SENG3011 – Epidemic Sandbox

Final report

– Team: WeByte!

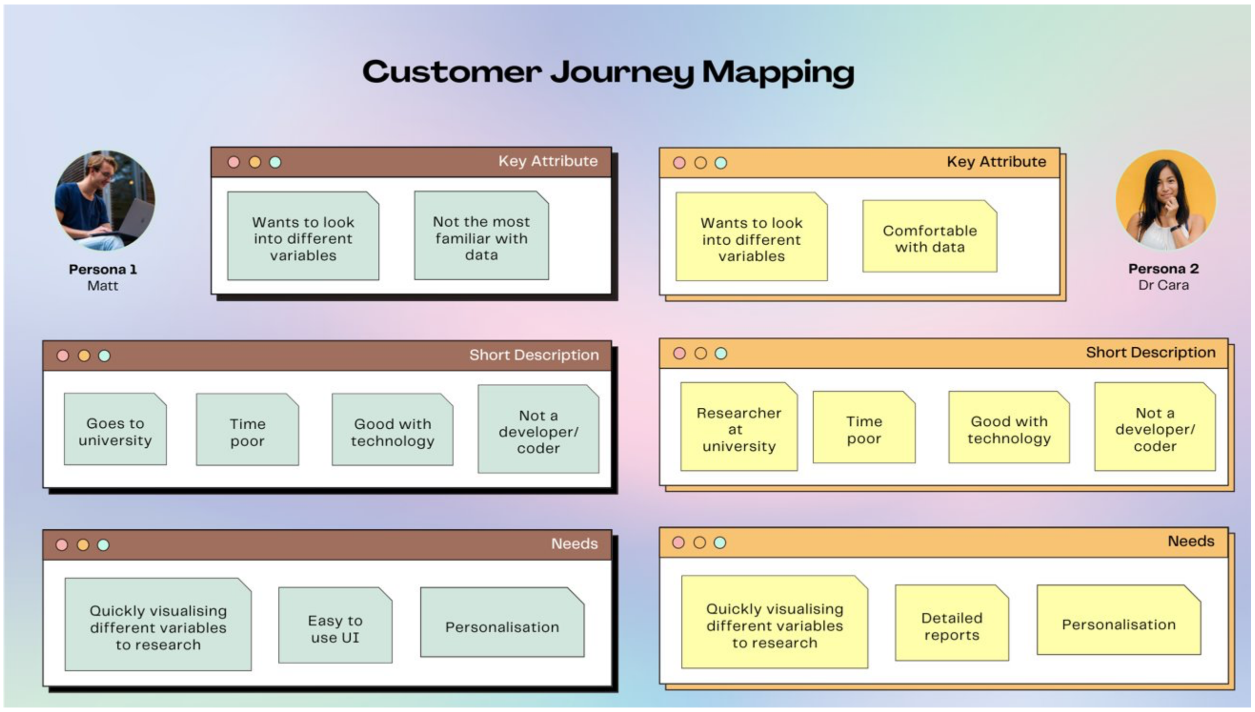
Team member’s name	zID
Harsimran Saini	z5208912
James Bradley	z5210562
Lachlan Harvey	z5258871
Sruti Desai	z5260319
Jaiki Pitt	z5213775

Table of Contents

Introduction

This report details the overview of the final product created by our team for the SENG3011 group project. We created a web application, called Epidemic Sandbox, for university students/researchers (our target users) who want to find relationships between different variables. Epidemic Sandbox provides visualisations, allows input of variables, performs analysis, provides predictions and creates reports on variables provided by us as well as user inputted variables.

Our user personas, as aforementioned, are university students/researchers and our application is designed with them in mind:



Our Persona 1, Matt, is a university student that is time poor, interested in connecting different variables and seeing them as visualisations but perhaps not the most familiar with data. Whereas, Persona 2, Dr. Cara, is a researcher at university and also time but she is comfortable with data and wants more personalisation and details. The user stories for this project attempt to find solutions for these wants while keeping the persona's requirements, such as an easy-to-use user interface, in mind.

