CASE STUDY

ON

AIRLINE RESEVATION SYSTEM

An Experiential Learning

Submitted in partial fulfilment of the requirements for the degree of

B.Tech. in Computer Science and Engineering

By: Group 27, Batch-07(C++)

Dept. CSE. C. V. Raman Global University, Odisha, Bhubaneshwar

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

C. V. Raman Global University, Odisha, Bhubaneswar

PIN -752054, India

TEAM MEMBERS

Sl No.	Registration Number	Name of Student
1	2201020093	Lipakhi Tripathy
2	2201020130	Sofia Akhtar
3	2201020340	Y.Sherisha

CERTIFICATE

This is to certify that the experiential learning report entitled "**Airline Reservation System**"submitted in partial fulfilment of the requirement for the award of Bachelor of Technology in **CSE** of the C. V. Raman Global University, Odisha during the year 2023-2024, is a faithful record of the bonafide work carried out by under my guidance and supervision.

Sir Mohammad Sikander
Cranes Varsity
C. V. Raman Global University, Odisha, Bhubaneswar, PIN- 752054

ACKNOWLEDGEMENT

We express our sincere gratitude to our mentor **Mohammad Sikander** for their invaluable guidance throughout this project. We also extend our thanks to the **faculty members of the Cranes Varsity** for their insights and encouragement. Lastly, we acknowledge the support from our peers and the university, whose encouragement has been instrumental in completing this project successfully.

Special thanks to our team of instructors, content creators, and reviewers for their tireless efforts in shaping this course. We would also like to acknowledge the support and encouragement from our colleagues and mentors throughout this journey.

Finally, we extend our heartfelt appreciation to all the learners who have embarked on this data structures adventure with us. Your enthusiasm and commitment to learning inspire us every day.

Thanking you

Group-27, Batch -07 (C++)

ABSTRACT

The **Flight Reservation System** is a desktop application developed using **Qt (C++ GUI framework)** to provide an efficient and user-friendly interface for booking, searching, and managing flight reservations. The system streamlines flight ticket reservations, making it easier for passengers to search available flights and book tickets.

Key Features:

- **User Registration & Authentication**: Secure login and signup mechanism for users.
- **Flight Booking Module**: Enables users to book flights with details like passenger name, flight number, source, destination, and seat type.
- **Search Functionality**: Allows users to search for flights and view booked tickets.
- **Cancellation & Modification**: Users can modify or cancel existing reservations.
- **GUI Components**: Uses **Qt Widgets (QLineEdit, QpushButton, QtableView, Qlabel, QcomboBox, QmessageBox)** for an interactive user experience.

The system ensures **data persistence** through integration with a **SQLite database**, allowing for seamless storage and retrieval of reservation details. The application is built using **C++** and **Qt Framework**, making it highly efficient and cross-platform compatible. Future enhancements could include payment gateway integration, seat selection, and real-time flight status tracking.

CONTENTS

Sl No	**TOPIC**	**PAGE NO**
1	INTRODUCTION	6
2	SOURCE CODE	7
3	SYSTEM AR CHITECTURE OVERVIEW	11
4	FUNCTIONALITY OVERVIEW	12
5	GUI DESIGN & USER INTERFACE ANALYSIS	13
6	SECURITY AND INTEGRATION ANLYSIS	14
7	CONCLUSION	15
8	REFERENCES	15

1.INRODUCTION

The **Flight Reservation System** is a modern and user-friendly application developed using **Qt (C++ GUI framework)** to streamline the process of booking and managing flight reservations. This project provides a simple yet efficient interface for passengers to book flights, search for reservations, and manage bookings with ease. Additionally, a reward system is integrated to enhance user engagement.

The main objectives of this project are:

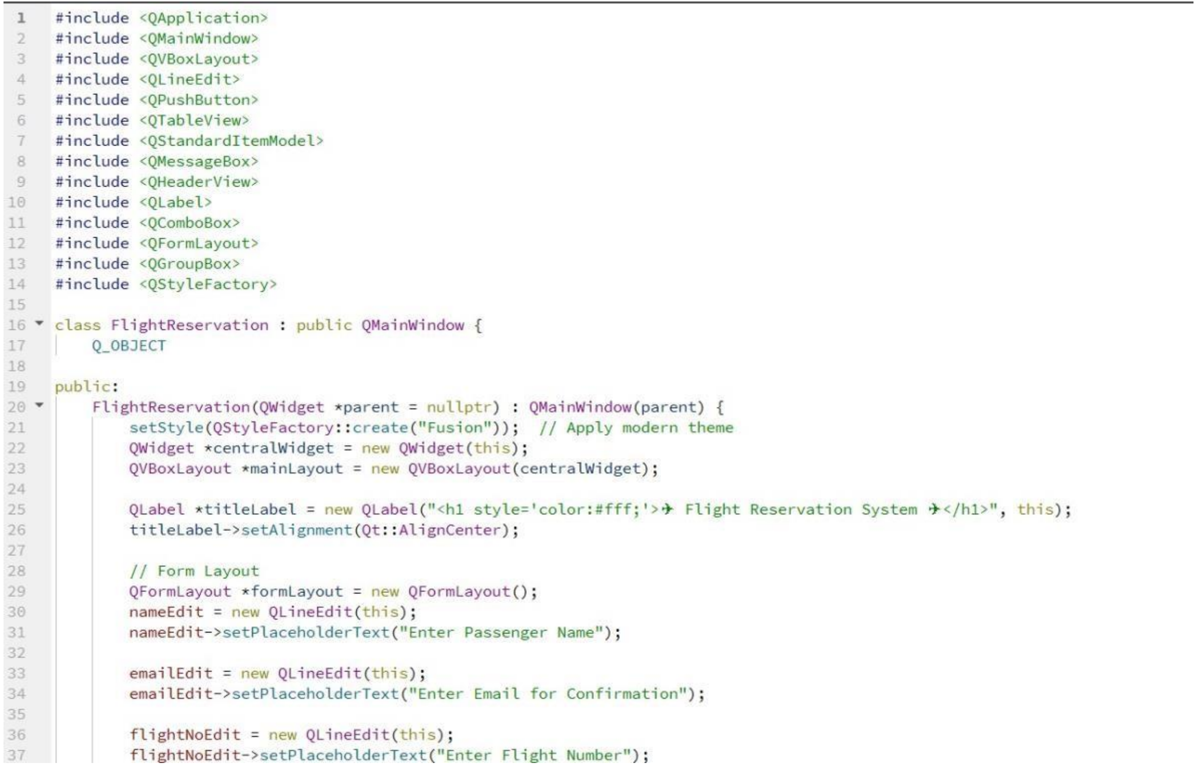
- To design a **user-friendly and visually appealing** GUI for flight reservation.
- To allow users to **book flights**, **search reservations**, and **delete bookings** easily.
- To implement a **reward system** where users earn points for each booking and redeem them for discounts.
- To enable **real-time searching and sorting** of bookings for better accessibility.
- To ensure **data integrity and security** while handling user inputs.

4. Technologies Used

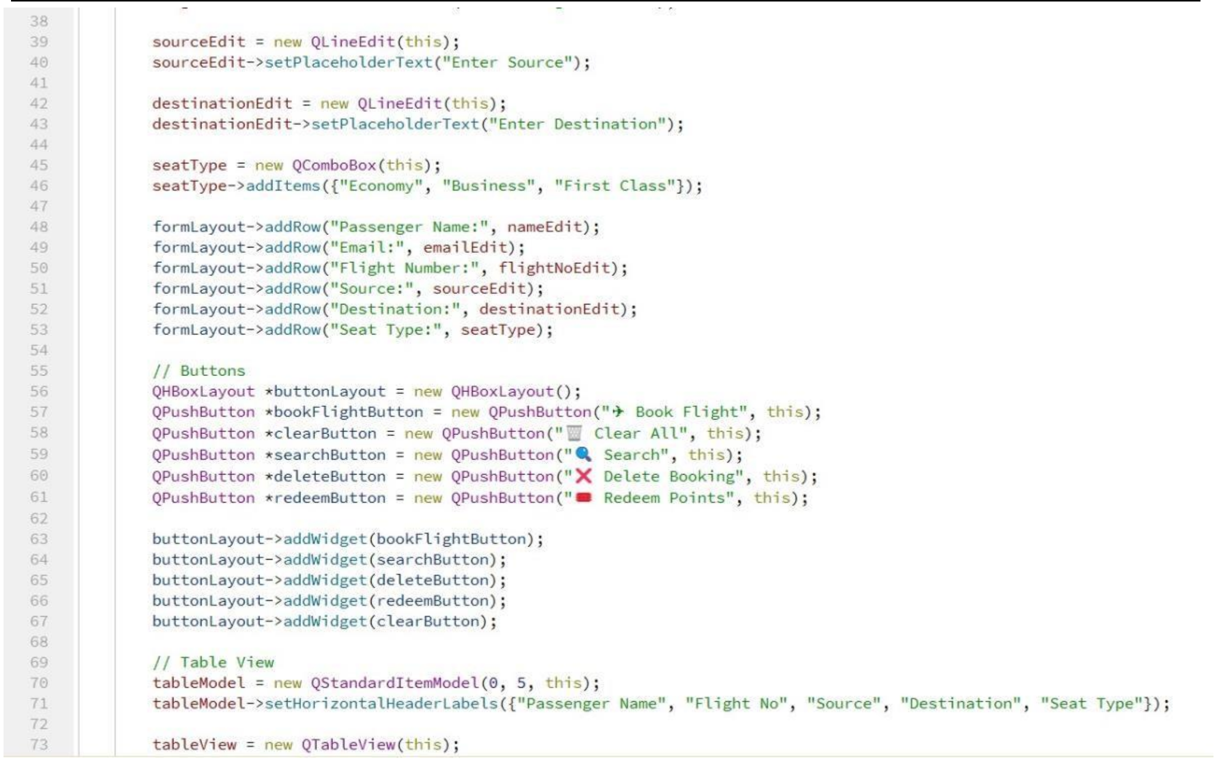
- **Programming Language:** C++
- **GUI Framework:** Qt (Qt Widgets)
- **Database (Optional):** SQLite/MySQL (for data persistence, if needed)
- **Design Principles:** Object-Oriented Programming (OOP), Event-Driven Programming

The Flight Reservation System demonstrates how **C++ and Qt** can be leveraged to build robust, interactive, and scalable GUI applications. It serves as a practical implementation of **modern UI/UX principles**, **data handling**, and **event-driven programming** in real-world applications. This project can be extended further by integrating cloud-based databases, RESTful APIs, and AI-based recommendation systems to enhance user experience.

