<<FLIGHT MANAGEMENT SYSTEM>>

21CSC101T – OBJECT ORIENTED DESIGN AND PROGRAMMING

Mini Project Report

Submitted by

M. JASWANTH (RA2211003011414)
B.Tech CSE
V. AMAN ROY (RA2211003011429)
B.Tech CSE

DEPARTMENT OF NETWORKING AND COMMUNICATION
SCHOOL OF COMPUTING
COLLEGE OF ENGINEERING AND TECHNOLOGY SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
(Under Section 3 of UGC Act, 1956)
S.R.M. NAGAR, KATTANKULATHUR – 603 203
KANCHEEPURAM DISTRICT

MAY 2023

SRM INSTITUTION OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR-603203

BONAFIDE CERTIFICATE

Certified that this Course Project Report titled **"FLIGHT MANAGEMENT SYSTEM"** is the bonafide work done by **M. JASWANTH (RA2211003011414) and V. AMAN ROY (RA2211003011429)** who carried out under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other work.

SIGNATURE
Faculty In-Charge
Mr. S. INIYAN
Assistant Professor
Department of Computing Technologies
SRM Institute of Science and Technology

HEAD OF THE DEPARTMENT
Dr.M. Pushpalatha
Professor and Head ,
Department of Computing Technologies
SRM Institute of Science and Technology

TABLE OF CONTENTS

ABSTRACT

The Flight Reservation System is a C++-based solution that allows users to quickly book flights and manage booking information, updates, and cancellations easily. It consolidates data from different airline carriers and thus provides all the necessary details and rates in real-time. In addition, administrators of flight data can also quickly view, create, and update any information about flights, bookings, routes, and schedules

INTRODUCTION

A sophisticated computerised system known as a flight management system aids aeroplane crews in managing and navigating a flight from takeoff to landing. To give the crew real-time information and automated direction, the FMS integrates data from several aircraft systems, including navigation, performance, and fuel.

OBJECTIVE

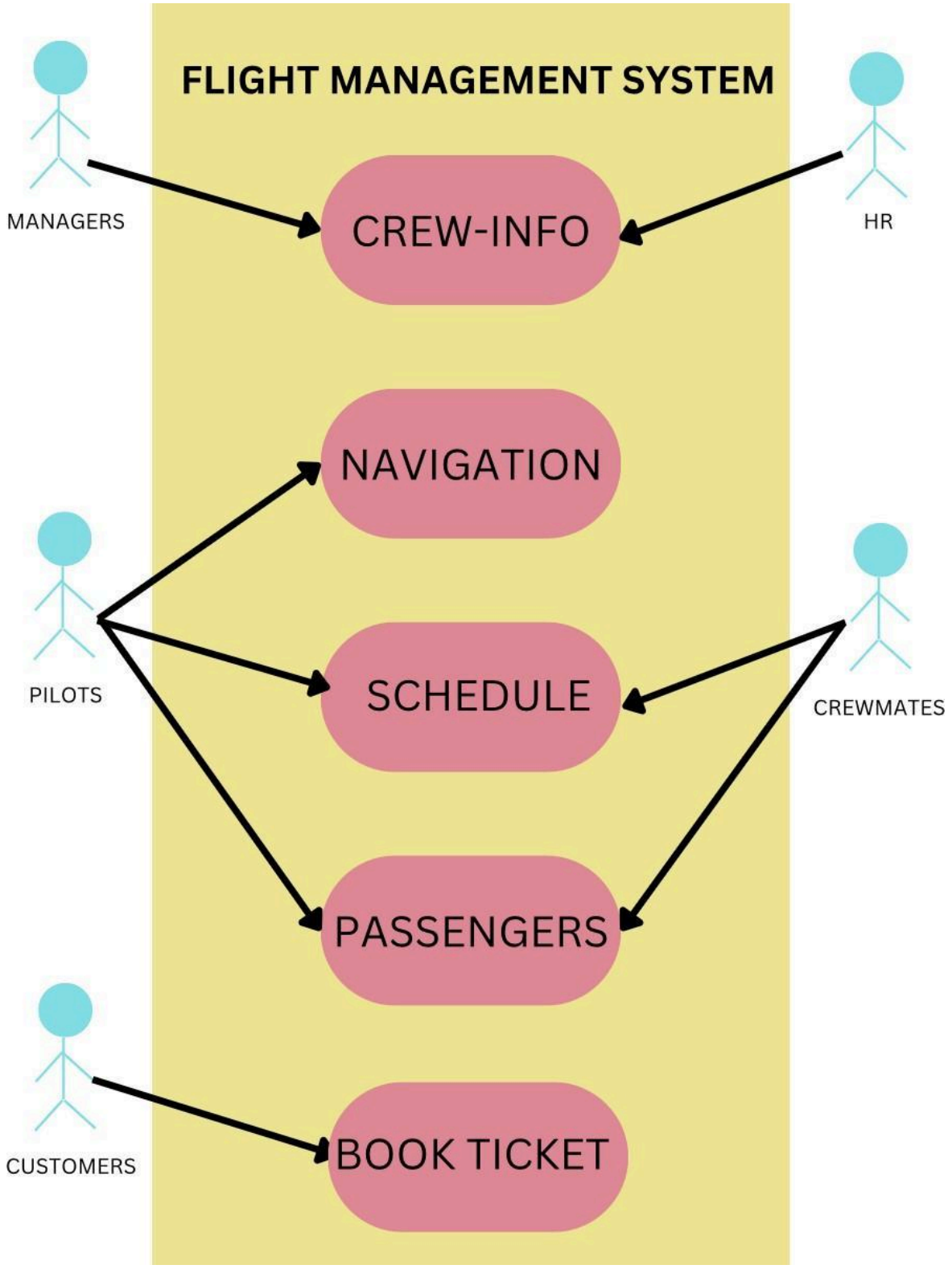
A flight management system (FMS) is designed to give the flight crew automate tools for managing the aircraft from takeoff through landing in an effective and safe manner. To give the crew real-time information and automatic direction, the FMS combines data from different aircraft systems, including navigation, performance, and fuel. The FMS assists the crew in planning and carrying out the flight path while taking into consideration variables including weather, air traffic control constraints, and aircraft performance. It informs the crew of the aircraft's position, altitude, speed, and fuel consumption and directs them as to when and where to turn and change altitude.