Use cases

Epic Story: 1

As a user, I want to see visualisations of the relationship between two continuous variables.

US: 1
Title: Scatter plot

Priority: Very High

Description: As a user I want to be able to get a scatter plot of covid cases and the number of news reports on covid in a particular country so that I can see a relationship between the two.

Acceptance Criteria:

- Scatter plot of points from number of reports and covid cases in different countries
- Independent variable: Number of covid cases in a particular country
- Dependant variable: Number of news reports on covid in a particular country

US: 2
Title: Map

Priority: Very High

Description: As a user I want to be able to get a map showing covid cases and the number of news reports on covid in a particular country so that I can see a relationship between the two.

Acceptance Criteria:

- Map showcases dots which the user can hover over for more info
- Covid cases and news report info provided on the map

US: 3
Title: Plot other diseases

Priority: Low

Description: As a user I want to be able to get a visualisation showing relationship cases of a disease within a time period of my choosing so that I can see a how fast a disease spreads.

Acceptance Criteria:

- Allow input of a disease name
- Allow input of a start date (in the past)
- Allow input of an end date (more than start date but in the past)
- Graph number of cases against days (with day 0 being the start date) as a scatter plot

US: 4

Title: Plot my input

Priority: Medium

Description: As a user I want to be able to get a visualisation showing relationship two continuous variables of my choosing so that I can see a relationship between any two variables.

Acceptance Criteria:

- Link to a google sheets template that the user can edit
- Google sheets allows for 4 different types of graphs (line, pie, bar and scatter)
- Visualisation of the selected type showcased with the data provided

US: 5

Title: Google trends

Priority: Low

Description: As a user I want to be able to get a visualisation showing relationship covid cases and search interest of any keyword, so that I can see the relationship between these 2 variables.

Acceptance Criteria:

- Allow input of any keyword and graph its interest using google trends against covid cases
- Time frame: the past year

Epic Story: 2

As a user, I want to see analysis of the relationship between covid cases and a dependant continuous variable so that I can see the results are scientifically attained.

US: 6

Title: Regression analysis

Priority: Very High

Description: As a user I want to be able to get a regression analysis of the relationship between covid cases and a dependant continuous variable.

Acceptance Criteria:

- Showcases at the 0.05 level whether there is a significant relationship between the two variables
- Showcases trendline on the scatter plot

US: 7

Title: Prediction

Priority: Medium

Description: As a user I want to be able to get a prediction on the number of cases of a disease on a particular data so that I can be informed.

Acceptance Criteria:

- Showcases at the 0.05 level whether there is a significant relationship between the two variables
- Predict using trendline what the number of cases on a particular day would be if the trend were to continue

Epic Story: 3

As a user, I want to create reports on my analysis so that I can download to make it easier to use the results.

