Project name: "COVID-19 Data Integration, Analysis, and Visualization Platform"

Student: Elvijs Melnis

Github repository: https://github.com/elvijs147/Bootcamp_Final_Project

"COVID-19 Data Integration, Analysis, and Visualization Platform" is a project designed to grasp wide range of possibilities that programming language Python offers together with data visualization framework Dash, Snowflake Data Cloud and MongoDB. The main topic of the project is COVID-19 which over the last couple of years has profoundly disrupted global society, causing widespread economic challenges, overwhelming healthcare systems, and prompting significant changes in daily life. The availability of diverse datasets has enabled researchers to uncover valuable insights, but issues such as data quality and reporting variations still present challenges. The main goal of this project is to show COVID-19 widespread around the globe in an appealing way by using an interactive world map.

The main dataset for the project was chosen World Health Organization's latest data on COVID-19. As a data scientist with a background in Economics, project's author feels that using the latest data ensures that the information accurately reflects the current state of the pandemic. Snowflake's provided data is more than three years old. Utilizing outdated information may not accurately capture the current state of the virus, hindering our ability to provide relevant insights. This approach allows for a comprehensive understanding of the ongoing situation around the world. The query for creating such table in Snowflake can be seen in Figure 1.

For the ability to add comments on COVID-19 metrics for different countries, MongoDB database is used. The user can click on a country and a pop-up window appears. It has text fields for user's name and a comment (Figure 2). The data is then stored in MongoDB in "project_db" database and "comments" collection. For the document schema see Figure 3. The usefulness is that the data scientist can see on which country and metric the comment was left.

For API implementation Flask framework is used because of its functionality and effectiveness. Flask serves as the backend server for the Dash web application. The use of Flask-Caching enhances the performance of the application by caching the results of Snowflake queries. Flask efficiently manages routing for different parts of the web application – querying COVID-19 data and submitting comments. Flask facilitates the handling of Dash callbacks – updating the choropleth map based on user-selected options, handling clicks on the map, and managing the display of the comment modal.

The main visual part of the project is the choropleth map of the world provided by Dash for displaying different metrics regarding COVID-19 (Figure 4). The useful features are that it is possible to switch between different metrics in the dropdown menu at the bottom to quickly see differences all around the globe or hover over a country and see the country's name and value for the metric right

away. Different colors provide the user with the ability to quickly assess the varying intensities of COVID-19 metrics across different countries. This visual representation allows users to see the overall situation in the world regarding COVID-19 at a glance. By using the query field, it is possible to filter countries or metrics. For example, in Figure 5 it is possible to see all the highlighted countries where COVID-19 cases per 1 million people are more than 100 thousand.

Although the dataset used for the main analysis of this project covers almost every country in the world, it has its limitations. It is the latest data and does not have timeseries. So, for forecasting another dataset is used from Snowflake's COVID-19 database's table ECDC_GLOBAL and the chosen country is Latvia for the year 2020. ARIMA (AutoRegressive Integrated Moving Average) time series forecasting model is implemented to see prediction for the next 30 days. Plot (Figure 6) displays the actual COVID-19 cases in Latvia, the model's predictions, and provides a visual assessment of the model's performance. The challenge using ARIMA model is that it is sensitive to data quality and it has limited capture of non-linear trends which we can clearly see in Figure 6. The p-value obtained from the ADF test being less than 0.05 indicates that the time series data was successfully differenced to achieve stationarity, which is a prerequisite for ARIMA modeling. However, despite achieving stationarity, the similarity between the mean and root mean squared error values suggests that the ARIMA model may not be capturing the underlying patterns and variability in the COVID-19 cases data effectively. The main problem could be complexity and non-linearity of the data.

Regarding optimal performance of Snowflake and API, a couple of techniques were implemented. First of all, a larger Snowflake warehouse was used to increase performance for data loading (Figure 7). It suspends after 5 minutes of inactivity to save resources and it aligns with Snowflake's elastic scaling model. It allows the platform to dynamically allocate and deallocate resources as needed. Snowflake does not support indexing because it uses a powerful and unique form of partitioning. Although this technique was not implemented, it is worth mentioning because it is crucial for enhancing query performance and overall database efficiency and can be used for many other databases like MySQL, MongoDB and others.

API caching's main purpose in this project is to enhance performance by leveraging Flask-Caching for frequently requested data. This optimization is beneficial for repeated queries to the Snowflake database, ensuring efficient and speedy data retrieval. The cache timeout is set to 60 seconds, providing an appropriate balance between data freshness and improved response times. Additionally, the caching mechanism was applied to the Snowflake query function, ensuring that the API caches and serves frequently requested data, contributing to a more responsive and resource-efficient application.

