CovidDectector System

**Abstract:** With the recent low number of community cases of covid-19, mass gathering is allowed. However, participants will have to take an antigen rapid test for Covid-19 and obtain a negative result before admission. Hence, event planner must ensure the participants safety/health condition before allowing participants to enter the event venue. This project proposes solution to this problem by using a machine learning model to determine the likelihood of participant having Covid-19. In addition, it includes a chatbot to allow the participants to know current information about Covid-19 and the event itself.

1. **Business Case**

This current law enforced mainly target huge events. Before a participant can enter the venue, the event organizer has to ensure that he/she is not contacted with Covid-19. To implement this, the event organizer will have to check with each participant individually on their health status before letting them in. As social distancing enforcement is a must, a long queue will be expected. Meaning, more manpower is needed to maintain the queue. If a shorter time can be implemented to handle these, the queue will be shorted thus lesser manpower is needed.

We propose that this can be handled with a web application (mobile-friendly) that has a survey for the participants to answer to get their health status. Participants with a mobile device would be able to answer the necessary questions before reaching the front of the queue. The result of the survey will let the queue maintainer either guide the participant to the swap test area or letting the participant into the venue. Moreover, it also includes a chatbot that allows the user to know (real-time) information about Covid-19 and the event itself. This may keep the participants occupied while waiting for their turn to enter the venue. The solution may be tailored to suit different type of events.

