**PROJECT REPORT ON MOBILE APPLICATION DEVELOPMENT (MAD)**

**COVID-19 TRACKER APP**

**ROLL NUMBERS: 20SW013 & 20SW135**

**Sections:**

* Real World Problem Identification
* Proposed Solution
* Responsive User Interfaces (Screenshots of your app on different screens & platforms)
* Data Storage (With justification for using a particular database)
* [Optional Section] APIs/Packages/Plug-ins (if used with justifications for using them).
* Issues and Bugs Encountered and Resolved during Development

***Real World Problem Identification***

The real-world problem identified was the need for a reliable and accessible source of real-time COVID-19 data. This data should encompass global statistics as well as country-specific information, including the number of cases, recoveries, deaths, and other relevant statistics. Additionally, the ability to track the virus's progression and containment measures in specific geographic locations was crucial to facilitating informed decision-making and public awareness.

***Challenges***

**Data Accuracy:** Ensuring that the data presented in the app is accurate and up-to-date, given the rapidly changing nature of the pandemic.

**Global and Local Data:** Gathering both global and country-specific data from diverse sources and presenting it in a user-friendly format.

**Real-time Updates**: Providing real-time updates to users, as delays in information could result in outdated statistics and recommendations.

**User Accessibility:** Creating a user-friendly and accessible app that can be used by individuals, healthcare professionals, and policymakers.

***Proposed Solution***

We gave a proposed solution of the below key features.

**Global Data Display**

**Global Statistics:** Displaying global COVID-19 statistics, including total cases, recoveries, deaths, active cases, and testing rates.

**Historical Data:** Providing historical data trends, such as charts and graphs, to help users understand the virus's progression over time.

**Country-specific Data**

**Country Selection:** Allowing users to select a specific country to view detailed COVID-19 statistics for that country.

**Country Statistics:** Displaying country-specific data, including total cases, recoveries, deaths, active cases, and testing rates.

**Geographical Tracking:** Utilizing a location tracker feature to provide real-time information about COVID-19 cases in the user's current location.

**Real-time Updates**

**API Integration:** Utilizing an API database to fetch real-time COVID-19 data from reputable sources.

**Push Notifications:** Sending push notifications to users for important updates, such as changes in local containment measures or significant developments in the pandemic.

**User Interface**

**Intuitive Design:** Creating an intuitive and user-friendly interface with easy navigation and clear data visualization.

**Search Functionality:** Allowing users to search for specific countries quickly.

***Our app could be used to address real-world problems:***

Public health officials could use the app to track the spread of the virus and to identify areas where there is a high risk of transmission. This information could be used to target public health interventions, such as testing and vaccination campaigns.

Researchers could use the app to study the virus and to develop new treatments and vaccines. For example, the app could be used to track the emergence of new variants of the virus and to assess the effectiveness of different treatments and vaccines.

Individuals could use the app to stay informed about the spread of the virus in their area and to make informed decisions about their own health and safety. For example, the app could be used to identify areas where there is a high risk of exposure to the virus and to decide whether or not to avoid those areas.

