Flight Tracker- A Real Time Flight Tracking Application

Meena Rao*

Parul Chaudhary#

*Associate Professor, Maharaja Surajmal Institute of Technology, Janakpuri, New Delhi, India

#Assistant Professor, Maharaja Surajmal Institute of Technology, Janakpuri, New Delhi, India

Abstract

This paper presents the development and implementation of a comprehensive Real- Time Flight Tracking Application that leverages numerous technologies and APIs to offer users a smooth and informative practice. This application integrates real-time flight data from external APIs, utilizes Google Maps for interactive visualizations, employs MongoDB for user account management, and implements a robust backend for authentication, all while presenting a user-friendly frontend developed with React.

Flight Tracker relies on external APIs to retrieve and display real-time flight data. Users gain access to live information on flight statuses, departure and arrival times, aircraft details, and current flight positions. The application employs the Google Maps API to enhance user engagement by visualizing flight routes and the precise positions of aircraft in real-time. MongoDB is the chosen database solution for user account creation and management. Users can effortlessly register accounts and login users.

The backend has stringent authentication protocols to safeguard user accounts and data. A user-friendly and responsive interface is developed using React for the frontend. This choice ensures that users, whether they are novices or aviation enthusiasts, can easily access and navigate the application's features.

Flight Tracker empowers users with the ability to monitor and track flights in real-time, providing a valuable tool for travelers, aviation enthusiasts, and anyone interested in global air traffic. By successfully integrating these technologies and APIs, the RTFTA emerges as a versatile application with numerous applications in the aviation industry and beyond.

Keywords: Flight Tracker, Flight search, Real time flight updates, Airport

1. Introduction

With the ever-increasing number of flights and passengers, there is a growing need for a reliable and user-friendly flight tracking application that can provide real-time information about flights to passengers, airlines, and other stakeholders [1]. This is because flight tracking applications can help to reduce anxiety and stress for passengers by providing them with up-to-date information about their flights, such as delays, cancellations, and gate changes.

The primary objective of a flight tracking application is to provide real-time information about flights to users. This information can include the flight's origin, destination, status, location, and estimated arrival time [2]. In addition to this basic information, flight tracking applications can also provide a variety of other features, such as:

- **Real-time flight updates**

  Flight tracking applications should provide real-time updates on flight delays, cancellations, and other changes.
- **Flight search**

  Flight tracking applications should allow users to search for flights by origin, destination, airline, and other criteria.
- **Flight information**

  Flight tracking applications should provide detailed information about flights, such as the aircraft type, departure and arrival times, and the flight's route.
- **Interactive map**

  Flight tracking applications should provide an interactive map that shows the location of flights in real time.

Utilising a flight tracking tool offers several advantages, such as:

- Reduced anxiety and stress for passengers
- Improved efficiency and customer service for airlines
- Increased productivity for business travelers
- Enhanced travel experience for all

Additionally, flight tracking applications can help airlines to improve their efficiency and customer service by providing them with real-time data about their flights, that can

assist them in making more informed decisions on staffing, scheduling, and routing.

2. Methodology

The development of the real-time flight tracking application followed a structured approach to ensure successful implementation. The methodology can be outlined as follows [3]:

- **Project Planning:** The initial phase involved comprehensive planning and requirement analysis. Key objectives and functionalities were identified, and a project plan with timelines and deliverables was created.
- **Technology Stack Selection:** The appropriate technologies were chosen for frontend and backend development. React.js was selected for the frontend due to its speed, flexibility and performance. On the backend, Node.js was used for building RESTful APIs.
- **User Authentication:** A secure user authentication mechanism was implemented using JSON Web Tokens (JWTs) to ensure user privacy and data protection. Calendar and more.
- **Frontend Development:** The frontend was developed, focusing on creating a user-friendly interface for smooth navigation and interaction.
- **Testing:** Rigorous testing, including unit testing and integration testing, was conducted to identify and fix bugs and ensure the application's functionality and reliability.
- **API Integration:** External APIs for real-time flight data and maps were connected and implement data retrieval and synchronization mechanisms were implemented.
- **Database Implementation:** MongoDB database was setup and configured for user account management and data storage.
- **Authentication:** Secure authentication mechanisms to protect user data and privacy were implemented.
- **UI Improvements:** The user interface was continuously refined to enhance the overall user experience.
- **Deployment:** Different hosting services were selected for deploying backend and frontend.

By following this methodology, Flight Tracker was successfully developed, offering

users an efficient platform for seamless real time flight tracking. The step-by-step approach allowed for the integration of essential features such as secure authentication, real-time messaging, and UI improvements [4]. This project has been built following the stages of the Software Development Life Cycle (SDLC). The six phases of the Software Development Life Cycle (SDLC) are: Feasibility study, Requirement Analysis, Design, Coding, Testing, Implementation

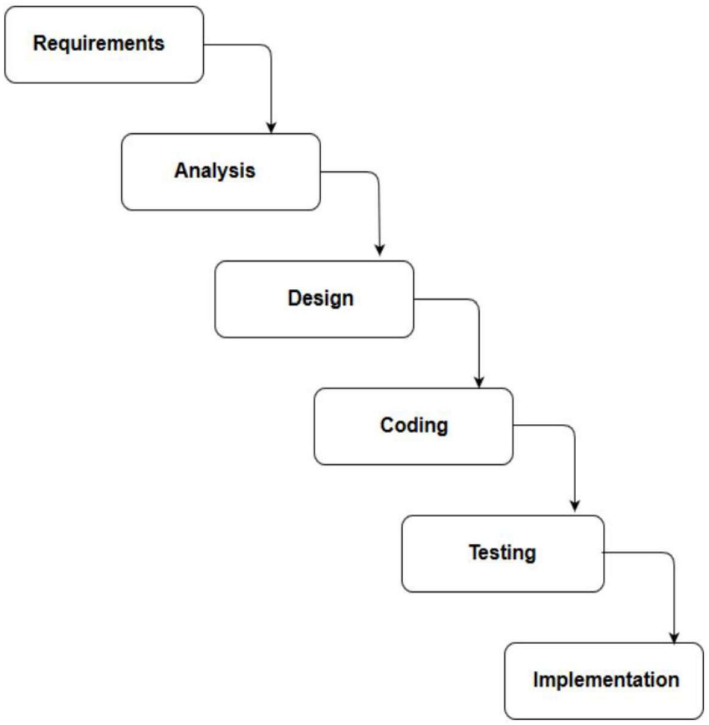

Fig.1. Waterfall Model

2.1 Use Case Diagram

A use case diagram illustrates the dynamic behaviour of a system. It encapsulates the system's functionality by integrating use cases, actors, and their interrelations [5]. It delineates the activities, services, and functionalities necessary for a system or subsystem of an application [6]. It illustrates the overarching operation of a system and delineates user interaction with the system.