1. **System Model**

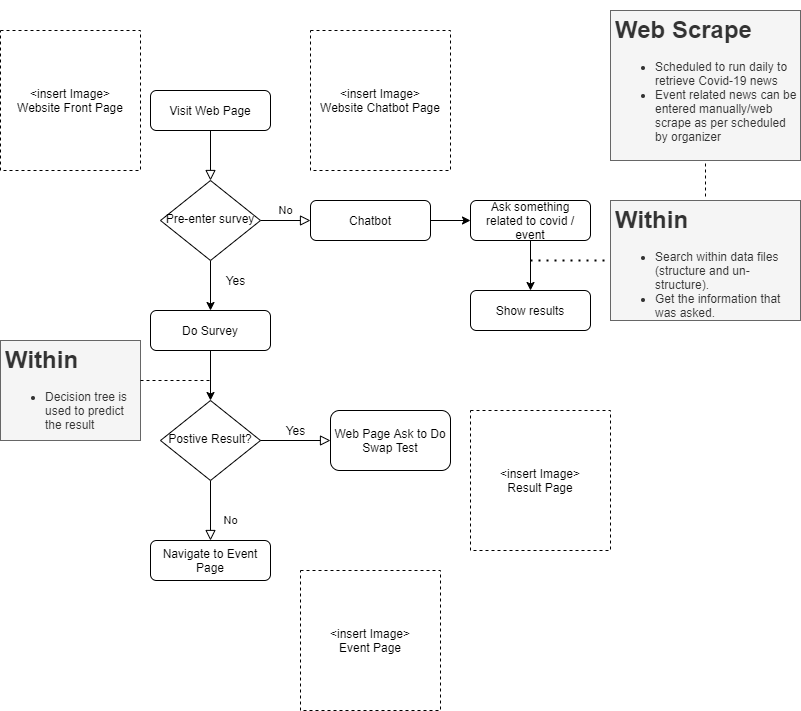
****

Figure 1 CovidDetector System - Flowchart