Overall, the objective of a flight management system is to enhance flight safety, efficiency, and accuracy while reducing crew workload.

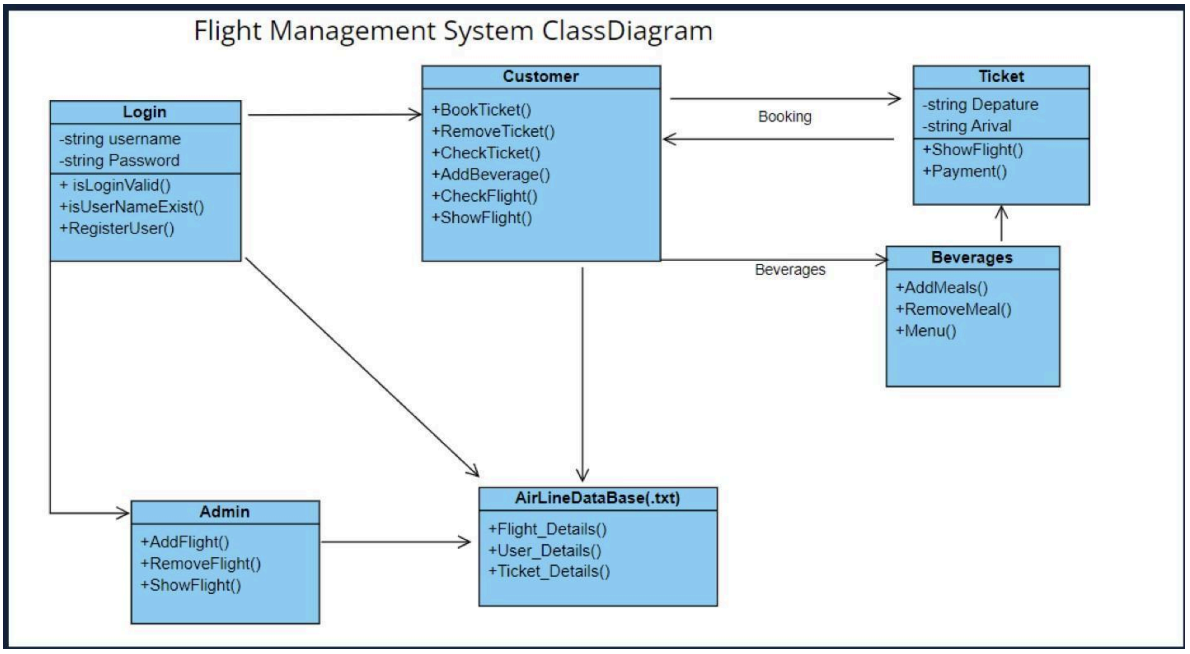
MODULES

1. **CREW-INFO** : THIS MODULE ALLOWS THE MANAGERS AND HR PEOPLE TO MANAGE ,EDIT, DELETE,ADD THE INFORMATION OF THE VARIOUS PEOPLE SUCH AS PILOTS,AIRCRAFT ENGINEERS ,CREW MEMEBERS ALSO THEIR SALARIES AND HOLIDAYS.
2. **NAVIGATION** : THIS MODULE ALLOW PILOTS,AIRCRAFT ENGINEER AND AIRCONTROL STAFF TO NAVIGATE THEIR PLANE ROUTE ,RUNWAY NUMBER , DIESEL INDICATION
   AFTER AND BEFORE LANDING , WEATHER FORCAST THIS CAN ONLY BE MODIFIED BY THE AIRCONTROL STAFF.
3. **SCHEDULE** : THIS MODULE ALLOWS PILOT TO VIEW THEIR TIMELY SCHEDULE , THEIR DAILY FLYING
   HOURS , CREWMATES.
4. **CUSTOMERS :** THIS MODULE ALLOWS CUSTOMERS TO USE BOOK TICKET , CHECK STATUS , CANCEL
   BOOKED TICKET , ADD BEVERAGES .