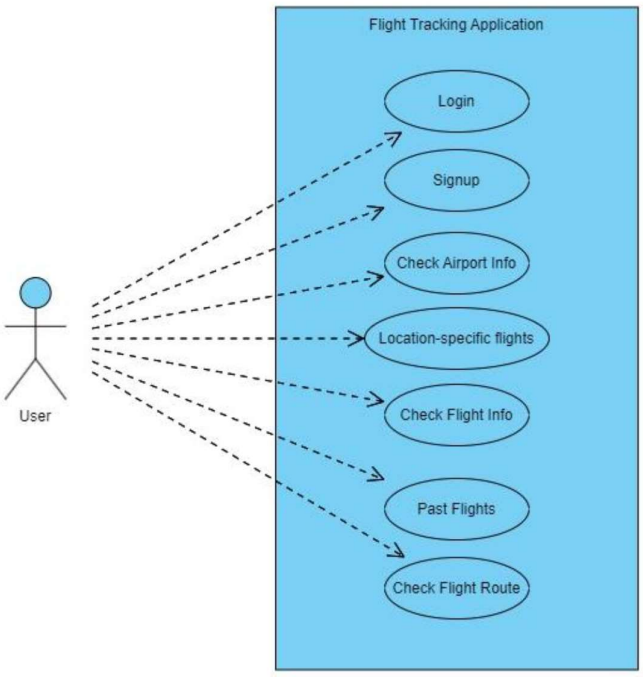

Fig. 2. Flight Track -Use Case Diagram

Actors:

- User- The user is the primary user of the flight tracking application. They use the application to track their flights, receive updates on flight delays and cancellations, and view other flight information[7][8].

Use cases:

- Login- User can login on the platform given their email and password
- Signup- Users can register on the platform by setting an email and password
- Check Airport Info- Users can read information about any airport on the map.
- Location-specific flights-Users can filter flights according to a country.
- View Stats- The user uses the flight tracking application to track a specific flight.
- Past flights – Users can review information about flights in the past
- Check flight route – Users can track the flight route from origin to destination

2.2 Activity Diagram

An activity diagram is a flowchart that illustrates the transition from one activity to another. The action can be characterised as a system operation. The control flow transitions from one operation to another. This process may be sequential, branching, or concurrent. Activity diagrams address various forms of flow control through the use of distinct elements, including fork and join. It encapsulates the system's dynamic behaviour. An event is established as an activity diagram with a collection of nodes linked by edges. Activities can be associated with any modelling element to represent their behaviour. It can represent use cases, classes, interfaces, components, and partnerships. It mostly simulates processes and workflows. It conceptualises the system's dynamic behaviour and develops a functional system that integrates both forward and reverse engineering [9][10]. The message component is excluded, indicating that message flow is not included in an activity diagram.

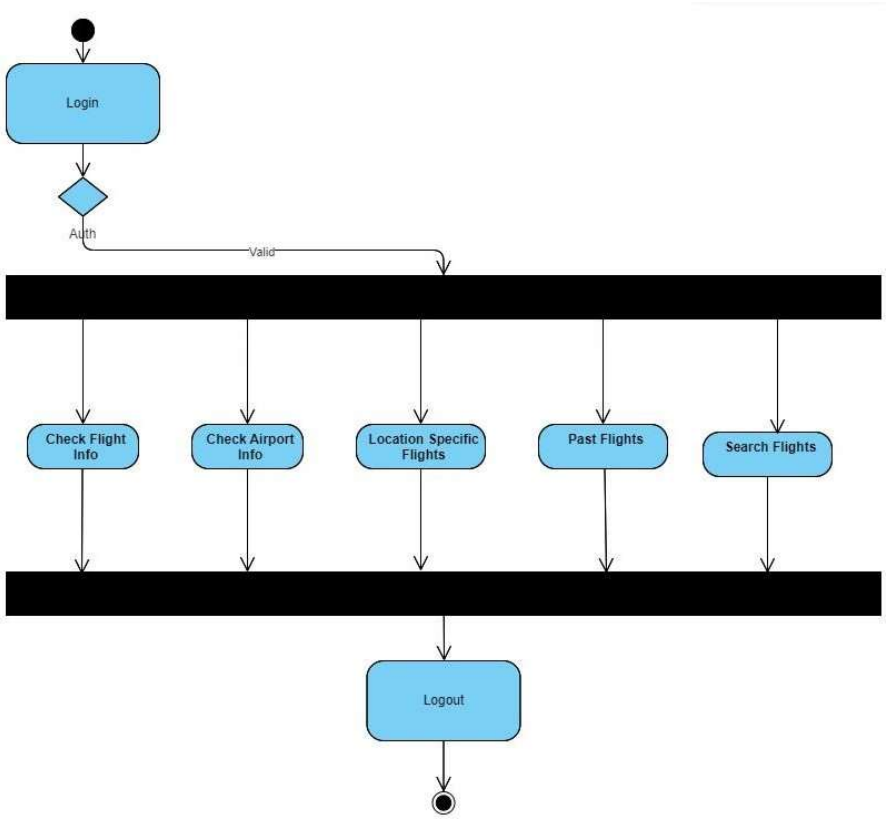

Fig.3. Activity diagram of flight tracking application

The above activity diagram shows five main activities:

- Check flight info

- Check airport info
- Check location-specific flights
- Past flights
- Search Flights

After the user logs in into the platform, he/she can perform any of the above five actions. Once the work is done, the user logs out.