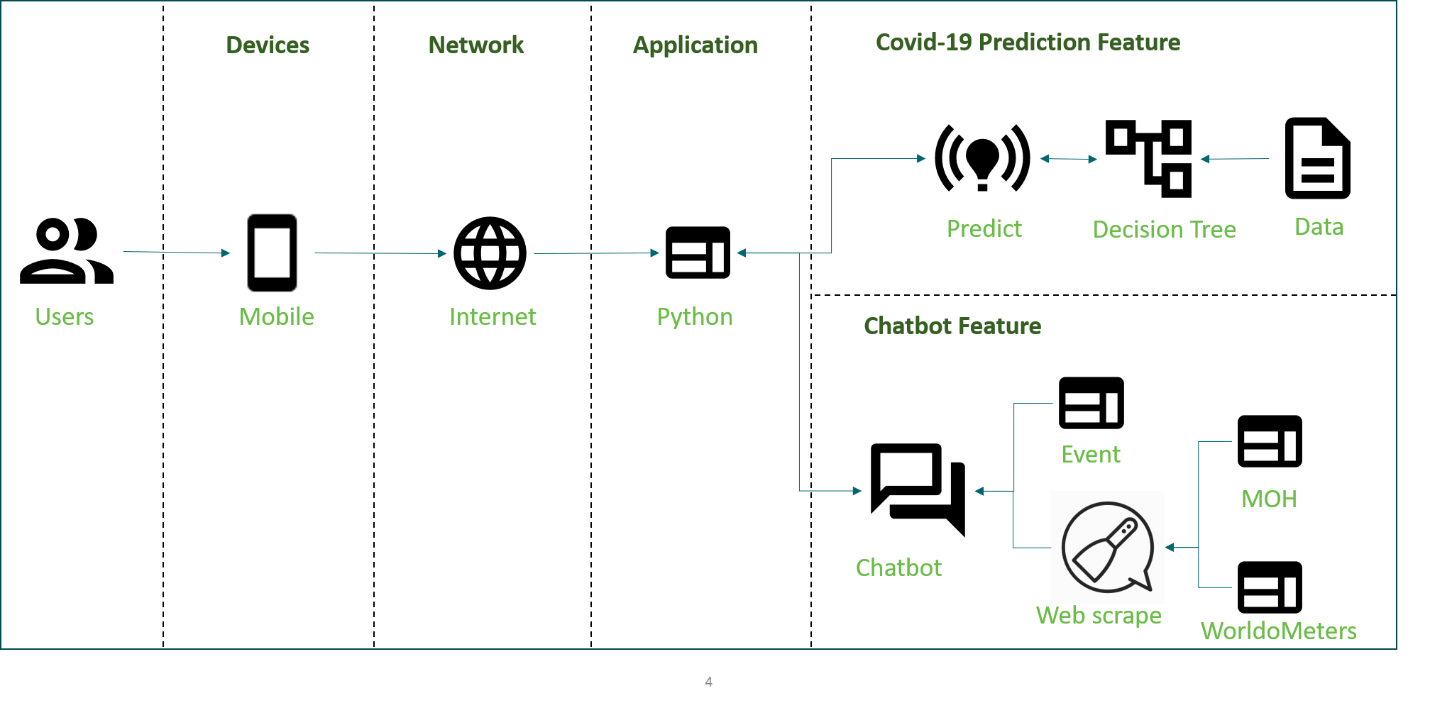
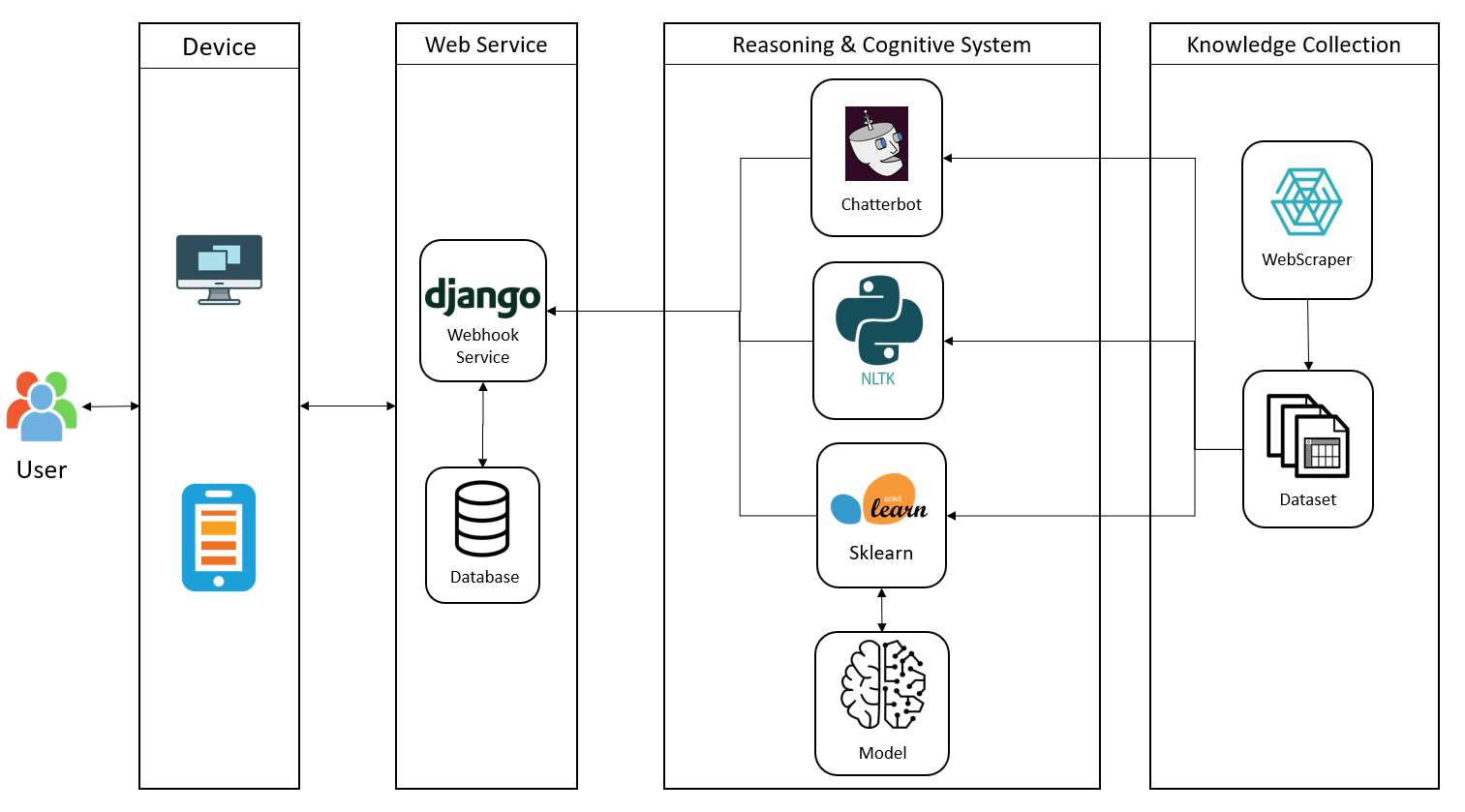