With reference to limitations of this project, there is a need to mention the primary dataset which lacks time-series information, restricting the ability to analyze trends over time comprehensively. Although the main goal of the project was to show the latest COVID-19 metrics around the globe which was achieved. One notable challenge encountered in the project lies in the limitations of the ARIMA forecasting model. Despite achieving stationarity through differencing, the model's performance suggests difficulties in capturing underlying patterns and non-linear variations in COVID-19 cases data effectively. This highlights the challenge of applying traditional time series models to complex and dynamic real-world scenarios.

For future development of the web app, there are several avenues to explore. Firstly, incorporating additional and more granular datasets, such as real-time COVID-19 data with time-series information, could enhance the platform's analytical capabilities. Integration with machine learning models, especially those adept at handling non-linear trends, could improve forecasting accuracy. Enhancements to the comment feature could involve sentiment analysis on user comments, providing an additional layer of insight.

Overall COVID-19 topic is very complex and statistics can vary significantly between different countries due to a combination of factors. Variances in testing capabilities and strategies among countries can lead to differences in reported cases. For example, USA with 334 million population has 110 million reported cases and China with 1.44 billion population has 503 thousand reported cases. In Africa the widespread of COVID-19 does not look believable as well. While in Europe many countries have tens of millions reported cases. (Figure 8). Some countries may be more conservative in reporting COVID-19-related deaths, leading to undercounts. Countries with higher population densities may experience faster virus spread although we can again see that it is not showed in the statistics. Another factor that might impact collecting useful information on COVID-19 is data transparency. Differences in it and reporting standards contribute to variations in statistics. Some countries may be more open in sharing data, while others might face challenges or be less transparent. The author of this project will leave in-depth COVID-19 metric analysis to professionals and experts in the field who can provide a thorough understanding of the complexities involved.

To summarize this project, it has been a good challenge to implement all the technical intricacies, merging diverse data sources, leveraging programming languages like Python, integrating databases such as Snowflake and MongoDB, and orchestrating a seamless collaboration between Flask and Dash frameworks. This endeavor required a meticulous balance between functionality and aesthetics to transform raw COVID-19 data into a useful and visually pleasing final product!

```sql
CREATE TABLE COVID_REAL (
    Country VARCHAR(255),
    ISO VARCHAR(3),
    population INT,
    total_cases INT,
    cases_per_1m INT,
    total_deaths INT,
    deaths_per_1m INT,
    total_recovered INT,
    total_tests INT,
    tests_per_1m INT
);
```

