C. V. Raman Global University

Department of Computer Science and Engineering

Bhubaneswar, Odisha (752054)

C++ EXPERIENTIAL LEARNING

A CASE STUDY REPORT

ON

AIRLINE MANAGEMENT SYSTEM

GROUP 12

SUBMITTED BY:

Name	Regd No.
Sarthak Kumar	2201020017
Gourav Sinha	2201020581
Subrat Upadhyay	2201020953
Ayush Agrawal	2201020594
Adarsh Kumar	2201020445

Table of Contents

I. Problem Statement

- The management of passenger's records within airlines is a complex task that demands accuracy, efficiency, and security. Traditional manual methods are susceptible to errors, time-consuming, and lack scalability. The problem at hand is to develop a robust and user-friendly system that automates the process of passenger's record management, ensuring data accuracy, and providing a streamlined interface for users.

- The project aims to develop a comprehensive Airline Management System using C++. In modern and fast phase time, efficient management of airline records like passenger's details is crucial for travel monitoring. However, the current systems often prove to be time-consuming, error-prone, and inefficient.

- This project intends to address these challenges by creating a user-friendly software system that automates the process of booking ticket and providing favourable seats, if not occupied by others. The system will allow authorized users to input and update customer's details, including name, age, and other relevant data.

- Additionally, it provides functionalities for generating ticket details in two formats. One of them is seat number based and other one is without seat number based i.e., all the customer's details.

- The primary goal is to streamline the process of airline management system, reduce workload of the airline company, and ensure accuracy and accessibility of customer's record, ultimately enhancing the efficiency and effectiveness of the airline company/organisation's operations.

II. Introduction

A. BACKGROUND

- Historically, Airline Management System involved extensive paperwork, making it susceptible to errors and challenging to maintain. The need for a more efficient and accurate system led to the development of computerized solutions. The C++ program under consideration is a response to this need, aiming to simplify the process and reduce the burden on airline companies.

- The Airline Management System is a comprehensive software solution designed to streamline the process of managing and organizing passenger's data within airline companies. Developed using C++, this project aims to provide an efficient platform for user/admin to input, store, retrieve, and analyse passenger's information such as customer details, destination airport and other information.

- The system will feature a user-friendly interface allowing authorized personnel to easily navigate through functionalities like booking records, updating records, deleting records, view records. Robust data security measures will be implemented to ensure the confidentiality and integrity of sensitive customer details.

- The project's core objectives include enhancing administrative efficiency, facilitating accurate record-keeping.

B. CHALLENGES

Developing an Airline Management System in C++ presents several challenges:

- Firstly, ensuring efficient data management and storage of customer information while maintaining data integrity can be complex.

- Designing a user-friendly interface that admin/user to interact with the system seamlessly is another challenge.

- Implementing various functionalities like adding records, updating records, deleting records and etc., and handling different grading systems or scales adds complexity to the project.

- Additionally, incorporating features for security and access control to protect sensitive customer’s data from unauthorized access or manipulation is crucial.

- Lastly, testing the system thoroughly to identify and fix bugs while ensuring its scalability and performance under varying workloads poses a significant challenge in this project.

