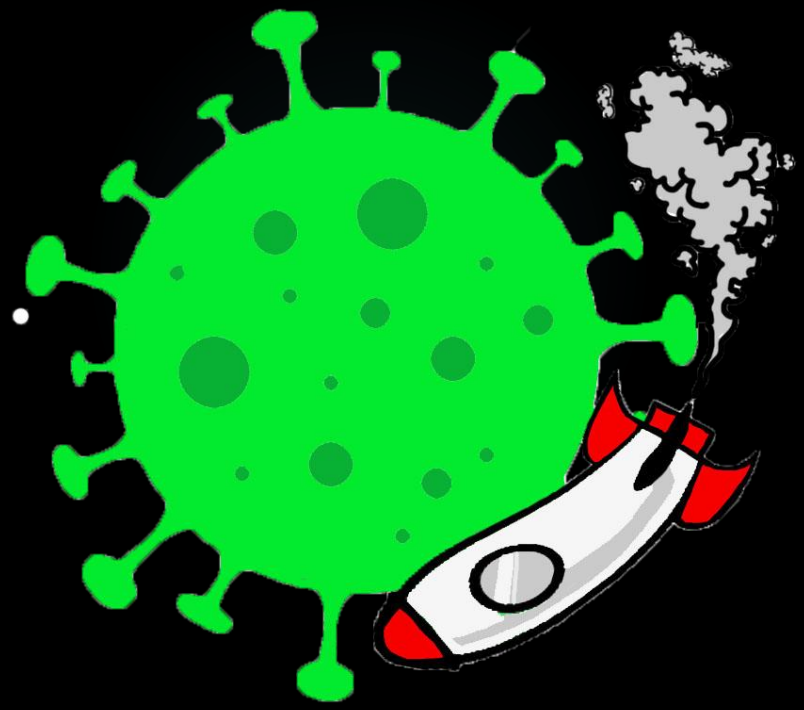
COMP47250: Team Software Project

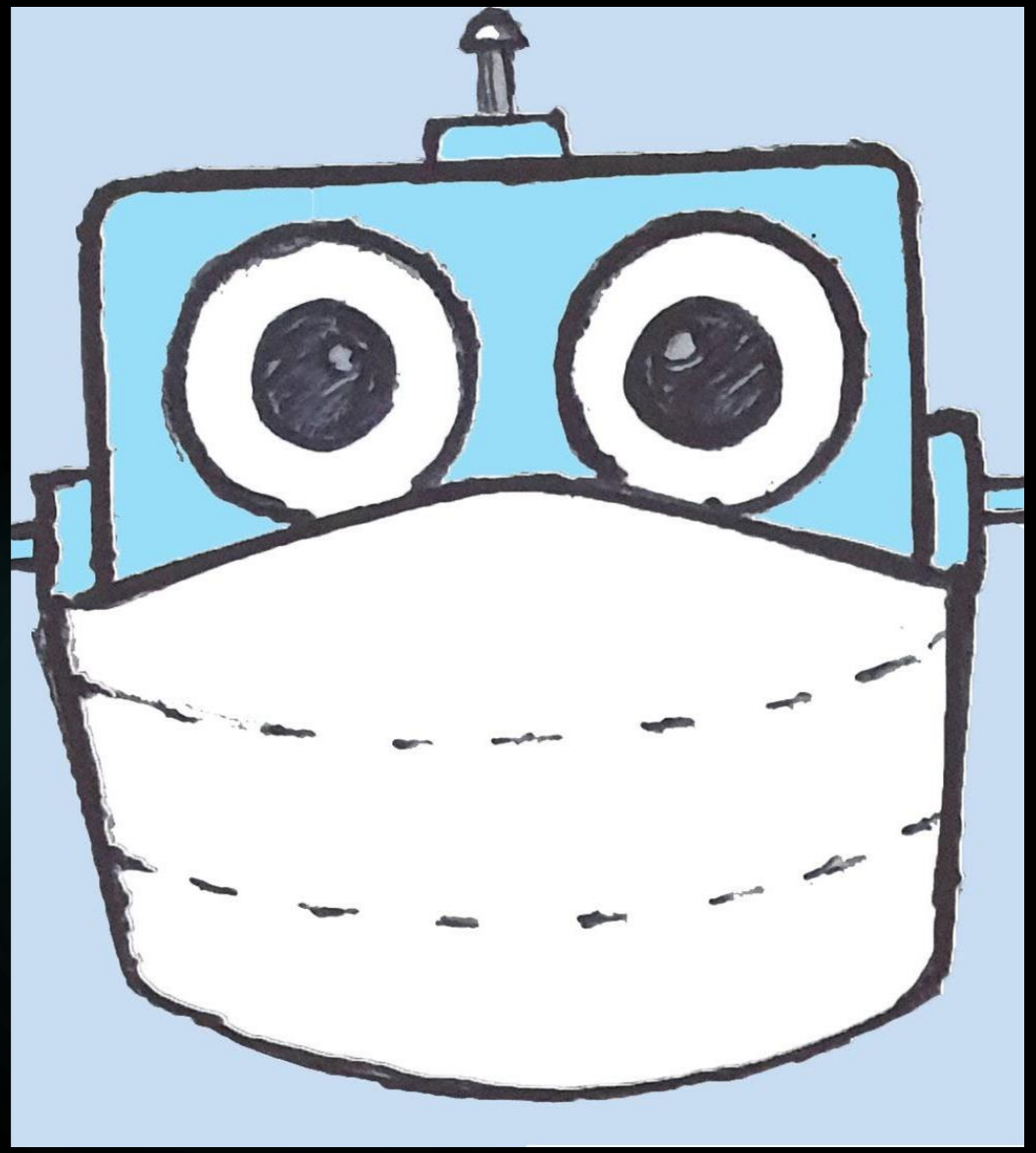
INTERIM PRESENTATION
29TH JUNE, 2020

Team Rocket

Aditya Krishnan	15316048
Apurva Deore	19200283
Ashwin Manikandan	19201000
Minaz Shaikh	19200277
Nripendra Singh	19200326
Sumit Kumar	19200701
Vishal Padma	19200174



Challenge 1: COVID-19



COVID-19 Interactive Visualizer

PROJECT OBJECTIVES

- ***What problem(s) is the project trying to solve/address? What are you trying to do?***

