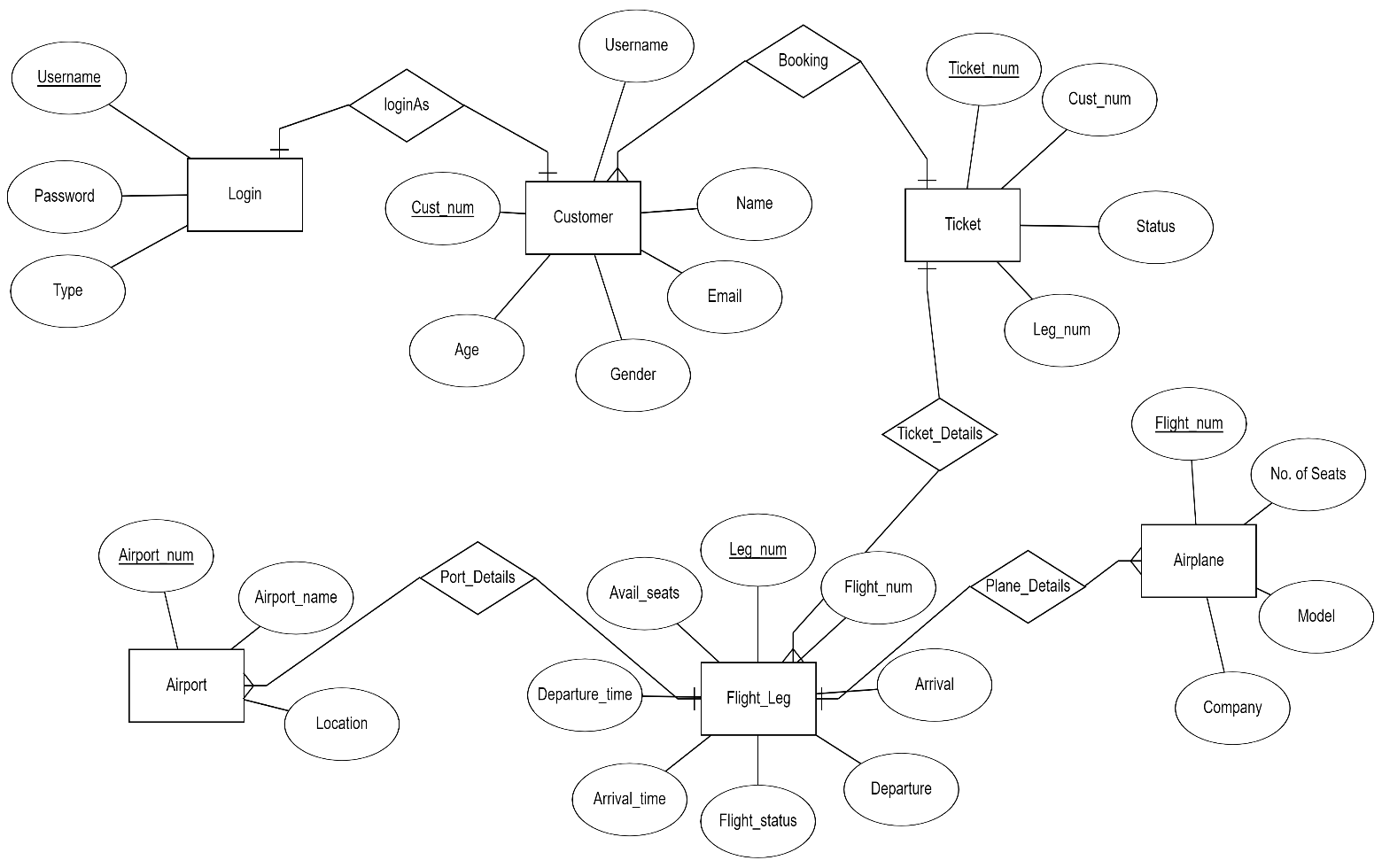
Flight\_Management

Database Queries:

|  |
| --- |
| CREATE DATABASE IF NOT EXISTS Flight\_Management;  use Flight\_Management;  Create table if not exists Logins (  Username varchar(20) ,  Password Varchar (20),  Type enum('admin','cust'),  Primary Key (Username)  );  Create table if not exists Customer (  Cust\_num INT not null auto\_increment ,  Username varchar(20) UNIQUE,  Name varchar(40) ,check (Name not regexp '[0-9]'),  Email varchar(20) ,  Gender enum("M" , "F") ,  Date\_of\_birth date not null ,  passport\_number varchar(15) not null ,  foreign key (Username) references Logins(Username) ,  Primary key (Cust\_num)  );  Create table if not exists Airplane (  Flight\_num int not null auto\_increment ,  num\_seats int not null, check (num\_seats >0),  model varchar(20) ,  company varchar(20) ,  primary key (Flight\_num)  );  Create table if not exists Airport (  Airport\_num int not null auto\_increment ,  Name Varchar(20) unique ,  Location varchar(20) unique ,  primary key (Airport\_num)  ) ;  Create table if not exists Flight\_legs (  Leg\_num int not null auto\_increment ,  flight\_num int ,  Arrival\_Airport int ,  departure\_airport int ,  flight\_status enum  ('ACTIVE','SCHEDULED', 'DELAYED','DEPARTED','LANDED','IN\_AIR','ARRIVED','CANCELLED', 'DIVERTED', 'UNKNOWN' ) ,  distance int, check(distance >0),  arrival\_time datetime ,  departure\_time datetime,  avail\_seats int,check(avail\_seats > 0),  cost int ,  primary key (Leg\_num),  foreign key (Arrival\_Airport) references Airport(Airport\_num) on DELETE CASCADE,  foreign key (departure\_airport) references Airport(Airport\_num) on DELETE CASCADE ,  foreign key (flight\_Num) references airplane(flight\_num) on DELETE CASCADE  );  Create table if not exists ticket (  ticket\_num int not null auto\_increment ,  cust\_num int not null ,  status enum( 'PENDING', 'CONFIRMED', 'CANCELLED') ,  leg\_num int not null,  seats int ,  foreign key (cust\_num) references Customer(Cust\_num) ,  foreign key (leg\_num ) references Flight\_legs (Leg\_num) on DELETE CASCADE ,  primary key (ticket\_num)  ) ;  create trigger loginsInsertCustomer after insert on logins for each row insert into customer(username,passport\_number,date\_of\_birth) values (new.username, "0000000","0001-01-01");  insert into logins values ('admin','admin','admin'); |

E-R Diagram:

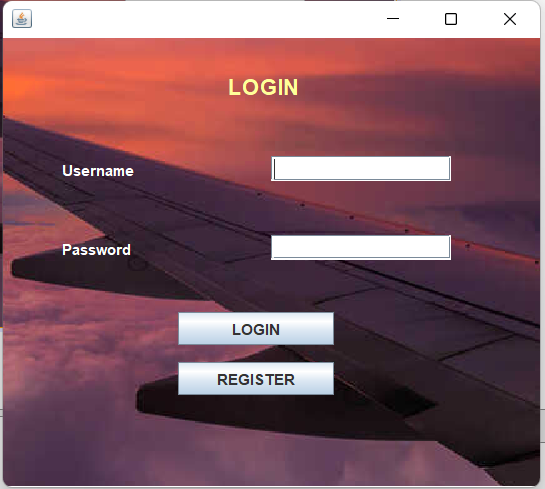


Java Classes and Functions:

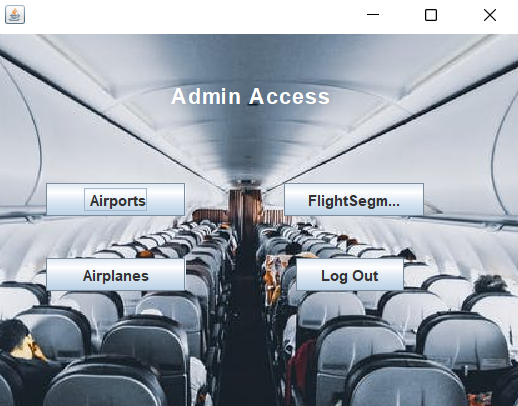
|  |
| --- |
| **Package: main**   * AddAirport.java : (For Admin) Add, Update and Delete Airports from database. * AddLeg.java : (For Admin) Add, Update and Delete Flight legs from database. * AddPlane.java : (For Admin) Add, Update and Delete Planes from database. * AdminDash.java : Dashboard for admin. * Book.java : Customer can search flights by arrival and destination and book them accordingly. * CustDash.java: Dashboard for admin. * Date\_checker.java : Checks for valid date. * Details.java : Customers can add or update their personal details. * Is\_number.java : Checks if a string is a valid number. * Main.java: Login and Register Page. The System will check if the user is admin or customer and shows the dashboard accordingly. * TicketCan.java : Customer can see all their past tickets booked or cancelled and they can also cancel any ticket. * time\_check.java : Checks for valid time. |
| **Package: TableClasses**   * Account.java :   + Parameters: Username, Password type * Airplane.java   + Parameters: Flight number, model, company, max seats available.   + check\_by\_num(int number): checks if the airport exists or not in the database. * Airport.java   + Parameters: Airport\_number, name, Location   + get\_airportnum\_byName(String a ): Checks in database and returns the airport number according to the name   + get\_airportLoc\_byNum(int a ): Checks in database and returns the airport location according to the number. * ConnectionManager.java: Details for MySQL like password, username, URL. * Flight\_leg.java   + Parameters: Leg\_num, Flight\_num, departure, arrival, dept\_time, arrival\_time,distance, cost, Status   + check(int leg\_num): Checks in database if that leg\_num is valid or not.   + getBy\_legnum(int lnum): Update parameters according to the leg\_num   + increment(int lnum,int seats\_taken): When the customer cancels their ticket the available seats in the corresponding leg increases.   + decrement(int lnum,int seats\_req): When the customer books their ticket the available seats in the corresponding leg decreases according to the number of seats the customer booked. * Flight\_status.java * Displays the Current Status of the Flight. * Person.java   + Parameters: id, name, email, passport\_num, gender,date\_of\_birth   + get\_by\_uname(String uname): Gets user details by their username. * Ticket.java   + Paramers: ticket\_num,Leg\_num, cust\_num,seats\_booked,status. |

Screenshots:

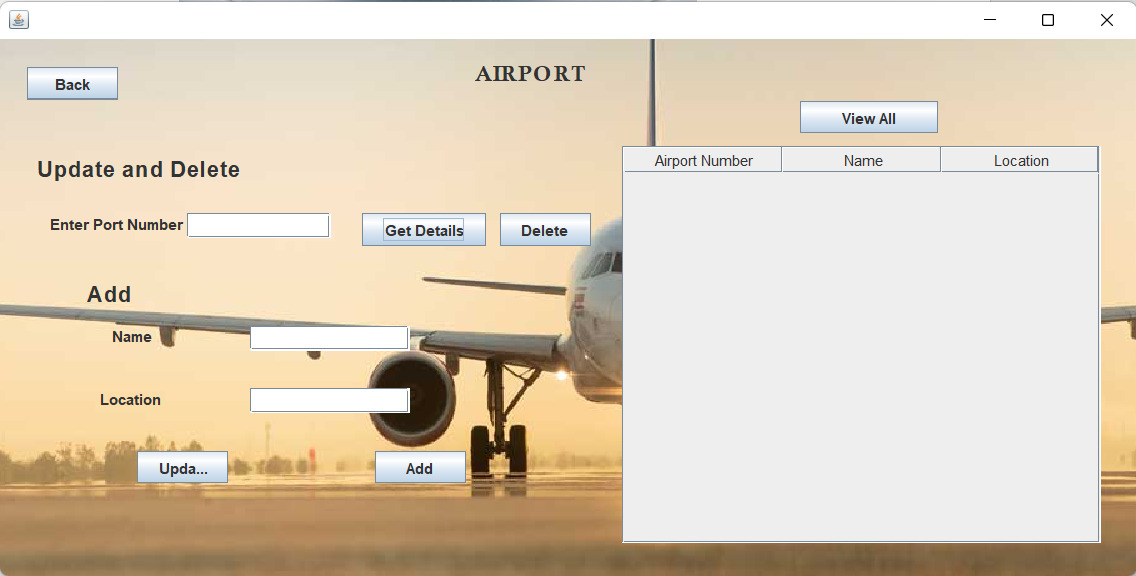
* Login Page



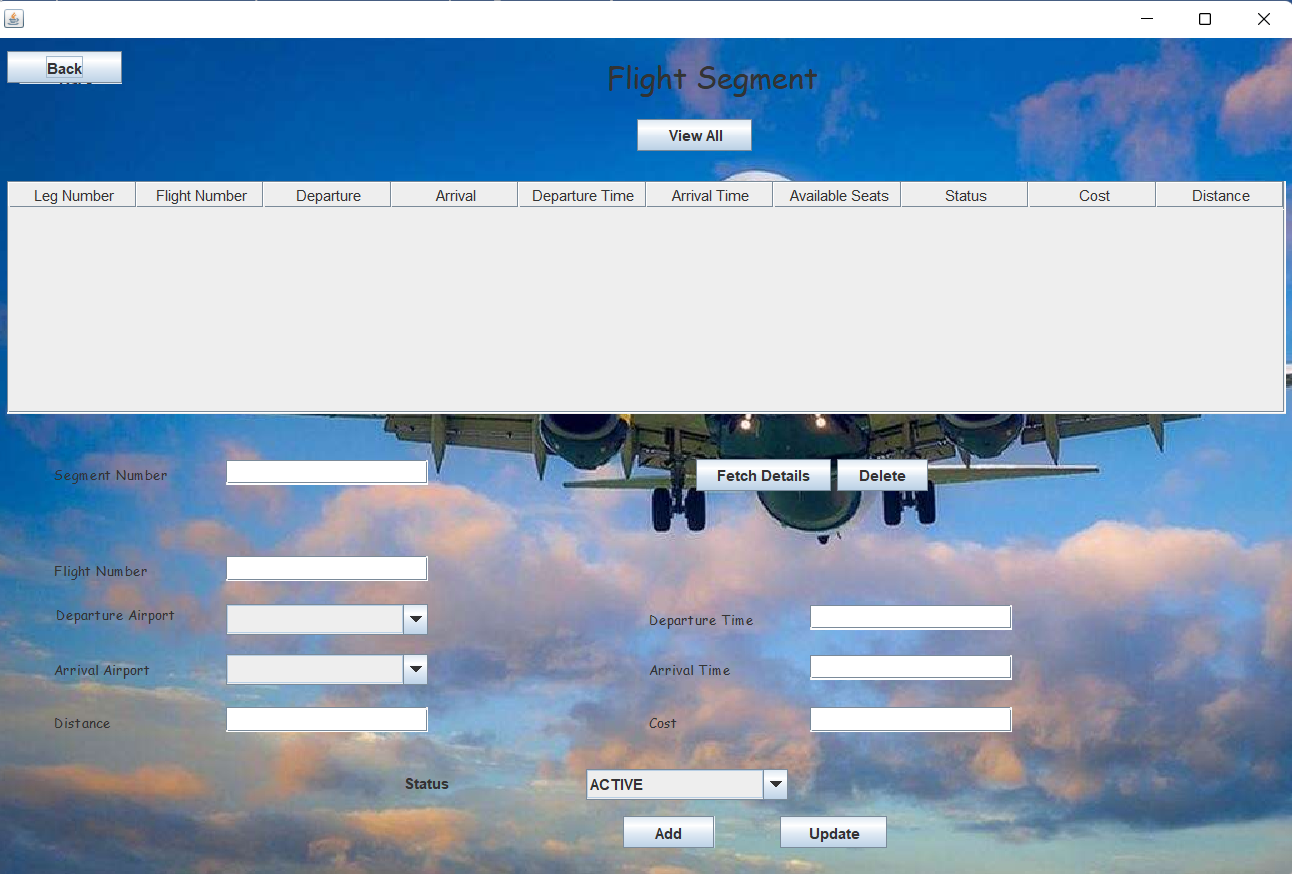
* Admin Dashboard



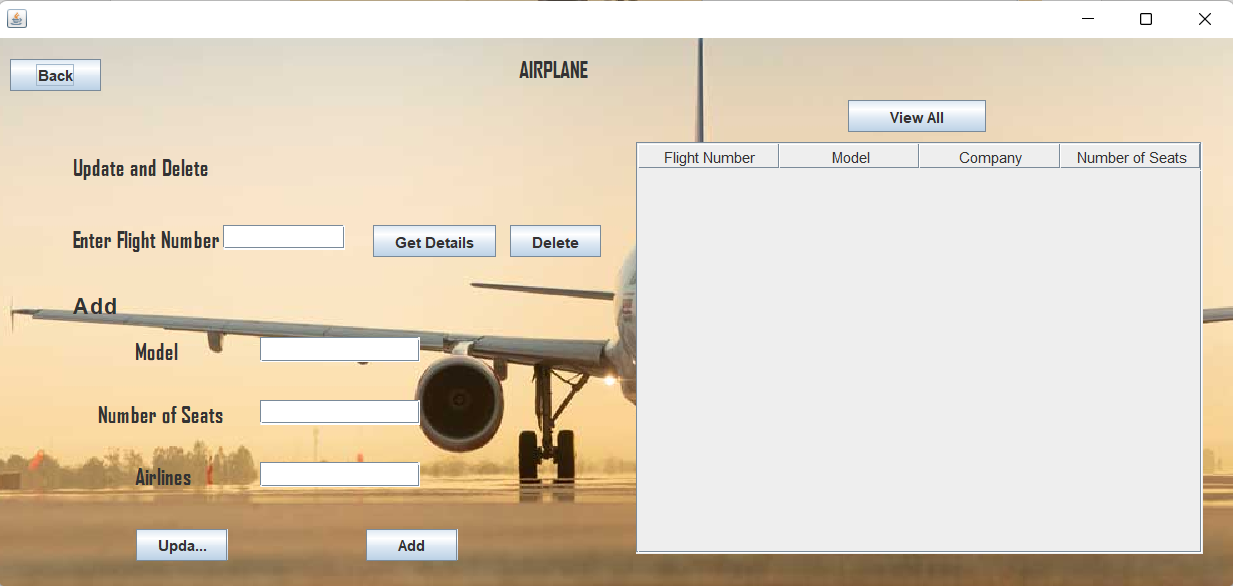
