Solent University Southampton

Faculty of Computing, Advance Database System

World Covid-19 Report

A Situation Analysis, Individual Report

September 2022

Author : 10152420 (Mohammad Sarker)

Course title : Advanced Database Systems

Course code : QHO541

Tutor’s name : Iftikhar Afridi

Word Count : 2000 words (+-10%)

Data source:

[COVID-19 Dataset Kaggle]

<https://www.kaggle.com/datasets/imdevskp/corona-virus-report?select=worldometer_data.csv>

Github Repo: https://github.com/sarkersh/covid\_19

Website Url: <https://kaggle-world-covid-19.herokuapp.com/>

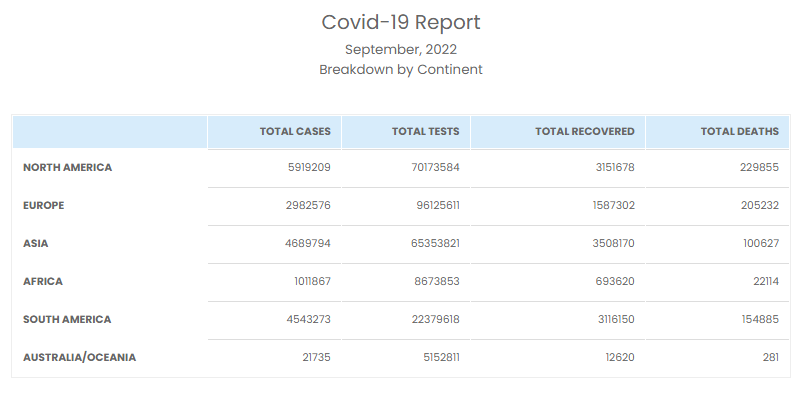
Introduction

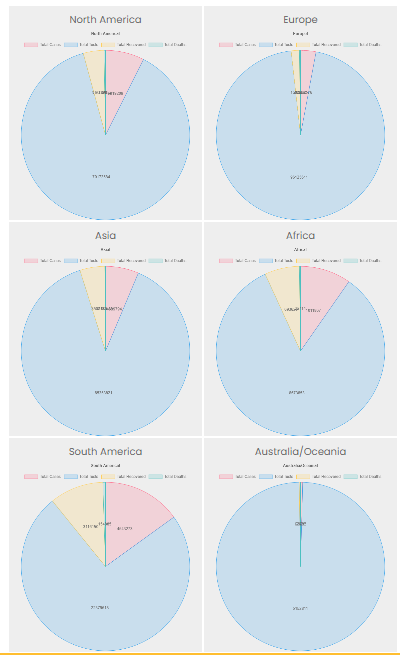
This project provides an analysis of the situation of covid-19 and its impact on various countries and regions around the world.

It provides the total number of cases, tests, recoveries and deaths per country. It then provides a comparison of these statistics by continent.

The data is based on a dataset downloaded from Kaggle. Although Kaggle is a well known source of statistical data I cannot totally verify the accuracy of this dataset.

As can be seen on the figure below the total number of covid-19 cases is lower in Africa and Australia/Oceania. From figure 2 below we can see that all continents have carried out a large amount of tests but Europe, North America and Australia/Oceania have a higher proportion of tests done. Again, looking at the pie charts, we can see that the death rates are quite higher in North America, South America and Europe