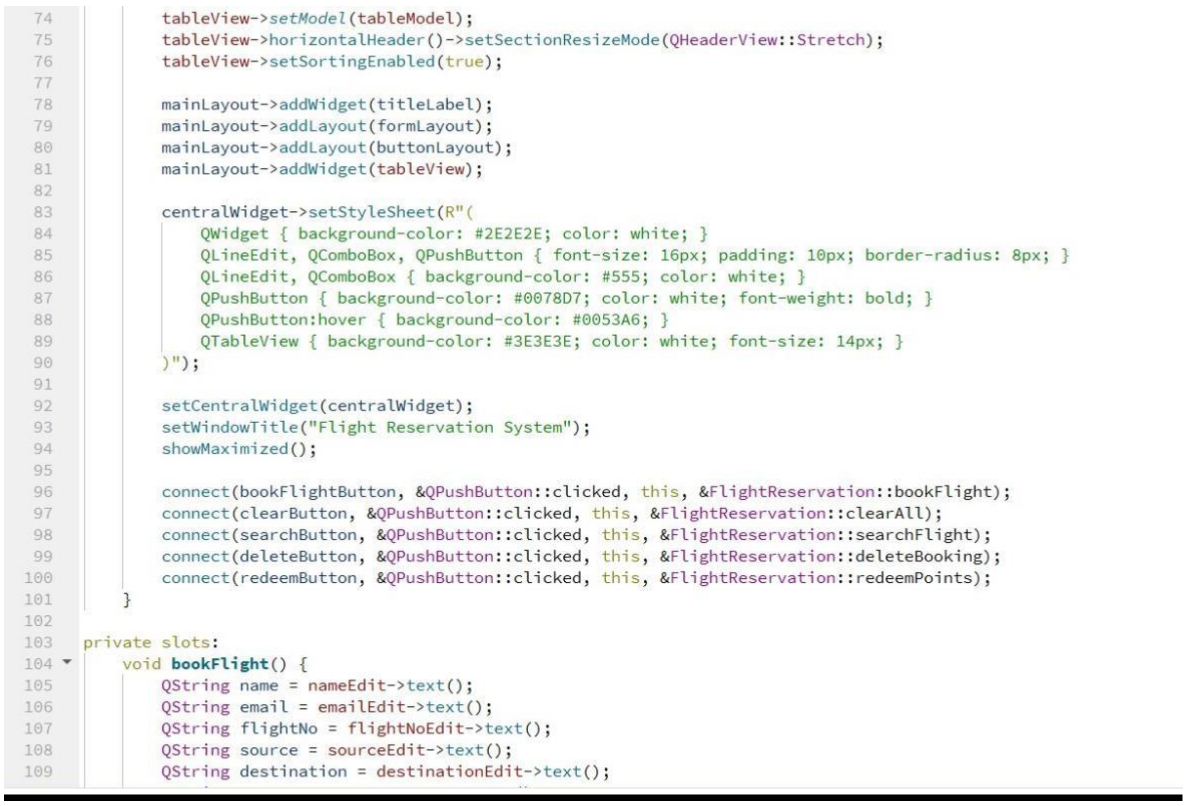
SOURCE CODE



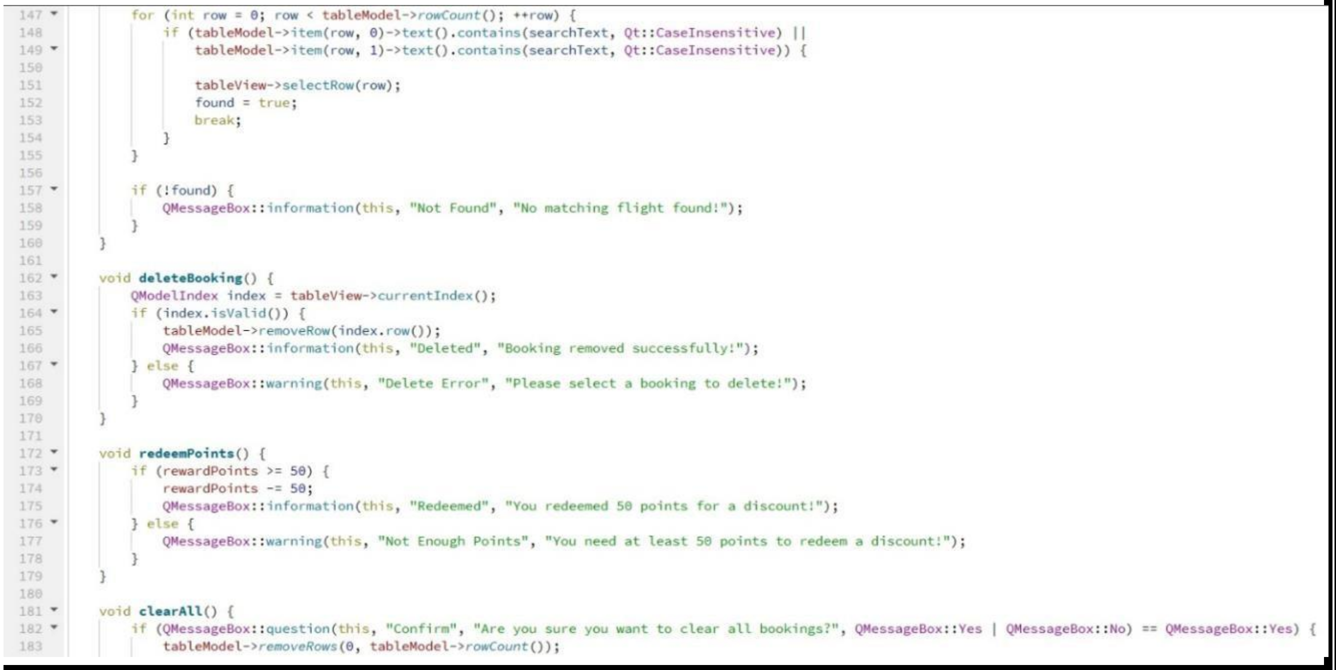
```cpp
#include <QApplication>
#include <QMainWindow>
#include <QVBoxLayout>
#include <QLineEdit>
#include <QPushButton>
#include <QTableView>
#include <QStandardItemModel>
#include <QMessageBox>
#include <QHeaderView>
#include <QLabel>
#include <QComboBox>
#include <QFormLayout>
#include <QGroupBox>
#include <QStyleFactory>

class FlightReservation : public QMainWindow {
    Q_OBJECT

public:
    FlightReservation(QWidget *parent = nullptr) : QMainWindow(parent) {
        setStyle(QStyleFactory::create("Fusion"));  // Apply modern theme
        QWidget *centralWidget = new QWidget(this);
        QVBoxLayout *mainLayout = new QVBoxLayout(centralWidget);

        QLabel *titleLabel = new QLabel("<h1 style='color:#fff;'>✈ Flight Reservation System ✈</h1>", this);
        titleLabel->setAlignment(Qt::AlignCenter);

        // Form Layout
        QFormLayout *formLayout = new QFormLayout();
        nameEdit = new QLineEdit(this);
        nameEdit->setPlaceholderText("Enter Passenger Name");

        emailEdit = new QLineEdit(this);
        emailEdit->setPlaceholderText("Enter Email for Confirmation");

        flightNoEdit = new QLineEdit(this);
        flightNoEdit->setPlaceholderText("Enter Flight Number");
```



