**Software Development Description (SDD)**

**For**

**UMBC Covid-19 Tracker**



**Our Mission:** To increase awareness and share data regarding the Covid-19 virus.

**Group 5:** David Kravets, Israel Morocho, Safia Shah, Duncan Taylor, Tom Tennant, Alex Wilson

**Due:** 3/25/2021

[**Scope.**](#_8h3d3lt852y2) **3**

[1.1 Identification.](#_p675lkjalg87) 3

[1.2 System overview.](#_z4okypuh4b7x) 3

[1.3 Document overview.](#_je5axlhwpit2) 3

[**2. Referenced Documents**](#_mbv20ipjx6be) **4**

[**3. CSCI-wide design decisions**](#_onu0zjg93ity) **4**

[3.1 Behavior:](#_ol1jcxszve8h) 4

[3.2 User Access:](#_ytmjpp1b33q2) 4

[3.3 Constraints:](#_n8973dpa9m0h) 5

[3.4 Safety, Security and Privacy:](#_34w349vdnd2v) 5

[**4. CSCI architectural design**](#_755jp33a2ywe) **5**

[4.1 CSCI components](#_vjuguwgx4uo6) 5

[4.1.1 Hardware usage](#_1iqxyfsx7c1n) 5

[4.1.2 Menu unit (CT\_ME)](#_lhp3y2ck0a1q) 6

[4.1.3 Statistics unit (CT\_ST)](#_u72zm7blw8ej) 6

[4.1.4 Map unit (CT\_MA)](#_5s6b2if9x8zl) 6

[4.2 Concept of execution](#_iifhv6kfb7yb) 6

[4.2.1 Flow of execution](#_9yo4ba5dlka4) 6

[4.2.2 Software unit control flow](#_midpwisnc0co) 7

[4.3 Interface design](#_wg03832c5gpw) 8

[4.3.1 Interface identification and diagrams](#_y1vkae1qymru) 8

[4.3.1.1 The Covid-19 Data Tracker API](#_ux7xue7qk4n3) 9

[4.3.1.2 DjangoDB API](#_k3xl7zgosg4z) 9

[4.3.1.3 Broadstreet Covid Data Interface](#_smza8wn4fn0) 9

[4.3.2 Project Unique Identifier for The Covid-19 Data Tracker API (see 4.3.1.1)](#_pcj7ocyievsv) 9

[4.3.3 Project Unique Identifier for DjangoDB API (see 4.3.1.2)](#_pcj7ocyievsv) 9

[4.3.4 Project Unique Identifier for BroadStreet Covid Data Interface (see 4.3.1.3)](#_pcj7ocyievsv) 10

[**5. CSCI detailed design**](#_2d32jjz2nsdo) **10**

[5.1 CT\_ME](#_3c3qmwbt843h) 10

[5.2 CT\_ST](#_pch3n6bgxv5a) 10

[5.3 CT\_MA](#_d2asag5t0ji8) 10

[**6. Requirements traceability.**](#_d1a63towdi17) **12**

[**7. Notes.**](#_m9em9tlipm8y) **13**

[A. Appendixes.](#_x1q3ygixg11t) **13**

# Scope.

The software to be produced is a web application called UMBC Covid-19 Data Tracker (CDT). This web application takes in verified Covid-19 case and vaccine data based on state and location to create a heat map to visualize the data. The tracker will not report on individual cases of Covid-19 where you would be able to tell if an individual has Covid-19. The web application will be using data from The Broadstreet Covid-19 Data Project to create the heat map, as their data is organized by state and county. Vaccine data and information will be received by a similar database. Settings will be implemented to allow a user to see cases, vaccinations, or both. Thus, creating an efficient software to aid in public safety and knowledge about the ongoing pandemic.

## 1.1 Identification.

This software will be used to spread awareness and give information about the Covid-19 virus. We have seen an overall lack of awareness of people about Covid-19 hot areas and ways to avoid the virus. We also think there is a lack of hubs for information with regards to virus information along with vaccine information all in one central place.

**Titles**: UMBC Covid-19 Tracker

**Identification Number**: 1.0

**Current Version of Software**: Covid-19\_Tracker1.0

**Release Number**: 0

## 1.2 System overview.

The purpose of this software is to spread awareness and information about the Covid-19 virus along with information about the vaccines. The system will be a heatmap of the virus “hot zones” along with vaccination centers either on the same map or a separate map.