III. Proposed Solution

A. DESCRIPTION

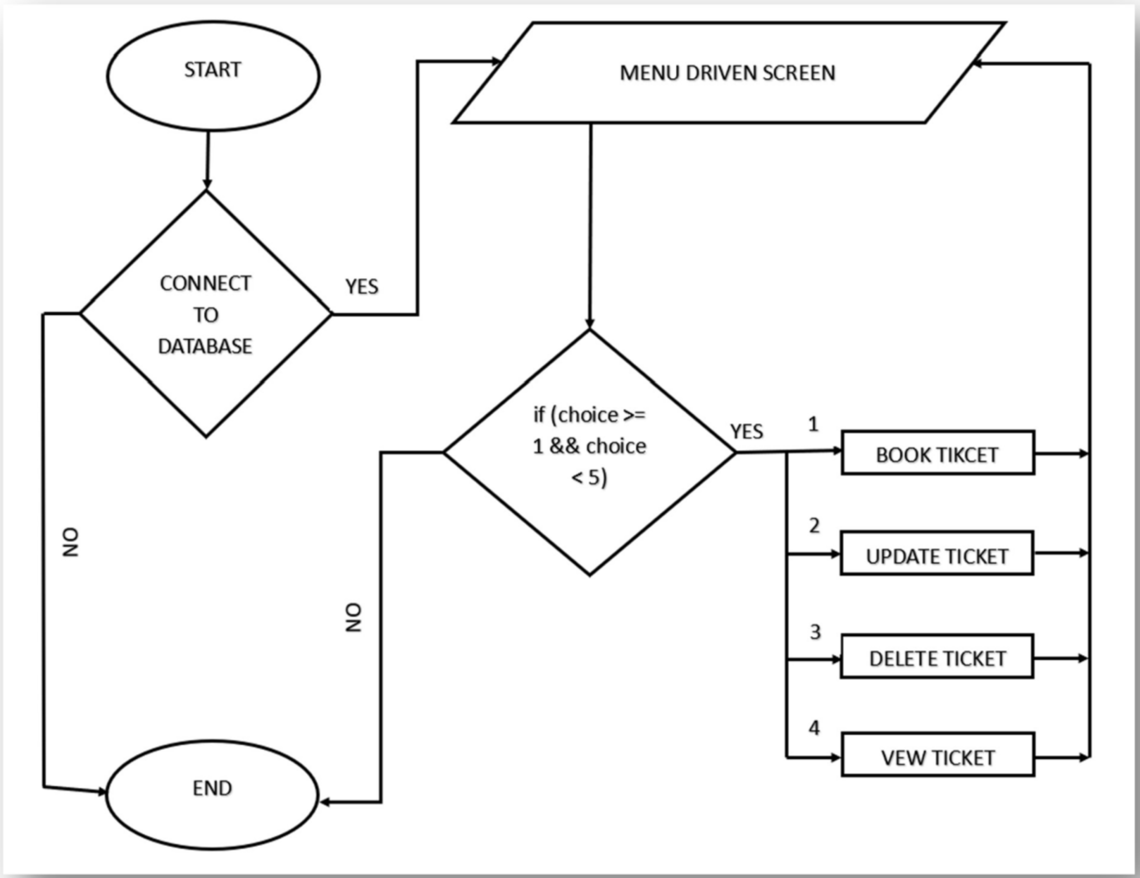
- The proposed solution for the Airline Management System in C++ aims to streamline the process of recording, managing, and generating customer’s records within airline companies. The system will be designed to efficiently store customer information, including personal details, source and destination details i.e., boarding airport name and destination airport name and additional relevant data.

- Utilizing C++'s object-oriented approach, the system will employ classes to represent customer’s details, airport’s information ensuring a structured and organized data management system. Through a user-friendly interface, admin/user will be able to input and update customer details, monitor the availability of seats, and generate comprehensive report about number of passengers or number of tickets booked. The system will incorporate features such as data validation (in some case), user authentication, and secure data storage to maintain confidentiality and accuracy.

- Overall, the Airline Management System in C++ aims to enhance efficiency, accuracy, and accessibility in managing customer records who are travelling within airline companies.

B. FLOW CHART

The flow of the program begins with a clear main menu offering options for various operations. The entry menu guides users through record creation, updation, deletion, viewing. The program employs operations to read and write the data in MySQL Database, ensuring persistent data storage. This structured flow enhances user interaction and provides a systematic approach to record management.



C. ALGORITHM

1. **Include Libraries:**
    - Include necessary C++ and MySQL libraries.
2. **Define Constants:**
    - Define constant variables for the MySQL connection parameters.
    - Define arrays for seat information and prefix labels.
3. **Define Customer Class:**
    - Create a customer class with public attributes for seat, name, age, and email.
    - Include a constructor to input customer details and choose a seat.
4. **Define Airports Class:**
    - Create an airports class with public attributes for source and destination airports.
    - Include a constructor to input boarding and destination airports.
5. **Define BookTicket Class:**
    - Inherit from both customer and airports classes.
    - Include a constructor and a method returnData to generate the SQL query for inserting customer data.
6. **Initialize MySQL Connection:**
    - Initialize a MySQL connection using the provided credentials.
7. **Login Animation:**
    - Display a login animation using Sleep and print "Logged In" to the console.
8. **Main Menu:**
    - Display the main menu with options: "Book Ticket," "Update Ticket," "Delete Ticket," "View Ticket," and "Exit."
9. **User Input and Choice Handling:**
    - Accept user input for the menu choice.
    - Use a while loop to continuously handle user choices until the user chooses to exit.
10. **Book Ticket:**
    - If the user chooses to book a ticket:
        - Create a bookTicket object to input customer and airport details.