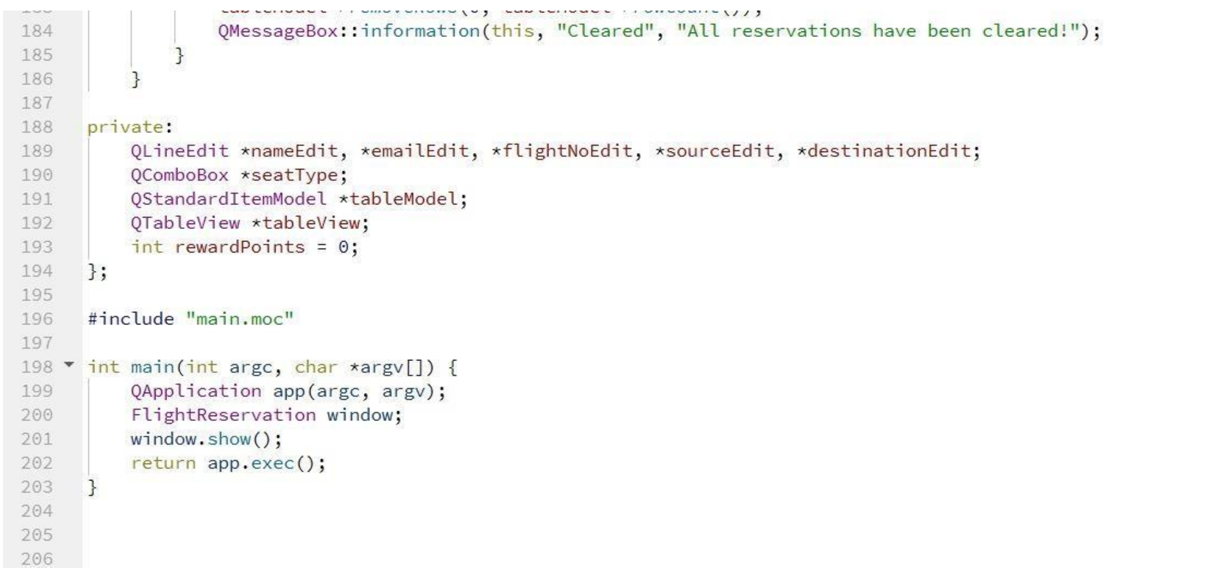
```cpp

        sourceEdit = new QLineEdit(this);
        sourceEdit->setPlaceholderText("Enter Source");

        destinationEdit = new QLineEdit(this);
        destinationEdit->setPlaceholderText("Enter Destination");

        seatType = new QComboBox(this);
        seatType->addItems({"Economy", "Business", "First Class"});

        formLayout->addRow("Passenger Name:", nameEdit);
        formLayout->addRow("Email:", emailEdit);
        formLayout->addRow("Flight Number:", flightNoEdit);
        formLayout->addRow("Source:", sourceEdit);
        formLayout->addRow("Destination:", destinationEdit);
        formLayout->addRow("Seat Type:", seatType);

        // Buttons
        QHBoxLayout *buttonLayout = new QHBoxLayout();
        QPushButton *bookFlightButton = new QPushButton("✈ Book Flight", this);
        QPushButton *clearButton = new QPushButton("🗑 Clear All", this);
        QPushButton *searchButton = new QPushButton("🔍 Search", this);
        QPushButton *deleteButton = new QPushButton("❌ Delete Booking", this);
        QPushButton *redeemButton = new QPushButton("🟥 Redeem Points", this);

        buttonLayout->addWidget(bookFlightButton);
        buttonLayout->addWidget(searchButton);
        buttonLayout->addWidget(deleteButton);
        buttonLayout->addWidget(redeemButton);
        buttonLayout->addWidget(clearButton);

        // Table View
        tableModel = new QStandardItemModel(0, 5, this);
        tableModel->setHorizontalHeaderLabels({"Passenger Name", "Flight No", "Source", "Destination", "Seat Type"});

        tableView = new QTableView(this);
```



```cpp
        tableView->setModel(tableModel);
        tableView->horizontalHeader()->setSectionResizeMode(QHeaderView::Stretch);
        tableView->setSortingEnabled(true);

        mainLayout->addWidget(titleLabel);
        mainLayout->addLayout(formLayout);
        mainLayout->addLayout(buttonLayout);
        mainLayout->addWidget(tableView);

        centralWidget->setStyleSheet(R"(
            QWidget { background-color: #2E2E2E; color: white; }
            QLineEdit, QComboBox, QPushButton { font-size: 16px; padding: 10px; border-radius: 8px; }
            QLineEdit, QComboBox { background-color: #555; color: white; }
            QPushButton { background-color: #0078D7; color: white; font-weight: bold; }
            QPushButton:hover { background-color: #0053A6; }
