



King Fahd University of Petroleum and Minerals
ICS 324:
Database Systems (212) Course Project
Group 36

Students	
Name	Serial number
Mohammed Alhassan	201831660
Abdullah Almohaimeed	201846220
Mohtady Alhelal	201825300

Abstract:

The purpose of this project is to learn how to design and implement a database for an airlines company and developing a user-interface application.

Constrains and Business Rules

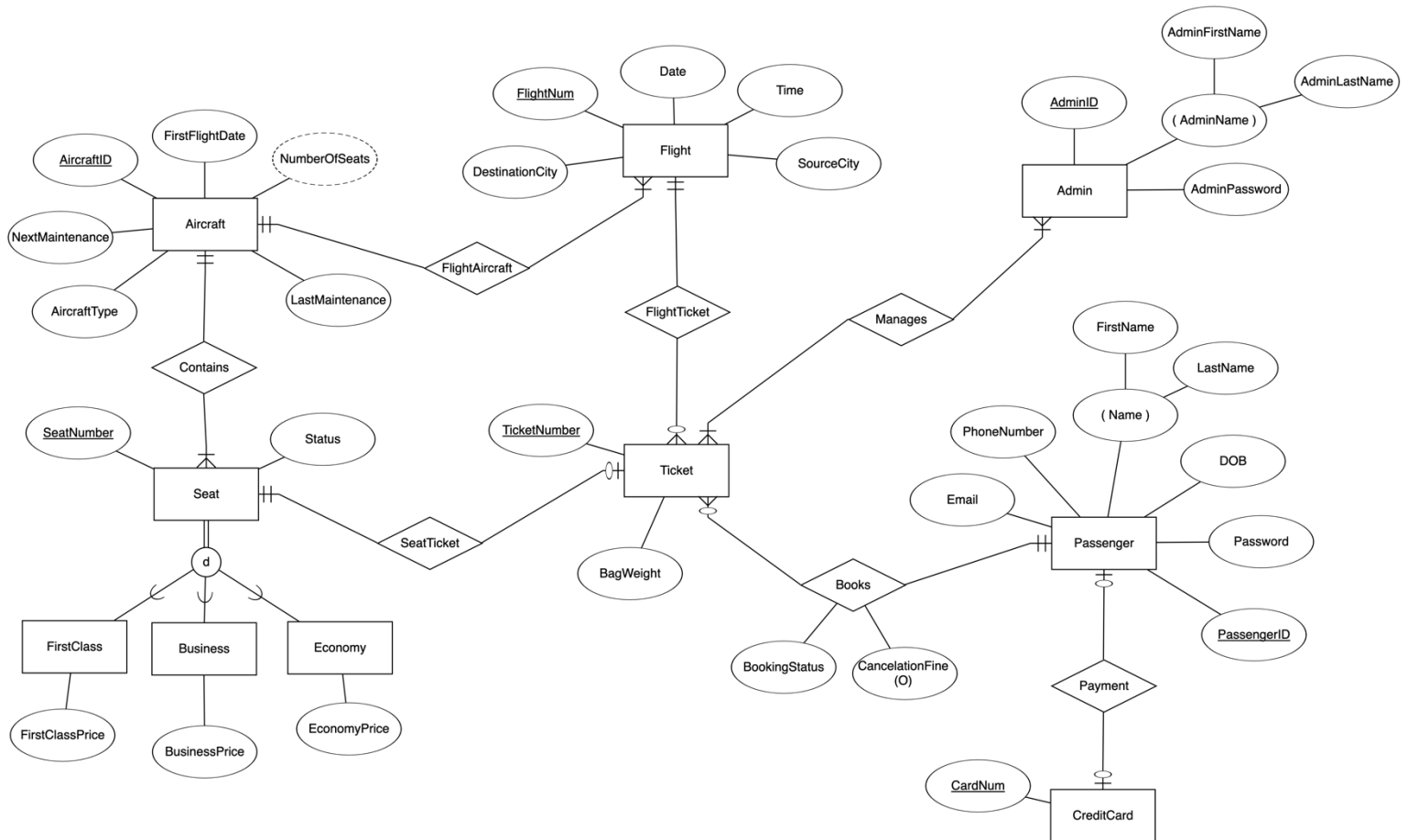
Business rules:

- A passenger is be able to search flights by dates, destination and source city.
- Each ticket has a unique identification number and other details including flight dates, weight, time etc.
- Passenger can book a seat if it is not booked already.
- The system maintains information of assigned plane for a flight. The planes have various attributes including aircraft type, date of first flight, total seats, last and next dates for maintenance etc.
- The aircraft details are maintained separately including aircraft type, number of seats in first class, business class and economy and price of seats for each class.
- The system maintains waitlisted passengers up to 10 seats in economy and 3 in other classes.
- There is a maximum limit (10) seats to be booked by passenger at any time in specific flight.
- There is a maximum limit (90) on how many days a member can reuse ticket whenever miss his/her flight.
- The system identifies fines for cancelation or missing flight.
- The system is able to send notifications whenever the seat booked and ticket successfully purchased.
- Every passenger's payment details should be kept.

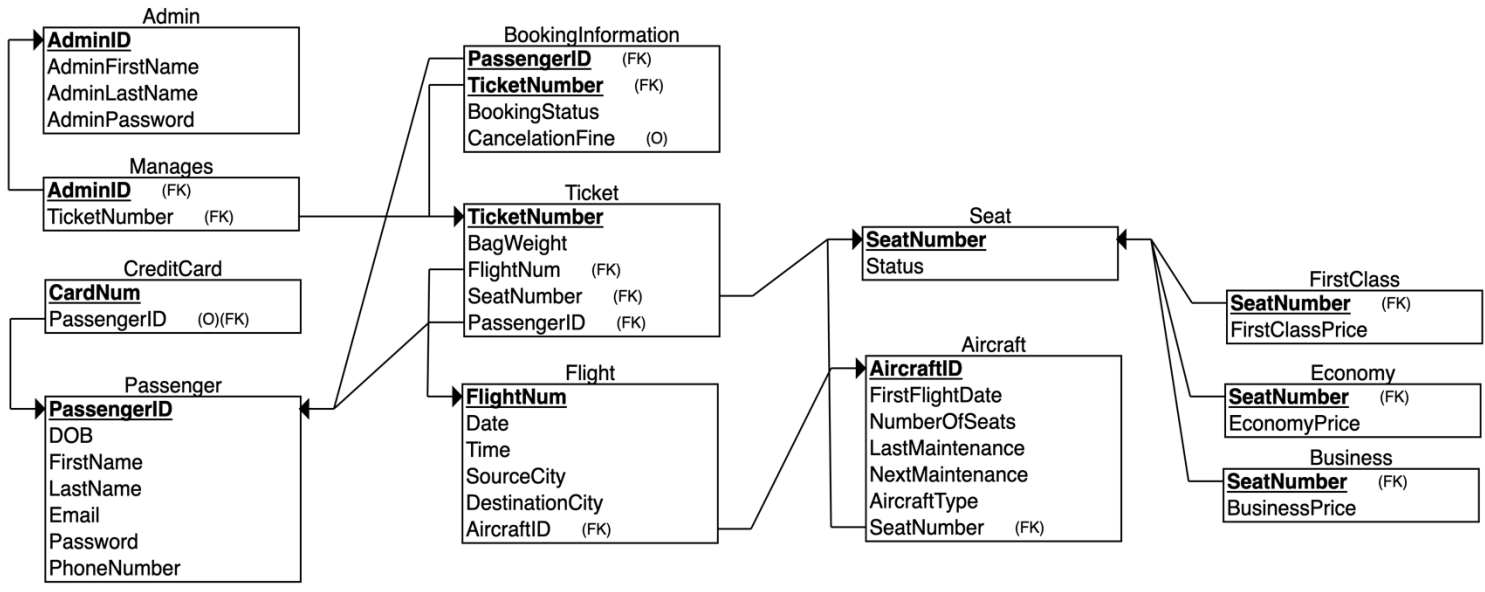
Technical constraints:

