UNSW
SYDNEY

SENG3011 T1 2021

Management Information

Team: Kool Kats

Mentor: Mirette Saleh

Team members:

Lachlan Stewart

z5207906

Pericles Telemachou

z5214738

Stephen Comino

z3470429

Joanne Chen

z5207491

Weiting Han

z5210581

Table of Contents

Web Application (DiSeeker)

Web Application Use Cases

Target Audience:

- Not-for-profits e.g. Doctors Without Borders, Red Cross, Médecins Du Monde
- Government health departments to track the disease response of other countries

Other possible uses:

- Individuals who are travelling interstate within Australia and need to identify travel restrictions
- Individuals looking to travel to different countries to see disease outbreaks and impact of COVID-19

Persona: Sam (Strategic Planner from Doctors without borders)

Goal: To determine where to allocate the organisation's resources to have the most impact for the delivery of vaccinations

- User Story 1: As a strategic planner I would like to be able to specify multiple different diseases so that I can see how to allocate resources in the world for different diseases
    Example
    - Sam also wants to find out about the countries that have been affected by Yellow Fever to plan his NFPs funding
    - He goes back to the "Disease", "Start Date" and "End Date" filters
    - He adds the disease Yellow Fever to the disease list and broadens the date range to the last year
    - The map automatically updates to reflect his changes
    - Sam can then see which countries have been worst affected by Yellow Fever and factor it into his NFPs budget planning

- User Story 2: As a strategic planner I would like to be able to see which areas in the world are worst affected by a given disease so that I can plan the budget of the NFP I work for effectively
    Example
    - Sam wants to find out which countries have been most affected by COVID in the last year, so that he can plan how to allocate his NFPs funding.
    - He goes onto DiSeeker and narrows the disease to "COVID-19" and restricts the date range to the last year
    - After the map updates, he clicks on the region that is the most heavily shaded to bring up more detailed information about it
        - Cases
        - Vaccination percentage (COVID)
        - Average change in cases by month

- User Story 3: Planning fundraising events in Australia and has to travel interstate.

I would like to be able to see each state's travel restrictions so that I can plan my travel ahead of time

Example

- Sam lives in Australia and will also be travelling interstate for his work, so he wants to find out what the travel restriction are between states
- He goes onto DiSeeker
- Clicks on Australia
- He clicks on the "Restrictions" button and select "VIC" from the drop down
- The current restrictions for that state come up
- He can then plan his travel around the restrictions

Web Application Design and Implementation

Final Web Application Architecture

- React JS framework for the main frontend development
- React-bootstrap for styling
- Yarn as the dependency manager for the web application
- AWS EC2 Instance for hosting the web application

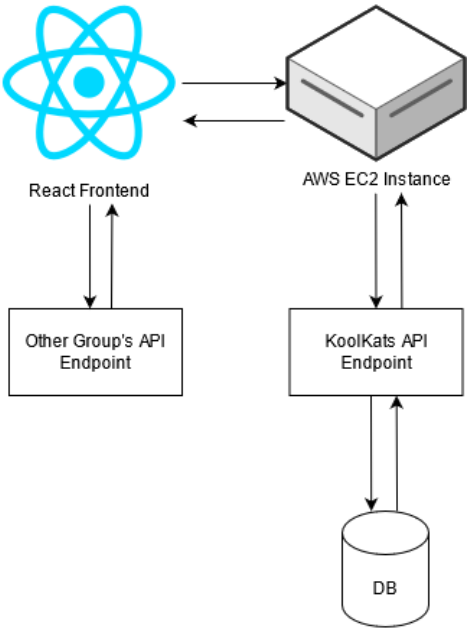

Figure 1 - Web Application Architecture

Choice of Implementation

AWS EC2

We have decided to use an AWS EC2 instance to host our web application because it is easily integratable into the instance we have to host our API endpoints.