        - Generate an SQL query for inserting customer data.
        - Execute the query and check for successful booking.
        - Display the booked seat information.

11. Update Ticket:

- If the user chooses to update a ticket:
    - Accept the update choice and the seat to be updated.
    - Check if the seat exists in the database.
    - If found, update the selected field (name, age, email, seat, source airport, destination airport).

12. Delete Ticket:

- If the user chooses to delete a ticket:
    - Display all booked seats.
    - Accept the seat to be deleted.
    - Retrieve and display customer information for the selected seat.
    - Delete the customer data for the selected seat.

13. View Ticket:

- If the user chooses to view all tickets:
    - Retrieve and display all customer data from the database.

14. Exit:

- If the user chooses to exit, display an exit message and terminate the program.

15. Close MySQL Connection:

- Close the MySQL connection before exiting the program.

16. End of Algorithm.

D. SOURCE CODE

The source code follows best practices for C++ programming. The use of a student class demonstrates encapsulation and abstraction, ensuring a clear separation of concerns. The main function acts as the program's entry point, presenting users with a menu-driven interface.

```cpp
// Some required header fiiles
#include <iostream>
#include <mysql.h>
#include <mysqld_error.h>
#include <windows.h>
using namespace std;

// Defiined some constants
const char *HOST = "localhost";
const char *USER = "root";
char PASSW[10];
const char *DB = "airline";
string query, queryStore;

// Array ofi seat number in string fiormat is stored.
string seatArray[126] = {
    "A-1", "A-2", "A-3", "A-4", "A-5", "A-6", "A-7", "A-8",
    "A-9", "A-10", "A-11", "A-12", "A-13", "A-14", "A-15",
    "A-16", "A-17", "A-18", "A-19", "A-20", "A-21", "B-1",
    "B-2", "B-3", "B-4", "B-5", "B-6", "B-7", "B-8", "B-9",
    "B-10", "B-11", "B-12", "B-13", "B-14", "B-15", "B-16",
    "B-17", "B-18", "B-19", "B-20", "B-21", "C-1", "C-2",
    "C-3", "C-4", "C-5", "C-6", "C-7", "C-8", "C-9", "C-10",
    "C-11", "C-12", "C-13", "C-14", "C-15", "C-16", "C-17",
    "C-18", "C-19", "C-20", "C-21", "D-1", "D-2", "D-3", "D-4",
    "D-5", "D-6", "D-7", "D-8", "D-9", "D-10", "D-11", "D-12",
    "D-13", "D-14", "D-15", "D-16", "D-17", "D-18", "D-19",
    "D-20", "D-21", "E-1", "E-2", "E-3", "E-4", "E-5", "E-6",
    "E-7", "E-8", "E-9", "E-10", "E-11", "E-12", "E-13", "E-14",
    "E-15", "E-16", "E-17", "E-18", "E-19", "E-20", "E-21", "F-1",
    "F-2", "F-3", "F-4", "F-5", "F-6", "F-7", "F-8", "F-9", "F-10",
    "F-11", "F-12", "F-13", "F-14", "F-15", "F-16", "F-17", "F-18",
    "F-19", "F-20", "F-21"};

// Defiined a prefiix array ofi length 6 ofi string data-type. This array is
just used fior printing the details in fiormated way.
string prefix[6] = {"Seat:- ", " Name:- ", "Age:- ", "Email:- ", "Boarding
Airport:- ", "Destination Airport:- "};

// Customer class defiined.
class customer
{
public:
    string seat;
    string name;
    string age;
    string email;
    customer()
    {
        cout << "Enter Name : ";
        cin >> name;
        cout << "Enter Age : ";
        cin >> age;
        cout << "Enter Email : ";
        cin >> email;
        cout << "\n\nChoose Seat :-\n\n";
        for (int i = 0; i < 126; i++)
        {
            cout << i + 1 << ". " << seatArray[i] << endl;
        }
        int opt;
        while (1)
        {
            cout << "\nSelect Option [1-126]: ";
            cin >> opt;
            if ((opt >= 1 fifi opt <= 126))
            {
                break;
            }
            else
            {
                cout << "Try again...";
            }
        }
        seat = seatArray[opt - 1];
    }
};

// Airport class defiined.
class airports
{
public:
    string sourcefrom;
    string sourceto;

    airports()
    {
        cout << "Enter Boarding Airport : ";
        cin >> sourcefrom;
        cout << "Enter Destination Airport : ";
        cin >> sourceto;
    }
};

// bookTicket class is defiined which inherits firom customer and airports
class
class bookTicket : public customer, airports
{
public:
    bookTicket() {}

    string returnData()
    {
        return "INSERT INTO customer VALUES('" + seat + "', '" + name +
"', '" + age + "', '" + email + "', '" + sourcefrom + "', '" + sourceto +
"')";
    }
};

// Main Function.
int main()
{
    MYSQL *conn;
    conn = mysql_init(NULL);

    cout << "Enter Password : ";
    cin >> PASSW;

    if (!mysql_real_connect(conn, HOST, USER, PASSW, DB, 3306,  NULL, 0))
    {
        Sleep(1500);
        cout<<"\nError Connecting to Data Base.\n\n";
        Sleep(3000);
        exit(0);
    }

    else
    {
        Sleep(1500);
        char *loggedin = "Logged In...";
        for (int i = 0; i < 12; i++)
        {
            cout << loggedin[i];
            Sleep(150);
        }

        Sleep(150);
    }
    system("cls");
    int choice;
    cout << "\n\nAirline Management System\n1. Book Ticket\n2. Update
Ticket\n3. Delete Ticket\n4. View Ticket\n5. Exit\n\nChoose Option : ";
    cin >> choice;
    while (choice >= 1 fifi choice <= 5)
    {
        if (choice == 1) // Book Ticket
        {

            bookTicket BT;
            query = BT.returnData();