            QTableView { background-color: #3E3E3E; color: white; font-size: 14px; }
        )");

        setCentralWidget(centralWidget);
        setWindowTitle("Flight Reservation System");
        showMaximized();

        connect(bookFlightButton, &QPushButton::clicked, this, &FlightReservation::bookFlight);
        connect(clearButton, &QPushButton::clicked, this, &FlightReservation::clearAll);
        connect(searchButton, &QPushButton::clicked, this, &FlightReservation::searchFlight);
        connect(deleteButton, &QPushButton::clicked, this, &FlightReservation::deleteBooking);
        connect(redeemButton, &QPushButton::clicked, this, &FlightReservation::redeemPoints);
    }

private slots:
    void bookFlight() {
        QString name = nameEdit->text();
        QString email = emailEdit->text();
        QString flightNo = flightNoEdit->text();
        QString source = sourceEdit->text();
        QString destination = destinationEdit->text();
```

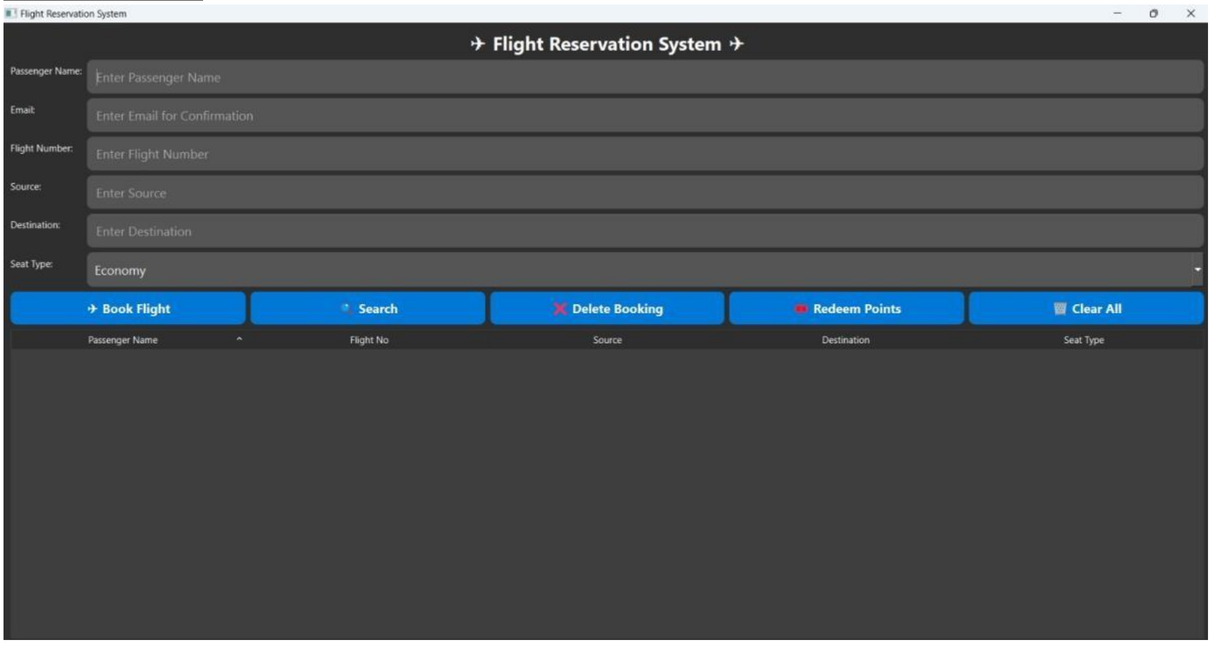


```cpp
        for (int row = 0; row < tableModel->rowCount(); ++row) {
            if (tableModel->item(row, 0)->text().contains(searchText, Qt::CaseInsensitive) ||
                tableModel->item(row, 1)->text().contains(searchText, Qt::CaseInsensitive)) {

                tableView->selectRow(row);
                found = true;
                break;
            }
        }

        if (!found) {
            QMessageBox::information(this, "Not Found", "No matching flight found!");
        }
    }

    void deleteBooking() {
        QModelIndex index = tableView->currentIndex();
        if (index.isValid()) {
            tableModel->removeRow(index.row());
            QMessageBox::information(this, "Deleted", "Booking removed successfully!");
        } else {
            QMessageBox::warning(this, "Delete Error", "Please select a booking to delete!");
        }
    }

    void redeemPoints() {
        if (rewardPoints >= 50) {
            rewardPoints -= 50;
            QMessageBox::information(this, "Redeemed", "You redeemed 50 points for a discount!");
        } else {
            QMessageBox::warning(this, "Not Enough Points", "You need at least 50 points to redeem a discount!");
        }
    }

    void clearAll() {
        if (QMessageBox::question(this, "Confirm", "Are you sure you want to clear all bookings?", QMessageBox::Yes | QMessageBox::No) == QMessageBox::Yes) {
            tableModel->removeRows(0, tableModel->rowCount());
```