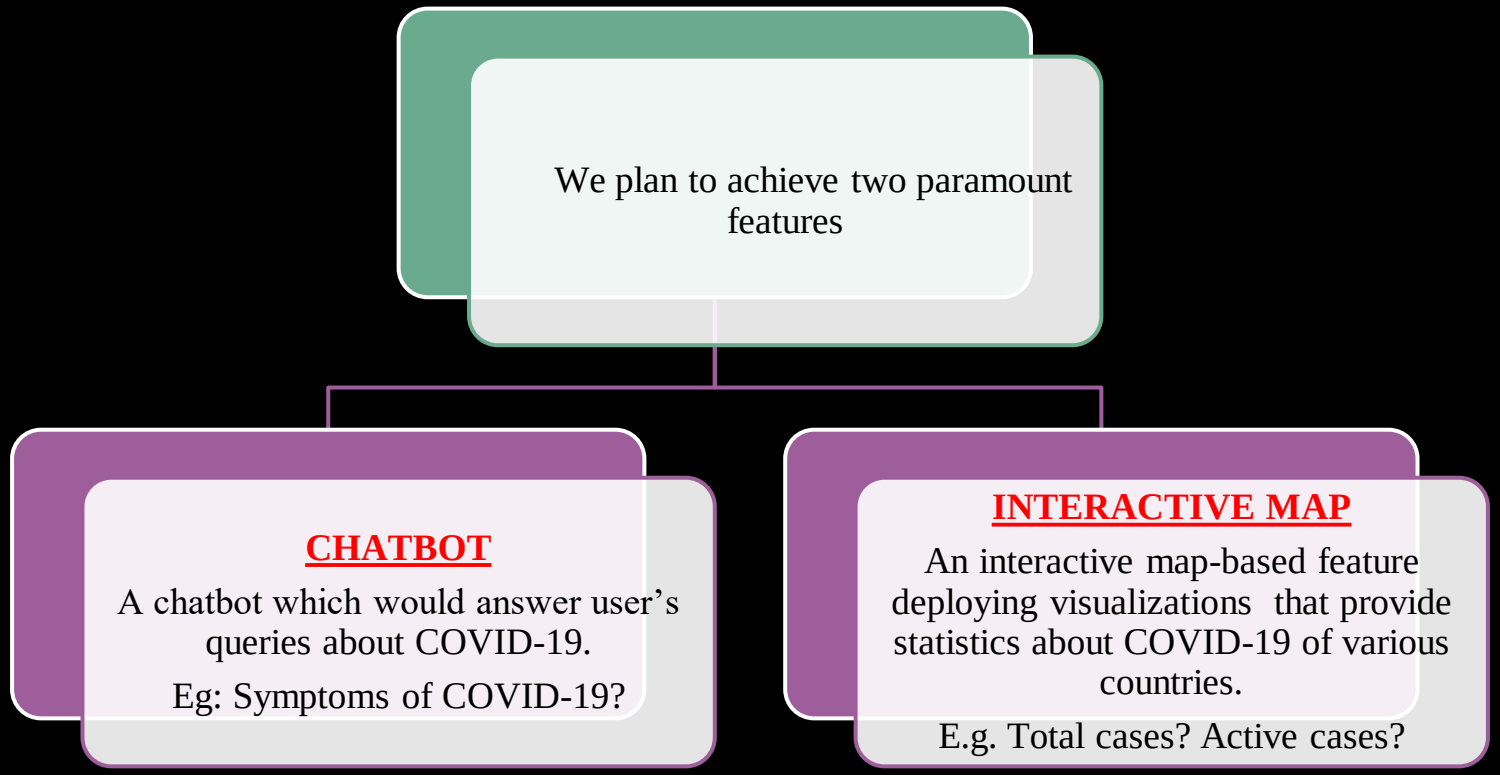
- Given the extent of the current situation regarding the COVID-19 pandemic, people are yet to grasp a good understanding about it. Due to the mind numbingly large sources of data(news articles, open source data sets, and huge social media traffic), it is difficult for the average user to get access to true and diverse information. Thus, we decided to tackle this topical COVID-19 challenge as part of the summer project.

- Considering the below mentioned project requirements:
  - Providing a (near)real time view of the current situation.
  - Allowing for geographically and/or temporal differentiated views of the situation.

- Thus, we pursued the idea of deploying an Interactive Map-Visualizer along with a Chat-Bot to help our users with accurate information related to COVID-19.

AIM

Creation of a **Web-based Chatbot Application and Map Visualizer** which would enable us to solve user queries regarding the COVID-19 pandemic across the world. We aim to create more awareness among people regarding the effects of the pandemic specifically in the health sector, i.e. Infected rates, Death rates, Recovery rates, etc, by letting users directly interact with the chatbot. The web page will use trusted sources to gather relevant information in the aforementioned categories which will be used to drive the responses of the bot and power the map-based visualizations of the world which will be also deployed on the same application.



- ***Who are we doing this for? Why do they need it? Link to the corresponding challenge(COVID-19)?***

- **Statistics**

To present users with statistical information about the total cases and deaths reported across the world.

- **Dashboard**

To provide them with dashboards (which would contain daily counts of cases and other reliable information), thus providing the users with a data visualization and data exploration resource at all times.

- **Reliable facts about COVID-19**

Providing users with a centralized location wherein all of the related COVID-19 information would be stored (either providing an information section on the web-page or by directly interacting with our Chat-Bot) which would aim them to stay protected, safe and healthy whilst also being able to monitor the situation across the world.

TECHNOLOGY STACK

- ***Team Management***

1. **Shared Google Drive Folder**: A file storage and synchronization service that provides with a systematic documentation of all resources (MOM's, PowerPoint presentations, Scripts) to be in one secured place. It allows simultaneous viewing and editing, thus providing enhanced availability and accessibility at all times for all the members.