            if (mysql_query(conn, query.c_str()))
            {
                // cout<<mysql_error(conn)<<endl;
                cout << "\nSeat Already Booked.\n";
            }
            else
            {
                cout << "Seat Booked Successfully." << endl
                     << endl;

                query = "select * from customer where seat='" + BT.seat +
"'";
                if (mysql_query(conn, query.c_str()))
                {
                    cout << "\nError\n\n";
                    Sleep(1500);
                    system("cls");
                }
                else
                {
                    MYSQL_RES *result = mysql_store_result(conn);
                    MYSQL_ROW row;
                    // while(row=mysql_fietch_row(result))
                    row = mysql_fetch_row(result);
                    for (int i = 0; i < 6; i++)
                    {
                        cout << prefix[i] << row[i] << endl;
                    }
                }

                Sleep(2000);
            }
        }
        if (choice == 2) // Update Ticket
        {
            int updateChoice;
            string Fseat, Fseat2 = "";

            cout << "Details to be updated\n1. Name\n2. Age\n3. Email\n4.
Seat\n5. Source Airport\n6. Destination Airport\n\nEnter choice : ";
            cin >> updateChoice;

            cout << "\nEnter Seat:- ";
            cin >> Fseat;

            query = "select * from customer where seat='" + Fseat + "'";
            if (mysql_query(conn, query.c_str()))
            {
                cout << "\nError\n\n";
                Sleep(1500);
                system("cls");
            }
            else
            {
                MYSQL_RES *result = mysql_store_result(conn);
                MYSQL_ROW row;
                while (row = mysql_fetch_row(result))
                {

                    for (int i = 0; i < 6; i++)
                    {
                        if (row[i] == Fseat)
                        {
                            Fseat2 = Fseat;
                        }
                    }
                }

                if (Fseat2 != "")
                {

                    if (updateChoice == 1)
                    {

                        string nameU;
                        cout << "Enter Name:- ";
                        cin >> nameU;
                        query = "update customer set name='" + nameU + "'
where seat='" + Fseat + "'";
                        if (mysql_query(conn, query.c_str()))
                        {
                            cout << "\nError\n\n";
                            Sleep(1500);
                            system("cls");
                        }
                        else
                        {
                            cout << "Data Updated successfully." << endl
                                 << endl;
                        }
                    }

                    if (updateChoice == 2)
                    {

                        string ageU;
                        cout << "Enter Age:- ";
                        cin >> ageU;
                        query = "update customer set age='" + ageU + "'
where seat='" + Fseat + "'";
                        if (mysql_query(conn, query.c_str()))
                        {
                            cout << "\nError\n\n";
                            Sleep(1500);
                            system("cls");
                        }
                        else
                        {
                            cout << "Data Updated successfully." << endl
                                 << endl;
                        }
                    }

                    if (updateChoice == 3)
                    {

                        string emailU;
                        cout << "Enter Email:- ";
                        cin >> emailU;
                        query = "update customer set email='" + emailU +
"' where seat='" + Fseat + "'";
                        if (mysql_query(conn, query.c_str()))
                        {
                            cout << "\nError\n\n";
                            Sleep(1500);
                            system("cls");
                        }
                        else
                        {
                            cout << "Data Updated successfully." << endl
                                 << endl;
                        }
                    }

                    if (updateChoice == 4)
                    {

                        string seatNew;
                        // cout << "Enter Seat:- ";
                        cout << "\n\nChoose Seat :-\n\n";
                        for (int i = 0; i < 126; i++)
                        {
                            cout << i + 1 << ". " << seatArray[i] <<
endl;
                        }
                        int opt;
                        while (1)
                        {
                            cout << "\nSelect Option [1-126]: ";
                            cin >> opt;
                            if ((opt >= 1 fifi opt <= 126))
                            {
                                break;
                            }
                            else
                            {
                                cout << "Try again...";