- Passenger:
 - PK: ID
 - FK: FlightNum, TicketNum, SeatNum, ReuseTicketNum
 - Not null: Fname, Lname, DOBirth, PhoneNum, Email
- Admin:
 - PK: AdminID
 - Not null: First, Last
- Seats:
 - PK: SeatNum
 - Not null: Price, Status
- Flight:
 - PK: FlightNum
 - Not null: Date, Destination, SourceCity
- Payment:
 - PK: CardNum
 - FK: ID
 - Not null: ID
- Ticket:
 - PK: TicketNum
 - FK: SeatNum
 - Not null: FlightDate, Weight, Time, SeatNum
- Aircraft:
 - PK: AircraftID
 - FK: SeatNum
 - Not null: LastMaintenance, DateFirstFlight, NextMaintenance, TotalSeats
- System:
 - FK: SeatNum, TicketNum, CanelTicketNum

EER Diagram



Relational Schema



Prototype



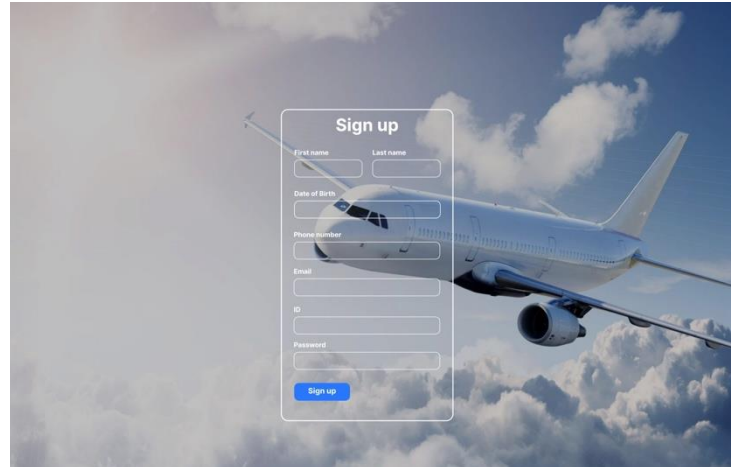
Login

Email

Password

[Login](#)

[Sign up](#)



Sign up

First name Last name

Date of Birth

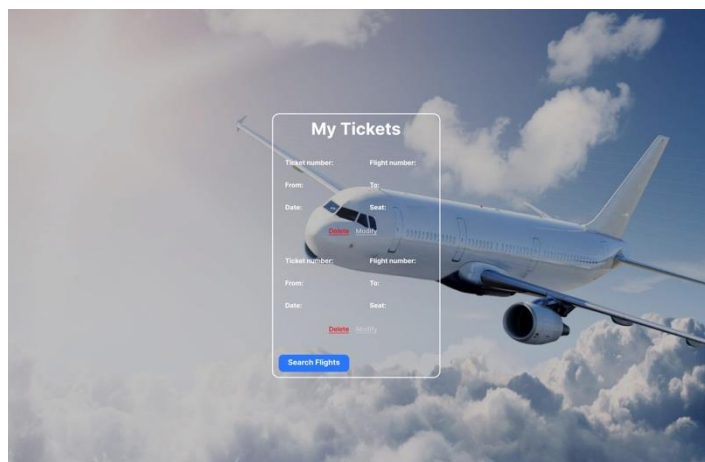
Phone number

Email

ID

Password

[Sign up](#)



My Tickets

Ticket number: Flight number:

From: To:

Date: Seat:

[Delete](#) [Modify](#)

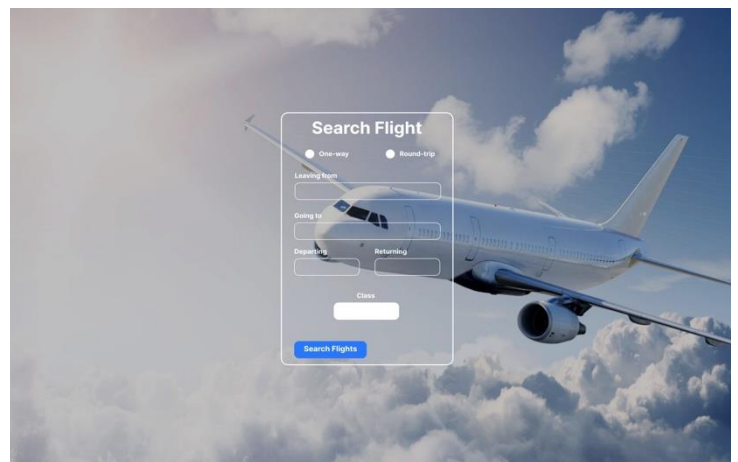
Ticket number: Flight number:

From: To:

Date: Seat:

[Delete](#) [Modify](#)

[Search Flights](#)



Search Flight

☐ One-way ☐ Round-trip


Leaving from

Going to

Departing Returning

Class

[Search Flights](#)



Available Flights

Flight number:

From: To:

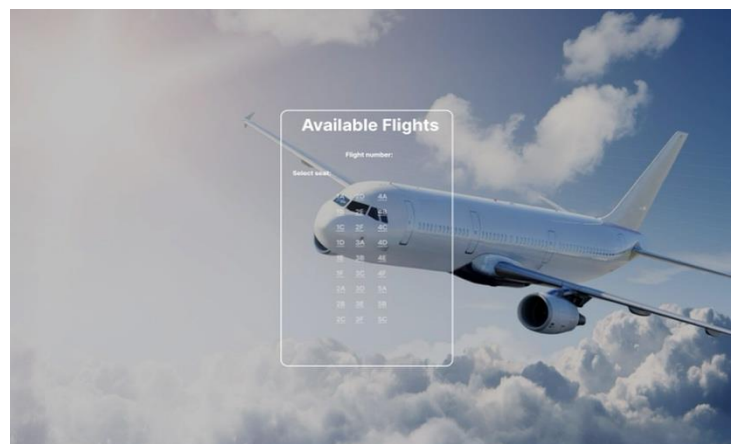
Date and time: Aircraft type:

Flight number:

From: To:

Date and time: Aircraft type:

[Select](#)