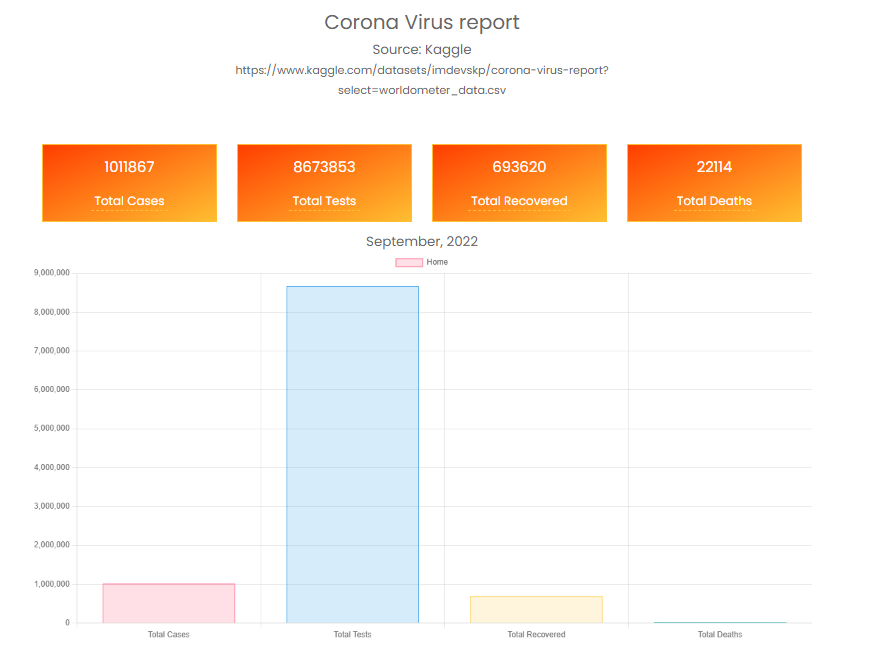


System Overview

This project and the related website use a dataset provided by Worldometer and downloaded from Kaggle, <https://www.kaggle.com/datasets/imdevskp/corona-virus-report?select=worldometer_data.csv>.

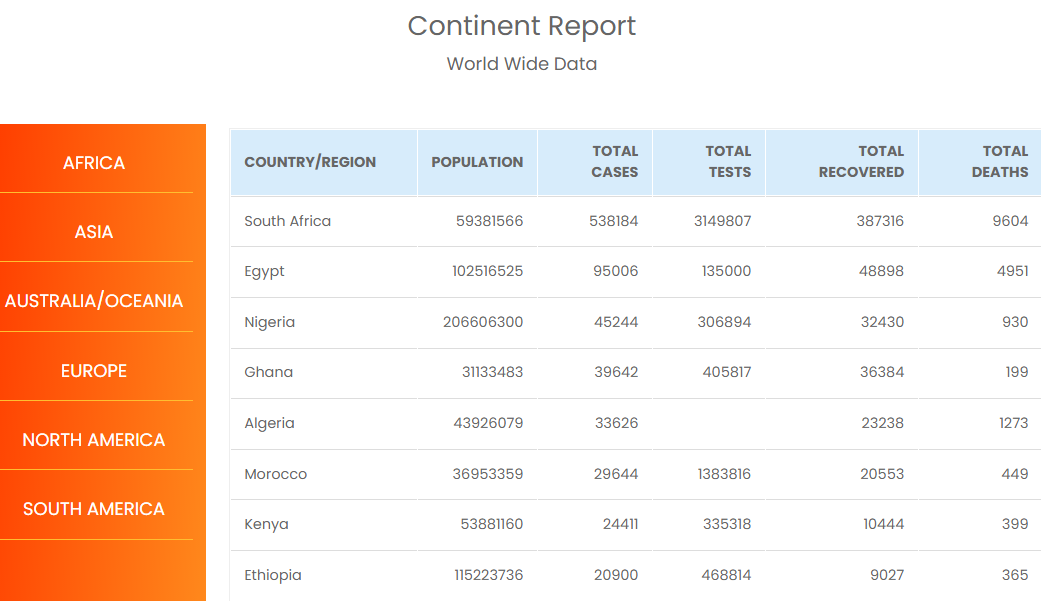
When a user visits the project website they are presented with a summary of the four vital statistics about Covid-19 that this project seeks to address – Total Cases, Total Tests, Total Recovered and Total Deaths. There is also a chart to visualize this information.

Towards the bottom of the page there is a per country breakdown of the statistics.