US: 8

Title: Reports

Priority: Medium

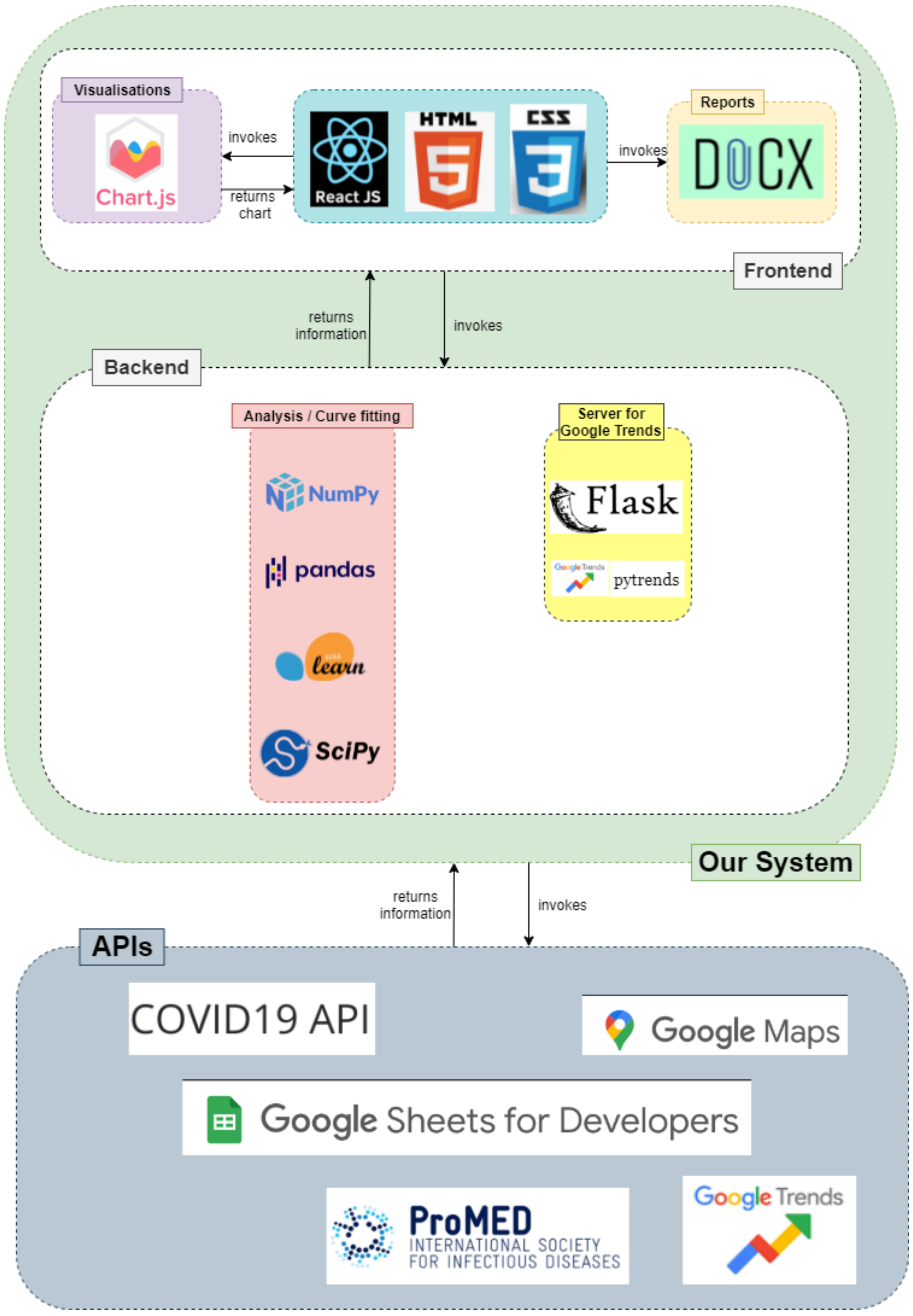
Description: As a user, I want to create reports on my analysis and/or prediction so that I can download to make it easier to use the results.

Acceptance Criteria:

- Downloadable as a word document
- Appendix contains tables of the raw data on the graphs
- All resources listed in the bibliography
- All graphs on the page available as images
- Summary of analysis added

System Design

System Overview:



Summary of the key benefits/achievements:

Key benefit/achievement	Implementation
Easy to use/update UI	React and bootstrapped was used in order to achieve this. React allows for easy updating and

	bootstrapping the front-end made it look profession without a lot of style sheets.
Ability to fit both polynomial curves and exponential curves. Giving the user flexibility and more choice when trying to fit data for analysis.	Combination of the Python3 analysis libraries SciPy and Numpy, with Pandas used to provide the infrastructure to easily manipulate the data to be used to fit.
Ability to dynamically render Chart.js graphs based on what data the user inputs and what graph they would like to display	Chart.js API (with React) allows for easy rendering and updating of charts.
Ability to view current covid statistics on a health map	Google Maps API has inbuilt advanced functionality and combined with our API from Phase 1 we were able to add covid information onto a health map.
Ability for the user to input user to add their own input	Users could edit the google sheets (implemented through google sheets API) and enter their own information using the template. This allowed for users to enter any variables they wanted to compare.
Ability to graph a multitude of different types of graphs	Users were able to graph pie, line, scatter and bar plots using their own input and the templates on google sheets. Implemented with the google sheets API and Chart.js
Multiple different data sources available for the user. This was done to ensure that users that did not have access to data could use the website to get data with the click of a button.	Many different APIs used (our Promed API, Covid19 API, Google Trends API) were all utilised so that users had a base of variables to play with without needing to provide their own data.
Easily creating graphs using google trends.	Implemented using Google Trends API, a flask server and Chart.js

