Comprehensive Monkeypox Dashboard

User's Manual

version 1.0

Prepared by
Saketh Dendi, Felix Huang, and Sakin Kirti (Dendi Huang Kirti Group)
CSDS 393
09 December 2022

Table of Contents

Introduction

1.1 Purpose

This document serves to provide simple instructions for anyone interested in using our Monkeypox Dashboard. Functions include viewing data in the form of a map, where the number of cases can be viewed according to the colors given in the legend. Additionally, the user can use the Chart View which shows the cumulative cases as well as 7-day moving average of the the new case counts. Lastly, the user can use the About page to view our data sources as well as some other helpful information. This document includes helpful information for each of these pages which allows the user to more easily and efficiently use our Monkeypox Dashboard.

1.2 Document Conventions

The formatting of this document is done the following way to stay consistent and allow any viewer to quickly orient themselves.

- Heading: Google Docs' *Heading 1* Format (20pt Calibri, black)
- Section Headers: Google Docs *Heading 2* Format (16pt Calibri, black)
- Subsection Headers: Google Docs' *Heading 3* Format (14pt Calibri, black)
- Body: 12pt Calibri, black

Accessing the Dashboard

In order to access the website, please clone our Git repository, download Python (v3.10.6) if necessary, and execute the following commands in the terminal at the root of the repository:

```
>>> cd backend
>>> pip install -r requirements.txt
>>> flask run
```

Monkeypox Dashboard can then be accessed at http://127.0.0.1:5000/

If this does not work, please also execute the following commands at the root of the repository.

```
>>> cd frontend
>>> npm install
>>> npm start
```

The dashboard should then be accessible at http://localhost:3000/.

Monkeypox Dashboard Usage

3.1 MapView

The MapView also doubles as the homepage. This is the page that is automatically rendered when you visit our page. Here, there are several pieces of information that one can find about monkeypox.

1. First, notice on the left side of the screen, a window denoting various state names. This window can be scrolled through to find a state of interest. Additionally, there is a search bar to allow a user to search for a state of interest. When a user clicks on the name of state, notice that the MapView auto-zooms in onto the state of interest and shows a miniature chart displaying the cumulative case counts for that state.
2. Notice, also, a small red dot near the center of each state. Clicking on this dot brings up the same pop-up showing a miniature display of case counts for that state.
3. Notice that each state is colored a different color. These are to bin each state by the number of cases. Notice the legend of the right side of the screen, linking a color with a bucket to the number of cases. This color can be used to shed light on states with greater or fewer cases.
4. Next, notice along the bottom a window titled “National Public Health Statistics for Today”. This displays the incidence rate, prevalence rate, and case-fatality ratio based on the new data pulled today.
5. Lastly, notice a similar window titled “National Predictive Statistics for Next Week”. This displays the same three statistics — incidence rate, prevalence rate, and case-fatality ratio for the next week based on our regression algorithm.

3.2 Chart View

The ChartView can be accessed by clicking on the link in the top right corner of the menu bar titled “Charts”. The ChartView allows the user a more complete view of the progression of monkeypox. Here, there are several functions that the user can take advantage of to learn more about monkeypox.

1. First, similar to the MapView, notice a window on the left side of the screen that allows the user to filter the Chart by state name. This can be done the same way as outlined in section 2.1 MapView point 1.
2. Next, notice at the top of the screen, two buttons titled “Cumulative” and “7-day Moving Average”. These allow the user to toggle between viewing cumulative case counts and the 7-day moving average of new cases.

3. Regardless of which view the user is using — cumulative or 7-day moving average — notice that two different pieces of data are being shown. First, there is the past data shown in dark blue and our 14-day predicted counts made in light blue. The 14-day forecast is made based on our predictive algorithm which used the previous data to predict how monkeypox might progress.
4. Lastly, on the right side, notice a constant count displaying the total number of cases in the United States on this day. This will not change for each state. This window also acts as a button that will allow a user to return to the cumulative and 7-day moving average charts for the entire United States.

3.3 Help

Lastly, we include a short help page that can be navigated to via the “Help” button in the top right corner. This displays some text that gives the user some more information about why this project was built, the data source, a small map guide, and a small chart guide. Notice, under the “Information about Data Sources” section, that the words “Global.Health” and “CDC Data” are also hyperlinks. Each of these links will take you to the page where we found our data.

AWS Tools

To make use of the most up-to-date data provided by the CDC, we pull data from each of our APIs daily. To do so efficiently, we made use of two AWS services.

4.1 AWS Relational Database

First, we used the Relational Database (RDS) service extensively. This gave us a central, cloud-based storage location to store the case counts, predictions, and public health stats.

4.2 AWS Elastic Compute Cloud

Because we update our RDS daily, we needed some automated way to perform this action. To do so, we made use of the Elastic Compute Cloud (EC2) service that AWS provides. Here, we leave the EC2 console running and once the update time is reached daily, our code is automatically run to get the newest data from the CDC and update our RDS.

Custom Flask API

Additionally, we built a custom API endpoint to retrieve data from the database to make it accessible for our frontend to access. This provides fast and efficient loading of the site.

Bug Tracking

In order to track bugs effectively and efficiently, we made use of GitHub's Issues, Dependabot, and Code Scanning softwares.

6.1 Issues

We used GitHub's built in bug tracking software to track any issues that arose manually. This allowed us to customize who we assigned the bug to, create labels and tags for the issue so that various team members could jump in whenever and wherever they could. Please see the "Issues" tab of our github page to see the list of issues that we logged.

6.2 Dependabot

We also made use of GitHub's Dependabot software which provided us with the ability to check for any dependencies that existed in our code. This gave us the ability to reduce these wherever possible to ensure that our code would work in case any of these were to go down. Please click the button titled "Dependabot" from the "Security" tab of our github page.

6.3 Code Scanning

We also used github's Code Scanning software which, each time we pushed code to the repository, the software would scan our code and notify us of any bugs that it found. This worked great as a backup in case we missed any bugs manually. Please click the button titled "Code Scanning" from the "Security" tab of our github page.