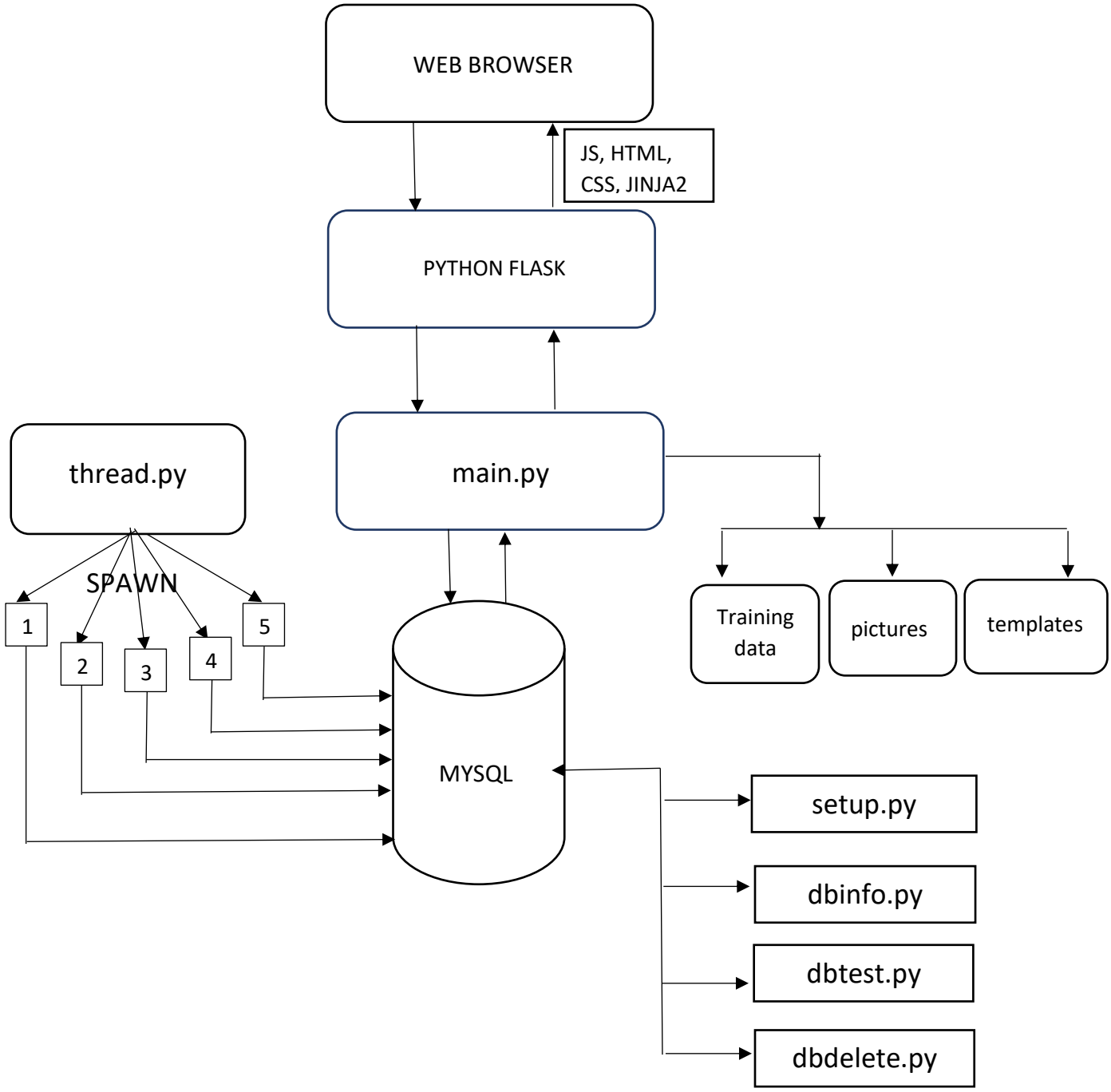
OSH Component Design Document

This document outlines the high level and component design of the One Stop Healthcare (OSH) Solution. OSH is a Python Flask application that provides help for all COVID19 related needs.

It consists of several modules that work together to provide the complete solution.