APIs used:

COVID-19 API

The Covid-19 API was used to get both present and past data about Covid-19 cases. It was also used to filter Covid-19 data by country. It allowed the user to have access to another dataset to analyze. It is closely linked to the analysis section, as the data is analyzed as soon as the user inputs their choice of location and date.

Google Trends

We utilised Google Trends data to provide additional data for analysis that would normally be difficult to obtain quickly for a researcher. Google Trends does not provide an official API, so to do this we used the ‘Pytrends’ library which serves as an unofficial API wrapper for Google Trends. This data was accessible dynamically via RESTful API endpoints on our flask server, and our frontend utilised this data to display a scatter plot comparing Trends data per country over the last year. With more time, this trends feature could be expanded to a line graph that shows keyword interest over time vs new covid cases per capita, which would hopefully provide more useful data than what is currently being shown.

Google Maps

The Google Maps API was utilised in displaying the health map segment of the application. The API allowed a map to be embedded onto the website, and gives the option for satellite, aerial and street maps. The map itself was displayed by using a unique Google Maps API key and the "GoogleMap" library in React. The versatility of the Maps API meant that it was easy to incorporate a custom marker pin and tooltip dialog onto the map (using marker clustering) in order to display the covid cases and news reports for each country.

Promed API (Updates)

The Promed API is the API our team developed in Deliverable 2 based upon our data source: promedmail.org. In order to meet the requirements of our analytics platform, extensions for the Promed API were necessary. This extension was the addition of a new route, which returned the sum of articles related to COVID within a given country between a given period. This extension was necessary as sections of our analytics platform aimed to find correlations between total COVID reports and COVID cases. As this extension was relatively simple, no changes were made to the database structure or API structure.

Google Sheets

Google Sheets served as an integral part of our final application as it allowed for us to expand upon this notion of a "sandbox", through allowing users to input whatever data they chose into whatever type of graph they wanted. The Sheets API itself allowed the main React application to link to a live google sheets (which users could edit) and thus created an easy-to-use interface for customising the sheet as one saw fit. In particular, a library called "Tabletop" was used in React to allow for easy extraction and manipulation of data to dynamically display user preferences and data on the web application.

Backend:

Flask

We implemented a simple flask server for the backend of Epidemic Sandbox. This was developed with the intention of providing trends and COVID-19 data to the frontend, however given more time could have been extended to provide other forms of data, as well as conduct data analysis dynamically and provide that to the frontend.

The flask server provides a RESTful API from which our frontend can interact with directly to obtain data. It is written in python and was chosen primarily for speed of development and leveraging team members prior experiences developing with it.

Python3 (analysis libraries)

SciPy-Optimize:

SciPy-Optimize is an optimization function within the SciPy python library. It was used to find a*, b, c* in $f_1(x) = ae^{bx}+c$ such that $|f_1(x) - f_2(x)|$ was minimized with $f_2(x)$ representing the curve of the inputted data. This effectively found an exponential curve that best fit the data using the "Sequential Quadratic Programming" (SLQP) algorithm. This was the simplest approach we found to allow for the users to fit exponential curves to their data.

Pandas:

The Pandas python library was heavily used due to its unique data structure it provided, Data Frames. This allowed for easy manipulation of the data so that it could be used in the curve fitting process. It also allowed for easy reading of csv files making it easy to connect with the google-sheets API. It also easily converted to Numpy arrays which was necessary for fitting both exponential and polynomial curves.

Numpy:

Numpy was greatly used in the fitting of n^{th} degree polynomials to the input data. Numpy provides a function called polyfit which uses the "Least Squares Polynomial Fit" algorithm to fit a given polynomial to a dataset. Numpy also provided a correlation suite, which allowed us to provide the Pearson, Spearman and Kendall correlation coefficient and their related p-values.