***Responsive User Interfaces (Screenshots of your app on different screens & platforms)***

**Splash Screen in Portrait and landscape mode**

**A person looking at a magnifying glass

Description automatically generated**

**Home Screen in Portrait and landscape mode**

**A blue circle with red center

Description automatically generatedA screenshot of a medical report

Description automatically generated**

**Preventions Screens**

**A person wearing a mask

Description automatically generatedA person holding a baby

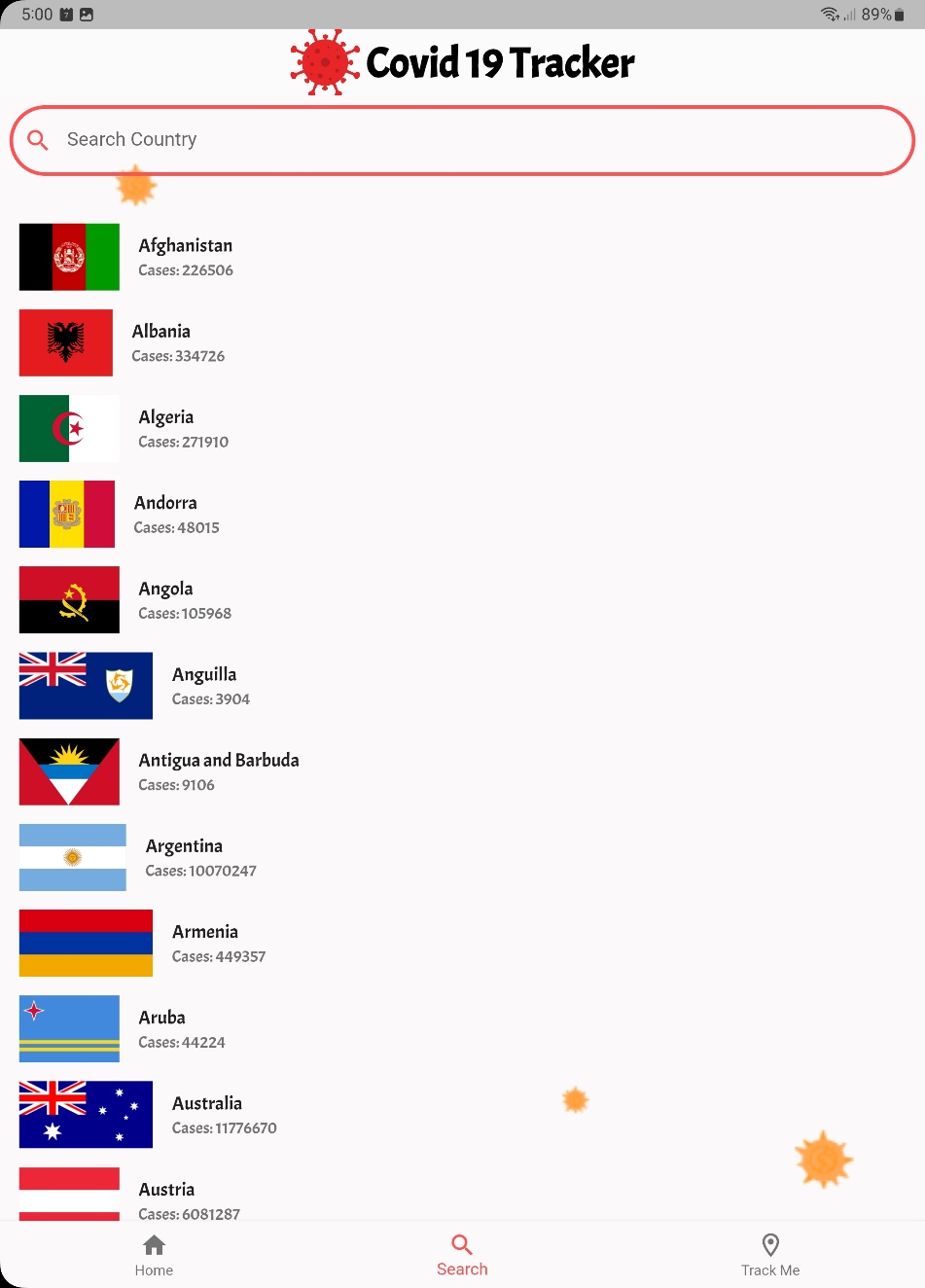
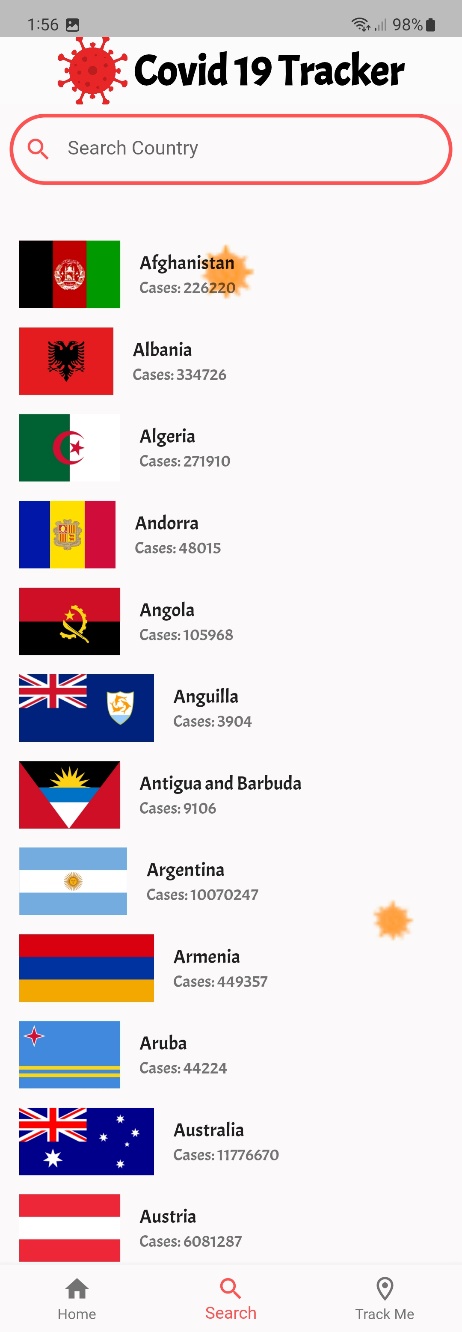
Description automatically generatedA person blowing her nose