We will be intaking raw data from The Broadstreet Covid-19 Data Project and transforming that into a heat map to where it will be easily accessible and readily available to all users starting with the Maryland area and then possibly extending to other states but do not plan to expand out of the United States of America. We want people to be able to get information quickly and easily in a way that makes sense to them and helps make informed decisions on what to do and what not to do regarding Covid-19.

## 1.3 Document overview.

● Section 1: Scope

● Section 2: Referenced documents

● Section 3: CSCI-wide design decisions

# 2. Referenced Documents

* Restful API: <https://www.tutorialspoint.com/restful/restful_introduction.htm>
* Django: <https://www.digitalocean.com/community/tutorials/how-to-create-a-django-app-and-connect-it-to-a-database>
* Framework from Medium: <https://medium.com/swlh/build-your-first-rest-api-with-django-rest-framework-e394e39a482c>

# 3. CSCI-wide design decisions

## 3.1 Behavior:

Our Covid-Tracker will function through displaying data of Covid-19 cases along with information about vaccines. This data will be organized and displayed on a map organized by state and county, as such expected inputs for our program would fall under the names of states and counties within the United States. This will be done through our Menu unit which serves as the user’s interface. Within the Menu unit, along with choosing location, users will also have the option to choose *cases, vaccines, vaccines and cases.* Unexpected inputs such as false names or names in which we hold no information on, would inform the user that we hold no information on this area. As for expected inputs, once a valid state and/or county have been given, the program will search the database to pull up the statistical information from the inputs given and take the user to their location visually with the Map unit, having the data displayed as a heatmap. Once arrived Covid-19 cases and/or vaccine information from the database will be displayed for the user to see of the area they had chosen. This would be what an expected run of our program would look like.

## 3.2 User Access:

Users will be able to access our application through their web browsers as our application will be designed as a browser based client-server architecture. The database for both covid and vaccine information will be stored on our end while the user will be asked to input the location they wish to see information on through our Menu unit. Once that input is given our database will send the corresponding information for that area onto the user’s side and be displayed as a visual map representation through our Map unit. Disadvantages to this approach consist of potential bandwidth and congestion issues which would cause the application to lag on the user’s end. However, the benefits would be an easy access to update our data as covid and vaccine information will be fluctuating.

## 3.3 Constraints:

Due to our large database with both covid and vaccine information we have to be cautious with our data storage capacity. Given that we may also plan to expand our database beyond just Maryland we must always take into account how much storage is being used and how much we plan to use. Our program will be open to the public, as such our bandwidth capacity would be another constraint to prevent slow response times in our queries.

## 3.4 Safety, Security and Privacy:

In terms of privacy our covid and vaccine data will only display numerical data, no personal information of patients, names of deceased, or any other private information will be displayed. This program will be open to the public without the need to create personal accounts so there is no fear in leaking our users information. In terms of safety, our goal with our data is to help educate the public and prevent the spread of the covid virus, we do not condone any type of misuse with our program.

# 4. CSCI architectural design

## 

## 4.1 CSCI components

The following subsections address the hardware and software components of the UMBC Covid-19 Tracker web application. These components refer to the required hardware needed to run the program and the interactive tools available to the user.

The hardware resources used are as follows:

* Local computers: The machine that hosts the Covid-19 Tracker app.

The software resources used are as follows:

* Menu Unit: Designed menu options that allows the user to search a specific area/ state. Once a state is chosen, decide between observing COVID cases, vaccines, or both. Tentatively, the user should have the option to select a hyperlink
* Statistics Unit: The number of cases, vaccines, and both for a given searched area.
* Map Unit: Once a state/ area is chosen, the heat map unit allows for a visual representation of the states covid and/or vaccine data

## 

### 4.1.1 Hardware usage

The UMBC Covid-19 Tracker should be able to work on a variety of devices that meet or exceed the requirements of:

* Operating System Requirements:

1. Windows OS (7 or higher)
2. MacOS (10 or higher)
3. Ubuntu or Linux (Newest versions)

* Software Requirements:

1. Modern web browser, such as Google Chrome

### 4.1.2 Menu unit (CT\_ME)

The menu starts out with one option: Search for a specific area of interest. Once a valid area is selected, another menu will pop up with three options: *cases*, *vaccines*, and *cases & vaccines* as tabs on the webpage.

### 4.1.3 Statistics unit (CT\_ST)

When a given area is selected, all the statistical data for cases and vaccines will be loaded. It will be up to the users to decide which data they want to visualize. For covid statistics the types of totals received will be among the total cases, deaths, probable cases (if applicable).

### 4.1.4 Map unit (CT\_MA)

When the location’s type of data to visualize is selected, a heatmap will be generated representing the requested data with the corresponding location.