1. **Google Meet/WhatsApp**: Google Meet is regularly used for meetings (which take place on a daily basis), WhatsApp is used for quick updates about work done and for quick short calls. Everybody is well-acquainted with both the apps thus making it easier to create groups and share files with one another.

1. **Project Management Jira**: The Jira workflows, issue types, and screens enable tailoring for almost any scenario and can easily be monitored and changed via the administration user Interface. The ease of use for configuring sprint and updating tasks makes it the right tool for the task.

1. **Version Control GitHub**: A Version-control system for tracking changes in source code during the development of the web application. Used by all members to push-commit their work onto the git so as to maintain a good presence on the platform, enabling efficient update and development of code within each sub-team.

- ***Data Processing***

1. **Jupyter Notebook:** Jupyter Notebook is an open-source web application which provides an environment to document the code, run it, visualize data and results without having to leave the environment. We run it using Python 3.7, it's the main tool for maintaining data scripts because of the enhanced productivity and collaboration that Jupyter notebook provides.
2. **Tableau:** Data visualization tool which helps us convert textual and numerical information to beautiful visualizations via interactive dashboards. It is the best way to change or transform the raw data into easily understandable data and is very user friendly and supports a high degree of customizability. Thanks to Tableau online, for additionally supporting easy embedding of dashboards onto a web platform, it's the primary tool of choice for visualizations.

- ***Back-End and Chat-Bot***

1. **Amazon Lex: Bot:** Since the Amazon Lex bot is powered by Automatic Speech Recognition (ASR), it can understand user input provided with text or speech and converse in natural language, by creating Lambda functions and add them as code hooks in your intent configuration to perform user data validation and fulfillment tasks.
2. **Django:** It is one of the most favoured web frameworks for Python which reduces the overall web application development time and helps build custom web applications rapidly. Since the project requires the application to be supported on various devices and platforms, django enhances the accessibility of web applications by supporting major operating systems like Windows, Linux and MacOS.