3. Implementation

a) **Frontend-**

This project is built using React as the major framework. Thus, the initial steps include setting up the app using: npx create-react-app ./

After the initial setup, the implementation of all the features is as follows:

b) Identity Management

Flight Tracker uses username and password for identity management. The usernames and hashed passwords are stored in the database on user registration and verified on login.

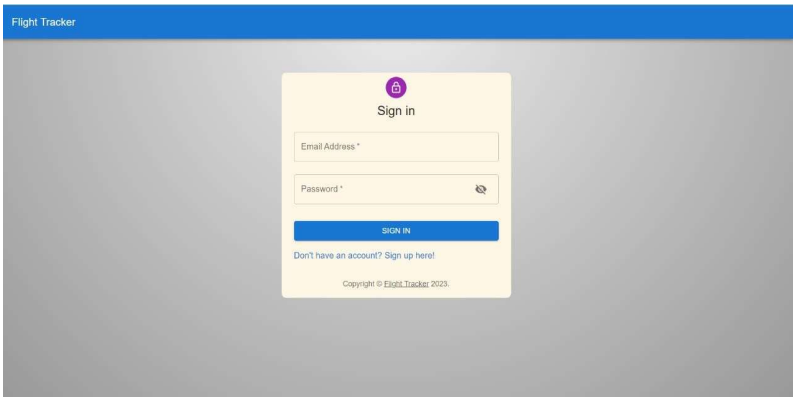

Fig.4. Identity Management

c) Filter flight results

Live flight results can be filtered on the basis of country or time

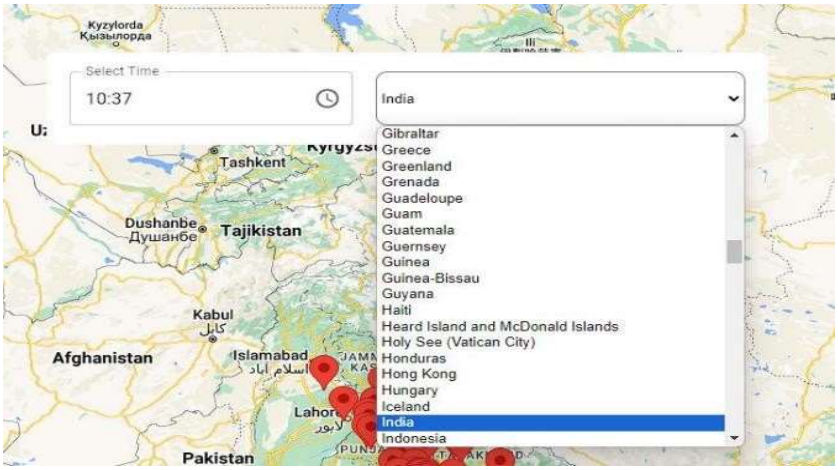

Fig.5. Filtering Results by Country

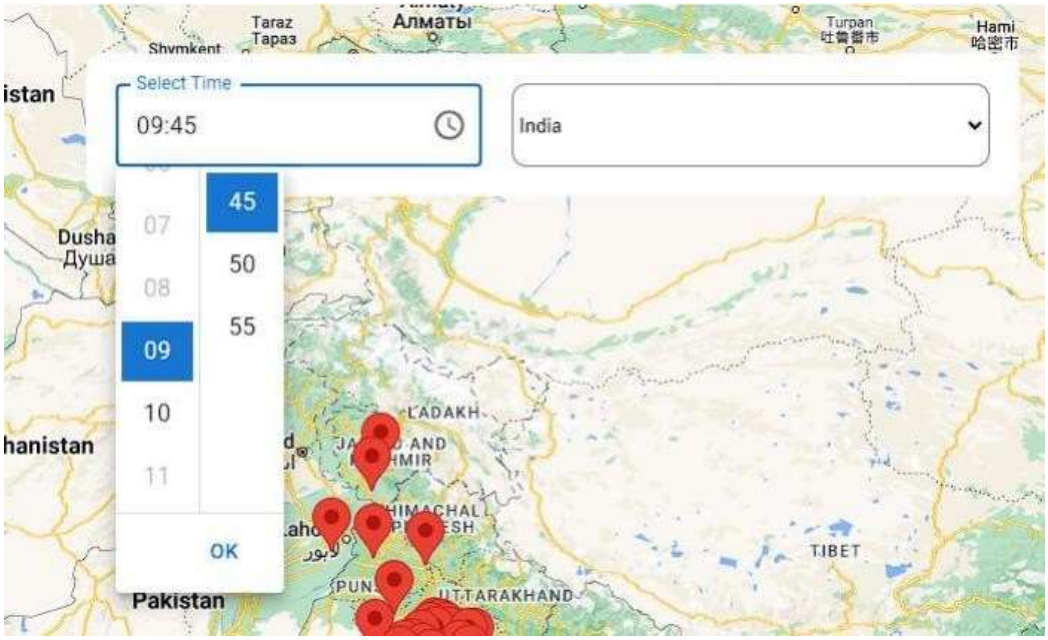

Fig. 6. Filtering results by time

d) Airport Markers

Markers representing airport are present on the map for the selected country. On clicking, the name of the airport is shown.