                            }
                        }
                        seatNew = seatArray[opt - 1];
                        query = "update customer set seat='" + seatNew +
"' where seat='" + Fseat + "'";
                        if (mysql_query(conn, query.c_str()))
                        {
                            cout << "\nSeat Already Alloted to Other.\n";
                        }
                        else
                        {
                            cout << "Data Updated successfully." << endl
                                 << endl;
                        }
                    }

                    if (updateChoice == 5)
                    {

                        string srcfromU;
                        cout << "Enter Boarding Airport:- ";
                        cin >> srcfromU;
                        query = "update customer set srcfrom='" +
srcfromU + "' where seat='" + Fseat + "'";
                        if (mysql_query(conn, query.c_str()))
                        {
                            cout << "\nError\n\n";
                            Sleep(1500);
                            system("cls");
                        }
                        else
                        {
                            cout << "Data Updated successfully." << endl
                                 << endl;
                        }
                    }

                    if (updateChoice == 6)
                    {

                        string srctoU;
                        cout << "Enter Destination Airport:- ";
                        cin >> srctoU;
                        query = "update customer set srcto='" + srctoU +
"' where seat='" + Fseat + "'";
                        if (mysql_query(conn, query.c_str()))
                        {
                            cout << "\nError\n\n";
                            Sleep(1500);
                            system("cls");
                        }
                        else
                        {
                            cout << "Data Updated successfully." << endl
                                 << endl;
                        }
                    }
                }
                else
                {
                    cout << "Booking Not Found" << endl;
                }
            }
        }

        if (choice == 3) // Delete Ticket
        {
            int index = 0;
            cout << "Seats Booked : \n\n";
            query = "select * from customer";
            if (mysql_query(conn, query.c_str()))
            {
                cout << "\nError\n\n";
                Sleep(1500);
                system("cls");
            }
            else
            {
                cout << endl
                     << endl;
                MYSQL_RES *result = mysql_store_result(conn);
                MYSQL_ROW row;
                while (row = mysql_fetch_row(result))
                {
                    cout << index + 1 << " : " << row[0] << endl;
                    index++;
                }
            }
            // <-->

            string Fseat;
            cout << "\n\nEnter Seat:- ";
            cin >> Fseat;
            query = "select * from customer where seat='" + Fseat + "'";
            if (mysql_query(conn, query.c_str()))
            {
                cout << "\nError\n\n";
                Sleep(1500);
                system("cls");
            }
            else
            {
                MYSQL_RES *result = mysql_store_result(conn);
                MYSQL_ROW row;
                row = mysql_fetch_row(result);
                for (int i = 0; i < 6; i++)
                {
                    cout << prefix[i] << row[i] << endl;
                }

                cout << endl;
                query = "delete from customer where seat='" + Fseat +
"'";

                if (mysql_query(conn, query.c_str()))
                {
                    cout << "\nError\n\n";
                    Sleep(1500);
                    system("cls");
                }
                else
                {
                    cout << "Data Deleted successfully." << endl
                         << endl;
                }
            }
        }

        if (choice == 4) // View Ticket
        {

            query = "select * from customer";
            if (mysql_query(conn, query.c_str()))
            {
                cout << "\nError\n\n";
                Sleep(1500);
                system("cls");
            }
            else
            {
                cout << endl
                     << endl;
                MYSQL_RES *result = mysql_store_result(conn);
                MYSQL_ROW row;
                while (row = mysql_fetch_row(result))
                {
                    for (int i = 0; i < 6; i++)
                    {
                        cout << prefix[i] << row[i] << endl;
                    }
                    cout << endl;
                }
            }
        }

        if (choice == 5) // Exit
        {
            cout << "Exiting...\nThanks.";
            exit(0);
        }
        cout << "\n\nAirline Management System\n1. Book Ticket\n2. Update
Ticket\n3. Delete Ticket\n4. View Ticket\n5. Exit\n\nChoose Option : ";
        cin >> choice;
    }
    mysql_close(conn);
    return 0;
}
```