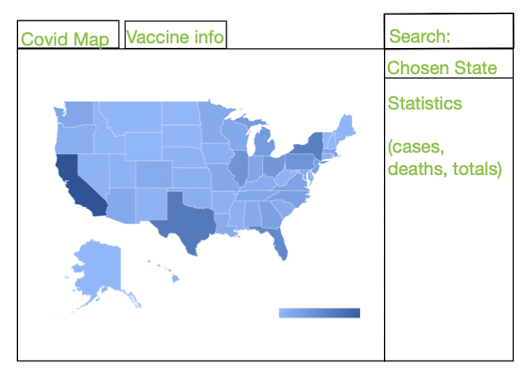


Figure 1

## 4.2 Concept of execution

### 4.2.1 Flow of execution

The state diagram below is a depiction of the transition between program states during execution. When the program loads up, it is initialized to the home screen. Data is loaded as requested, such as locations and specific data of interest.



Figure 2

### 4.2.2 Software unit control flow

The control flow of the state diagram below shows how different software units pass around control in the program

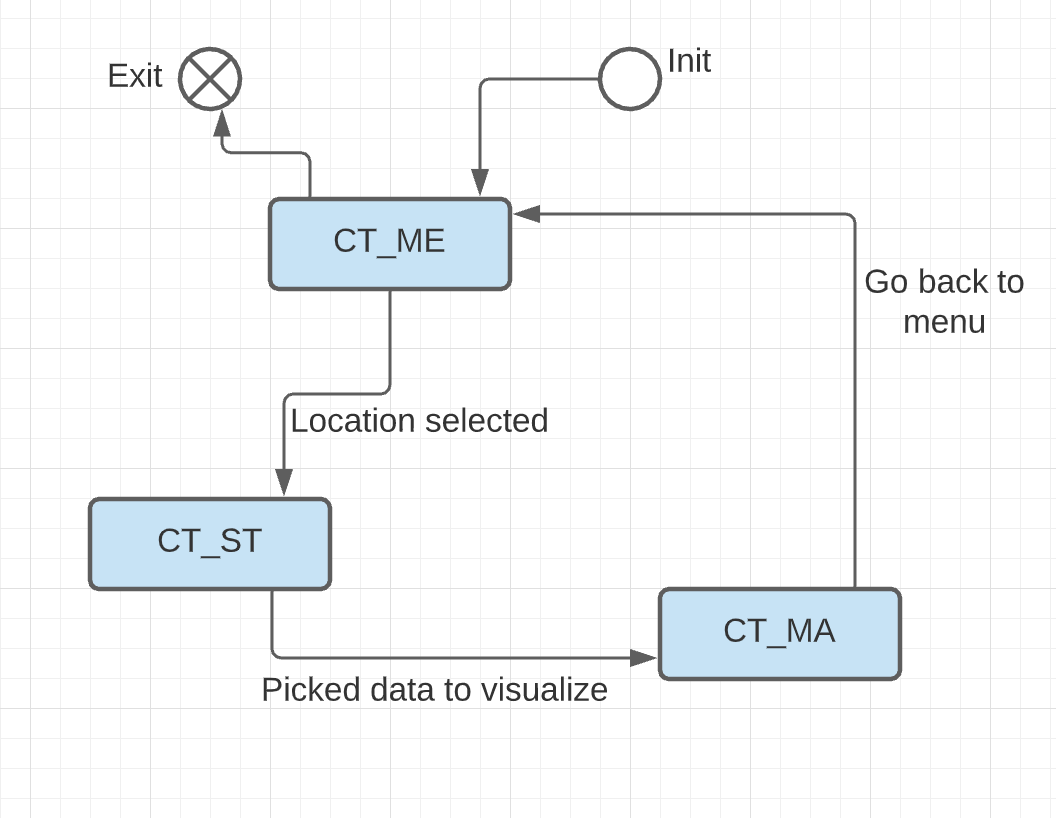


Figure 3

## 4.3 Interface design

### 4.3.1 Interface identification and diagrams

The UMBC Covid-19 Data tracker consists of 3 interfaces.