Figure 2 CovidDetector System - System Architecture

1. **System Development & Implementation**
   1. **System Architecture**

**3.1.1. Overview**



In order to build up a system which contains front-end and back-end and can integrate with machine learning libraries, as a result, Django, a python-based web framework, is decided to use in this project. The other reason that we choose Django is it is built with SQLite database by default which is easier to help us deploy model.

Our eco-system is constituted by two components, predictive model for covid cases and chatbot for covid frequent questions. We choose to leverage scikit-learn which is a python library for predictive data analysis to build a model to predict if a user gets Covid-19 based on his symptoms and his profile. The chatbot consists of two types, one uses chatterbot which is a Python library that makes it easy to generate automated responses to a user’s input, and the other one is developed by scikit-learn and nature language tool kit to find out the most similar pattern from dataset based on user inputs. All of the knowledge and data are coming from web scraper which would run daily to refresh the knowledge base.

* 1. **Data Sources**
     1. **Covid Symptoms and User Profile Data Set**

Our dataset comes from the Israeli Ministry of Health that publicly released data of all individuals who were tested for SARS-CoV-2 via RT-PCR assay of a nasopharyngeal swab. During the first months of the COVID-19 pandemic in Israel, all diagnostic laboratory tests for COVID-19 were performed according to criteria determined by the Israeli Ministry of Health. While subject to change, the criteria implemented during the study period included the presence and severity of clinical symptoms, possible exposure to individuals confirmed to have COVID-19, certain geographical areas, and the risk of complications if infected. Except for a small minority who were tested under surveys among healthcare workers, all the individuals tested had indications for testing. Thus, there was no apparent referral bias regarding the vast majority of the subjects in the dataset used in this study. In addition, all negative and positive COVID-19 cases this dataset were confirmed via RT-PCR assay. Therefore, we can consider this dataset is genuine and reliable.

The following list describes each features of the dataset:

A. Basic information:

1). Sex (male/female)

2). Age ≥60 years (true/false)

B. Symptoms:

3). Cough (true/false)

4). Fever (true/false)

5). Sore throat (true/false)

6). Shortness of breath (true/false)

7). Headache (true/false)

C. Other information:

8). Known contact with an individual confirmed to have COVID-19 (true/false)

* + 1. **Chatbot FAQ and Related Information**

To synchronize the knowledge base used by the chatbot with the frequently updated online webpage, the data extraction of the system is required to be executed daily. This can be done with the help of Cronjob external time scheduler, keeping the information updated automatically. This allows users to get the most accurate answers at any point of time while using the chatbot. The following table shows the websites we used to scrape the latest Covid-19 information.

|  |  |
| --- | --- |
| Websites | Data to Crawl |
| *https://www.who.int/news-room/q-a-detail/q-a-coronaviruses* | The WHO provides answers to frequently asked question regarding the coronavirus. |
| *https://www.worldometers.info/coronavirus/* | Infection Status across all countries |
| *https://www.moh.gov.sg/covid-19* | Latest updates & advisories from Ministry of Health, Singapore |

* 1. **Knowledge Elicitation and Extraction**
     1. **Manual Extraction**
     2. **Web Scraper**
  2. Beautiful Soup Python library:
     1. Visit the websites to retrieve Covid-19 related information.
     2. Study the html website and identify the information to extract.
     3. Html parser is utilised to extract the identified information.   
        Extract relevant information from the html and write to the data file.  
        Extract relevant urls and visit those websites to get relevant information.
  3. **Reasoning System**
     1. **Overview**
     2. **Predictive Model**
  4. **Cognitive System**
     1. **Chatterbot Library**
     2. **NLTK Chatbot**
  5. **Web Service**
  6. Web application

Web application uses Django Web Framework for these functions:

* + 1. UI for user to check event information and answer the survey question before entering the venue.   
       This UI is form based, requiring information for decision tree to predict the result.
    2. User may navigate to the event tab to get event information (e.g queue waiting time etc).
    3. User may navigate to the chatbot tab to seek information with regards to the event and Covid-19 that is not stated in the website.

1. **Challenge and Conclusion**
   1. **challenges**
   2. **Future Improvements**

If we have a longer timeframe to work on this project, we would have work on these areas:

* + 1. **Include database to store data**

A database could be added to store the survey answer, result and personal information (e.g name and last 4 char of NRIC) of every participant. If one of the participants happened to be a covid-19 confirmed case, necessary information could be provided to the relevant authorities for contact tracing.

* + 1. **Real-time API for web scrape**

Currently, the web scrape could only be triggered by schedule. With a real-time API and adding more relevant and trustable websites for web scrape, the user could be more informed of the latest news every time they ask a relevant question via the chatbot.

* + 1. **Chatbot icon to be shown at the bottom right of the webpage**

Currently, the chatbot is within a tab. However, the ideal placement for the chatbot should be at the bottom right of every web page. The reasons being, most of the people are right-handed and they will not be comfortable with the bottom left placement because they are used to things being catered for right-handed people. In addition, having a chatbot not within the webpage, user might forget the question to ask regarding the content when they clicked on a new tab.

* 1. **Conclusion**