Figure 1. Table creation query in Snowflake

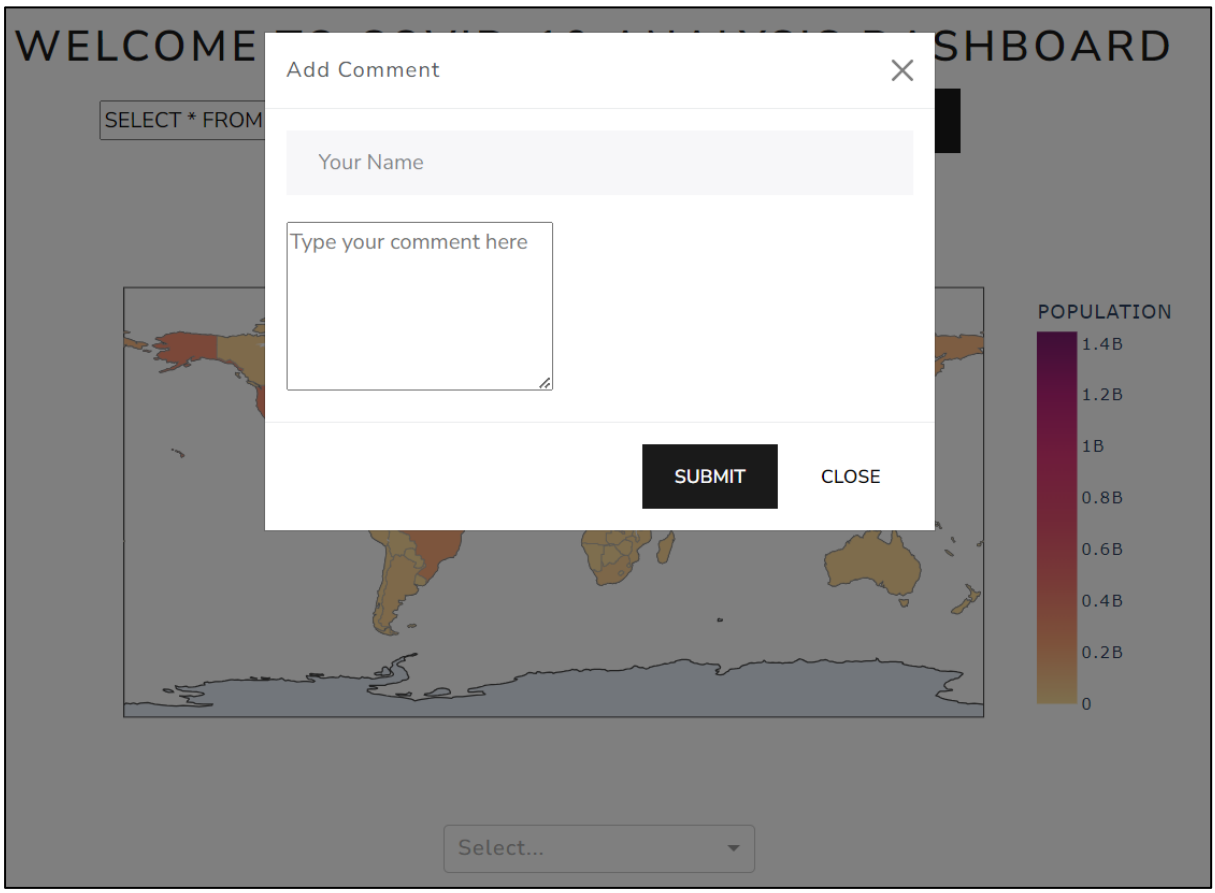

Figure 2. Comment section in the Dash map


```
_id: ObjectId('65b3a2c3f44593347d0f9ba8')
country: "China"
column: "CASES_PER_1M"
author: "Elvijs Melnis"
comment: "Interesting that China has only 347 COVID-19 cases per 1 million of in…"
```