1. The Covid-19 Tracker API (4.3.1.1)
2. DjangoDB API (4.3.1.2)
3. Broadstreet covid data Interface (4.3.1.3)

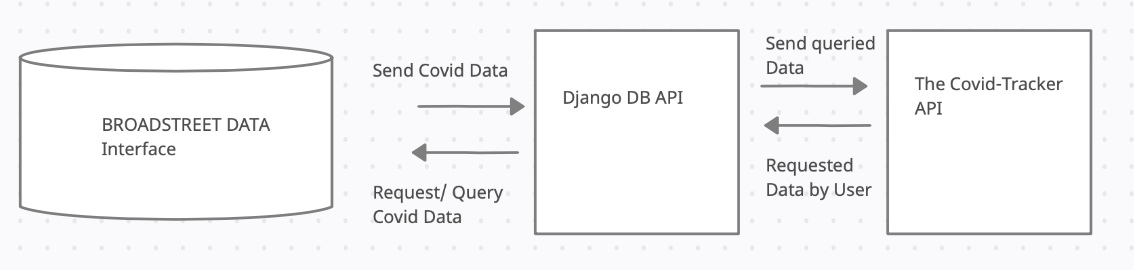


Figure 4

#### 4.3.1.1 The Covid-19 Data Tracker API

The Application Interface is the main request initializer for data queries and application display. This API will be the interface that requests data from the other interfaces and is the actual web application available to the user. The Covid-19 Data Tracker API shall request a specific states data among three different options: vaccine, covid cases, cases and vaccines.

After receiving the data from the Broadstreet and DjangoDB APIs , the specific data will be loaded and displayed in a heat map. Optional/ tentative task is displaying vaccine information based on the different types available.

#### 4.3.1.2 DjangoDB API

The DjangoDB API is the interface that will allow the CDT (UMBC Covid Data Tracker) to load data (see 4.3.1.1). This interface will start by aggregating covid case and vaccination and storing them into its database. Because, the data will be of different types, such as case data which will have categories of date, confirmed cases, deaths, etc, this will slightly simplify the aggregation and organization. Same idea applies to the vaccine data.

#### 4.3.1.3 Broadstreet Covid Data Interface

The Broadstreet API is the interface of the collected and refined covid-19 related data. It provides the data that is queried by the DjangoDB API (see 4.3.1.2). This interface includes github repositories and data sheets that are organized by region, and state.

\*The following paragraphs go into more detail about the above interfaces characteristics

### 4.3.2 Project Unique Identifier for The Covid-19 Data Tracker API (see 4.3.1.1)

The CDT API characteristics will include the following items as part of the user experience and overall design of the web application. We will define this interface as a level 2 importance since it is the first and only interaction between the user and the previously defined menu tab options. However, the display and data is dependent on the received covid and vaccine information given by the DjangoDB, which is also dependent on the Broadstreet API. This will serve as the main user interface and a retrieval mechanism.

The CDT API will take in an input string (can be empty as a default) that will be stored as the requested region/state variable. The output of this API will be the heat map of the given state as well as the statistics for that state. The statistics will be in string and integer format for state/ county names and totals by categories mentioned above (see 4.1.3) respectively.

### 4.3.3 Project Unique Identifier for DjangoDB API (see 4.3.1.2)

The DjangoDB API will load the data for the CDT from the Broadstreet API. Will organize data by Covid cases, vaccinations, and both combined.

### 4.3.4 Project Unique Identifier for BroadStreet Covid Data Interface (see 4.3.1.3)

The BroadStreet Covid Data Interface will include the following characteristics as part of the CDT web application interfaces. We shall identify this interface as a priority level 1 since it contains the data that is needed in order for the tracker to output updated and useful information about the Covid-19 pandemic. This interface is a collection of data sheets that will be aggregated, sent, and loaded into the DjangoDB API (see 4.3.1.2) in order to be used in this web application. This will be used in real time data transfer/ collection (tentative as we work out the details of efficiency when loading large sums of data). This data sheet includes strings made up of the states, counties, collected categories (based on the previous two), and the numerical values of said categories.

# 5. CSCI detailed design

## 5.1 CT\_ME

The menu unit is responsible for interacting with the other software units and processing user input. The menu unit receives user input specifically text entered in the geographical selector and clicking of the different option buttons. Once the user inputs the location information, the state of the Map and Statistical units are changed and updated with the selections surrounding covid/vaccination data. An example of user interaction is described in the use case diagram of Figure 5 and further mapped out in the activity diagram of Figure 6. All the units are planned to be written as individual react modules in Javascript.

## 5.2 CT\_ST

The statistics unit displays relevant statistical information about the default region (the U.S. / Maryland) or the user selected region. This will include all available information provided by 3rd party covid case and vaccination databases. This includes case count, total number of deaths, and hospitalizations. When vaccination data is selected, the total number of vaccines given out will also be displayed. All the units are planned to be written as individual react modules in Javascript.