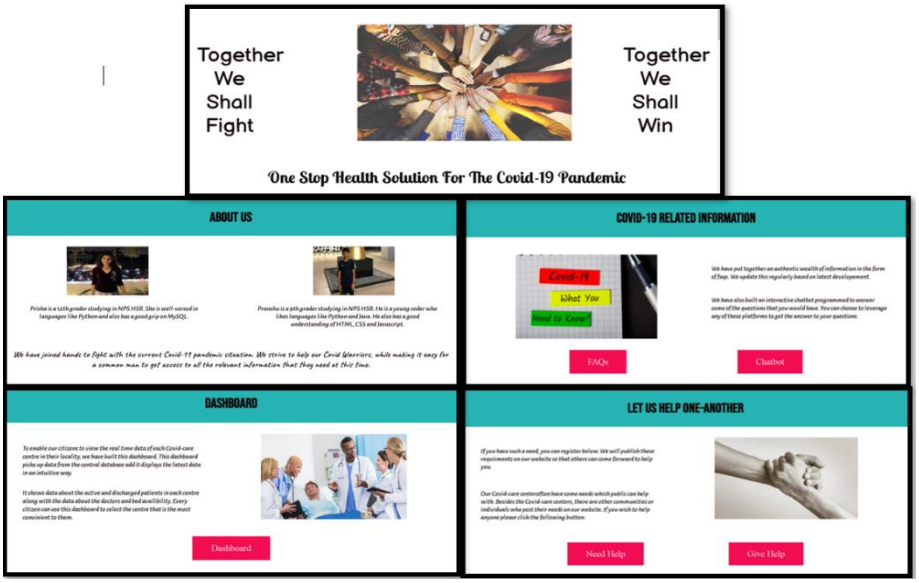
The following diagram outlines the overall component design of the OSH solution.



These are the components of One Stop Healthcare solution:

main.py

This is the primary component which works with Flask to provide a web-based interface that can be accessed by a web browser (preferably chrome). It connects with MySQL database in the backend, and fetches all the real time data from there. It then generates a Javascript/HTML/CSS page and sends it to the browser through the Python Flask framework. This resultant page can be rendered on any web browser, though it works best with Google Chrome.



OSH web interface consists of a total of 5 screens as shown above. These are:

- Home screen- This is the main landing page with a scrollbar to scroll down to the other pages
- About Us- It has small description about the creators - Prisha and Pranshu
- Dashboard- It gives the link for our Covid-care centres' patient, bed and doctor data
- COVID-19 information- It provides the most reliable and accurate information about the pandemic, and gives the links for an inbuilt chatbot application and a FAQ page
- Let us help one another- It gives links to the donations page where one can make donations, and to the page where one can request for help during these COVID times

setup.py

This program is used to seed the required data in the MySQL database. This needs to be run after the product is installed, and before it is used. This created the required tables and also some sample data in those tables. Setup.py performs the following actions:

- Creates a database - "one_stop_healthcare"
- Creates the required tables - ccinfo, donators, help_needed
- Adds some basic values for the tables

dbinfo.py

This file contains information about the MySQL database connection. This contains the default configuration which comes with WAMP installation. If you have set non-default values for MySQL

username and password, then please edit this file to contain the right values. If this is not set to the correct values, then the rest of the program will not work as expected.

dbtest.py

This program should be run before executing the main.py program. This tests the MySQL database connectivity. If the MySQL database is installed properly and the connection parameters are entered correctly in the dbinfo.py file, then this program should return success.

dbdelete.py

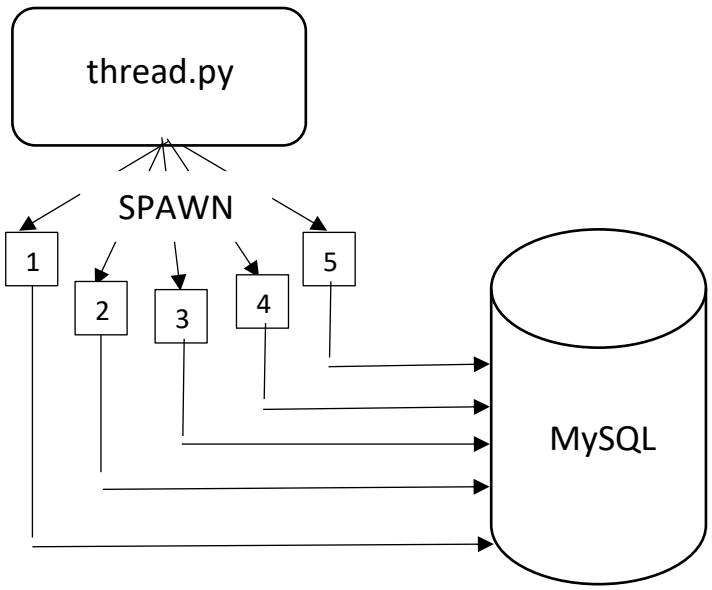
This program cleans up the database tables. This should be run in these cases:

- If you wish to uninstall the program, and no longer need the tables and data in the MySQL database
- If the database is corrupted, and you want to delete and reset the database values

thread.py

This program simulates the working of a set of COVID care centres. For the demo purpose, we have set this value to 5, and it can be increased to any other number. This is a multithreaded program that spawns a set of threads, where each thread represents the working of a particular COVID care centre. As we know, each centre has a constant inflow of new patients, and there is a workflow of patients getting discharged after the treatment. Therefore, each thread mimics that functionality through these steps:

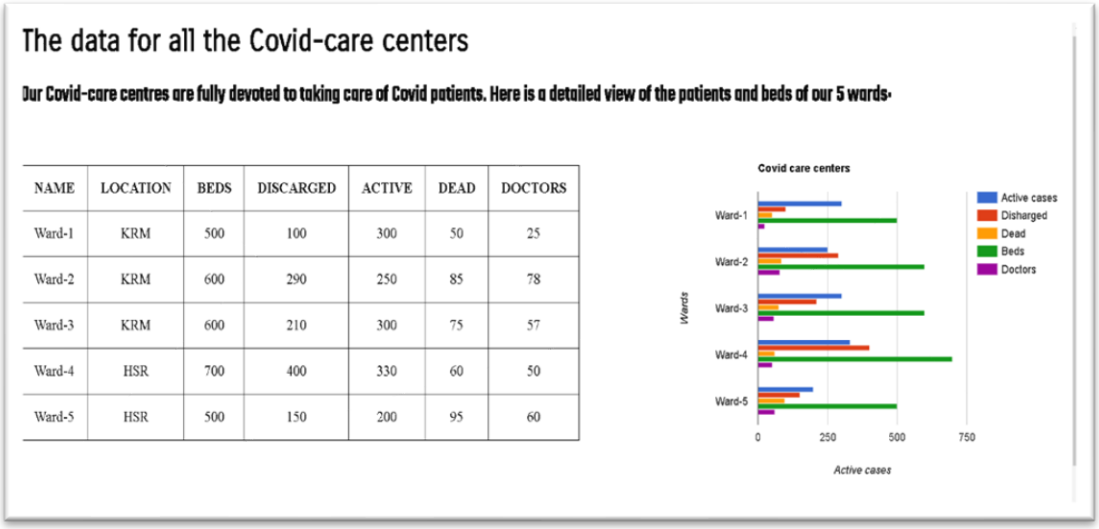
- Each thread runs in an infinite loop until it is interrupted using Ctrl-C
- In each loop, the thread sleeps for 7 seconds
- It then generates a random number, which is used to determine the parameter to be changed. These parameters include:
    - Number of inpatients
    - Number of discharged patients
    - Number of mortalities
    - Number of doctors/beds
    - Any help required by the centre related to PPE kits, masks, blankets, etc.



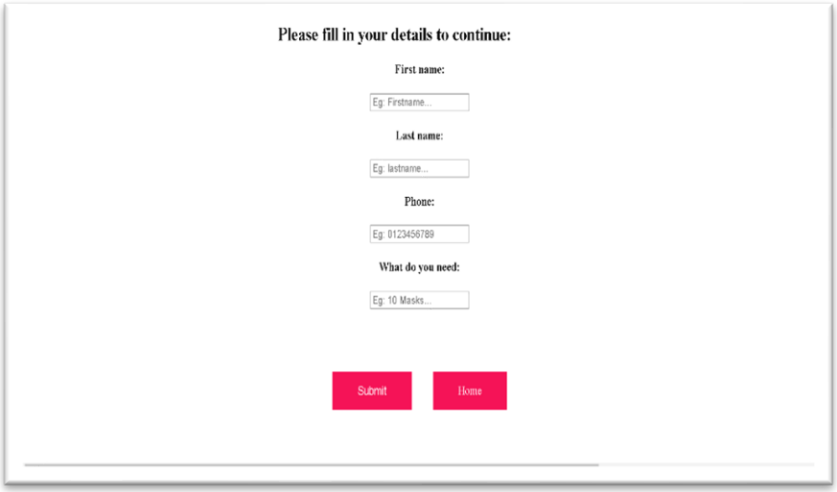
Besides the above Python programs, the OSH solution also bundles a set of HTML files. These are:

1. Chatbot.html
    - This displays the chatbot interface
    - It gathers the input data and sends it back to main.py to get an appropriate response