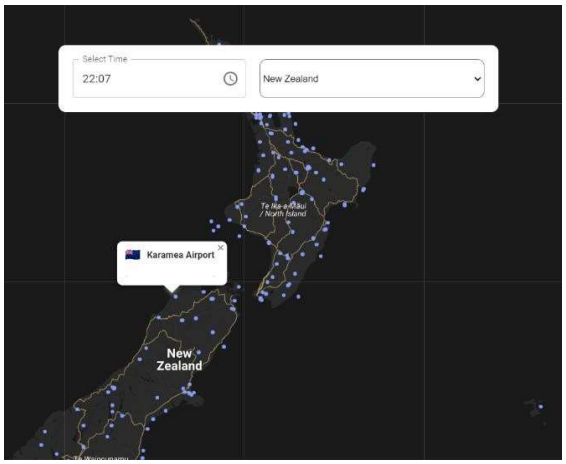

Fig. 7. Airport Markers

e) Live Flights Information

Live flights for the selected country or time are shown with symbols of different colours that are decoded in the information box that appears on clicking [11][12]. Other information such as the departure airport, arrival airport, altitude, speed, direction (in degrees) is shown.

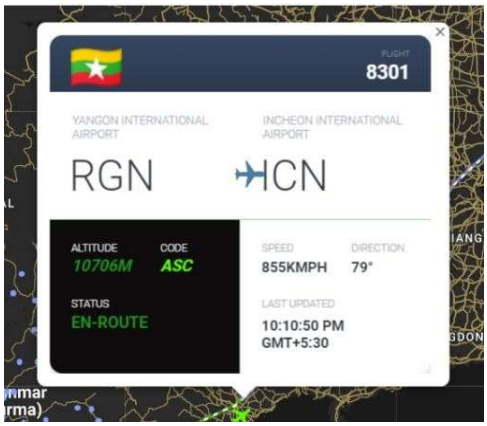

Fig. 8. Live Flights Information

f) Flight Route

On clicking on the flight icon, line connecting waypoints from departure airport to the current location and further till the arrival airport is shown to indicate the flight path [13][14].

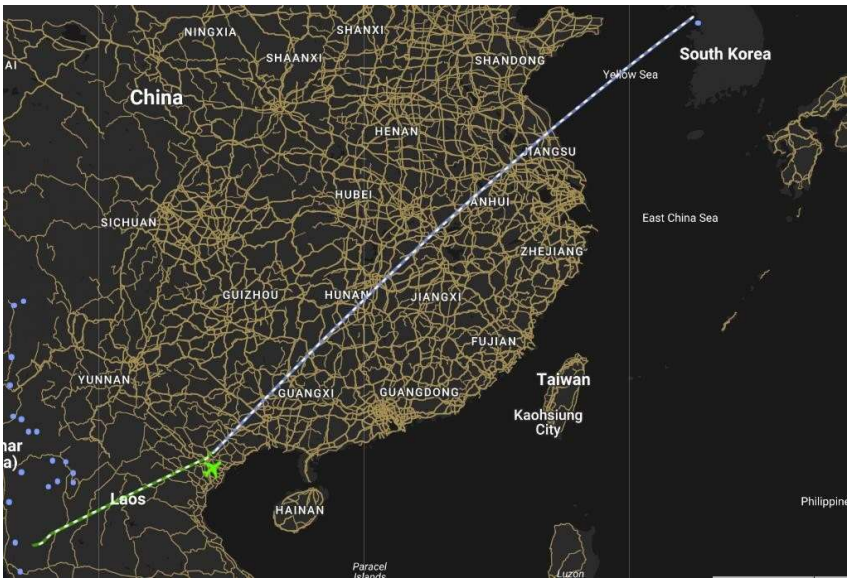

Fig. 9. Flight Route

g) Toast Notification

Toast notification was implemented for error handling and better user experience. The error message received from backend are shown if any unusual behavior occurs such as:

"User already exists, please login instead" when an existing user signs up.