## 5.3 CT\_MA

The map unit visualizes relevant case data through a heatmap layered on top of an interactable map. This map will most likely be restricted to visualizing the U.S. as quality of data collection is most consistent within the country. Once user input is received from the CT\_ME the heatmap data will be re-rendered for the region and zoom you in to a more narrow field of view. All the units are planned to be written as individual react modules in Javascript.

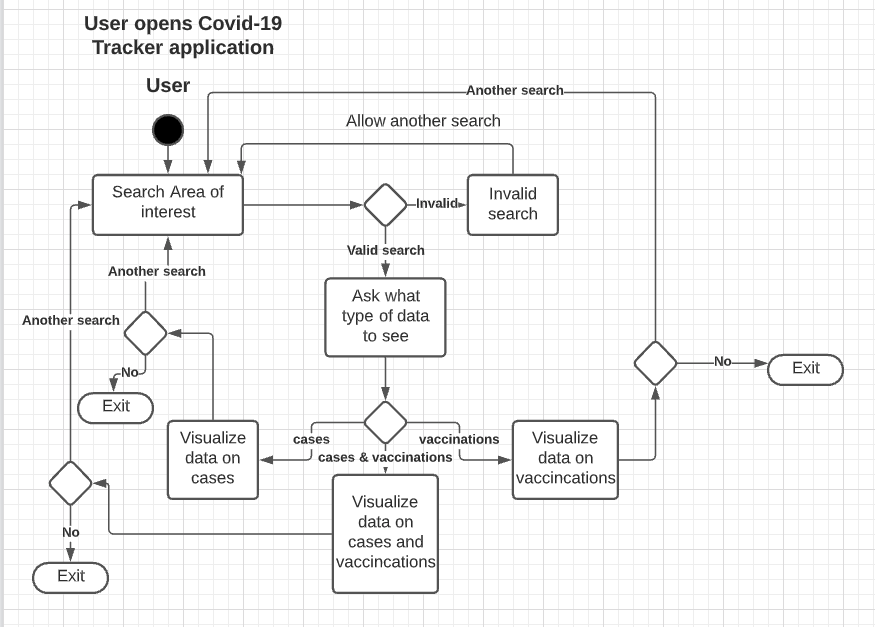


Figure 5

| Name | Access covid cases with the UMBC covid-19 tracker |
| --- | --- |
| ID | CT\_ME\_UserInputRegion |
| Description | The user accessed the cases dataset using their current region as a reference point. |
| Actors | User |
| Organizational Benefits | This use case is necessary for the program to serve any helpful function. Without input of region, the map could not be interacted with in a meaningful way |
| Frequency of Use | High Use. Users will want to narrow their result findings down to a specific region at a time |
| Triggers | User has navigated to the home page of the covid tracker and input a region they wish to view |
| Preconditions | Geographical data has been given permission in the browser |
| Postconditions | Covid case data is sent to the user based off of their desired location |
| Main Course | 1. User selects a region in which they wish to receive covid data 2. Data is queried to find matching region data for cases 3. The data is returned and mapped to a heatmap on the web app 4. User can interact with the visual representation |
| Alternative Courses | 1. Users can change the region and query more data 2. Users can change filtering for confirmed cases / deaths/ hospitalizations |
| Exceptions | If there is no regional data for the area requested by the user, the heatmap will fail to populate. The user can continue requesting other regional data as a result. |

Figure 6

# 6. Requirements traceability.

This section shall contain:

1.Traceability from each software unit identified in this SDD to the CSCI requirements

allocated to it.

2.Traceability from each CSCI requirement to the software units to which it is allocated.

| **Req.** | **Description** | **SRS Par. #** | **Software Unit** |
| --- | --- | --- | --- |
| *CT\_F\_1* | Provide GUI allowing the user to view current covid data | 3.2.1 | CT\_MA |
| *CT\_F\_2* | Database queryable in an acceptable time | 3.2.2 | CT\_ME |
| *CT\_F\_3* | Database queryable as a result of user input | 3.2.3 | CT\_ME |
| *CT\_P\_1* | The application should remain operational | 3.3.1 | CT\_ME, CT\_ST, CT\_MA |
| *CT\_C\_1* | The application should query the database and get the newest data upon web access | 3.4.1 | CT\_ME |
| *CT\_C\_2* | The application should provide data access to users | 3.4.2 | CT\_ME |

Table 1

# 7. Notes.

We may add functionality based on our conversation with our client Sara. Any additions will be reflected in an updated version of this SDD.

# A. Appendixes

There are no appendices for this document as of yet.