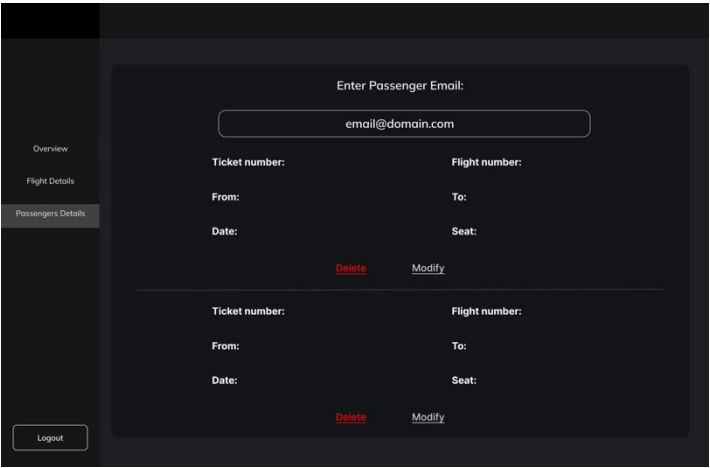
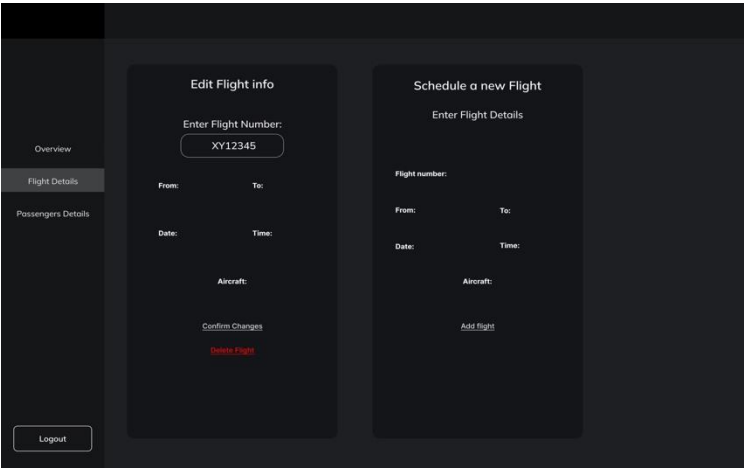
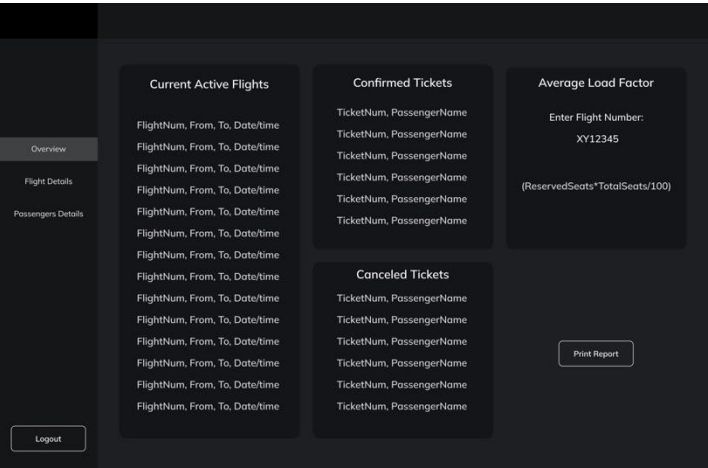
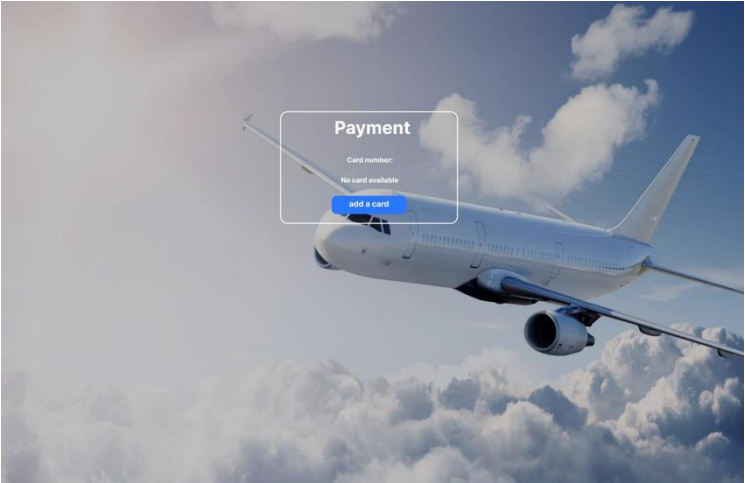
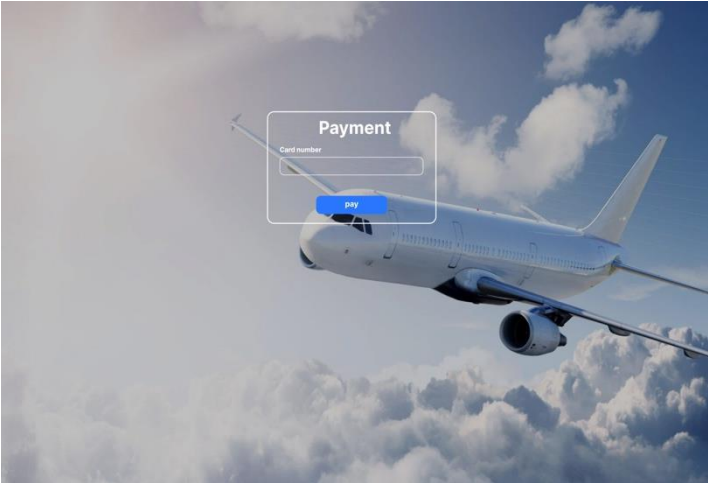


Available Flights

Flight number:

Select seat:

1A	2F	3C	4D
5E	6A	7B	8C
9D	10E	11A	12B
13C	14D	15E	16A
17B	18C	19D	20E



Note: prototypes were made before implementation

SQL Queries

Initializing Tables

```
CREATE TABLE Passenger
(
  PassengerID CHAR(10) NOT NULL,
  DOB DATE NOT NULL,
  FirstName VARCHAR(32) NOT NULL,
  LastName VARCHAR(32) NOT NULL,
  Email VARCHAR(50) NOT NULL UNIQUE,
  Password VARCHAR(26) NOT NULL,
  PhoneNumber CHAR(10) NOT NULL,
  PRIMARY KEY (PassengerID)
);
```

```
CREATE TABLE CreditCard
(
  CardNum INT NOT NULL,
  PassengerID CHAR(10),
  PRIMARY KEY (CardNum),
  FOREIGN KEY (PassengerID) REFERENCES Passenger(PassengerID)
);
```

```
CREATE TABLE Seat
(
  SeatNumber INT NOT NULL,
  Status VARCHAR(10) NOT NULL,
  PRIMARY KEY (SeatNumber)
);
```

```
CREATE TABLE FirstClass
(
  FirstClassPrice INT NOT NULL,
  SeatNumber INT NOT NULL,
  PRIMARY KEY (SeatNumber),
  FOREIGN KEY (SeatNumber) REFERENCES Seat(SeatNumber)
);
```

```
CREATE TABLE Business
(
  BusinessPrice INT NOT NULL,
  SeatNumber INT NOT NULL,
  PRIMARY KEY (SeatNumber),
  FOREIGN KEY (SeatNumber) REFERENCES Seat(SeatNumber)
);
```

```
CREATE TABLE Economy
```



```
(
    EconomyPrice INT NOT NULL,
    SeatNumber INT NOT NULL,
    PRIMARY KEY (SeatNumber),
    FOREIGN KEY (SeatNumber) REFERENCES Seat(SeatNumber)
);
```

CREATE TABLE Admin

```
(
    AdminID CHAR(10) NOT NULL,
    AdminFirstName VARCHAR(32) NOT NULL,
    AdminLastName VARCHAR(32) NOT NULL,
    AdminPassword VARCHAR(26) NOT NULL,
    PRIMARY KEY (AdminID)
);
```

CREATE TABLE Aircraft

```
(
    FirstFlightDate DATE NOT NULL,
    AircraftID CHAR(10) NOT NULL,
    NumberOfSeats INT,
    LastMaintenance DATE,
    NextMaintenance DATE,
    AircraftType VARCHAR(32) NOT NULL,
    SeatNumber INT,
    PRIMARY KEY (AircraftID),
    FOREIGN KEY (SeatNumber) REFERENCES Seat(SeatNumber)
);
```

CREATE TABLE Flight

```
(
    FlightNum CHAR(5) NOT NULL,
    Date DATE NOT NULL,
    Time CHAR(5) NOT NULL,
    SourceCity VARCHAR(58) NOT NULL,
    DestinationCity VARCHAR(58) NOT NULL,
    AircraftID CHAR(10) NOT NULL,
    PRIMARY KEY (FlightNum),
    FOREIGN KEY (AircraftID) REFERENCES Aircraft(AircraftID)
);
```

CREATE TABLE Ticket

```
(
    TicketNumber CHAR(5) NOT NULL,
    BagWeight INT,
    FlightNum CHAR(5) NOT NULL,
    SeatNumber INT NOT NULL,
    PassengerID CHAR(10) NOT NULL,
    PRIMARY KEY (TicketNumber),
    FOREIGN KEY (FlightNum) REFERENCES Flight(FlightNum),
    FOREIGN KEY (SeatNumber) REFERENCES Seat(SeatNumber),
    FOREIGN KEY (PassengerID) REFERENCES Passenger(PassengerID)
);
```

CREATE TABLE Manages

```
(
  AdminID CHAR(10) NOT NULL,
  TicketNumber CHAR(5) NOT NULL,
  PRIMARY KEY (AdminID),
  FOREIGN KEY (AdminID) REFERENCES Admin(AdminID),
  FOREIGN KEY (TicketNumber) REFERENCES Ticket(TicketNumber)
);
```

CREATE TABLE BookingInformation

```
(
  BookingStatus VARCHAR(10) NOT NULL,
  CancelationFine INT,
  PassengerID CHAR(10) NOT NULL,
  TicketNumber CHAR(5) NOT NULL,
  PRIMARY KEY (PassengerID, TicketNumber),
  FOREIGN KEY (PassengerID) REFERENCES Passenger(PassengerID),
  FOREIGN KEY (TicketNumber) REFERENCES Ticket(TicketNumber)
);
```