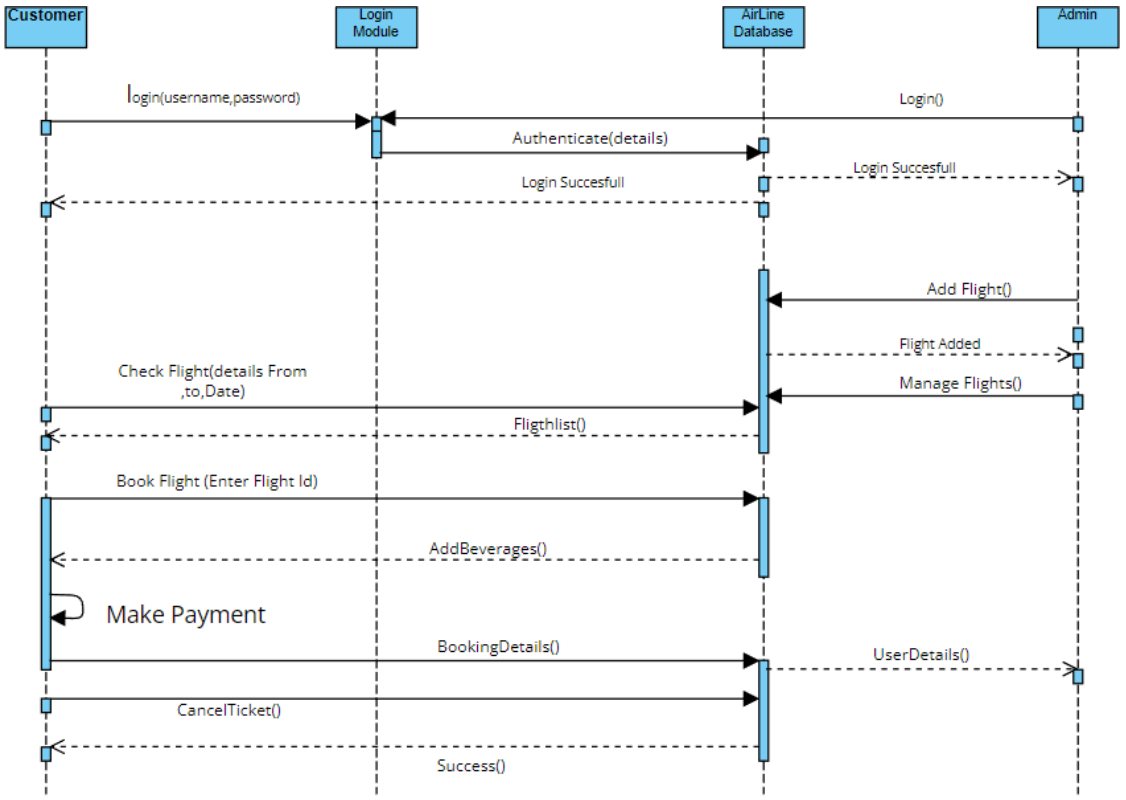
USE CASE DIAGRAM



CLASS DIAGRAM

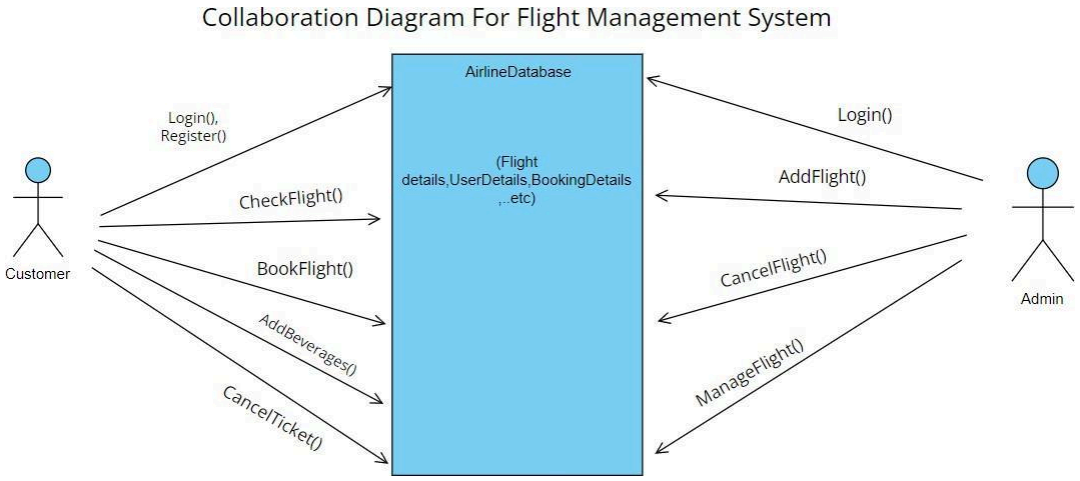


SEQUENCE DIAGRAM

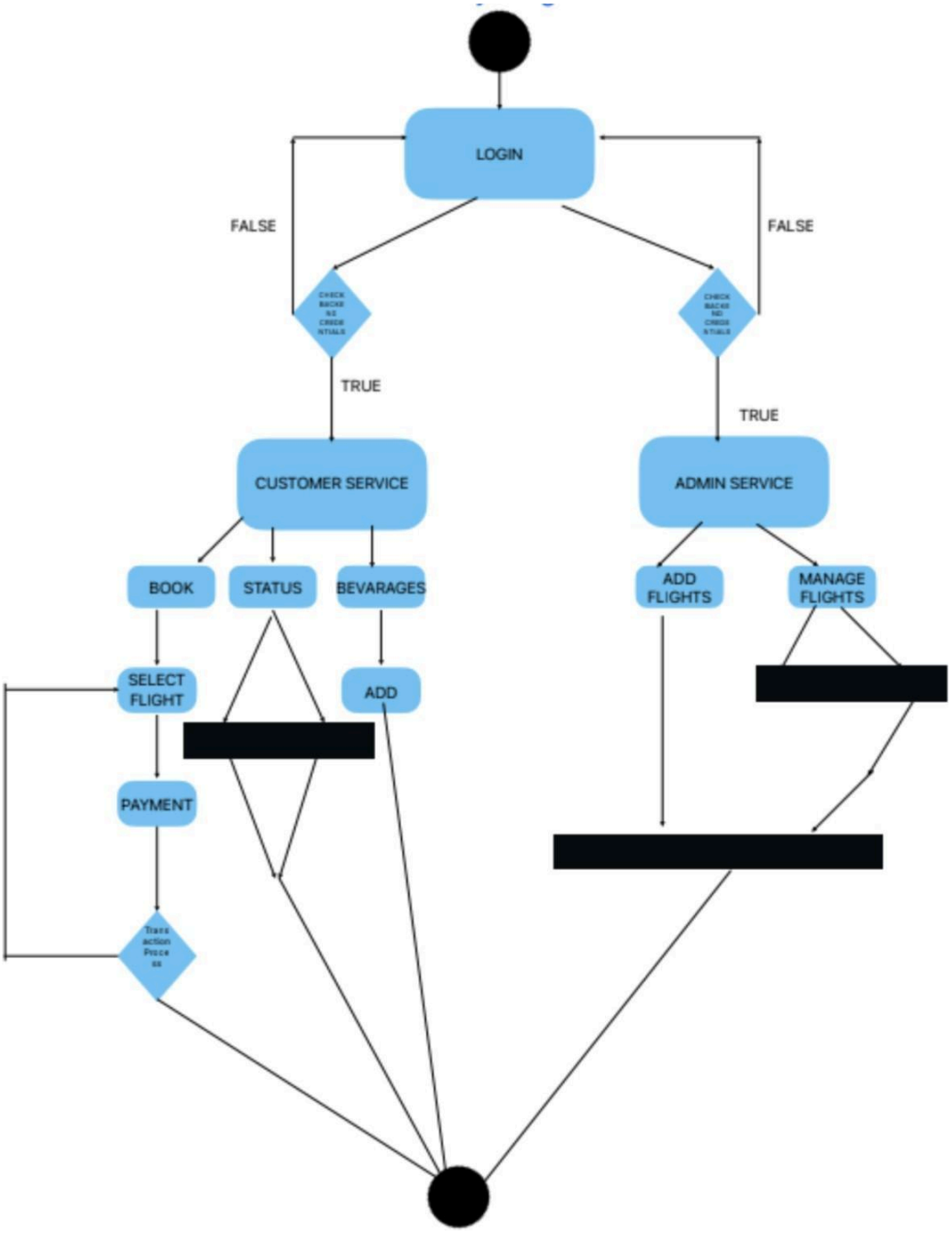


Sequence Diagram Flight Management System

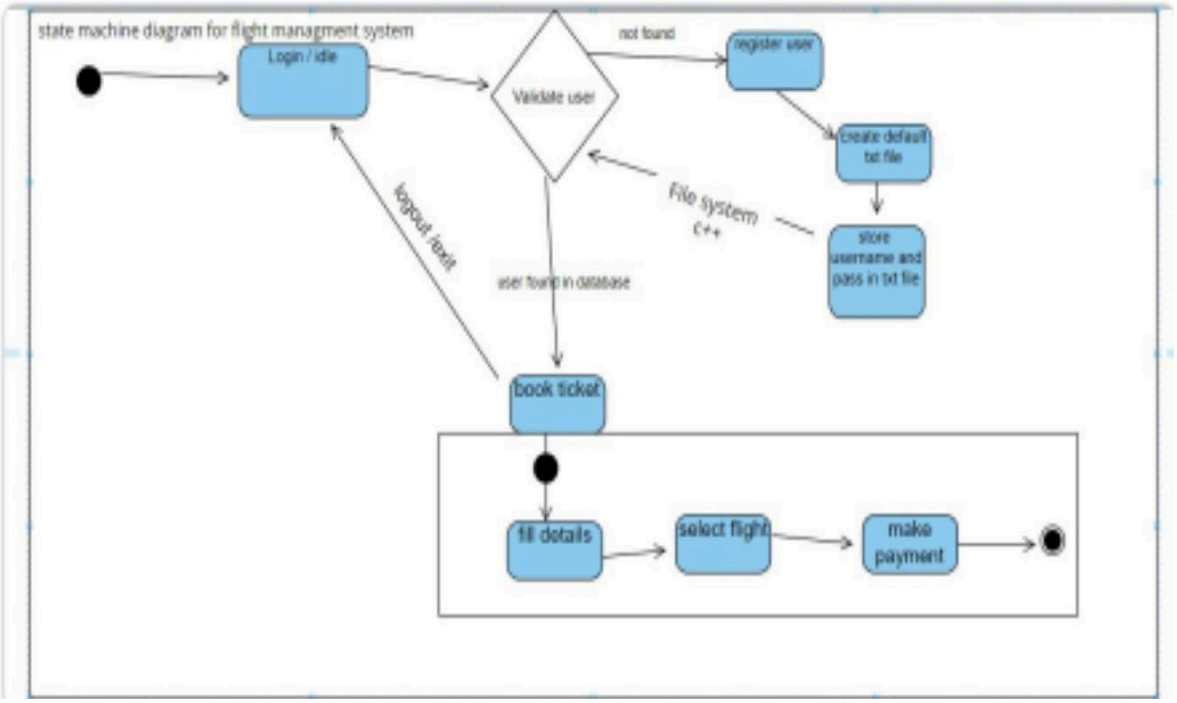
COLLABORATION DIAGRAM



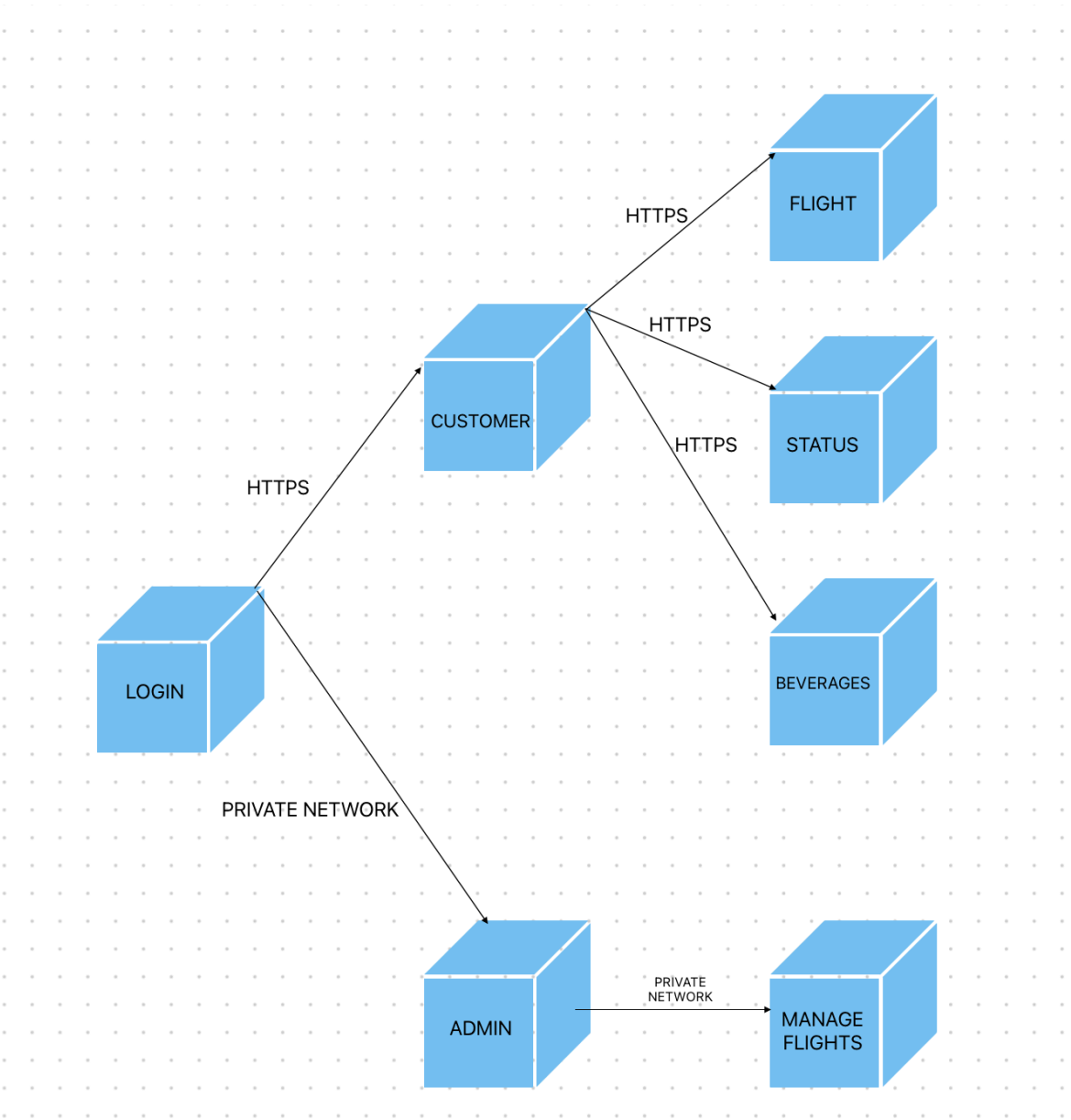
ACTIVITY DIAGRAM



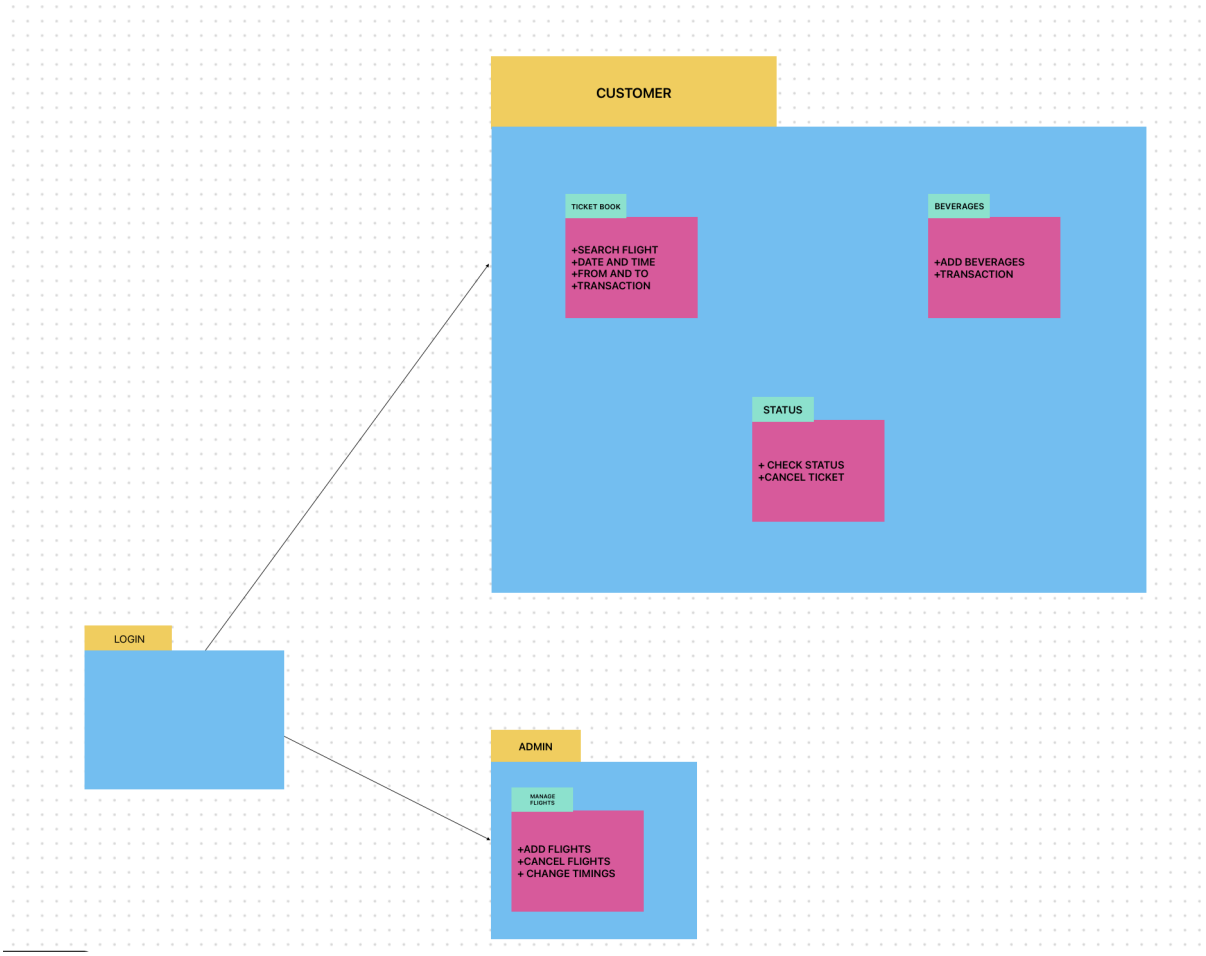
STATE-CHART DIAGRAM



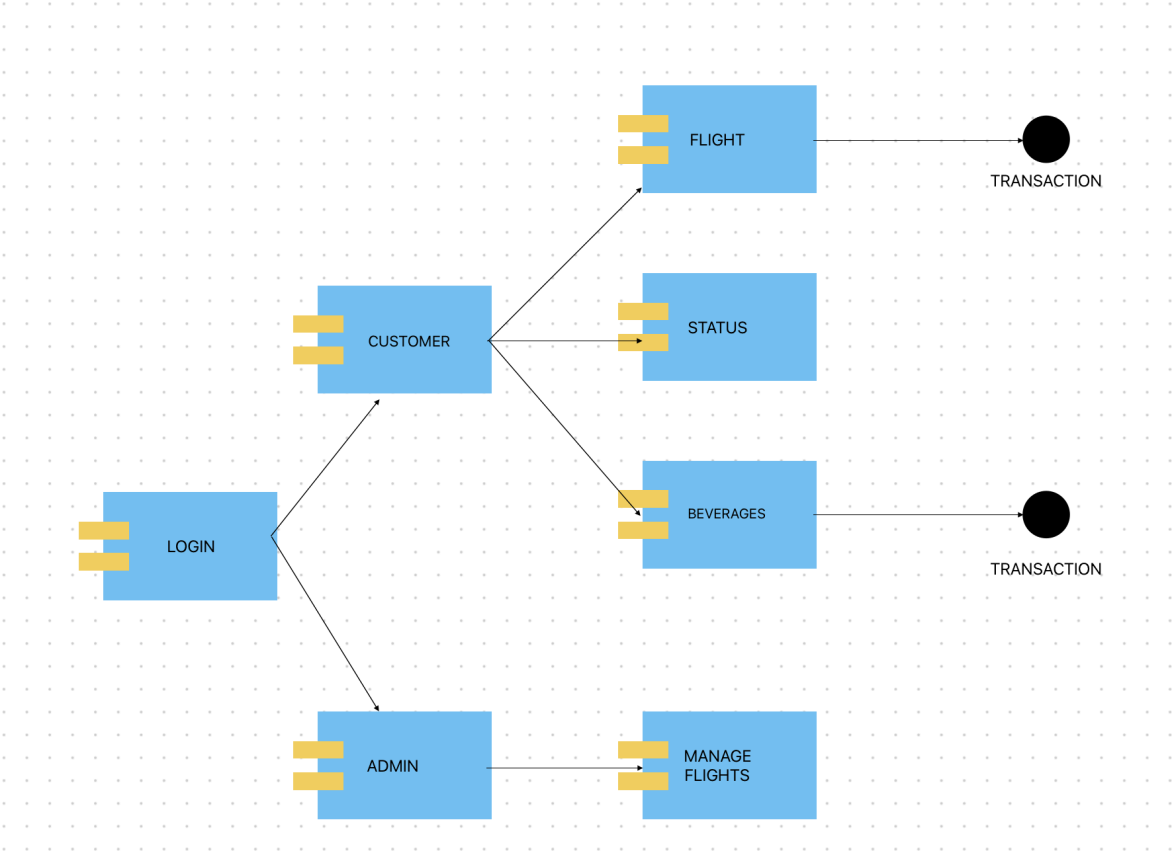
DEPLOYMENT DIAGRAM



PACKAGE DIAGRAM



COMPONENT DIAGRAM



IMPLEMENTATION

```
#include <iostream>

#include <fstream>

#include <string>

#include <chrono>

#include <thread>

using namespace

std;

const int MAX_USERS =

100; bool

viewAllFlights(){

   ifstream file("flight.txt");

   string line;

   while (getline(file, line)) {

       cout << line << endl;

   }

   file.close();

   return

   false;}

bool removeFlight(string

   flightno){ ifstream

   file("flight.txt"); string line;

   while (getline(file, line)) {

       size_t found =

       line.find(flightno); if (found !=

       string::npos) {

           file.close();

           return true;

           cout<<"                                    "<<endl;
```

```
cout<<"Processing Cancellation ....." << endl;

using namespace std::this_thread; // sleep_for, sleep_until
```

```
            using namespace std::chrono; // nanoseconds, system_clock, seconds

            sleep_for(nanoseconds(10));

            sleep_until(system_clock::now() +

            seconds(4)); cout << "Cancellation

            Successful" << endl; return true;}}

    file.close();

    return

    false;}

bool removeTicket(string

