# Technical Report

By Cameron Doggett, Jeffrey Moulckers,
Shrivu Shankar, Vassi Gianitsos, and Yash Shenvi Kakodkar

## Summary & Motivations

The coronavirus has negatively impacted millions of people and hugely altered the world in which we live. Our team is interested in dissecting the different outcomes brought by contracting the coronavirus relative to the many locations the virus reaches. As a result, we built our website to serve as a database exposing different countries, potential risks, and case statistics surrounding the coronavirus. We hope that users realize the reality of the data we provide. We believe this site will inform users with relevant data that will hopefully motivate people to be careful about social distancing and spreading viruses in general. Our site allows users to categorize data based on country, making relevant statistics, such as total cases or total deaths, easily comparable across locations around the world.

## User Stories

We were given 5 user stories by our customer and they are listed as follows: capital, news page, graphs, testing info, and a map UI. We incorporated these stories by embedding them within the attributes of various models and by giving a few their own separate pages. The capital of a country was presented in each individual country instance. We list its name, show a picture of it, and provide a map of the area as well. For the news page user story, we created a separate news page that can be accessed through the navbar. This news is a compiled list of various COVID-19 related news from around the world and is presented in a card-style UI. News for a specific country can also be accessed within that specific country's instance. To meet the graph requirement, we display a line graph using the NOVI module that shows daily cases in 2020 per country in each case data instance. We plan to add more graphs for other types of data in future phases. To meet the testing info user story, we put total test data as an attribute within the case instances. Lastly, we put maps in several places within the site. The first being the splash page and then a map of the country or capital in each instance page of each model. Although the map UI user story asks to develop a heat map that highlights countries based on their COVID-19 case totals, this will be a story that will be pushed onto the next page.

## Our REST API

Our RESTful API was designed first-and-foremost to meet the needs of our frontend application. The demands of the frontend portion of this project necessitated the creation of six endpoints, which are shown in the table below. Each model was assigned two endpoints that respond to GET requests in order to reflect the demands to retrieve all instances of that model and another to retrieve a single instance of the model by a unique country identifier. The former is very useful for presenting all instances of that model on a single page, such as the model pages with tables/grids, whilst the latter saves on server bandwidth and helps meet the need to display information on a single instance of that model, such as on the individual instance pages. Across all of these endpoints, an optional attribute query parameter enables the frontend client to specify exactly which fields they require in the response. This simultaneously lowers server bandwidth and serves to specify exactly which attributes the client can expect in the response. These endpoints and parameters were sufficient for phase one, but we do expect to design more as the needs of the frontend increase.

API endpoints table

| Endpoint | HTTP Method | Description | Path Parameters | Query Parameters |
|---|---|---|---|---|
| /countries | GET | Get all countries | N/A | attributes |
| /countries/:identifier | GET | Get a country | identifier | attributes |
| /case-statistics | GET | Get all instances of case statistics | N/A | attributes |
| /case-statistics/:identifier | GET | Get an instance of case statistics | identifier | attributes |
| /risk-factor-statistics | GET | Get all instances of risk factor statistics | N/A | attributes |
| /risk-factor-statistics/:identifier | GET | Get an instance of risk factor statistics | identifier | attributes |

The second goal we had in mind while designing the API was to allow any external client to easily interface and retrieve data from our database. To that end, our API does not currently require any kind of API key in order to make requests. Additionally, we provide

thorough documentation of the COVID-19 DB API on Postman (see tools section for more details) and our project's repository, so that anyone can learn to make requests to our API.

## Our Models

Each model targets a specific grouping of data and statistics to localize information for a user to their specific country. This data is essential in understanding trends on a country to country basis and providing context as to why certain states have handled the pandemic more effectively than others.

### Countries

As COVID-19 is a global issue, we decided to create a model representing each country to effectively group statistics for local reference. Each country instance contains information on its location, currency, time zone, bordering countries, naming, and reference details, and subregion relative to the world. Additionally, each country instance has a 1:1 relationship with that country's corresponding case statistics and risk factors.

### Case Statistics

Our case statistics include quantitative information specific to a country including total and active cases, tests, deaths, and recoveries. We maintain updated statistics by presenting their current values and comparing them to those of yesterday. Along with day to day statistics we also display the rate of change of many factors and present the country's case totals in an interactive graph. Each case statistics instance has reference to its corresponding country, as well as the country's risk factors.

### Risk Factors

Risk factor statistics represent some of the health and human development-related factors associated with a country and may be utilized to indicate if portions of the country's population are more susceptible to a fatal case of COVID-19 or if the country is less prepared to handle large outbreaks. Examples of attributes found within risk factor statistics instances include GDP per capita, GINI, diabetes prevalence, cardiovascular disease death rate, share of male smokers, life expectancy, hospital beds per thousand, and more.

## Tools Used

### Postman
In this first phase, Postman was used to generate an API documentation collection and to make requests to our API data sources for scraping. Documentation was created on Postman by defining an OpenAPI specification which outlines our API design and then generating a collection that corresponds to that definition. The API specification is named api-spec.json in our public repository. Second, we made GET requests to the API data sources from Postman and saved the responses to be used for our model instances.

### React
We use Facebook's React UI library for all of the UI shown on the site. The framework allows us to build modular and composable components that can be adjusted and adapted to different parts of the site. We write components in JSX (a mixture of JavaScript and inline HTML) which are then webpacked into JavaScript bundles and served to the user.

### AntDesign
We utilize the pre-made and dynamic components provided by the AntDesign React UI library to create a more standardized interface and aesthetic across the site. Many of our most important elements, including the tables, grids, and buttons, are backed by this library in our Phase I implementation.

### React-Bootstrap
Another UI Library supporting our implementation is React-Bootstrap, which supports our main layout, navbar, and site navigation with an added React-Bootstrap-Router plugin. We used this framework to get a quick and light static implementation up, but we may shift away from it as we take on the next phases.

### GitLab
All of our code, including both frontend and backend, is hosted in a public repository on GitLab. GitLab enables us as developers to collaborate efficiently because it has a wide range of useful features including: an issue tracker, support for tracking a multitude of separate development branches, continuous integration, and more. We will continue to develop within the same repository throughout the lifecycle of the project.

### Flask

While backend development has yet to begin, it is worth noting that we will be using Python's Flask and Flask-Restless frameworks to build it in the upcoming phases.

## Hosting

We chose the Google Cloud Platform (GCP) for hosting because of its developer friendliness and available credits. Within GCP, we use two separate App Engine services for serving static web packed frontend assets and for running our Flask-based REST API. App Engine handles SSL certificate generation, orchestration, and deployment for us. We are using GoDaddy for managing our domain's nameservers and have a setup A, AAAA, and CNAME directives for routing traffic to GCP machines. App Engine's dispatching feature is used to direct this traffic to either the API or frontend service based on a set of pattern matching rules.