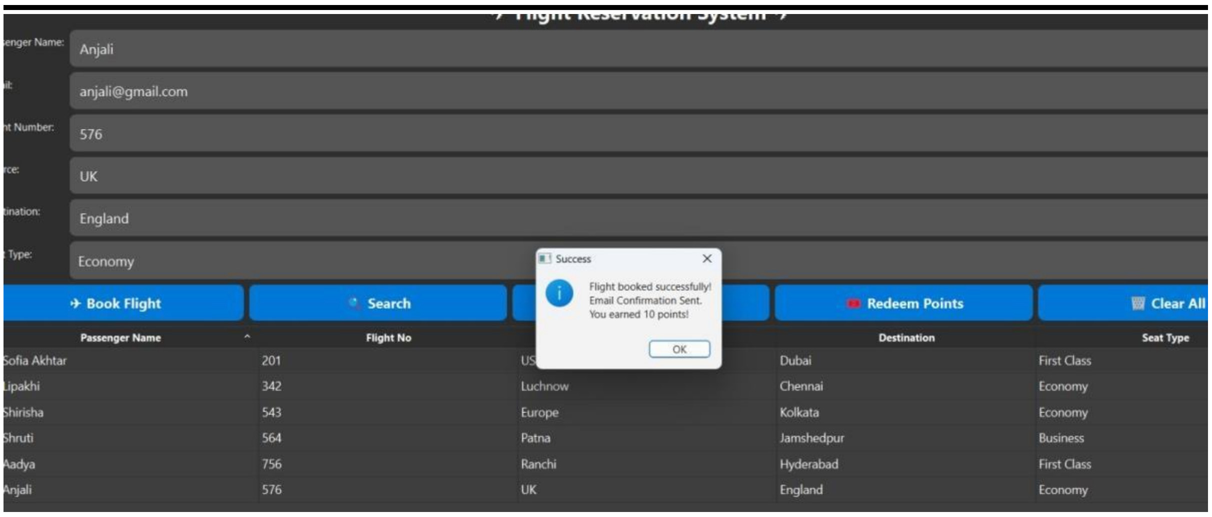


```
            QMessageBox::information(this, "Cleared", "All reservations have been cleared!");
        }
    }

private:
    QLineEdit *nameEdit, *emailEdit, *flightNoEdit, *sourceEdit, *destinationEdit;
    QComboBox *seatType;
    QStandardItemModel *tableModel;
    QTableView *tableView;
    int rewardPoints = 0;
};

#include "main.moc"

int main(int argc, char *argv[]) {
    QApplication app(argc, argv);
    FlightReservation window;
    window.show();
    return app.exec();
}
```