    flightno){ ifstream

    file("ticket.txt"); string line;

    while (getline(file, line)) {

        size_t found =

        line.find(flightno); if (found !=

        string::npos) {

            file.close();

            return true;

            cout<<"                                "<<endl;

            cout<<"Processing Cancellation ....." << endl;

            using namespace std::this_thread; // sleep_for, sleep_until

            using namespace std::chrono; // nanoseconds, system_clock, seconds

            sleep_for(nanoseconds(10));

            sleep_until(system_clock::now() +

            seconds(4)); cout << "Cancellation

            Successful" << endl; return true;}}

    file.close();

    return false;}

bool addFlight(){

    string flightno, source, destination, date, time, price;

                                cout<<"                                "<<endl;
```

```
                        cout << "Enter flight number: ";
                        cout<<"                                "<<endl;
                        cin >> flightno;
                        cout<<"                                "<<endl;
                        cout << "Enter source: ";
                        cout<<"                                "<<endl;
                        cin >> source;
                        cout << "Enter destination: ";
                        cout<<"                                "<<endl;
                        cin >> destination;
                        cout << "Enter date:
                        "; cin >> date;
                        cout<<"                                "<<endl;
                        cout << "Enter time: ";
                        cout<<"                                "<<endl;
                        cin >> time;
                        cout<<"                                "<<endl;
                        cout << "Enter price: ";
                        cout<<"                                "<<endl;
                        cin >> price;
   ofstream file("flight.txt", ios::app);
   if (file.is_open()) {
       file << flightno << " " << source << " " << destination << " " << date << "
" << time << " " << price << endl;
       file.close();
       cout << "Flight Added Successfully!" <<
       endl; return true;
   } else {cout << "Failed to open users file for writing." << endl;
       return false;}}
```

```
bool isUsernameExists(string username, string filename)
   { ifstream file(filename);
   string line;
   while (getline(file, line)) {
       size_t found =
       line.find(username); if (found !=
       string::npos) {
           file.close();
           return
           true;}}
   file.close();
   return
   false;}
bool checkticket(string username, string filename)
   { ifstream file(filename);
   string line;
   while (getline(file, line)) {
       size_t found =
       line.find(username); if (found !=
       string::npos) {
           file.close();
           return
           true;}}
   file.close();
   return
   false;}
bool registerUser(string username, string password, string filename)
   { if (isUsernameExists(username, filename)) {
       cout << "Username already exists. Please choose a different username." <<
endl;
```

```
    return false;}
ofstream file(filename, ios::app);
if (file.is_open()) {