Figure 3. MongoDB document schema

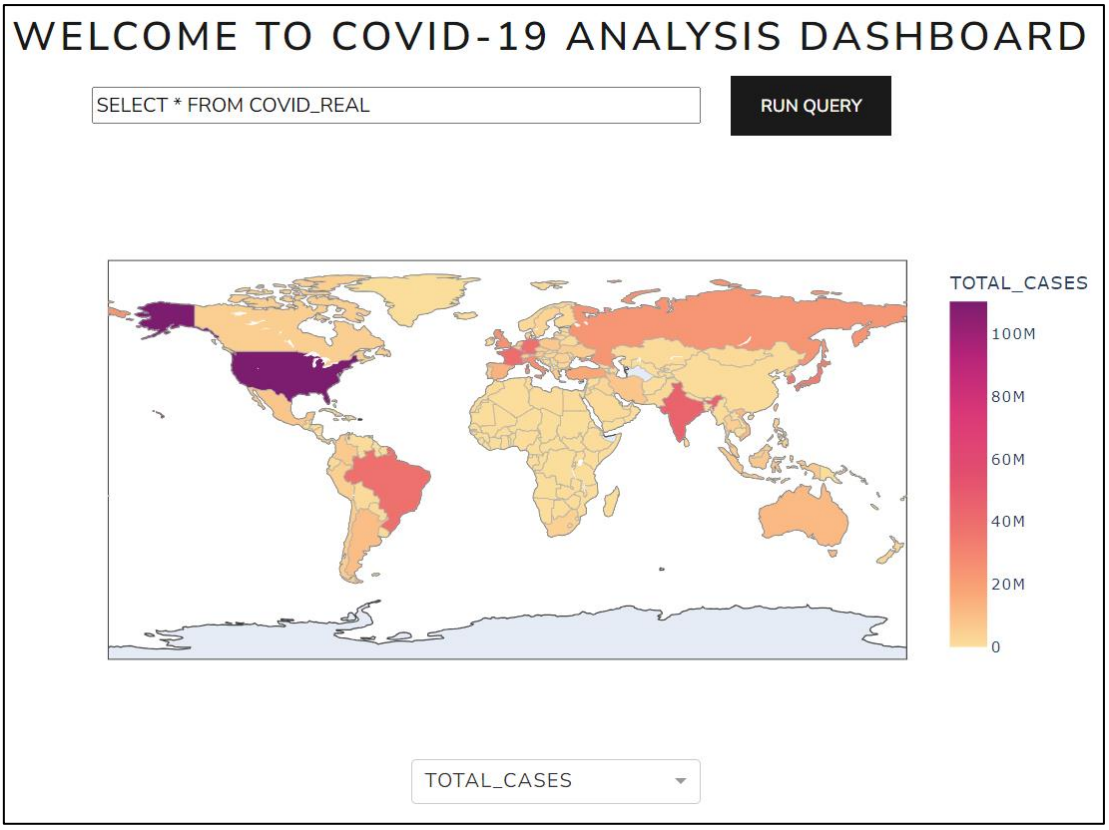

Figure 4. Visual dashboard

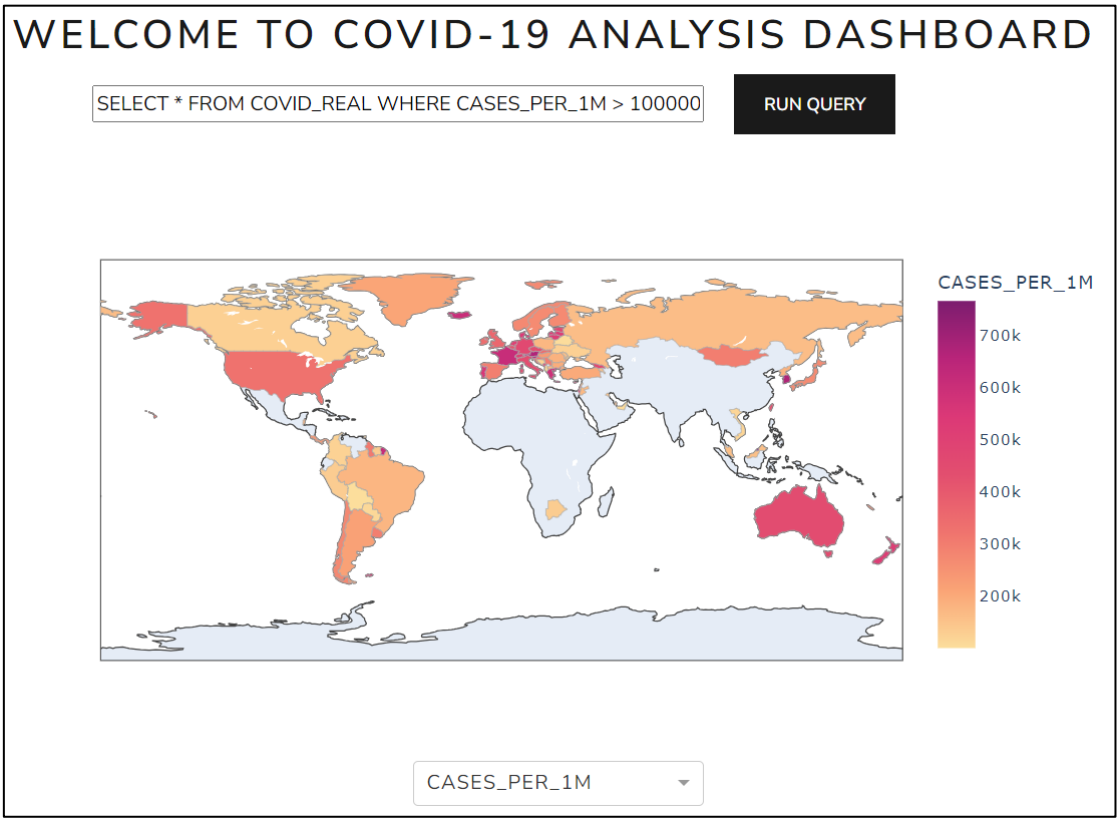

Figure 5. SQL query example

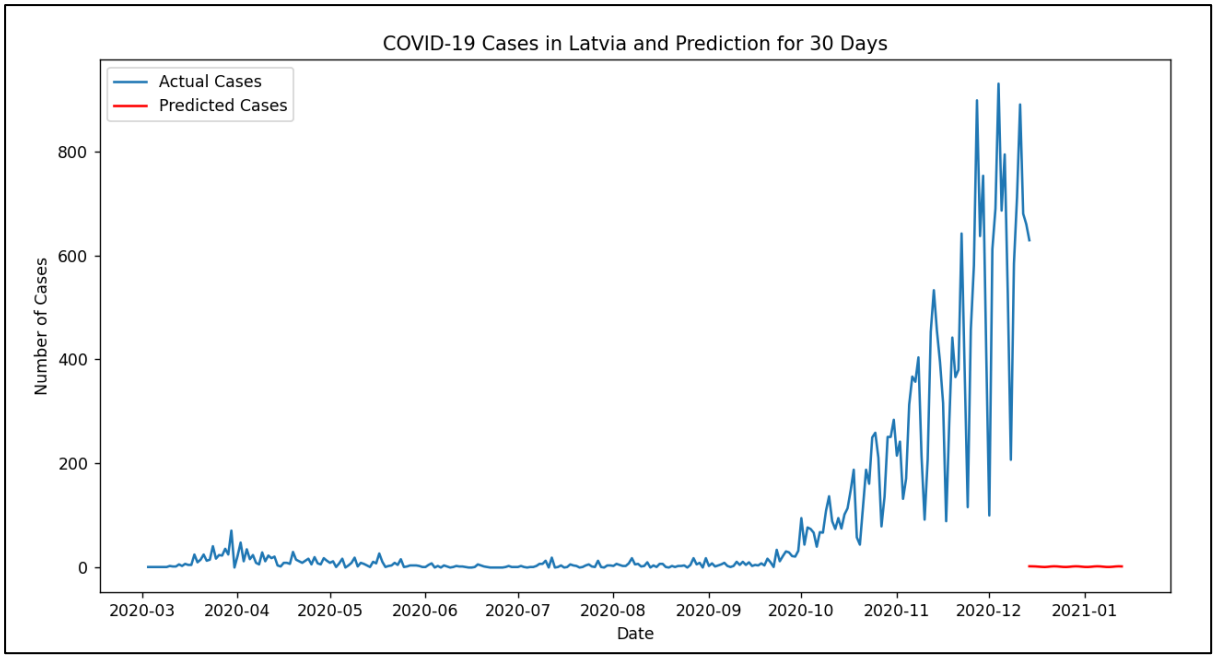

Figure 6. ARIMA model prediction for the next 30 days in Latvia

```