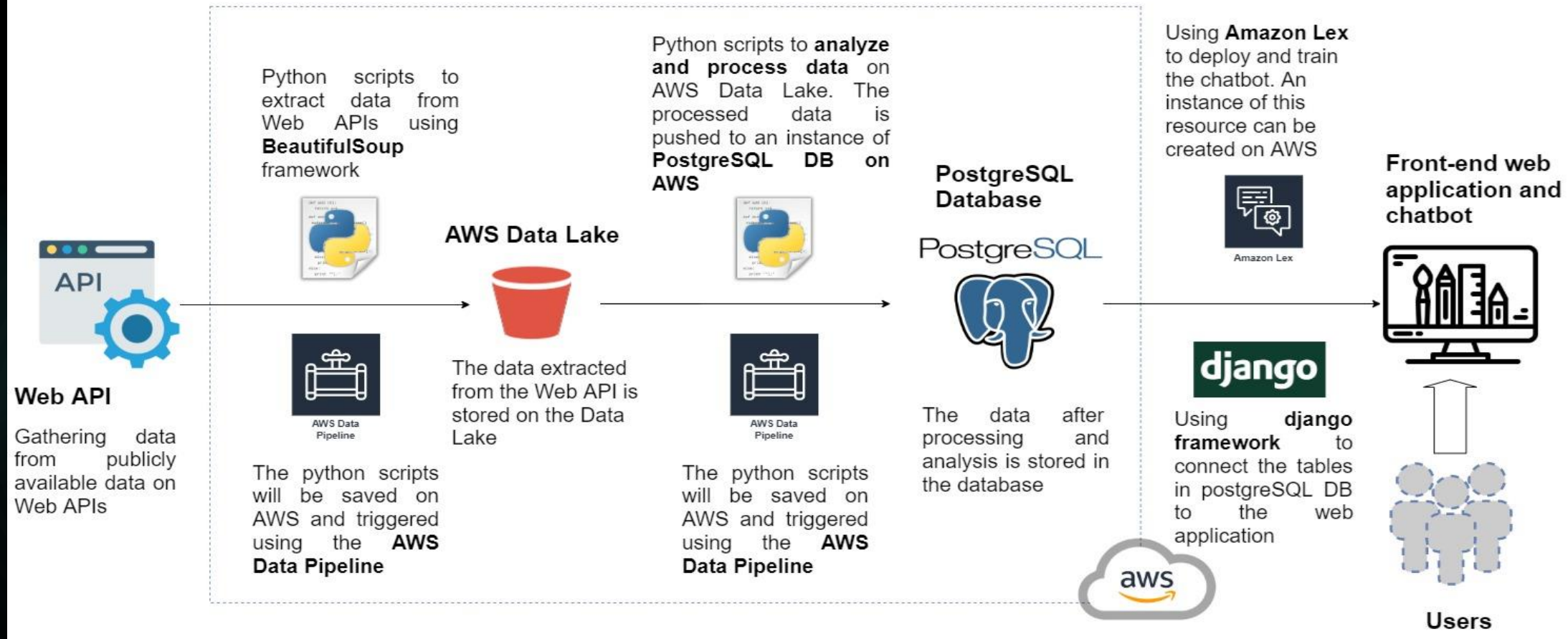
- ***Web Development***

1. **Atoms and Brackets:** As a text editor Atom was released by GitHub. Both of the text editors come with a great set of visual tools and preprocessor supports. The availability of shortcuts which immediately produce chunks of predefined code, and it's git integration makes the atom a very user-friendly editor.

Both text editors support smart auto-completion, and the Inline Editors allow easy opening and editing of HTML, Javascript and CSS files. Given the live preview feature which supports real-time connection to your browser to immediately see the changes made on the file reflect on the screen.