Description automatically generated**

**A person holding a ball

Description automatically generated**

**Search Screen in Portrait and landscape mode**

******

**Track Me Screen in Portrait mode and Land Scape mode.**

**A screenshot of a cell phone

Description automatically generatedA screenshot of a computer

Description automatically generated**

**Country information screen in Portrait mode and Landscape mode.**

**A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated**

***Data Storage (With justification for using a particular database)***

In our COVID-19 tracker project, we have implemented a robust solution for handling data locally using the Sqflite database. This ensures that crucial information related to COVID-19 statistics is readily available even when the device is offline or the API is inaccessible.

***Justifications for Using Sqflite:***

***Lightweight and Efficient:***

Sqflite is a lightweight and efficient database that perfectly aligns with the resource constraints of mobile devices. This ensures optimal performance even on devices with limited processing power and storage.

***Offline Accessibility:***

One of the key requirements of our project is to provide users with seamless access to COVID-19 data even when they are offline. Sqflite's ability to store data locally allows our application to function independently of network availability, enhancing the user experience.

***Ease of Integration:***

Sqflite is seamlessly integrable with Flutter, the framework we used for developing our COVID-19 tracker app. Its integration is straightforward, allowing our development team to focus on building features rather than dealing with complex database setup.

***Support for Structured Data:***

Sqflite provides support for structured data storage, enabling us to organize COVID-19 statistics in a manner that aligns with our application's data model. This structured approach enhances data retrieval efficiency.

***Cross-Platform Compatibility:***

Our decision to use Sqflite is reinforced by its cross-platform compatibility. Since our COVID-19 tracker is developed using Flutter, having a database that works seamlessly across both Android and iOS platforms is crucial for a consistent user experience.

***Implementation Details:***

In the implementation of our COVID-19 tracker project, Sqflite is used to create tables that mirror the structure of the data obtained from the API. The database is queried efficiently to insert, update, and retrieve information. This implementation ensures a reliable and persistent storage mechanism for our COVID-19 statistics.

***APIs/Packages/Plug-ins (if used with justifications for using them).***

***APIs:***

In the development of our COVID-19 tracker application, we have strategically leveraged external APIs to provide real-time and accurate COVID-19 statistics. The three main APIs utilized in our project are:

**Global COVID-19 Data API:**

***https://disease.sh/v3/covid-19/all***

**Justifications:**

**Comprehensive Global Overview:**This API endpoint provides a comprehensive overview of the global COVID-19 statistics, including total cases, deaths, recovered cases, and more. It serves as the foundation for presenting worldwide data to users.

**Efficiency:**By accessing a single endpoint for global data, we optimize efficiency in data retrieval, reducing the number of API calls needed for a holistic global snapshot***.***

**Countries COVID-19 Data API:**

***https://disease.sh/v3/covid-19/countries***

**Justifications:**

**Country-Specific Data:**This API endpoint allows us to fetch COVID-19 statistics for all countries individually. It enables our application to present users with detailed information on the pandemic's impact in specific regions.

**Flexibility*:*** The ability to fetch data for individual countries provides flexibility in designing country-specific screens and enhances the user's ability to focus on regions of interest.

**Country-Specific COVID-19 Data API:**

***https://disease.sh/v3/covid-19/countries/{country}***

**Justifications:**

**Detailed Country Insights:** This API endpoint is instrumental in providing in-depth insights into the COVID-19 situation in a specific country. It includes information such as cases, deaths, recovered cases, and more, allowing users to access detailed and accurate data for any chosen country.

**Dynamic Content:**The dynamic nature of this endpoint ensures that our application stays up-to-date with the latest statistics for any selected country, providing users with real-time information.

**Implementation Insights:**

In our COVID-19 tracker project, these APIs are seamlessly integrated into the application's data retrieval mechanism. The data obtained from these APIs is then processed and presented to the user through an intuitive and user-friendly interface.

By relying on these APIs, we ensure that our application stays current with the rapidly evolving global and country-specific COVID-19 statistics. The use of specific endpoints tailored to our application's requirements enhances efficiency, flexibility, and the overall user experience.