    file << username << " " << password << endl;
```

```
        file.close();
        cout << "Registration successful!" << endl;
        return true;
    } else {
        cout << "Failed to open users file for writing." << endl;
        return false;}}
bool addBeverage(string username, string filename)
    { if (isUsernameExists(username, filename)) {
        cout << "U CAN AD ONLY ONE ITEM IN THIS FLIGHT." << endl;
        return false;}
    ofstream file(filename, ios::app);
    if (file.is_open()) {
        file << username<< endl;
        file.close();
        cout << "Processing Payment ..." << endl;
        using namespace std::this_thread; // sleep_for, sleep_until
        using namespace std::chrono; // nanoseconds, system_clock, seconds
        sleep_for(nanoseconds(10));
        sleep_until(system_clock::now() + seconds(4));
        cout << "Payment Successful" << endl;
        return true;
    } else {
        cout << "Failed to open beverage file for writing." << endl;
        return false;}}
bool isLoginValid(string username, string password, string filename)
    { ifstream file(filename);
    string line;
    while (getline(file, line)) {
```

```
        size_t found = line.find(username + " " + password);
        if (found != string::npos) {
            file.close();
            return
            true;}}
    file.close();
    return
    false;}
bool showflight(string filename) {
    ifstream file(filename);
    string line;
    while (getline(file, line)) {
        cout << line ;}
    file.close();
    return
    false;}
bool checkflight(string username, string password, string filename) {
    ifstream file(filename);
    string line;