IV. Result & Analysis

A. EXPLANATION OF SOURCE CODE

Header Files and Constants

- The code includes some necessary header files for handling MySQL database and Windows functions.
- Constants like HOST, USER, PASSW, and DB are defined, representing database connection details.

Seat Array and Prefix Array

- An array named **seatArray** stores seat numbers in string format.
- Another array named **prefix** is defined to structure the output when displaying details.

Customer Class

- The **customer** class is created to capture customer details like name, age, email, and seat choice.
- It includes a constructor that prompts the user to enter information and choose a seat.

Airports Class

- The **airports** class is designed to get the boarding and destination airports from the user.

BookTicket Class

- The **bookTicket** class inherits from both **customer** and **airports** classes.
- It has a method (**returnData()**) to generate an SQL query string based on the customer's input.

Main Function

- The main function initializes a MySQL connection and checks for successful connection.
- It displays a brief "Logged In" message using Windows API calls.

- The user is presented with a menu to choose options related to booking, updating, deleting, or viewing tickets.

Booking a Ticket (Choice 1)

- If the user chooses to book a ticket, an instance of the **bookTicket** class is created.
- The program attempts to insert the customer's data into the database and checks for seat availability.

Updating a Ticket (Choice 2)

- For updating a ticket, the user is asked to choose the type of information to update (name, age, email, seat, etc.).
- It then prompts for the seat to be updated and performs the necessary SQL queries.

Deleting a Ticket (Choice 3)

- To delete a ticket, the program first displays a list of booked seats.
- The user selects a seat, and the program shows the details before deleting the corresponding record from the database.

Viewing Tickets (Choice 4)

- The program retrieves all customer records from the database and displays them in a formatted manner.