React JS

React JS allows us to create reusable components, allowing more robust applications. Also, React JS has a virtual DOM which allows the UI to update more quickly and efficiently. Furthermore, all of our team members have used React JS before and are more familiar with it. As a result, there would be less of a learning curve allowing more features to be implemented.

React Modules

React simple maps

React simple maps provides pre-made components for us to easily create a zoomable map that we can modify using the data from our API

React high charts

React high charts provide a template component for us to easily insert information and create graphs. By using this, we are able to dynamically update our charts (pie chart, line graph) as required.

Bootstrap

Bootstrap is a frontend framework with templates for UI elements such as buttons and forms. Bootstrap is open source and free, allowing us to easily design the user interface.

Yarn

We have decided to use Yarn as our dependency manager of npm due to its speed and efficiency. Yarn also allows us to autoclean to clear dependencies that aren't necessary, saving storage in the node_modules.

APIs

We decided to use 4 APIs, including the one we created, 2 from SENG3011 groups and 1 external API.

KoolKats API (SENG3011 Group)

We decided to use KoolKats API because we know the API well and have designed some of its endpoints to suit our needs. Our API also aggregates data such as case numbers to provide more useful insights.

Sour Dough's API (SENG3011 Group)

We decided to use Sour Dough's API to provide COVID vaccination percentage data for each country. We feel that this API will provide further valuable information to our users that'll enable them to not only recognise where diseases are most prevalent but also which countries have or haven't effectively rolled out COVID vaccinations.

Brickwalls API (SENG3011 Group)

We decided to use Brickwalls API to provide data for Australia interstate restrictions.

Disease.sh API

We decided to use disease.sh API as it provides up to date information on COVID-19 data on the world and different countries. Disease.sh uses data sources from John Hopkins University.

Shortcomings

As a lot of the API's used are external, we have little control of how updated their data is as well as the consistency of the data formats.

Challenges

One of the main challenges of developing DiSeeker was the development of all the visual graphs and charts as none of us had experience dealing with charts in the frontend before.

Using the APIs of other groups was also a challenge that we faced. Both groups had the CORs policy issue meaning we could not use their API. Only one of the two groups ended up fixing the issue so for the other group we had to use static data structured from their example response.

Another challenge we had was managing information received from external data sources as it was difficult to obtain valid data form the various APIs used.

We were also alerted that the WHO website that we were scraping would no longer be accessible from April 2021. As a result we had to speed up the process of web scraping so that we had the latest articles in our database before the web page was taken down.

Key Achievements/Benefits of Design

- Achievement: Scraping all articles from both the old WHO website and the new one
- Achievement: Incorporating 4 APIs into our design in total including our API, two other SENG3011 groups' APIs and a 3rd party API
- Benefit: Gives the user a visual representation of disease spread
- Benefit: Gives the user the ability to adjust the output with the "Disease" and "Time Period" parameters
-

Implementation

The application was built using React and Bootstrap due to their simplicity. The React Multiselect Dropdown package was used for the disease input field and the React Date Picker package was used for the end and start date selector. The React HighCharts

library was used for all graphs in the information panel such as the pie graph. React Simple Maps was used for the heat map.

Some example (figures 2-7) uses of DiSeeker:

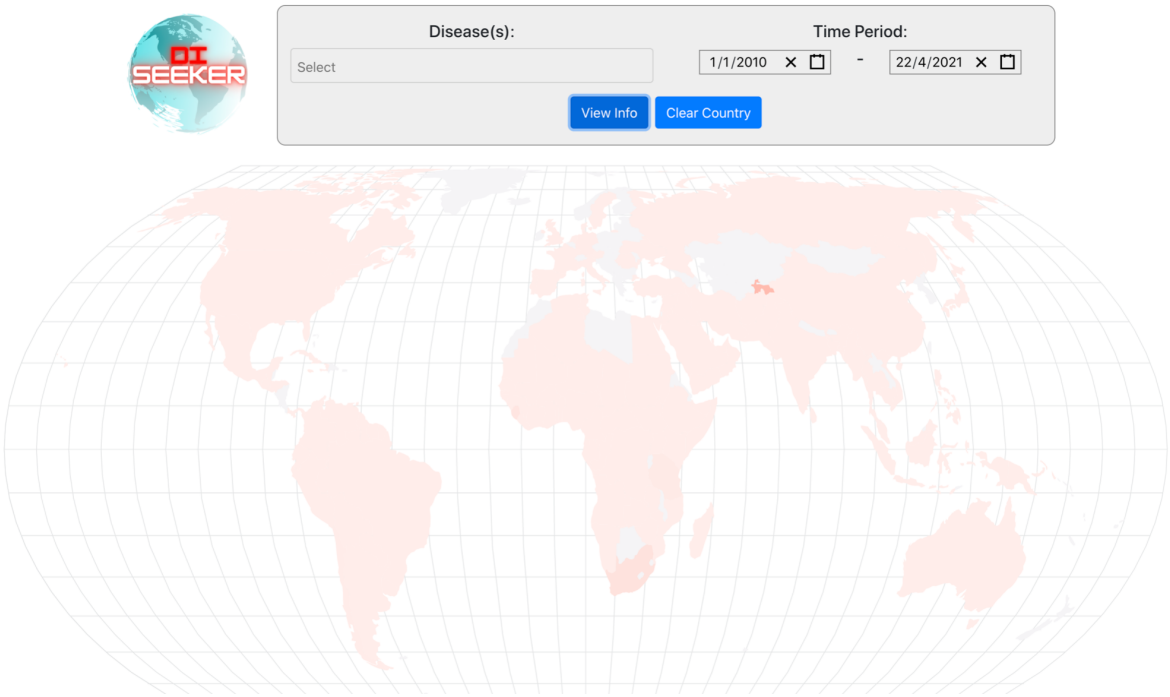

Figure 2 - DiSeeker home page

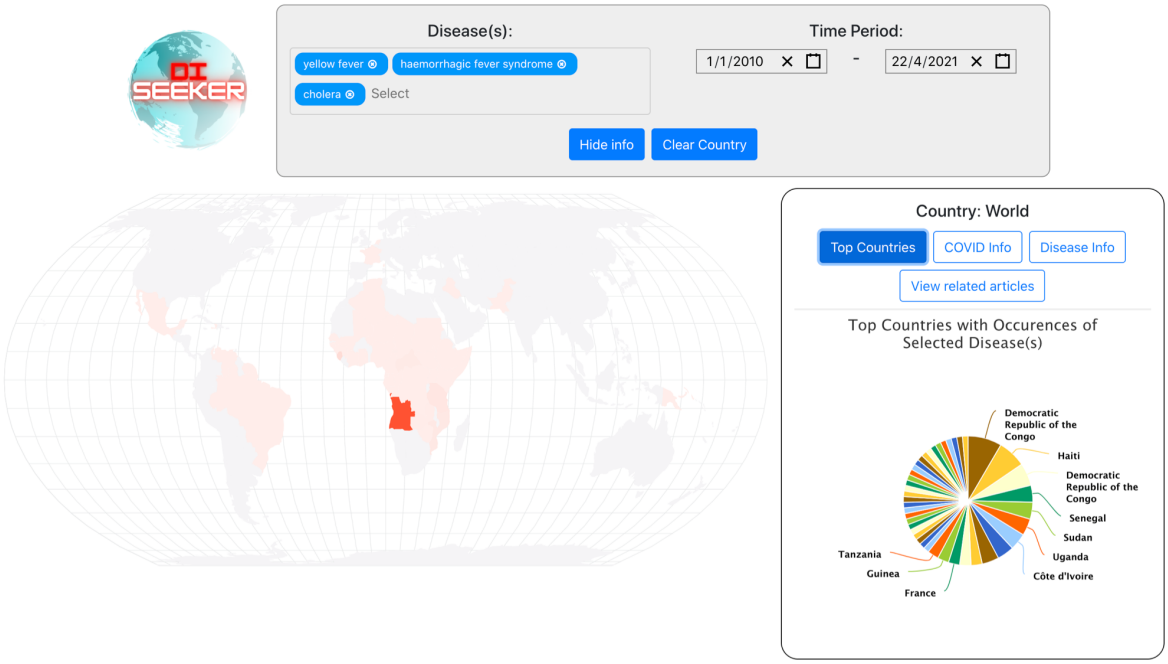

Figure 3 - DiSeeker