Flight Reservation System Project Overview

This project presents a console-based Flight Reservation System developed in C, simulating core airline booking functionalities. It uses arrays to manage flights and linked lists for dynamic passenger information, enabling efficient booking, cancellation, and passenger display.

The system serves as an educational tool to demonstrate data structures, dynamic memory allocation, and pointer manipulation in C, providing a foundational understanding of reservation platforms.

Background and Objectives

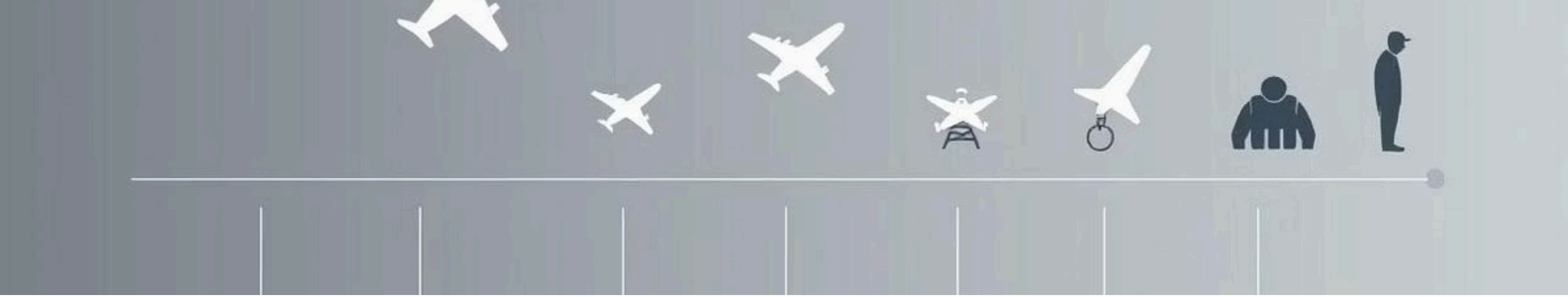
Background

Traditional manual booking methods are error-prone and slow. Computerized systems improve efficiency and customer satisfaction by managing flight reservations digitally.

This project leverages C programming to simulate flight management using arrays and passenger handling with linked lists.

Objectives

- Design a flight reservation system in C.
- Use arrays for flight data management.
- Employ linked lists for passenger details.
- Implement booking, cancellation, and display functions.
- Enhance understanding of data structures and memory management.



Existing Systems and Their Limitations

Traditional Airline Systems

Initially manual and error-prone, later replaced by Computer Reservation Systems (CRS) like SABRE and Galileo, which automated bookings but had limited passenger autonomy.

Limitations

- Passengers couldn't select seats or print boarding passes independently.
- Lack of real-time updates on flight changes.
- Data silos hindered information sharing across airlines.
- Security vulnerabilities in centralized databases.

Modern Advances

Improvements include user-centered design, microservices, AI for predictive analytics, and blockchain for secure transactions.

Software Requirement Analysis

Functional Requirements

- Flight management: add, update, delete flights.
- Passenger management: book, update, cancel reservations.
- Seat allocation based on availability.
- Search flights by destination, date, or number.

Non-Functional Requirements

- Performance: process requests within 2 seconds.
- Usability: intuitive interface.
- Reliability: less than 1% downtime monthly.
- Maintainability: modular design.

System Design and Architecture

1

User Interface Module

Handles user input and menu navigation with clarity and ease.

2

Flight Management Module

Manages flight data using arrays, including validation and updates.

3

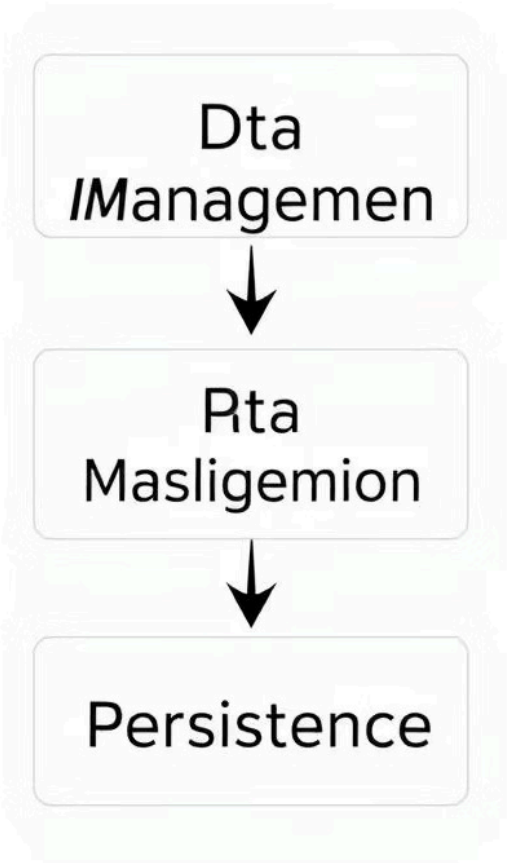
Passenger Management Module

Uses linked lists to book, cancel, and list passengers, ensuring seat availability.