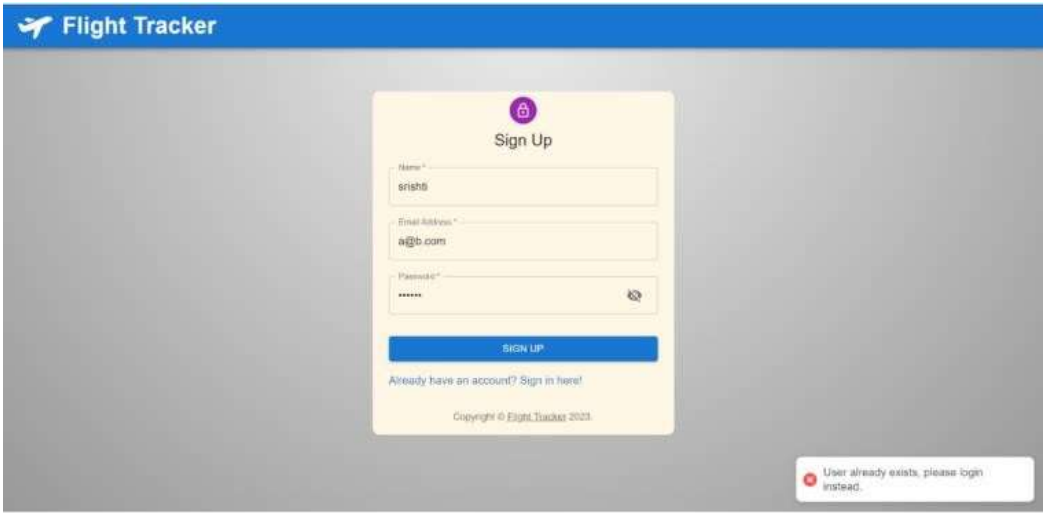

Fig. 10. Unsuccessful Login

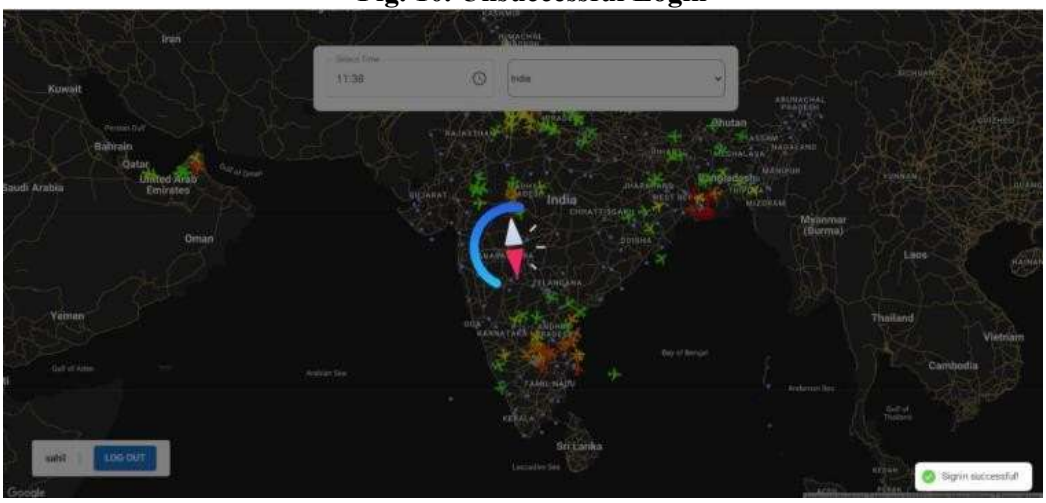

Fig. 11. Successful login

h) JWT-based Authentication

JSON Web Tokens (JWT) were chosen as the authentication mechanism for this project. JWTs were generated upon successful login and sent to the client, allowing them to be included in subsequent requests for authentication purposes. Token verification was performed on the backend to ensure the validity of requests. The token received and user Id is stored in the local storage on login and deleted on logout**.**

Fig. 12. JWT-based authentication

i) Database Setup and Configuration

A database was set up to store user information and authentication-related data. We chose MongoDB as the database management system due to its compatibility with the project requirements[15][16]. Objects were created to store user profiles, hashed password, and other relevant data.

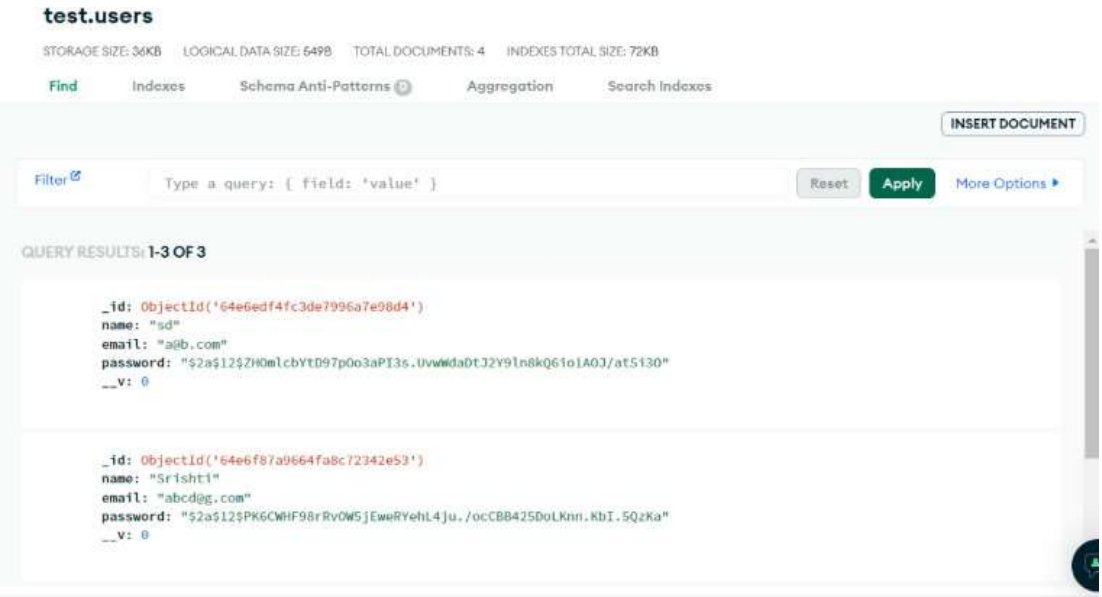

Fig. 13. Database setup and configuration

4. Results

After implementing the above components, the final website is as follows:

a) Landing Page

On opening the flight tracking application, the user lands on a signup/ login page. Registered users can signing with their email and password. New users can sign up with their name, email id and password.