diseases specified and information panel

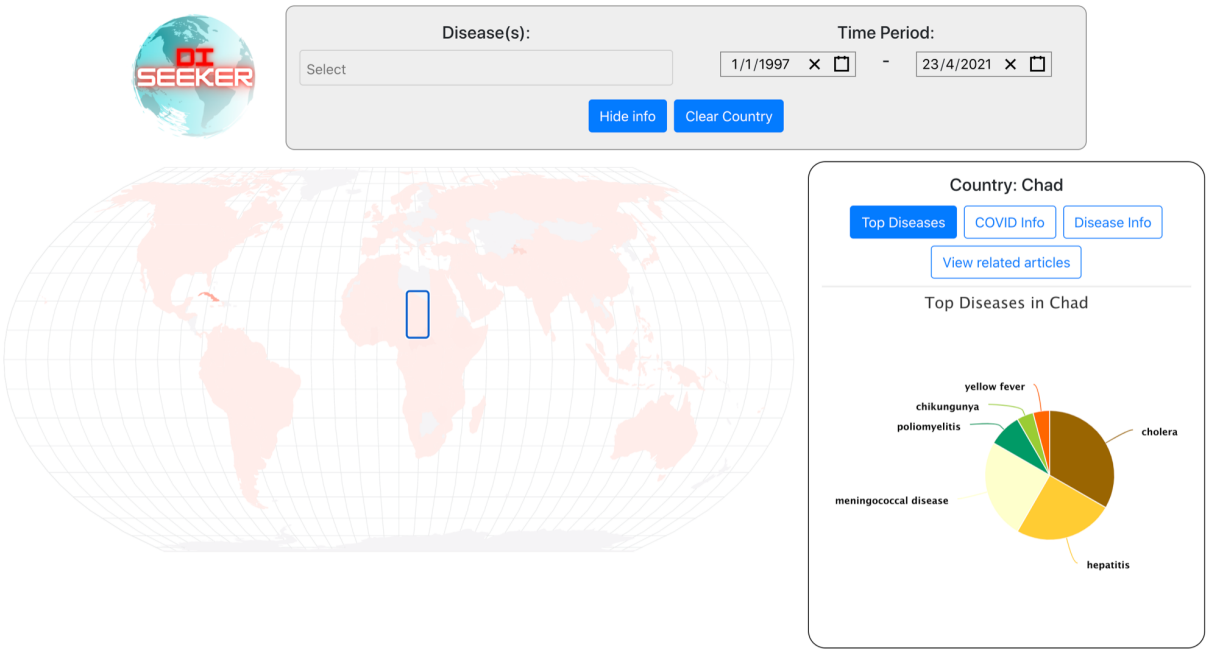

Figure 4 - DiSeeker information panel with country selected