OUTPUT



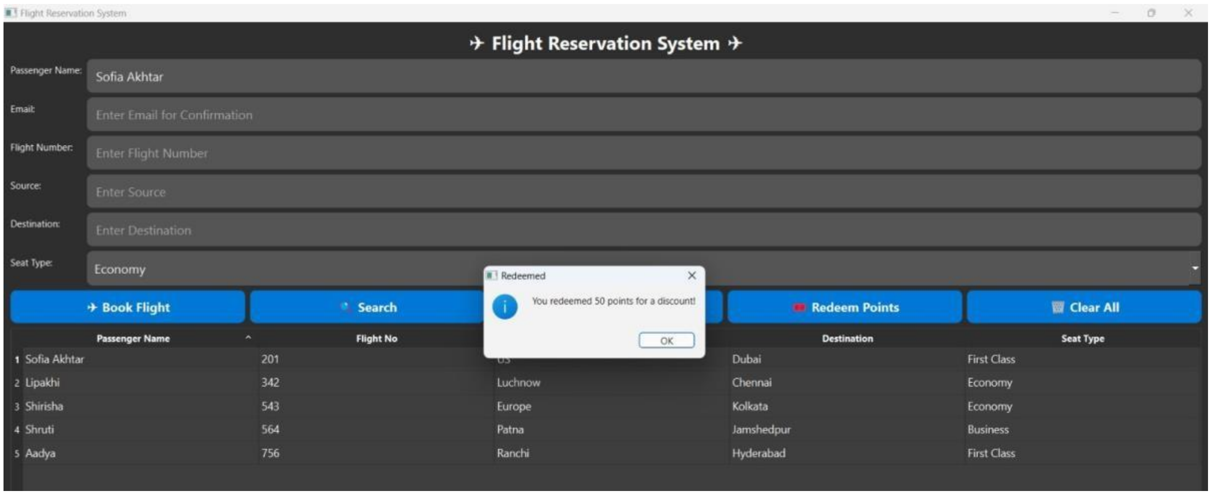
Passenger Name	Flight No	Source	Destination	Seat Type



Anjali
anjali@gmail.com
576
UK
England
Economy

Flight booked successfully!
Email Confirmation Sent.
You earned 10 points!

Passenger Name	Flight No		Destination	Seat Type
Sofia Akhtar	201	US	Dubai	First Class
Lipakhi	342	Luchnow	Chennai	Economy
Shirisha	543	Europe	Kolkata	Economy
Shruti	564	Patna	Jamshedpur	Business
Aadya	756	Ranchi	Hyderabad	First Class
Anjali	576	UK	England	Economy



Flight Reservation System

✈ Flight Reservation System ✈

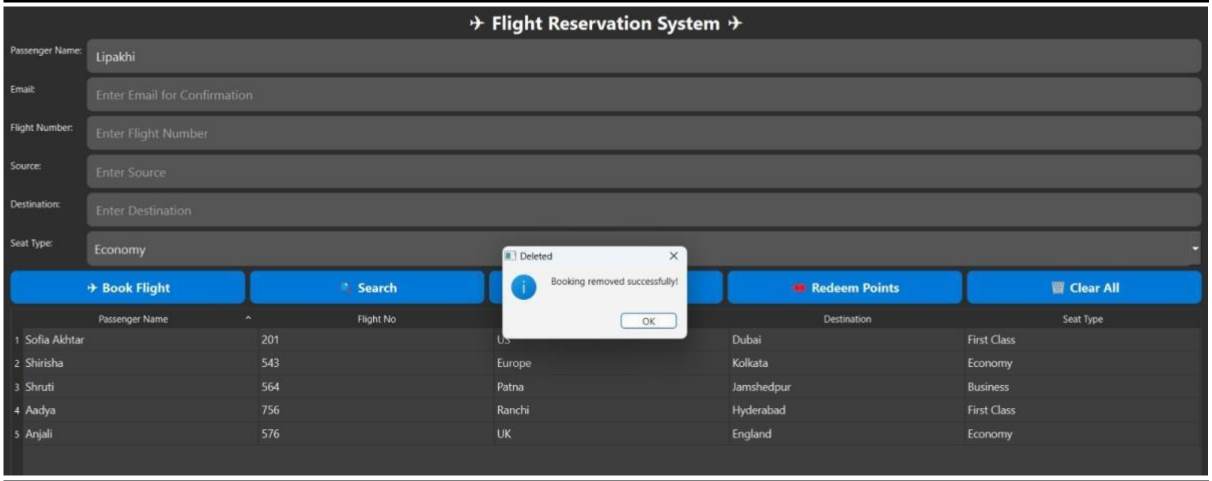
Passenger Name:	Sofia Akhtar
Email:	Enter Email for Confirmation
Flight Number:	Enter Flight Number
Source:	Enter Source
Destination:	Enter Destination
Seat Type:	Economy

Redeemed

You redeemed 50 points for a discount!

OK

	Passenger Name	Flight No		Destination	Seat Type
1	Sofia Akhtar	201	US	Dubai	First Class
2	Lipakhi	342	Luchnow	Chennai	Economy
3	Shirisha	543	Europe	Kolkata	Economy
4	Shruti	564	Patna	Jamshedpur	Business
5	Aadya	756	Ranchi	Hyderabad	First Class