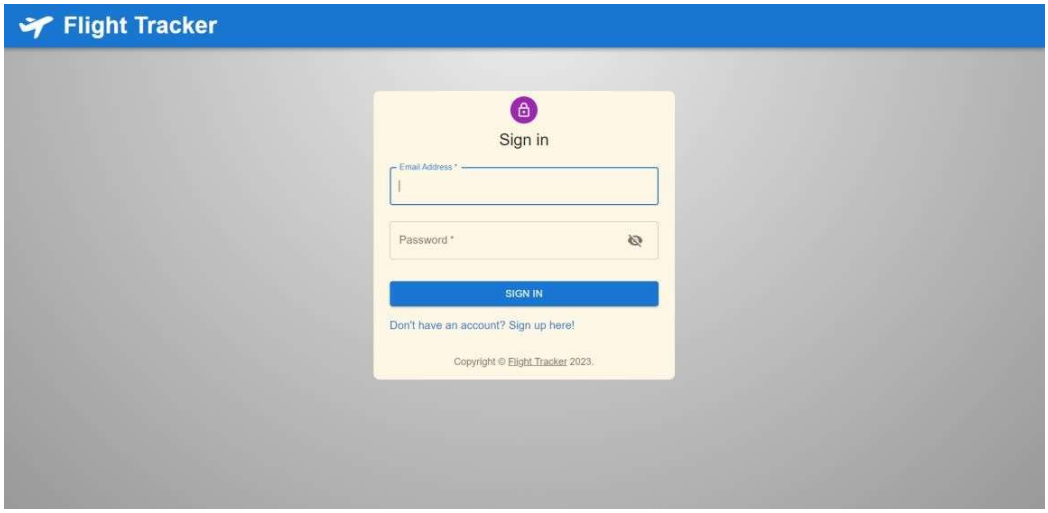

Fig. 14. Landing page-I

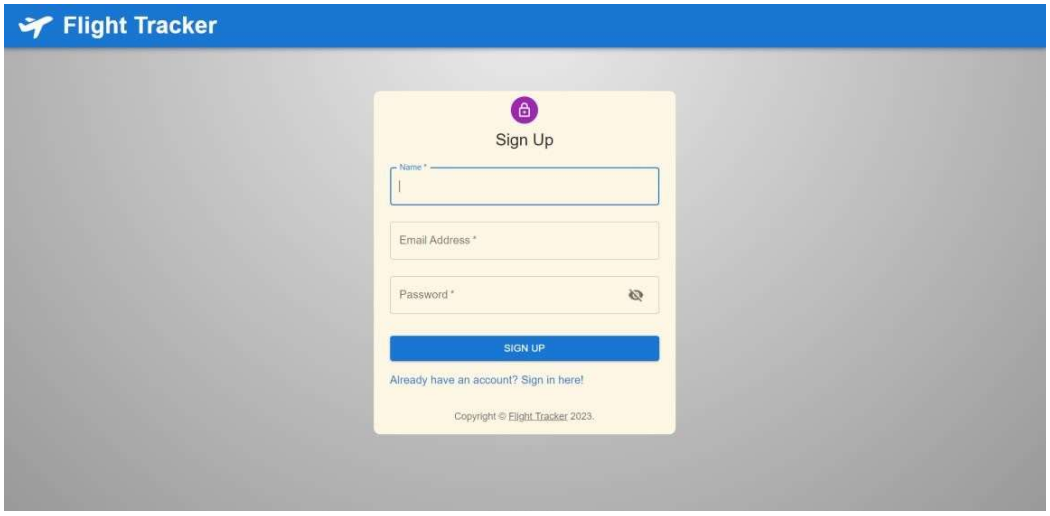

Fig. 15. Landing page-II

b) Home page

On successful login, a map is shown with current flights, a search bar to filter results and a logout button. The map is marked with dots to represent airports. The flight symbol colour and direction is decided on the basis of direction and altitude.

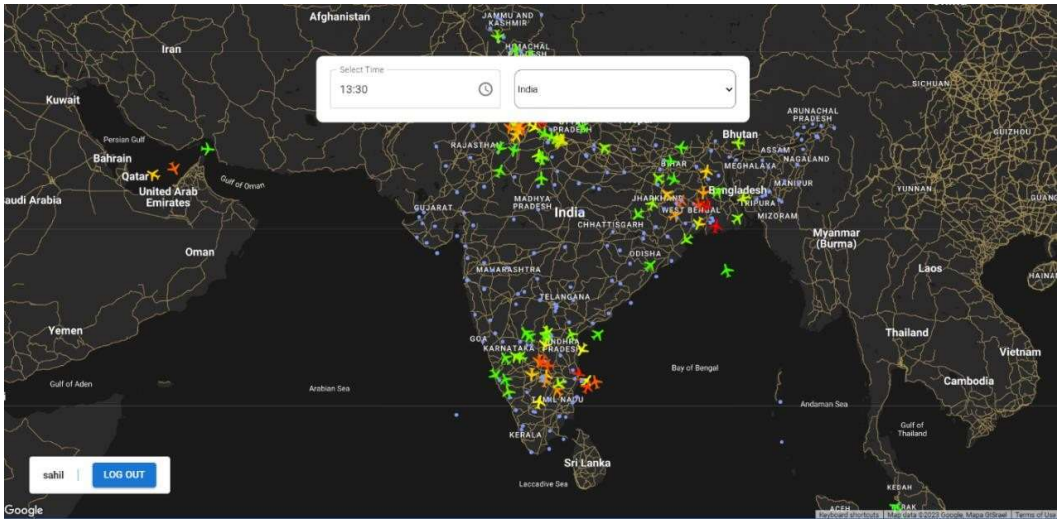

Fig. 16. Home page

c) Filter flight results

The search bar allows users to filter the flight results by country or time as required.

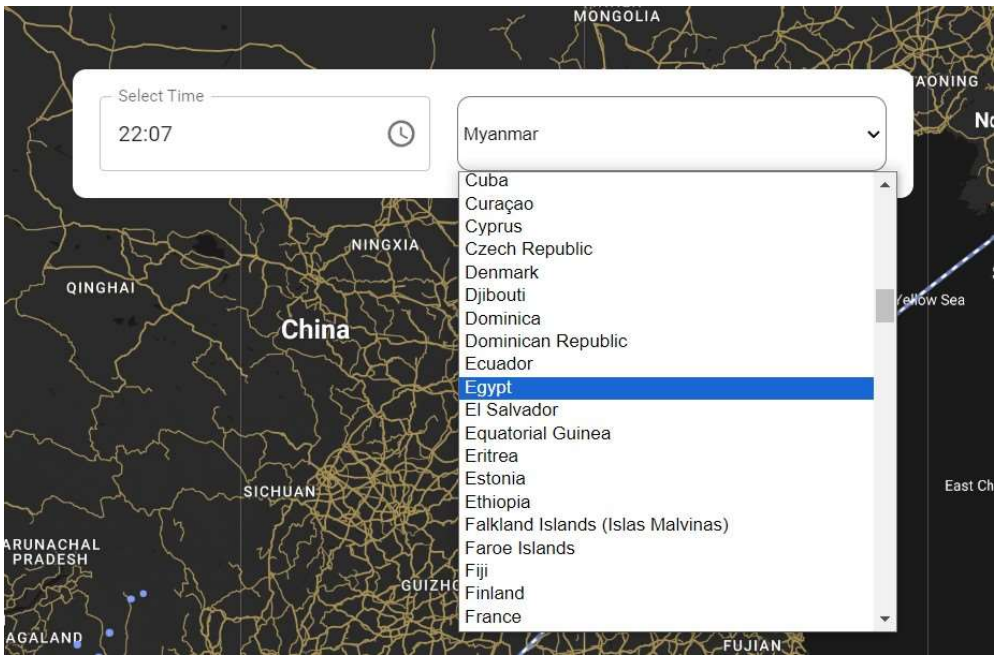

Fig. 17. Filter flight results

d) Airports

Airports are represented by a blue dot, distributed over the entire map. On clicking, a popup box shows the airport name along the flag of the country where it is located.