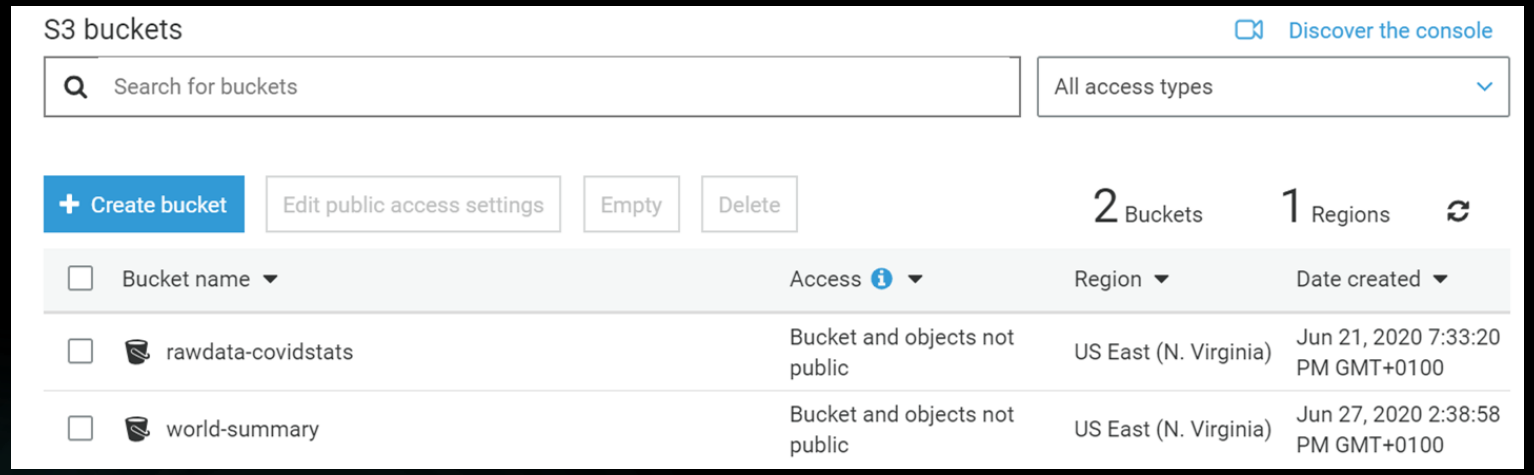
ARCHITECTURE

Team Rocket - Covid-19 Project Architecture

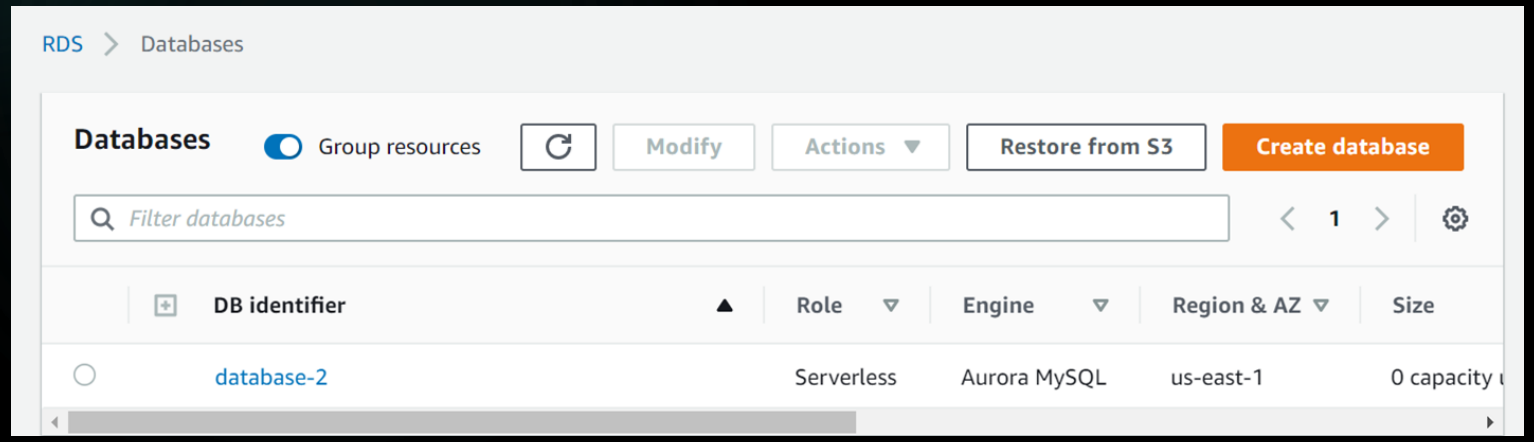


- We have AWS Student accounts set up and have limited access to AWS resources (Resource usage restrictions and a $100 credit limit).

-Created S3 buckets that contains data from different APIs.



-Created an instance of Amazon RDS with an Aurora Serverless API that uses a MySQL engine for processing queries.



DATA

Considering that we need the data that'd answer **statistical** and **theoretical** questions asked to the Chat-Bot; we further divided the data into two broad sections.

Dynamic Data

-Data that provides the user with statistical information such as follows:

-Real-time data required at all times, hence would need a mechanism that'd allow frequent updates of this data.

- Total cases
- Active cases
- Recovered cases
- Deceased cases
- Tests done

Static Data

Data that's loaded in once and doesn't need frequent updates. This would answer the theoretical queries of users such as follows:

- Symptoms of COVID-19?
- Healthy habits to be followed?
- Whom to contact for testing related situations (some helpline numbers)?

DATA SOURCES

After having decided on these two broad sections of data, we went through a couple

of APIs that would allow us to collect and present data in the most efficient way possible. For the static data, we are maintaining a document wherein we keep adding information about COVID-19 that one would like to know from a user perspective. For the dynamic/statistical data, we are currently using the **John Hopkinns** API as it has in store the stats for many countries displayed in various forms, has set-up real-time updation of statistics everyday (If we're getting the data up-until date 27th June, we'd get the data updated until 26th June).

Some of the links we're considering currently:

- https://api.covid19api.com/summary (summarised statistics across the world)
- https://api.covid19api.com/countries
- https://github.com/CSSEGISandData/COVID19/tree/master/csse_covid_19_data/csse_covid_19_time_series (Time series Dataset)

Going forward, we have decided to take into account some more region-specific detailed APIs such as the one offered by European Centre for Disease Prevention and Control: COVID-19, Tweets from twitter, Population data for various countries to draw some more related insights.

Data Collection and Storing

We've set-up our Data Scripts in place for Data Collection from time-to-time. We've also been performing Data-cleaning and pre-processing as and when required.