Back-end Queries

#insert user info (sign up)

```
INSERT INTO Passenger (  
FirstName, LastName, DOB, PhoneNumber, Email, PassengerID, Password)  
Values ($1, $2, $3, $4, $5, $6, $7)
```

#fetch the user email and password (login)

#note: the ID is used to store in browser cookies to keep user logged in

```
SELECT email, password, passengerid FROM passenger  
WHERE email = ${email} AND Password = ${password}
```

#fetch the admin ID and password (login)

```
SELECT adminid, adminpassword FROM Passenger  
WHERE Email = ${adminID} AND Password = ${adminPassword}
```

#display all of the user's tickets

```
SELECT * FROM (  
SELECT * FROM ticket JOIN flight ON ticket.flightnum = flight.flightnum  
AND passengerid = ${passengerid}) AS dt
```

#display all of the user's (active tickets/ confirmed payments)

```
SELECT * FROM (SELECT * FROM ticket JOIN flight ON ticket.flightnum = flight.flightnum  
AND passengerid = ${passengerid}) AS dt  
WHERE flightNum = (SELECT flightNum FROM bookingInformation WHERE status = 'confirmed')
```

#display all of the user's (active tickets/ confirmed payments)

```
SELECT * FROM (SELECT * FROM ticket JOIN flight ON ticket.flightnum = flight.flightnum  
AND passengerid = ${passengerid}) AS dt  
WHERE flightNum = (SELECT flightNum FROM bookingInformation WHERE status = 'canceled')
```

#add a new aircraft

```
INSERT INTO aircraft VALUES (${firstFlightDate}, ${aircraftID}, ${countSeats()} , ${lastMaintenance},  
${nextMaintenance}, ${aircraftType}, NULL);
```

#add a new flight

```
INSERT INTO flight VALUES (${flightNum}, ${flightDate}, ${flightTime}, ${srcCity}, ${dstCity},  
${aircraftID});
```

#add a new ticket

```
INSERT INTO tickets VALUES (${ticketID}, ${baggageWeight}, ${flightNum}, ${seatNum}, ${passengerID});
```

#display all (active tickets/ confirmed payments)

```
SELECT * FROM flight WHERE flightNum = (SELECT flightNum FROM bookingInformation WHERE status =  
'confirmed');
```

#display all canceled tickets

```
SELECT * FROM flight WHERE flightNum = (SELECT flightNum FROM bookingInformation WHERE status =  
'canceled');
```

Tools and Resources

Communication	<ul style="list-style-type: none">• Whatsapp• MS teams
Sharing data	<ul style="list-style-type: none">• Github• Whatsapp
Diagram modeling	<ul style="list-style-type: none">• ERDPlus• Draw.io
Prototyping	<ul style="list-style-type: none">• Figma
Application Type	<ul style="list-style-type: none">• Web
DBMS	<ul style="list-style-type: none">• PostgreSQL
Database Software	<ul style="list-style-type: none">• pgAdmin
Front-end	<ul style="list-style-type: none">• ReactJS
Back-end	<ul style="list-style-type: none">• NodeJS
Frameworks	<ul style="list-style-type: none">• Bootstrap
Resources and inspiration	<ul style="list-style-type: none">• Youtube• Udemy• W3Schools• Stackoverflow• KFUPM ICS 324 slides• Behance• Figma community

Github Repository for the project:
<https://github.com/M-Alhassan/Airlines-Booking-System>

Work Distribution

Phase 1

Task	Mohammed Alhassan	Abdullah Almohaimeed	Mohtady Alhelal
Constraints	33%	33%	33%
EER Diagram	20%	45%	35%
Relational Schema	30%	30%	40%
Report	60%	20%	20%
Total	36%	32%	32%

Phase 2

Task	Mohammed Alhassan	Abdullah Almohaimeed	Mohtady Alhelal
Prototype	100%	0%	0%
EERD and Relational Schema	100%	0%	0%
Frontend	100%	0%	0%
Backend	100%	0%	0%
Database	100%	0%	0%
SQL Queries	100%	0%	0%
Total	100%	0%	0%