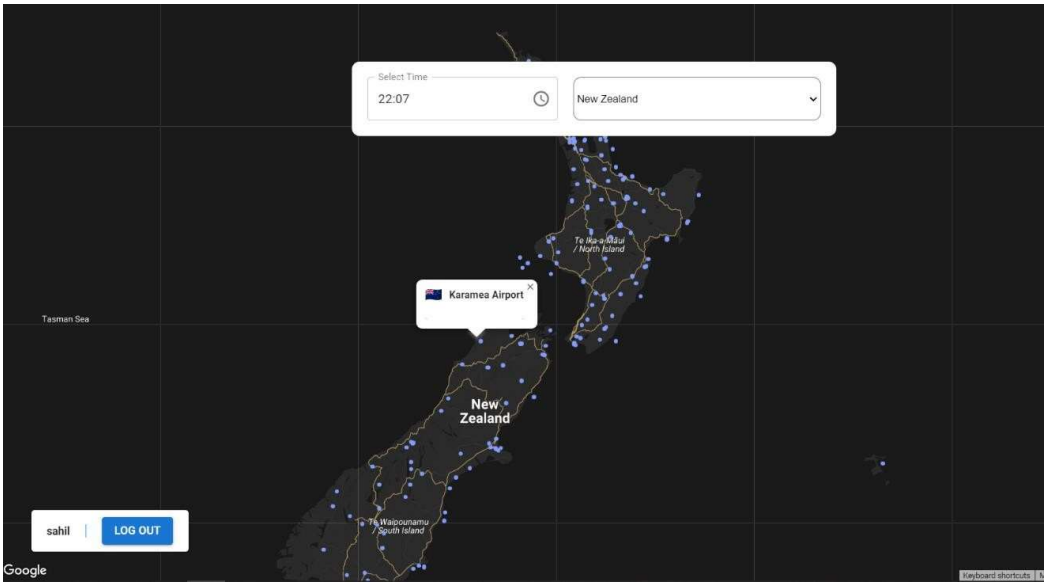

Fig.18. Airports

e) Live Flight Information

This application shows live flights flying over the selected area. The flight symbols can be clicked to get all the information about the flight such as: departure and arrival airports, flight number, altitude, symbol color code, speed, direction, etc.

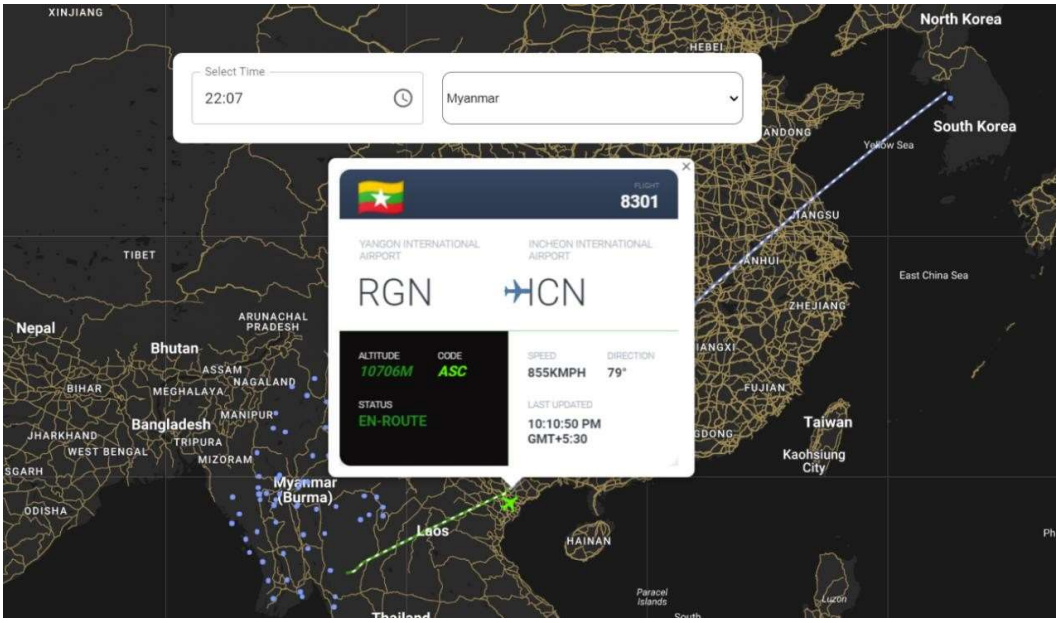

Fig. 19. Live Flight Information

f) Flight route

Flight routes are visualized with a line from the departure airport to the arrival airport. The colour difference clearly shows the path travelled and the path to be travelled. The path is a result of connecting all the waypoints from source to destination.

