Design Details

API Module

Beams will be utilising the following technologies to develop the API module:

- NodeJS
- ExpressJS
- Typescript
- Firestore
- Puppeteer

The backend will be constructed in Express with NodeJs. Express is useful in creating server applications and is considerably simple, flexible and easy to use. Given the short time frame of the project, Beams decided to utilise NodeJS + Express to aid in creating the base of the backend in a shorter time frame. This allows for more time to be dedicated to other opportunities such as improving our scraping technologies and future web-app features.

Moreover, Beams will be developing in Typescript, a superset of the Javascript language which enforces strict data types. We utilise ts-node to compile our TS code into ES2020 compatible JS on-the-fly, thus allowing us to take advantage of the latest JS features and produce more compact, readable code. Strong typing will also provide further rigidity in our code which will help us detect issues before run time, and, together with strict ESlint integration, allow us to keep our codebase consistent with potential to scale.

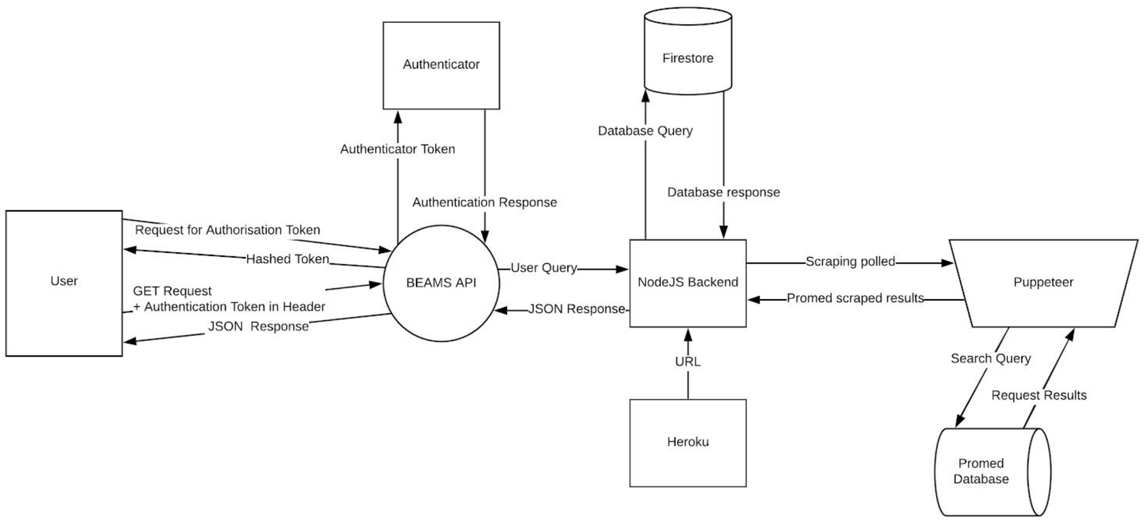
Firebase's Firestore database tool will be used in conjunction with our web scraper whose results will be formatted and stored in our Firestore database for fast retrieval by user API requests. Firestore's online interface opens opportunity to test data structures online, have multiple tenants interacting with the same data, and provides real-time updates to data changes both in-app and through our servers, bringing us opportunity to experiment and produce results rapidly.

Beams is using Puppeteer, a Google Developer Tool, which instantiates a Headless Chrome component that allows developers to interact with the browser DOM without the GUI of Chrome. This allows us to generate server-side rendered content to interact with the ProMed site, including entering terms into ProMed's search system and scraping relevant articles and post results. We can further process and parse this raw data, along with other data received from other sources, into JSON format and structure it as an accessible API endpoint.

Additionally, Beams will be integrating CircleCI into our GitHub repository to ensure our commits and pull requests abides by preset linting standards, and passes unit tests which will be developed overtime. CD methods are open to discussion and may be implemented in future iterations.

In conjunction with CI, Beams will also be using commit hooks with husky to ensure code is neat and passes tests before it enters the CI pipeline. We will be utilising Airbnb's ESLint specification (https://github.com/airbnb/javascript) since it provides a rigorous guideline to follow and because it will allow us to adapt to industry-level code consistency and style. This will ensure mistakes are caught before they are pushed into the master branch and will save time when there are several branches being tested and built through our CI pipelines.

System Design Diagram



Hosting

Beams will be using Heroku for hosting due to its simplicity and cheap cost.

API Design

At the current state of development, we envision our API to serve response to users using the standard URL query method. The API will collate and format data scraped from https://promedmail.org/ using a web-scraper.

API Request Format Form:

```
https://designated-host-service-url/?keyterms=firstterm[,secondterm,thirdterm…]&startdate=2018-01-01T08:45:00&enddate=2018-01-01T08:45:00&location=somelocation&key=authorization_token
```

An example request for searching for coronavirus and SARS during January, 2020 in China with an existing authorisation token:

```
https://designated-host-service-url/?keyterms=coronavirus,sars&startdate=2020-01-01T00:00:00&enddate=2020-02-01T00:00:00&location=china&key=4ba3f313d29993940b4abd48200e56477
```

An example request for an authorization token:

```
https://designated-host-service-url/?newuser=true
```

As of now, the API will enforce strict URL parameters so that the following conditions can be met:

- At-least one key term is provided with an optional comma separated list of other search terms (case insensitive).
- The start date meets the format `yyyy-mm-ddTHH:MM:SS` where:
    - yyyy is the year
    - mm is the month
    - dd is the day
    - T is the upper ascii character, used to separate date from time
    - HH is the hour in 24-hour format
    - MM is the minutes
    - SS is the seconds
- A single location must be provided.

Any deviation from the above conditions will result in a malformed request response from the API. We intend to provide dynamic error checking that can inform the API users of bad endpoint format, missing queries or unauthenticated access in the future.

The API has been designed to allow for future extensions or modifications as such:

- Location could later be extended to accept a list of locations.
- Given our allocation of the resource https://promedmail.org/, search queries by time are not supported so the time supplied is arbitrary. The above design meets design specifications but it can later be extended to support only dates being provided by users.
- Further, `startdate` and `enddate` could be extended to support variations of provided dates as follows:
    - `yyyy` must be supplied in which case `mm` default to January (`mm=01`) of that year (`yyyy`) and the day will default to the first (`dd=01`). If `yyyy` and `mm` is supplied `dd` will default to the first of the provided month (`dd=01`).
    - Year provided but not month or day: `startdate|enddate=yyyy` → `startdate|enddate=yyyy-01-01`
    - Year and month provided, but not day: `startdate|enddate=yyyy-mm` → `startdate|enddate=yyyy-mm-01`
    - Ignore requirement for `HH:MM:SS` which will default to the start of the day.
- `startdate` and `enddate` may also be open to take input in unix timestamp form in the future to alleviate issues with timezone incompatibilities and make the request format less prone to mistakes.
- A sentiment analysis field may also be added so that users could search by sentimentality (positive, negative or neutral).

Moving Forward

With the final goal of a user friendly website to display information and predict epidemics, the development of the API is stage one. Our API satisfies and goes beyond the constraints and required specifications with providing users data from ProMed, our new source. We also have the intention of expanding the API such that users are registered and authenticated to track user usage and request statistics.

During stage two, the API can be used in conjunction with other developed APIs, news sources and social media to expand our range of data sources. Stage two will allow Beams to present this information in a visually pleasing way with integrated news sentiment analysis, interactive graphs, and more.