***Packages:***

1. ***google\_fonts Package:***

**Justifications:**

Typography Enhancement: The google\_fonts package enables us to effortlessly integrate Google Fonts into our application, enhancing the visual appeal and readability of text across various screens.

Customization: This package offers a wide range of font styles and weights, allowing us to tailor the typography to our application's design, contributing to a cohesive and aesthetically pleasing user interface.

1. ***http Package:***

**Justifications:**

**API Interaction:**The http package is instrumental in making HTTP requests to fetch real-time COVID-19 data from external APIs. Its simplicity and effectiveness streamline the process of data retrieval and integration into our application.

**Asynchronous Operations:**Leveraging the asynchronous capabilities of the http package ensures that our application remains responsive during data fetching, preventing any delays in user interaction.

1. ***smooth\_page\_indicator Package:***

**Justifications:**

**Page Navigation Enhancement:**This package enhances the user experience by providing smooth and visually appealing indicators for page navigation. It is particularly useful in the presentation of informative content, such as COVID-19 symptoms, ensuring an engaging and intuitive user interface.

1. ***dart:async Library:***

**Justifications:**

**Asynchronous Programming:** The dart:async library is a fundamental component in managing asynchronous operations in Dart. It enables the effective handling of tasks such as asynchronous API calls, ensuring a responsive and seamless user experience.

1. ***syncfusion\_flutter\_charts Package:***

**Justifications:**

**Data Visualization:**The syncfusion\_flutter\_charts package facilitates the creation of visually compelling charts to represent COVID-19 statistics. Its versatility allows us to present data in a clear and informative manner, aiding users in comprehending complex information through graphical representations.

1. ***dart:convert Library:***

**Justifications:**

**JSON Parsing:**The dart:convert library is crucial for parsing JSON data retrieved from external APIs. It enables us to efficiently convert JSON responses into Dart objects, making the integration of real-time data into our application straightforward and reliable.

1. ***sqflite Package:***

**Justifications:**

**Local Data Storage:** The sqflite package is employed for local data storage, allowing us to store COVID-19 data locally. This ensures that users can access relevant information even when offline, contributing to a resilient and reliable application.

1. ***path Package:***

**Justifications:**

**Path Handling:** The path package simplifies path manipulation, aiding in the creation and management of local databases. It ensures that file paths are handled consistently across platforms, enhancing the stability and compatibility of our application.

1. ***geolocator Package:***

**Justifications:**

**Location Services:** The geolocator package is utilized to obtain the device's current location. It plays a crucial role in determining the user's country dynamically, ensuring that COVID-19 statistics are personalized based on the user's geographical location.

1. ***geocoding Package:***

**Justifications:**

**Reverse Geocoding:** The geocoding package complements the geolocator package by providing reverse geocoding capabilities. It translates geographic coordinates into human-readable addresses, facilitating the identification of the user's country for a personalized experience.

***Issues and Bugs Encountered and Resolved during Development***

1. **Network Connectivity Handling:**

**Issue:** The app needed to handle scenarios where the device had no internet connectivity or faced intermittent connection issues.

**Resolution:** Implemented a mechanism to check for network connectivity before making API calls. Added error dialogs to notify users when there was no internet connection.

1. **API Request Failures:**

**Issue:** Occasionally, API requests to retrieve COVID-19 data failed due to server issues or unexpected responses.

**Resolution:** Implemented robust error handling for API requests, displaying meaningful error messages to users and logging detailed errors for developers. Added retry mechanisms for transient failures.

1. **Location Services and Permissions:**

**Issue:** Obtaining user location information was challenging due to location service disabled or denied permissions.

**Resolution:** Implemented a comprehensive solution to check and request location permissions. Provided informative messages to guide users in enabling location services.

1. **User Interface Responsiveness:**

**Issue:** Ensuring a responsive UI across various screen sizes and orientations posed a challenge.

**Resolution:** Tested and adjusted UI layouts for responsiveness. Utilized Flutter's layout mechanisms to create adaptive user interfaces. Ensured a consistent user experience on different devices.

1. **Country Information Retrieval:**

**Issue:** Retrieving detailed information for a specific country from the API posed challenges, especially when data was not available.

**Resolution:** Implemented additional error handling for country-specific API requests. Displayed user-friendly messages in case of no data or API errors. Added logging for developers to trace issues.

1. **Geolocation Errors:**

**Issue:** Errors related to geolocation services, such as inaccurate coordinates or failure to retrieve location information.

**Resolution:** Enhanced error handling for geolocation services. Implemented mechanisms to handle edge cases and provide accurate location data.