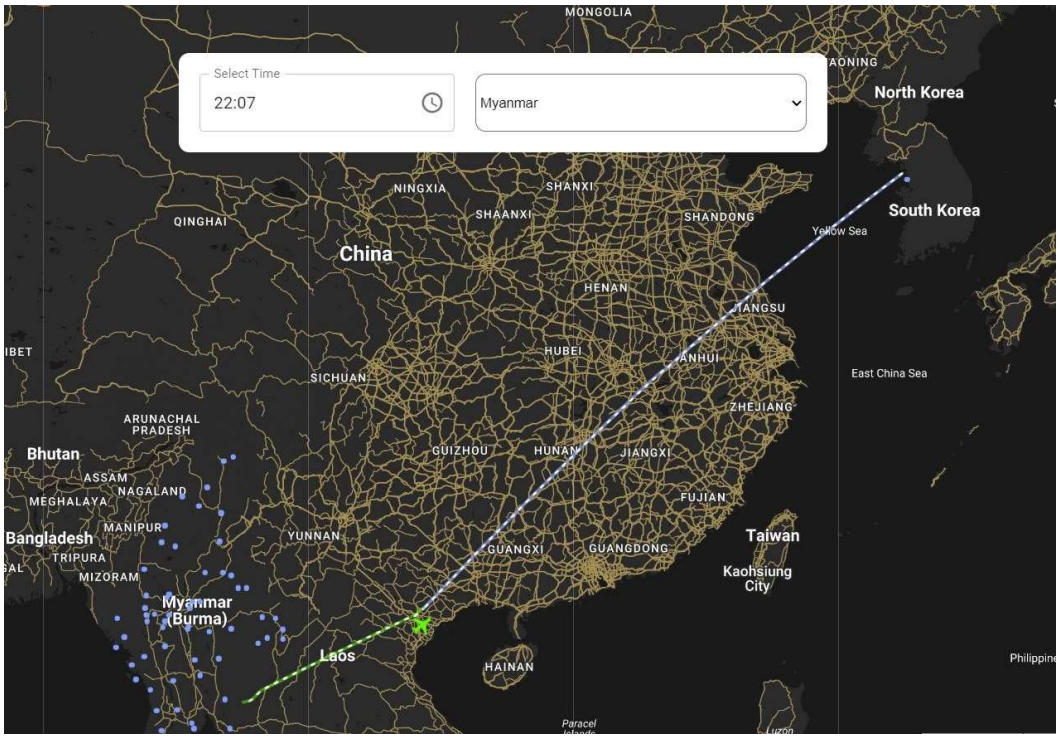

Fig. 20. Flight route

5. Conclusion and Future Scope

A comprehensive Real- Time Flight Tracking Application was implemented. Also, results were analyzed and observed.

As the real-time flight tracking application continues to evolve, there are several potential areas of improvement and expansion that can enhance the overall user experience and security. These are as follows:

a) User Preferences

Expand the data base to store user preferences and other personalized travel choices.

b) Forgot Password

Implement a password recovery tool, such as a "Forgot Password" feature, to aid users in restoring access to their accounts. Implement a password recovery method, such as a "Forgot Password" feature, to aid users in restoring access to their accounts in case of password loss.

c) Voice Assistants

Integrate voice-controlled features to make the application more accessible to users with disabilities.

By addressing these future work areas, the application can further elevate its usability, security, and user satisfaction, ensuring it remains a reliable and sought- after platform for real-time flight tracking.

References

1. Li, Y., Zhang, Y., Wang, L., & Guan, X. (2022). Research on potential ground risk regions of aircraft crashes based on ADS-B flight tracking data and GIS. *Journal of Transportation Safety & Security*, *14*(1), 152-176.
2. Zhang, X., & Mahadevan, S. (2020). Bayesian neural networks for flight trajectory prediction and safety assessment. *Decision Support Systems*, *131*, 113246.
3. Chen, M., Xiong, S., & Wu, Q. (2019). Tracking flight control of quadrotor based on disturbance observer. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *51*(3), 1414-1423.
4. Murça, M. C. R., Hansman, R. J., Li, L., & Ren, P. (2018). Flight trajectory data analytics for characterization of air traffic flows: A comparative analysis of terminal area operations between New York, Hong Kong and Sao Paulo. *Transportation Research Part C: Emerging Technologies*, *97*, 324-347.
5. Xu, B., Yang, C., & Pan, Y. (2015). Global neural dynamic surface tracking control of strict-feedback systems with application to hypersonic flight vehicle. *IEEE Transactions on Neural Networks and Learning Systems*, *26*(10), 2563-2575.
6. Teoh, R., Schumann, U., Majumdar, A., & Stettler, M. E. (2020). Mitigating the climate forcing of aircraft contrails by small-scale diversions and technology adoption. *Environmental science & technology*, *54*(5), 2941-2950.
7. Ma, B., Liu, Z., Zhao, W., Yuan, J., Long, H., Wang, X., & Yuan, Z. (2023). Target tracking control of UAV through deep reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems*, *24*(6), 5983-6000.
8. Horri, N., & Pietraszko, M. (2022). A tutorial and review on flight control co-simulation using matlab/simulink and flight simulators. *Automation*, *3*(3), 486-510.
9. Alhafnawi, M., Salameh, H. A. B., Masadeh, A. E., Al-Obiedollah, H., Ayyash, M., El-Khazali, R., & Elgala, H. (2023). A survey of indoor and outdoor uav-based target tracking systems: Current status, challenges, technologies, and future directions. *IEEE Access*, *11*, 68324-68339.
10. Baharuddin, A. D., & Basri, M. A. M. (2023). Trajectory tracking of a quadcopter uav using pid controller. *ELEKTRIKA-Journal of Electrical Engineering*, *22*(2), 14-21.
11. Dhand, G., Rao, M., Chaudhary, P., & Sheoran, K. (2025). A secure routing and malicious node detection in mobile Ad hoc network using trust value evaluation with improved XGBoost mechanism. *Journal of Network and Computer Applications*, *235*, 104093.
12. Rao, M., Chaudhary, P., Sheoran, K., & Dhand, G. (2023). A secure routing protocol using hybrid deep regression based trust evaluation and clustering for mobile ad-hoc network. *Peer-to-Peer Networking and Applications*, *16*(6), 2794-2810.
13. Zubairi, J. A., & Er, A. (2012, December). Fault tolerant aviation data tracker design. In *High Capacity Optical Networks and Emerging/Enabling Technologies* (pp. 047-051). IEEE.
14. Pycinski, B., Czajkowska, J., Badura, P., Juszczyk, J., & Pietka, E. (2016). Time-of-flight camera, optical tracker and computed tomography in pairwise data registration. *Plos one*, *11*(7), e0159493.
15. Bodie, K., Tognon, M., & Siegwart, R. (2021). Dynamic end effector tracking with an omnidirectional parallel aerial manipulator. *IEEE Robotics and Automation Letters*, *6*(4), 8165-8172.
16. Kumar, M., & Mondal, S. (2021). Recent developments on target tracking problems: A review. *Ocean Engineering*, *236*, 109558.