    while (getline(file, line)) {
        size_t found = line.find(username + " " + password);
        if (found != string::npos) {
            file.close();
            return
            true;}}
    file.close();
    return
    false;}
bool booktticket(string id, string filename)
    { if (checkticket(id, filename)) {
```

```
        cout << "Flight already in list " << endl;
        return false;}
    ofstream file(filename, ios::app);
```

```
   if (file.is_open()) {

       file << id << endl;

       file.close();

       cout << "Processing Payment ....." << endl;

       using namespace std::this_thread; // sleep_for, sleep_until

       using namespace std::chrono; // nanoseconds, system_clock, seconds

       sleep_for(nanoseconds(10));

       sleep_until(system_clock::now() + seconds(4));

       cout << "Payment Successful" << endl;

       return true;

   } else {

       cout << "Failed to open users file for writing." <<

       endl; return false;       }}

bool cancelFlight(string id,string

   filename){ ifstream file(filename);

   string line;

   while (getline(file, line)) {

       size_t found = line.find(id);

       if (found != string::npos) {

           file.close();

           return true;

           cout<<"                              "<<endl;

           cout<<"Processing Cancellation ....." << endl;

           using namespace std::this_thread; // sleep_for, sleep_until

           using namespace std::chrono; // nanoseconds, system_clock, seconds

           sleep_for(nanoseconds(10));

           sleep_until(system_clock::now() +

           seconds(4)); cout<<"                  "<<endl;
```

