are aiming at dockerizing non-docekrized web applications using our method and create a dockerized form of this application which will create a docker file for the application that can be run on any platform of our choice.

Firstly for the environment setup, we have the python 3.7 or above installed. The preference being the latest version. For dockerization we need Docker desktop be installed, within which we can execute the docker commands. In cases where docker desktop cannot be installed we make use of Docker client for lower versions of Operating systems.

2 Execution Steps

- Begin with starting the docker interactive shell with the command "start.sh"
- Import the source code of any python web application(non-dockerized version) on which we are going to apply dockerization.
- We import our 2 main python files, "dockermaker.py" and "runcommands.py" within the same folder.
- Within dockermaker we make use of tkinter for python with which we create a UI to accept input parameters for the imput web application.
- We input the project name, project directory, modules, and then the directory of python installation, then the app name and the port number on which we want to expose this application.
- At this point if docker installation is not done right, the process will fail right away. If installed, the performDocker commands will be called from within "runcommands.py" file.
- Within this file we take care of creating a "run.bat" file.
- In run.bat , we create a python virtual environment and we install all the modules required within virtual env to run that particular application.

- Then we perform steps within the bat file to create a directory with name "pip_cache" which will be copied into docker.
- We collect statistics , that will combine all the htmls.
- Finally we do a docker build to create a docker image of the python web app.

In python we have django, flask web frameworks. These are excellent frameworks but we need a load balancer.The load balancer which we have used is "ngnix". Hence while making a docker file, we need to take care of installing ngnix as well. This is achieved from within runCommands.py file.

3 Result

Last step is to create docker file and have the following steps run from within

1. Download python
2. Perfrom ngnix installation .
3. Copy source and install dependencies which we generated in the virtual environment using pip_cache.
4. create project folder, copy all the requirements, stats files within that.
5. Then run the install command. We need to note that here the install command is running within docker.
6. Then start the server, We mention the port on which the URL must be exposed.

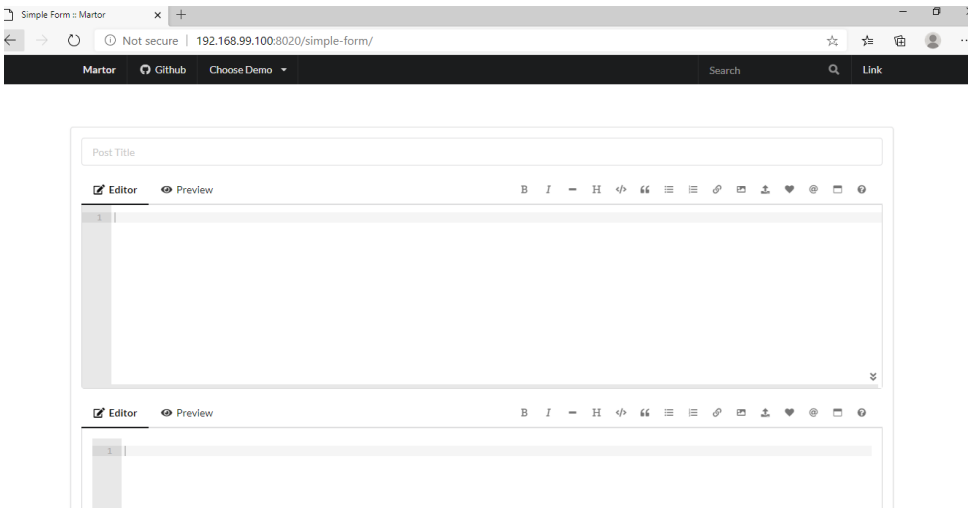

Figure 1: Dockerized application

The final build gives us a docker file which we can run in our local system as well to check the application is running or not. To deploy on cloud, we build image, deploy to docker repo and deploy to any cloud as well.