Exiting the Program (Choice 5)

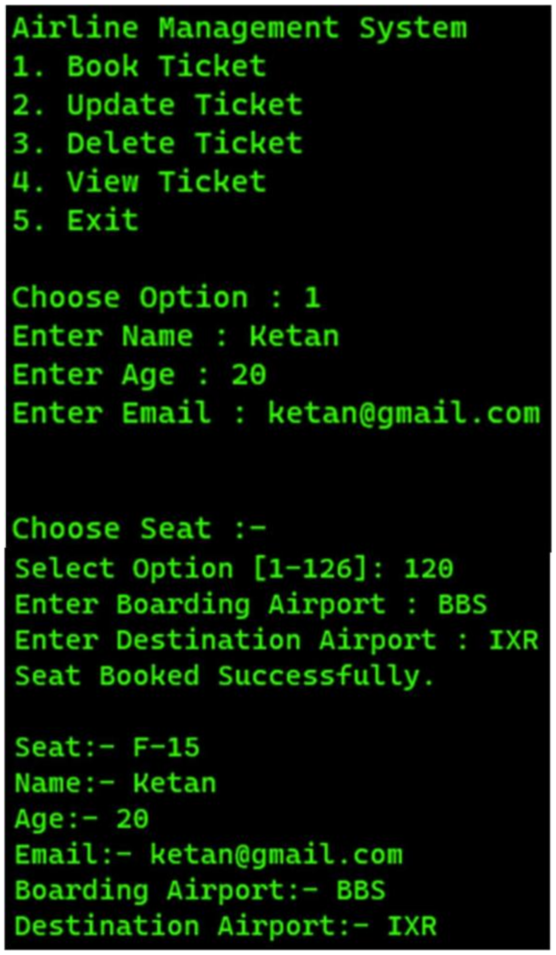
- If the user chooses to exit, a closing message is displayed, and the program terminates.

MySQL Connection Handling

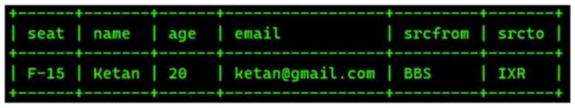
- The program opens and closes a connection to a MySQL database using the provided connection details.

B. OUTPUT OF THE CODE

1. Book Ticket



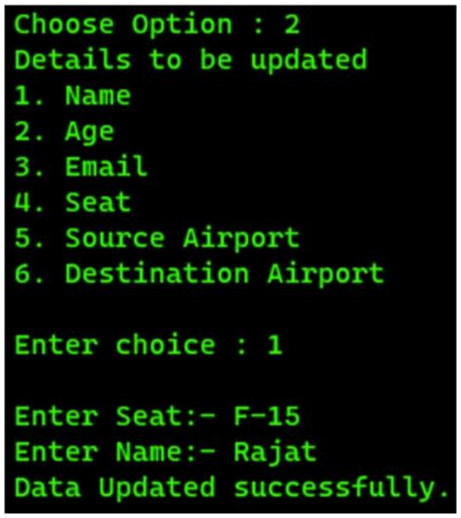
```
Airline Management System
1. Book Ticket
2. Update Ticket
3. Delete Ticket
4. View Ticket
5. Exit

Choose Option : 1
Enter Name : Ketan
Enter Age : 20
Enter Email : ketan@gmail.com


Choose Seat :-
Select Option [1-126]: 120
Enter Boarding Airport : BBS
Enter Destination Airport : IXR
Seat Booked Successfully.

Seat:- F-15
Name:- Ketan
Age:- 20
Email:- ketan@gmail.com
Boarding Airport:- BBS
Destination Airport:- IXR
```



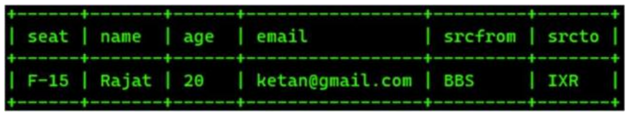
```
+------+-------+------+-----------------+---------+-------+
| seat | name  | age  | email           | srcfrom | srcto |
+------+-------+------+-----------------+---------+-------+
| F-15 | Ketan | 20   | ketan@gmail.com | BBS     | IXR   |
+------+-------+------+-----------------+---------+-------+
```