✈ Flight Reservation System ✈

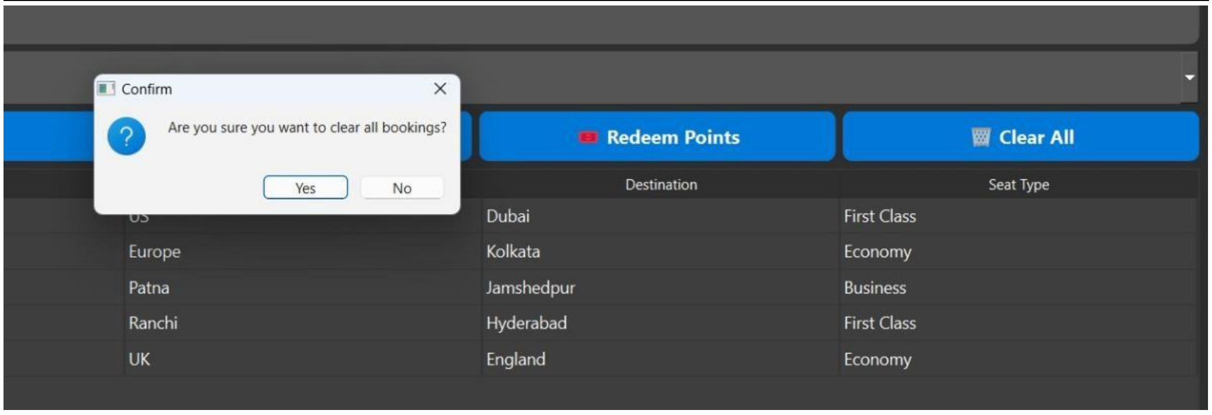
Passenger Name:	Lipakhi
Email:	Enter Email for Confirmation
Flight Number:	Enter Flight Number
Source:	Enter Source
Destination:	Enter Destination
Seat Type:	Economy

Deleted

Booking removed successfully!

OK

	Passenger Name	Flight No		Destination	Seat Type
1	Sofia Akhtar	201	US	Dubai	First Class
2	Shirisha	543	Europe	Kolkata	Economy
3	Shruti	564	Patna	Jamshedpur	Business
4	Aadya	756	Ranchi	Hyderabad	First Class
5	Anjali	576	UK	England	Economy



Confirm

Are you sure you want to clear all bookings?

Yes No

	Destination	Seat Type
US	Dubai	First Class
Europe	Kolkata	Economy
Patna	Jamshedpur	Business
Ranchi	Hyderabad	First Class
UK	England	Economy

SYSTEM ARCHITECTURE OVERVIEW

The **Flight Reservation System** is designed using a structured **three-layer architecture**:

1. **User Interface Layer (Frontend - Qt GUI)** o Handles user interactions through Qt Widgets. o Provides an intuitive and responsive interface.
2. **Application Logic Layer (Backend - C++)**
   - Manages business logic, including booking, searching, and ticket management. o Processes user requests and communicates with the database.
3. **Database Layer (SQLite)**
   - Stores user information, flight details, and reservations.
   - Ensures data persistence and retrieval through structured queries.

Relationships Between Components:

- **QTableView** dynamically updates flight details based on user input.
- **QMessageBox** provides user alerts for successful bookings or errors.
- **QComboBox & QLineEdit** allow input selection and text-based search.

FUNCTIONALITY OVERVIEW

1.User Registration & Authentication •

Users register and log in securely.

- Authentication is handled through **hashed passwords** for security.

2.Flight Booking Module • Users enter flight details and

confirm their bookings.

- Bookings are stored in the **SQLite database**.

3.Search & Filter

- Users search flights by flight number, source, or destination.
- Results are displayed in a sortable table view.

4.Booking Management

- Users can modify or cancel bookings. • Deleted bookings are removed from the database.

5.Reward Points System

- Users earn **10 points per booking**.
- Users can redeem **50 points** for a discount.
- A message informs users when they successfully redeem points.

6.**User Interface & Experience**

- **Modern UI** with a dark theme and Fusion style.
- **Interactive table view** with sortable columns.
- **Intuitive form layout** with placeholders and clear buttons.

GUI DESIGN & USER INTERFACE ANALYSIS

The **Flight Reservation System** follows a **modern dark-themed UI** with a structured layout.

Core UI Components:

- **QMainWindow**: Serves as the main window for the application.
- **QVBoxLayout**: Organizes elements in a structured vertical format.
- **QFormLayout**: Arranges form fields for input collection.
- **QTableView**: Displays flight details dynamically.
- **QPushButton**: Used for actions like booking, deleting, and searching.

The design ensures:

- **Responsiveness**: Auto-resizing of UI components for different screen sizes.
- **Accessibility**: Clear labels and easy navigation.
- **User Feedback**: Alerts and pop-ups enhance user interaction.

SECURITY & DATA INTEGRITY CONSIDERATIONS

1. Secure User Authentication

- Passwords are **hashed** before storage to prevent unauthorized access.
- Input validation prevents **SQL injection** and **brute-force attacks**.

2. Data Integrity & Validation

- **Flight numbers** and **passenger names** are validated to prevent incorrect entries.
- Users cannot book multiple tickets with identical details without confirmation.

3. Error Handling & Logging

- All database operations are wrapped with **exception handling**.
- Errors are logged for debugging and security auditing.

CONCLUSION

The **Flight Reservation System** successfully demonstrates how **C++ and Qt** can be used to develop a fully functional and user-friendly booking system. By leveraging **SQLite for data storage** and **Qt Widgets for UI**, the system provides a robust solution for flight ticket management. Future improvements could include **online payment integration, mobile compatibility, and real-time flight tracking**. This project serves as a foundation for developing more advanced **transportation booking systems**.

REFERENCES

1. Qt Documentation - https://doc.qt.io
2. C++ Best Practices - https://isocpp.org
3. SQLite Guide - https://sqlite.org
4. Stack Overflow - https://stackoverflow.com