4

Data Persistence Module

Reads and writes data to files, maintaining consistency across sessions.



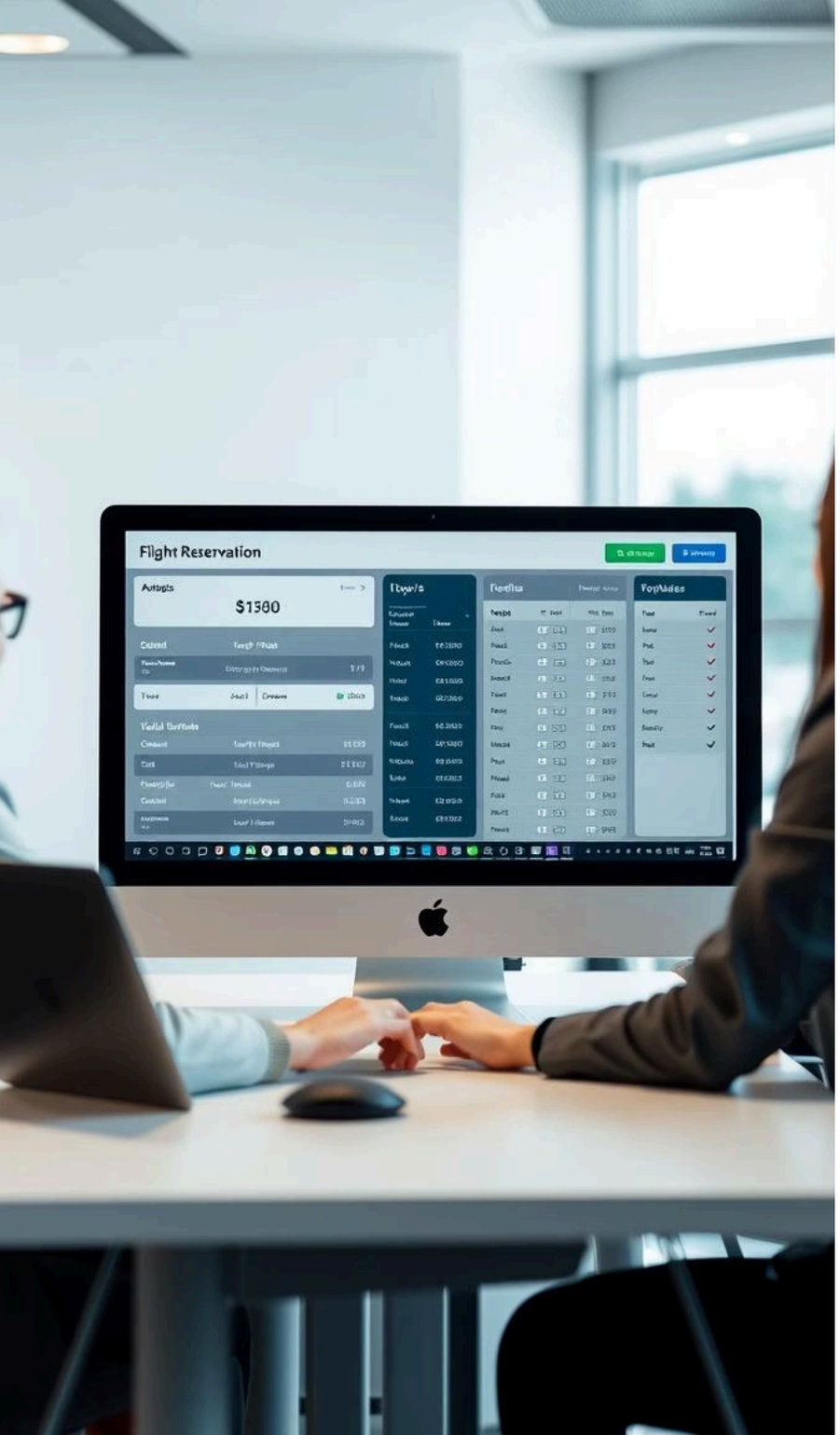
Implementation Details and Data Structures

Flight Structure

Array of structures storing flight number, origin, destination, times, and seat availability.

Passenger Structure

Singly linked list nodes containing passenger name, passport number, seat number, and pointer to next.



Project Achievements and Advantages

Efficient Data Management

Arrays and linked lists organize flight and passenger data effectively.