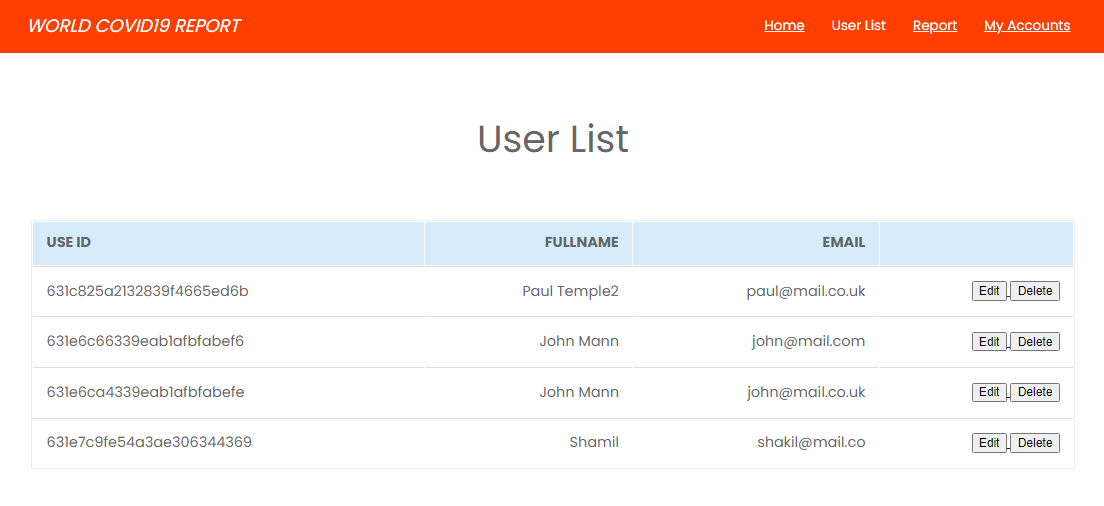


To see more information the user must log in. The figure below shows how a logged in user is able to view statistics by continent. When the user clicks a continent, the screen will change to display data including chart and table relating to that continent.

The pie charts we looked at earlier are also only visible to users who are logged in.



Another piece of data that is available to only logged in users is the user list that shows a list of all registered users. An admin user will also be able to edit and delete the currently registered users – CRUD operations.



Key Design Decisions

**Database Design**

The database used in this project is Mongodb. Mangodb is an efficient, document-based database system that is easy to use and quite scalable.

The database is hosted on Atlas in the cloud.

**FullStack**

The website is a full stack application based on node js, espress and ejs. Node JS provides the runtime environment and server-side processing while ejs templating, javascript, css and html are used on the client-side to present data coming from the server.

**MVC**

To keep the code clean and easy to manage, the MVC design pattern is used for the most part. The routes receive a request and pass it to the **Controller**. The controller then handles the request, connecting with the **Model** if necessary and passing data to the **View**.

An example of how this is implemented in this project is shown in the following code screenshots.

The website user sends a request to use ***list*** route. This request is passed on to the ***userController*** and handled by the ***listUsers*** function.

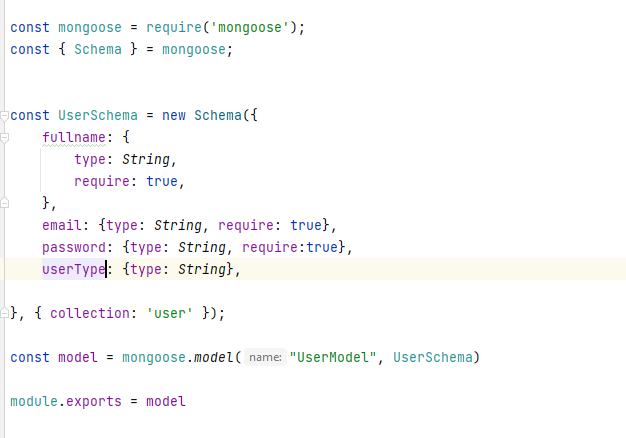


*Fig user route file*

The ***listUsers*** function of the **userController** will query the database for a list of all the users via the ***UserModel.*** The data retrieved is then passed on to the ***user-list*** view.



*Fig user controller file*



*Fig. user model file*

When the data reaches the ***user-list.ejs*** view file, it is rendered with the help of the ejs templating engine, html, javascript and css.