Currently, we have in store two collected/cleaned and processed datasets:
1) Summarized statistical Data for all countries until the day of collection.
2) Time-series Data showing the gradual incrementation of cases for all countries overtime (dated from 22nd January to the latest update)

All of this data has been collected multiple times so as to maintain accurate data if some data related mishap were to happen in the near future. Updated versions are stored on our local systems as well as committed onto git frequently.

EVALUATION

How will you evaluate your system?

Below are the three basic evaluation questions/measures which will be used to evaluate the chatbot:

- **Information-Retrieval Perspective:** As part of this evaluation, we would be looking at the questions answering service offered by the chatbot, so we can measure their performance in terms of accuracy, e.g. using precision, recall, F-measure (Whichever is suited). Mostly, this would allow us to check the relevancy of the data retrieved with respect to the user query.

- **User-Experience Perspective:** For this, we'd be looking at the Chat-Bot which is an interactive software application and judge it on human factors or usability point of view, taking into consideration measures such as task completion and user satisfaction.

- **Measure of Performance with respect to time**: As part of this perspective, we'd be evaluating the Chat-Bot on the basis of the actual time it takes to provide answer to a particular query in order to check the speed that the Chat-Bot offers and what changes could be made to improve the speed.

What question(s) are you trying to answer with your application and how will you evaluate if it is successful?

We aim to answer information related to COVID-19 with the help of statistical and textual data that we've collected.
To give a gist, some of the queries that we plan to answer with the help of our Chat-Bot are as follows:

- What is the condition of "**country**" right now?
- What are the recovered cases of "**country**"?
- What are the death cases of "**country**"?
- What is the current number of infected/confirmed cases in "**country**"?
- What are the symptoms of COVID-19?

As we move forward with further planned sprints, we'd be answering more complex queries. The evaluation would be performed using the earlier 3 perspectives taken into consideration. As of now, our Chat-Bot is set-up and running. It's in the learning phase and for now is able to greet the user.

How will you evaluate your system?

Below are the three basic evaluation questions/measures which will be used to evaluate the chatbot:

- **Information-Retrieval Perspective:** As part of this evaluation, we would be looking at the questions answering service offered by the chatbot, so we can measure their performance in terms of accuracy, e.g. using precision, recall, F-measure (Whichever is suited). Mostly, this would allow us to check the relevancy of the data retrieved with respect to the user query.

- **User-Experience Perspective:** For this, we'd be looking at the Chat-Bot which is an interactive software application and judge it on human factors or usability point of view, taking into consideration measures such as task completion and user satisfaction.

- **Measure of Performance with respect to time**: As part of this perspective, we'd be evaluating the Chat-Bot on the basis of the actual time it takes to provide answer to a particular query in order to check the speed that the Chat-Bot offers and what changes could be made to improve the speed.

CURRENT STATUS

We have now achieved the MVP version of our web application.

VERSIONS/MILESTONES:
Achieved by breaking down the tasks to be done into sprints and creating milestones for the project

v1.0 (8th June – 15th June'20)
First view of the rough front-end page of the web-page along with main statistical summarized data collected.

v1.1 (16th June – 22nd June'20)
Integration of Django to the front-end and a basic implementation of the Chat-Bot.

v1.2 (24th June – 1st July'20) Milestone 1 / MVP
Core implementation of the paramount tasks (prepare the map visualizations and basic chatbot) before the interim presentation and subsequent preparation of the report.

After having achieved V1.2 and the delivery of MVP , we aim to further breakdown tasks and reach goals to design a finalized sprint plan and plan the release versions accordingly.

SPRINT PLAN

Since we have achieved the MVP of the web application, we are on schedule and hope to achieve the reach goals we set out at the beginning they include:

- Integrating the chatbot to the main web application so that both entities would exist as one.
- Implementing a more dynamic version of the application where the bot responses regarding certain queries, such as What is the total active cases in Ireland, the map would refresh from a world view to a view of Ireland automatically.

Immediate next sprint plan:

- **Begin to integrate the chatbot with the web application**
- **Build data pipeline to facilitate transfer of data from S3 to database.**

STAY SAFE!

THANK YOU!

-Team Rocket