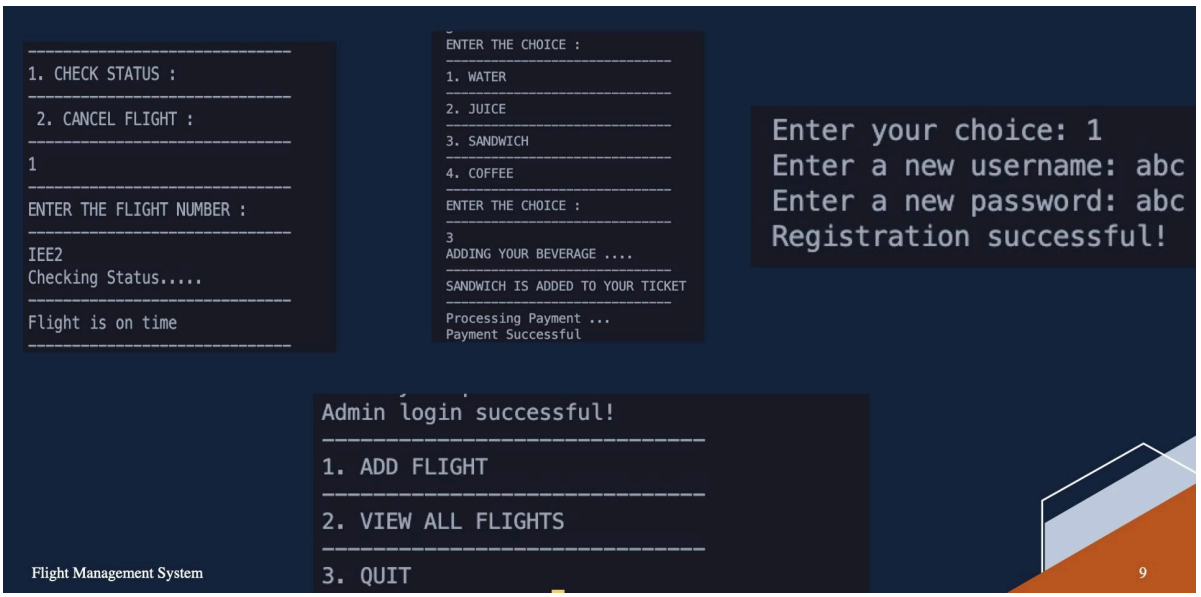
```
            cout << "Cancellation Successful" << endl;
            cout<<"                                "<<endl;}}
    file.close();
    return
    false;}
int main() {
    string username,password;
    while (true) {
        cout<<"                                "<<endl;
        cout << "Welcome to the Login/Register System!" <<
        endl; cout<<"                          "<<endl;
        cout << "1. Register as Customer" << endl;
        cout<<"
        "<<endl;
        cout << "2. Customer Login" << endl;
        cout<<"                                "<<endl;
        cout << "3. Admin Login" << endl;
        cout<<"                                "<<endl;
        cout << "4. Quit" << endl;
        cout<<"                                "<<endl;
        cout << "Enter your choice:
        "; int choice;
        cin >> choice;
        switch (choice)
        {
            case 1: {
                // Register as Customer
                ifstream
                file("customers.txt");
                file.close();
```

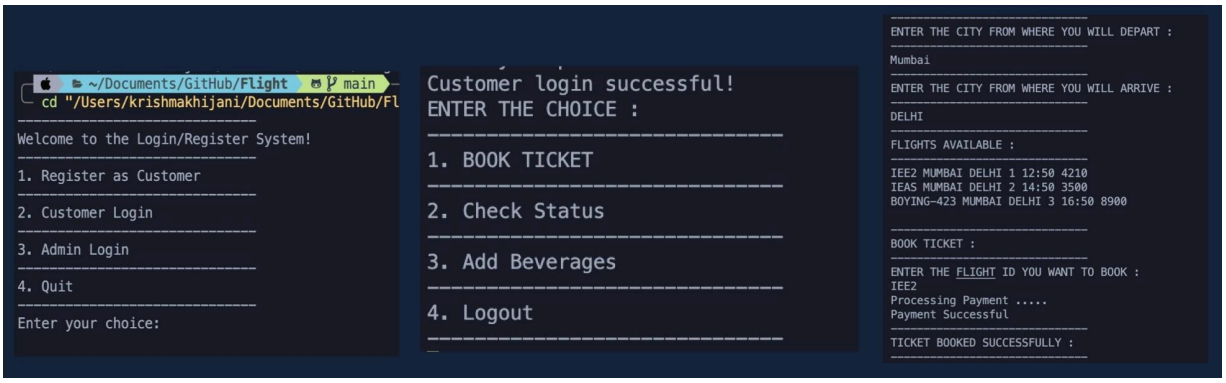
```
string newUsername, newPassword;

cout << "Enter a new username: ";
```

```
cin >> newUsername;
cout << "Enter a new password: "; cin >>
                newPassword;
                if (registerUser(newUsername, newPassword, "customers.txt")) {
                    break;
} else {
cout << "Registration failed. Please try again." << endl; break;}}
case 2: {
cout << "Enter your username: "; cin >>
                username;
cout << "Enter your password: "; cin >>
                password;
                if (isLoginValid(username, password, "customers.txt")) {
                   cout << "Customer login successful!" << endl;
int select;
cout << "ENTER THE CHOICE : "<<endl;
cout<<"             "<<endl;
cout << "1. BOOK TICKET" << endl;
cout<<"             "<<endl;}}}}
```

RESULTS


1. CHECK STATUS :
2. CANCEL FLIGHT :
1
ENTER THE FLIGHT NUMBER :
IEE2
Checking Status.....
Flight is on time
ENTER THE CHOICE :
1. WATER
2. JUICE
3. SANDWICH
4. COFFEE
ENTER THE CHOICE :
3
ADDING YOUR BEVERAGE ....
SANDWICH IS ADDED TO YOUR TICKET
Processing Payment ...
Payment Successful
Enter your choice: 1
Enter a new username: abc
Enter a new password: abc
Registration successful!
Admin login successful!
1. ADD FLIGHT
2. VIEW ALL FLIGHTS
3. QUIT
Flight Management System
9


~/Documents/GitHub/Flight main
cd "/Users/krishmakhijani/Documents/GitHub/Fl
Welcome to the Login/Register System!
1. Register as Customer
2. Customer Login
3. Admin Login
4. Quit
Enter your choice:
Customer login successful!
ENTER THE CHOICE :
1. BOOK TICKET
2. Check Status
3. Add Beverages
4. Logout
ENTER THE CITY FROM WHERE YOU WILL DEPART :
Mumbai
ENTER THE CITY FROM WHERE YOU WILL ARRIVE :
DELHI
FLIGHTS AVAILABLE :
IEE2 MUMBAI DELHI 1 12:50 4210
IEAS MUMBAI DELHI 2 14:50 3500
BOYING-423 MUMBAI DELHI 3 16:50 8900
BOOK TICKET :
ENTER THE FLIGHT ID YOU WANT TO BOOK :
IEE2
Processing Payment .....
Payment Successful
TICKET BOOKED SUCCESSFULLY :