User-Friendly Interface

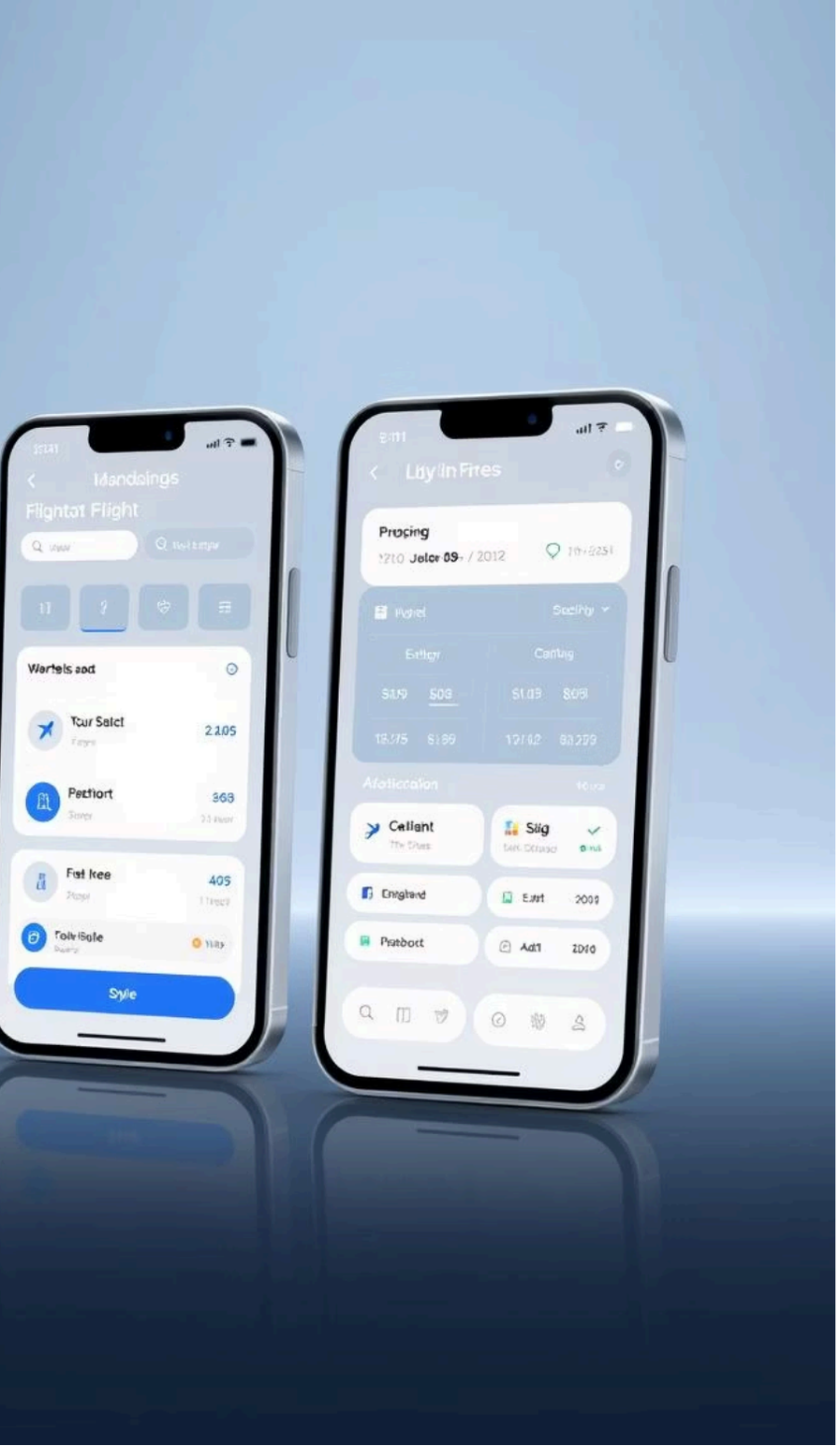
Console-based menu system enables easy interaction for booking and cancellations.

Data Persistence

File handling preserves data across sessions, ensuring reliability.

Scalability and Accuracy

Designed to handle multiple flights and passengers with consistent data integrity.



Future Enhancements and Conclusion

1 Graphical User Interface

Develop GUI for improved usability using libraries like GTK or Qt.

2 Database Integration

Replace file storage with relational databases for better data management.

3 Authentication and Real-Time Data

Implement user login and integrate APIs for live flight updates.

4 Mobile and Multilingual Support

Create mobile apps and support multiple languages and currencies.

The project demonstrates core data structure concepts and provides a foundation for developing more advanced, user-centric flight reservation systems.