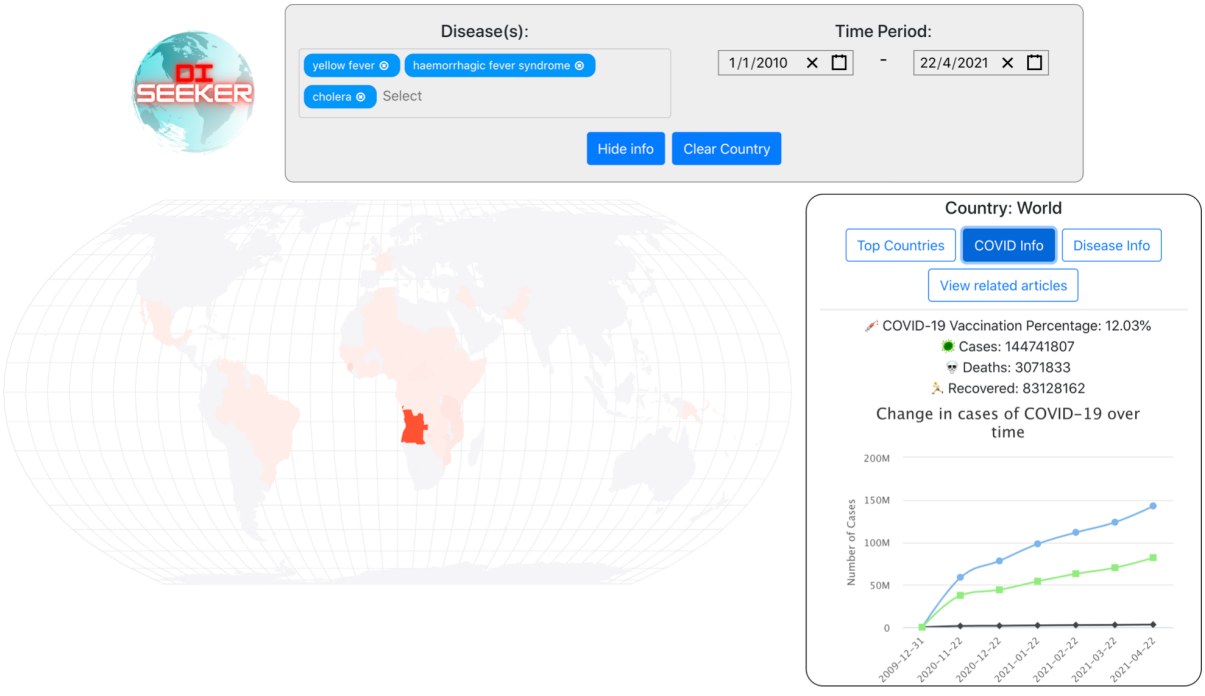

Figure 5 - DiSeeker COVID information panel

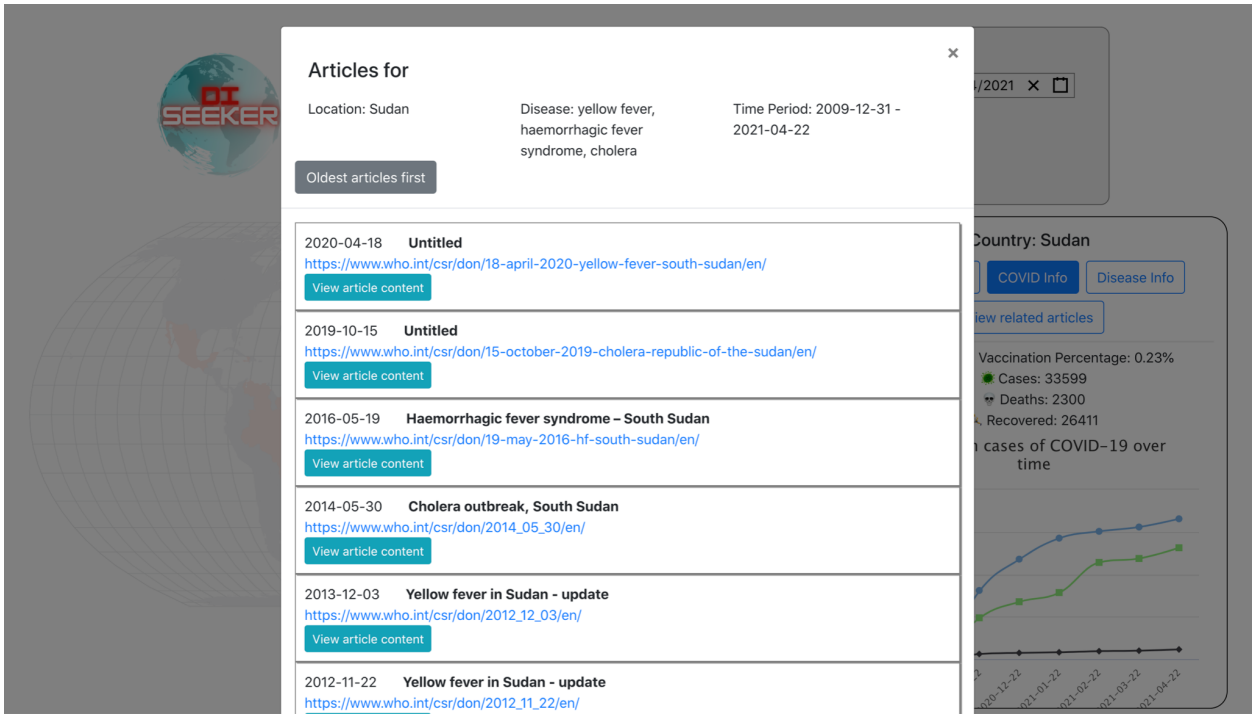