CREATE WAREHOUSE IF NOT EXISTS PROJECT_WH
  WAREHOUSE_SIZE = 'Large'
  MAX_CLUSTER_COUNT = 3
  AUTO_SUSPEND = 300
  AUTO_RESUME = TRUE;
```

Figure 7. Snowflake large warehouse creation query

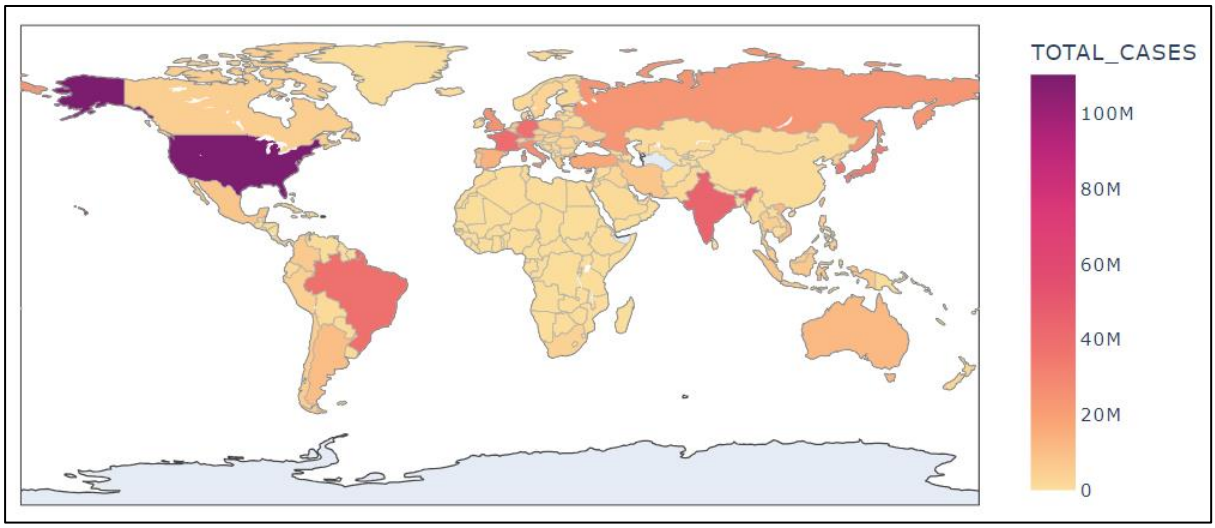

Figure 8. Differences in total COVID-19 cases among countries