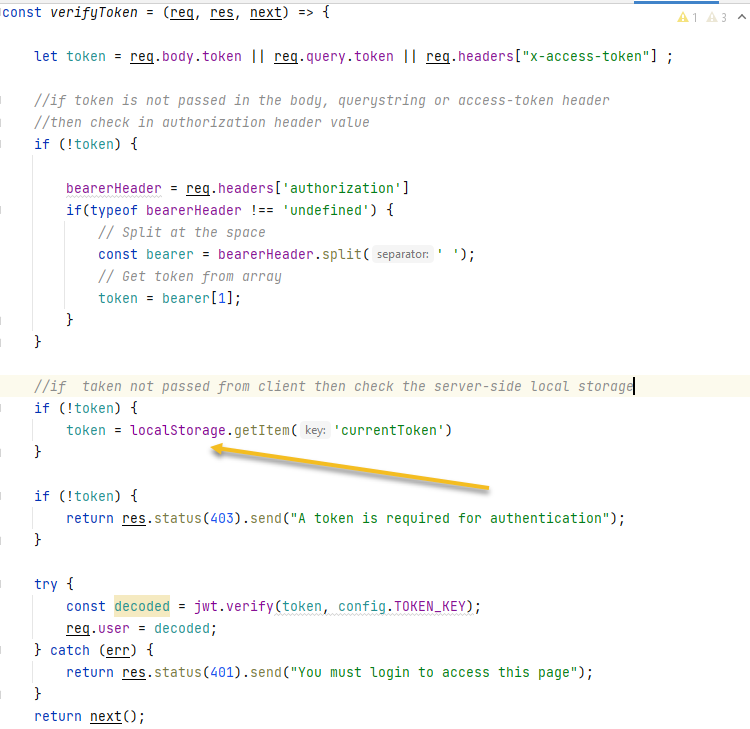
*Fig view file*

**Security and Scalability**

The system uses a token-based authentication using JSON Web Tokens (TWT). This basically allows the user to request a token from the web server and present that token for any request that requires access to a protected route.

For additional security the generated token is not stored in the client browser but on the server. So to authenticate, the system retrieves the token from local storage on the server and verify it.

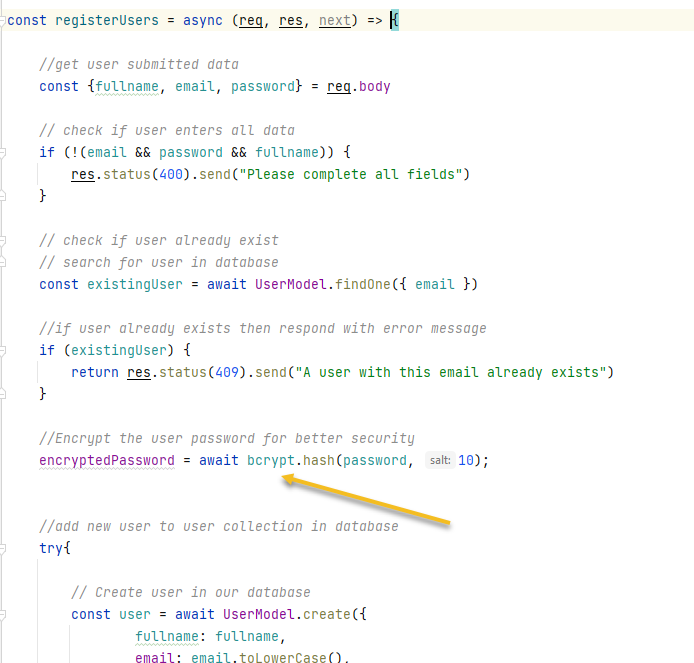
The screenshot below shows how the verification is implemented.



*Fig. jwt token verification code*

As can be seen in the code the verification code will allow tokens send in the post body, as query string or in the request header. However, in my implementation, I chose the option to store token on the server.

To improve security further, all passwords are encrypted to a high level with the help of ***bcrypt*** Node package. See code snippet below.



One of the reasons for using Momgodb is to take advantage of its high scalability features. Mongodb is very scalable both vertically and horizontally.

Some of the scalability features of Mangodb are **sharding** and **replica sets.** **Sharding allows horizo**ntal scaling by spreading data across multiple nodes. Each node containing a part of the overall data..

**Testing**

A good amount of manual testing has been carried out in this project. However no automated testing scripts are included with this project.

Conclusion and Reflection

To conclude, it must be noted that the data used in this project is mainly for educational purposes only. The Worldometer data is regularly updated and I am not quite sure about the scope or period covered by the dataset. However, I do believe that the data is valid from a country and region wise comparison.

To satisfy CRUD operations the project allows user registration which allows the creation of new records/document in the database. Data retrieval is achieved by querying statistics data from the database to present as chart and tables on the browser. When logged in as admin, the user can delete user records and edit and update them.

Given the time I would have included automated testing scripts for unit testing, integration testing, end-to-end etc.

The design UI also need further work.

Mobile responsiveness is good but large tables do break the design when you scroll the screen to reveal more of the table.

Due to token-based authorization, I could not able to save temporary token on remote Heroku and therefore the authentication (login and registration does not work in Heroku). Everything works fine on my local install as you can see in my presentation.