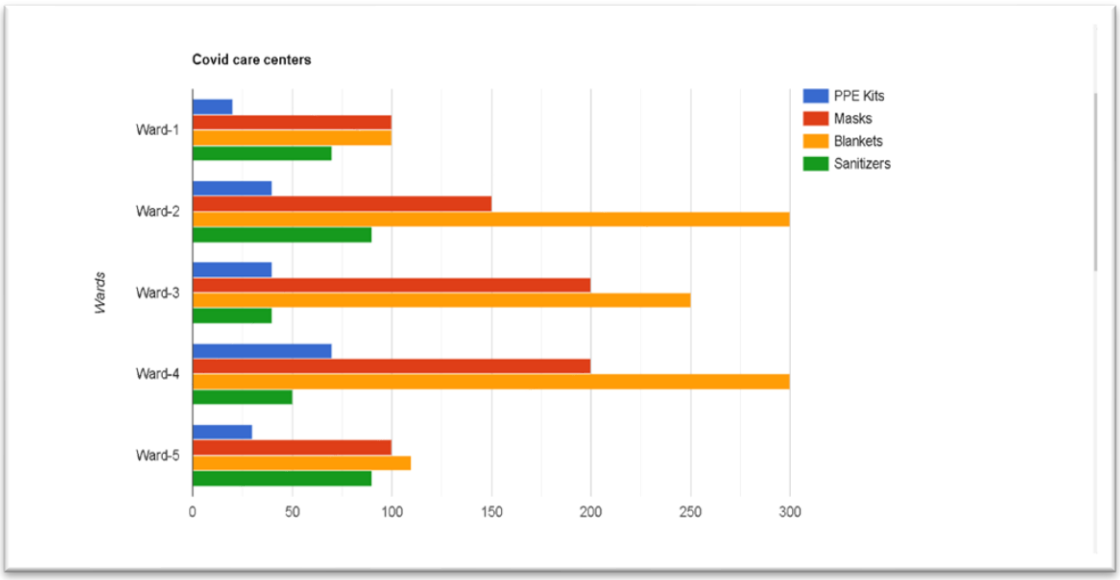
2. Dashboard.html
    - This basically shows current active, dead, recovered, bed availability etc in the form of a table and a bar chart (which is rendered with the help of google charts)
    - It is a dynamic page which when refreshed can show the new values which have changed in the background due to the multithreaded simulation program



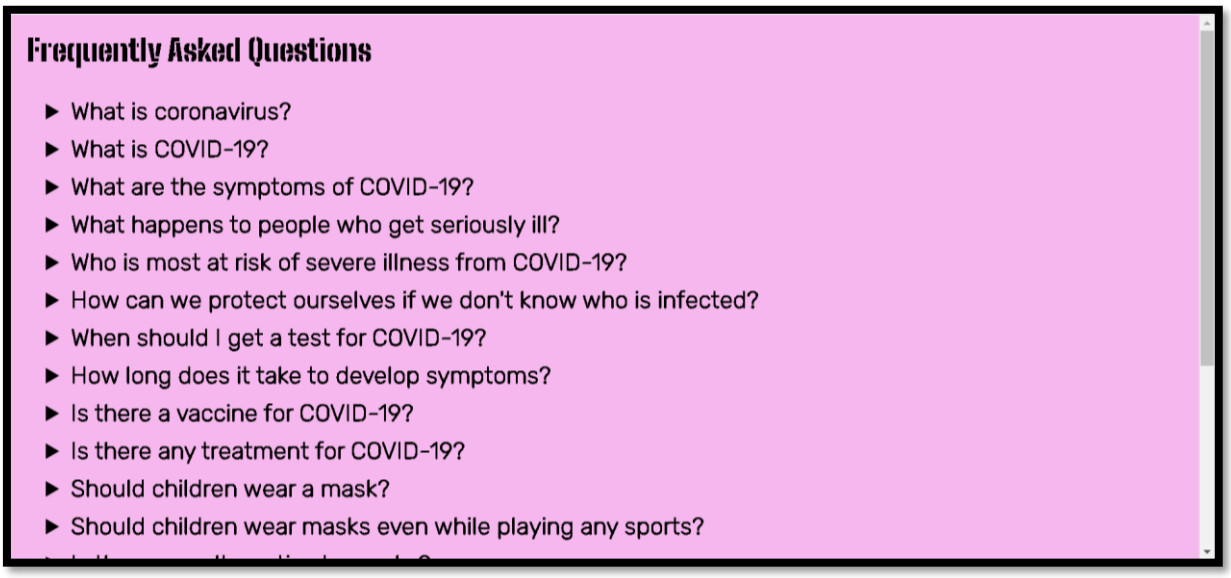
3. Details.html and Info.html
    - Both of these pages gather details about the people who wish to donate and who need help, and then passes these details back to main.py to add to the database



4. Donation.html
    - This page uses Google Charts to render a bar graph which displays the values of the supplies required by are simulated Covid care centres.
    - It also displays a table which has the data of the people who require any help and have asked for it on our website.



5. Faq.html
    - This page is a rich display of the frequently asked questions regarding COVID-19 and the coronavirus



6. Index.html
    - Index.html is the home page which has the links to all the other child pages.

7. Thanks.html
    - Thanks.html serves a dual purpose of showing an exit page after one has input all required details in details.html and info.html