Sci-Kit Learn:

Sci-Kit Learn provided the r^2 score. It compared the fitted curve vs the input curve, with a score closer 1 representing the fitted curve predicts/fits the inputted data exactly.

Frontend:

React

Chart.js

Chart.js is an open-source library that allows you to display a range of interactive and animated charts including (bar, line, pie and scatter). In our application, we linked chart.js with a range of APIs (team promed API, Google Sheets API and the COVID-19 API) in order to display a multitude of datasets in different formats. The dynamic nature and aesthetic appeal of chart.js meant it was suitable for our requirements and also allowed for the application to change the type of chart displayed if user required to do so.

Docx.js

Docx.js is a JavaScript library that was used to create the word document reports. Docx.js was compatible with the react front end and compatible with Chart.js and therefore was easy to use in our technology stack. Other libraries such as python-docx were also evaluated, these contained more functionality, however they did not offer inbuilt as front-end compatibility with Chart.js and as we only required basic functionality for reports, Docx.js was chosen.

Team Organisation

Responsibilities of team members:

Team Member's Name	Responsibilities
Harsimran Saini (z5208912)	• Meeting coordinator • Developer: Responsible for Epic story 3 (Reports) • Created User stories and business requirements • Created business pitch • Report writing
Sruti Desai (z5260319)	• Front-end developer • Report writing • Developed UI • Created Health Map with dynamic custom markers • Used chart.js to display dynamic graphs. • Used COVID-19 API to display current COVID statistics and vaccinations statistics • Added functionality of user being able to import their own data and change the type of graph shown. • Linked Google Sheets API, Google Maps API, COVID-19 API to React application
Jaiki Pitt (z5213775)	• Front-end developer • Report writing • Implemented Promed API extension • Used chart.js to display dynamic graphs
James Bradley (z5210562)	• Back-end developer • Report writing • Creating the analysis backend to fit the given data
Lachlan Harvey (z5258871)	• Front-end and back-end developer • Report writing • Managed the Trello board • Implemented flask server • Implemented google trends frontend and backend • Developed prototype UI for D3 demo

Project Reflection:

Major achievements in project. Issues/problems encountered.

Achievements:

- As a group, we were able to create a functioning API that is updated every day and is hosted on AWS (a platform we previously did not have any experience with)
- As a group, we had multiple productive team meetings per week that allowed us to effectively create timelines and help one another complete our project to a high level of technical difficulty
- Our team managed to complete, in one week, our user stories that we started out with and added more user stories to our existing plan (such as trendline, prediction and google trends). This can be accredited to our teamwork which was definitely a major achievement for us.

- As a group, we were able to generate a strong and creative idea for our analytics platform, that combined our own API with interesting APIs and analysis.

Issues/problems countered:

- AWS free tier had restrictions that gave us some trouble, such as time limits that our web scraper (that we used to scrape the whole PROMED website) was limited by. To counteract this, we built 2 versions of the scraper – one that we ran on our local machine to scrape all the articles originally and put them into the database and another to just update the database each day.
- Originally, we were using a NOSQL database for our API but we had to switch half way due to speed concerns (it was slower that using a SQL database is what we found)
- The AWS database seemed to keep changing passwords on random occasions.
- Swagger – because we allowed returning a huge amount of information, Swagger sometimes crashed. Introduced Summary functionality to counteract this.
- For the application, getting google sheeting editable on the page was a challenge, we ultimately linked the sheet to a different tab.
- Rate limiting with the COVID API was a significant issue, especially as it was a hard limit meaning no requests would even go through. This issue was solved by purchasing the premium membership and caching the API as .csv files.

What kind of skills you wish you had before the workshop?

- As the final product relied heavily on CSS, bootstrapping workshops may have been helpful.
- Better introduction in AWS and other hosting options, there were lectures on this topic however they were more generalised.

Would you do it any differently now?

- We would not do anything differently. While we did change our technology stack quite a bit as we worked on this project, the discovery/process was very important to finding our final solution.