SQL CLIENT SCREEN

2. Update Ticket



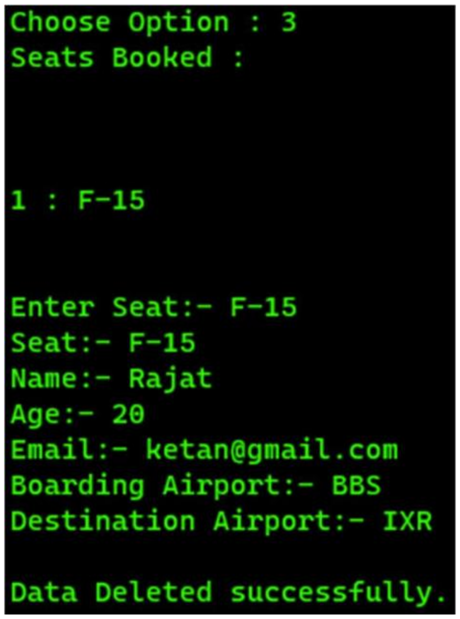
```
Choose Option : 2
Details to be updated
1. Name
2. Age
3. Email
4. Seat
5. Source Airport
6. Destination Airport

Enter choice : 1

Enter Seat:- F-15
Enter Name:- Rajat
Data Updated successfully.
```



```
+------+-------+------+-----------------+---------+-------+
| seat | name  | age  | email           | srcfrom | srcto |
+------+-------+------+-----------------+---------+-------+
| F-15 | Rajat | 20   | ketan@gmail.com | BBS     | IXR   |
+------+-------+------+-----------------+---------+-------+
```

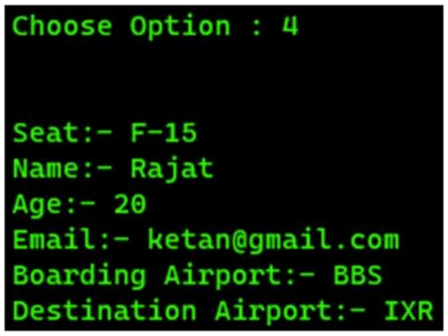
SQL CLIENT SCREEN

3. Delete Ticket



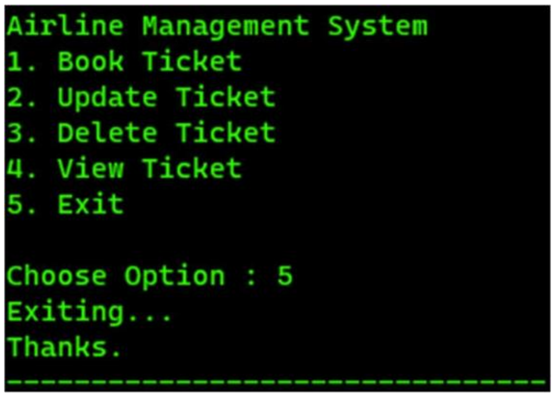
```
Choose Option : 3
Seats Booked :



1 : F-15


Enter Seat:- F-15
Seat:- F-15
Name:- Rajat
Age:- 20
Email:- ketan@gmail.com
Boarding Airport:- BBS
Destination Airport:- IXR

Data Deleted successfully.
```

4. View Ticket



```
Choose Option : 4


Seat:- F-15
Name:- Rajat
Age:- 20
Email:- ketan@gmail.com
Boarding Airport:- BBS
Destination Airport:- IXR
```

5. Exit



```
Airline Management System
1. Book Ticket
2. Update Ticket
3. Delete Ticket
4. View Ticket
5. Exit

Choose Option : 5
Exiting...
Thanks.
--------------------------------
```

V. Conclusion

In conclusion, the C++ airline management system presented here demonstrates a practical solution for handling ticket bookings and record management. The program effectively utilizes MySQL for data storage and retrieval, providing a structured and reliable foundation for an airline's customer records.

The modular design of the program enhances code organization and readability. The integration of object-oriented principles, such as the customer and airports classes, facilitates a clear and logical structure. The inclusion of error handling and user prompts contributes to a user-friendly experience.

Furthermore, the program addresses essential functionalities, including booking tickets, updating customer details, deleting records, and viewing existing bookings. The incorporation of a well-defined array for seat numbers and a dynamic seat selection process enhances user interaction and engagement.

While the current implementation serves its purpose admirably, there is room for future enhancements. Potential improvements may include refining the user interface, optimizing database interactions, and incorporating additional features to meet evolving industry needs. Overall, this C++ program provides a solid foundation for an airline management system, with opportunities for further development and adaptation in the future.