Figure 6 - DiSeeker articles popup

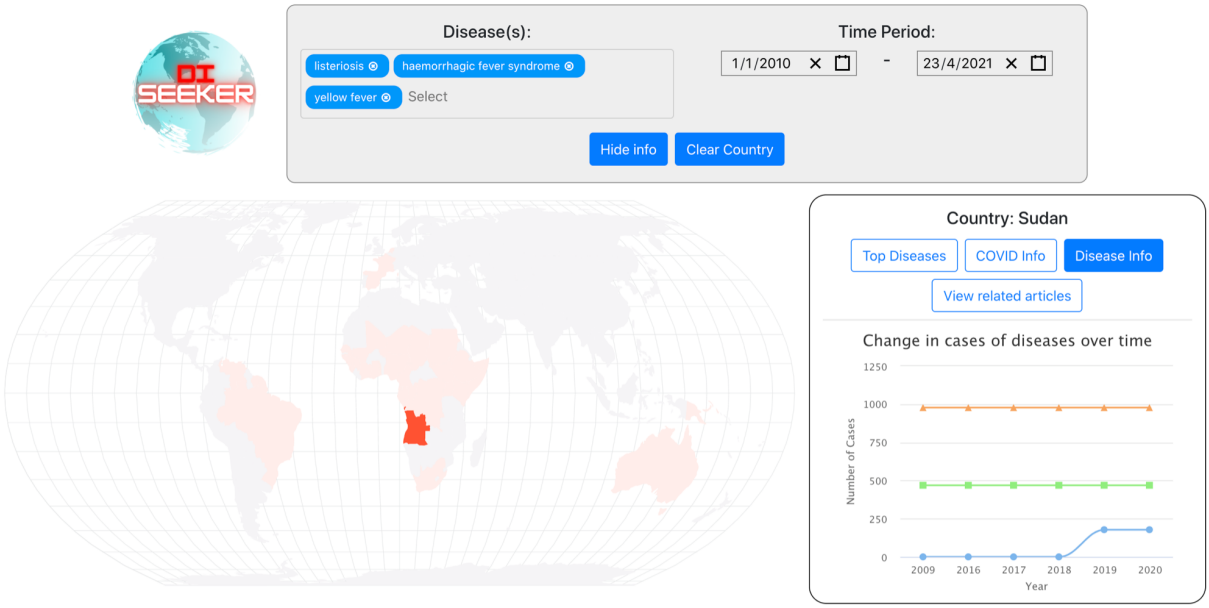

Figure 7 - DiSeeker information panel with case changes over time