* Airport Page



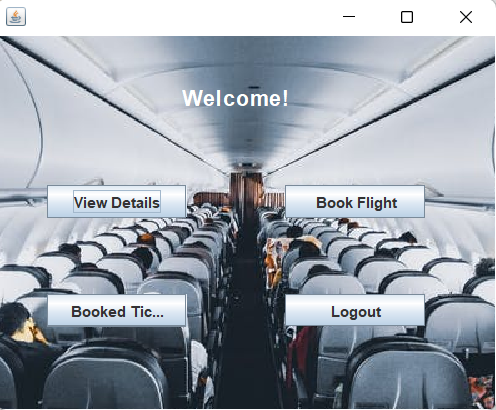
* Flight Segment



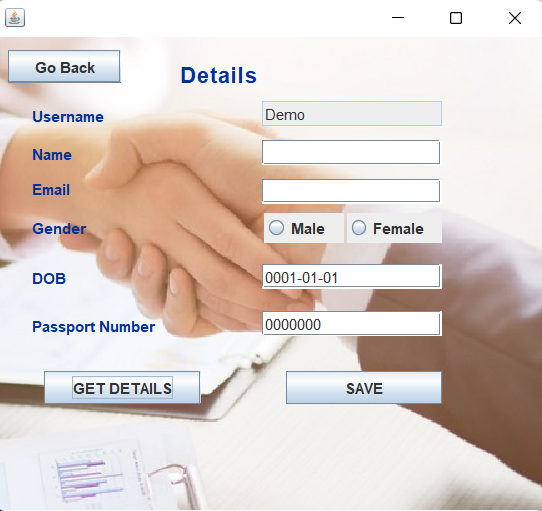
* Airplane Details



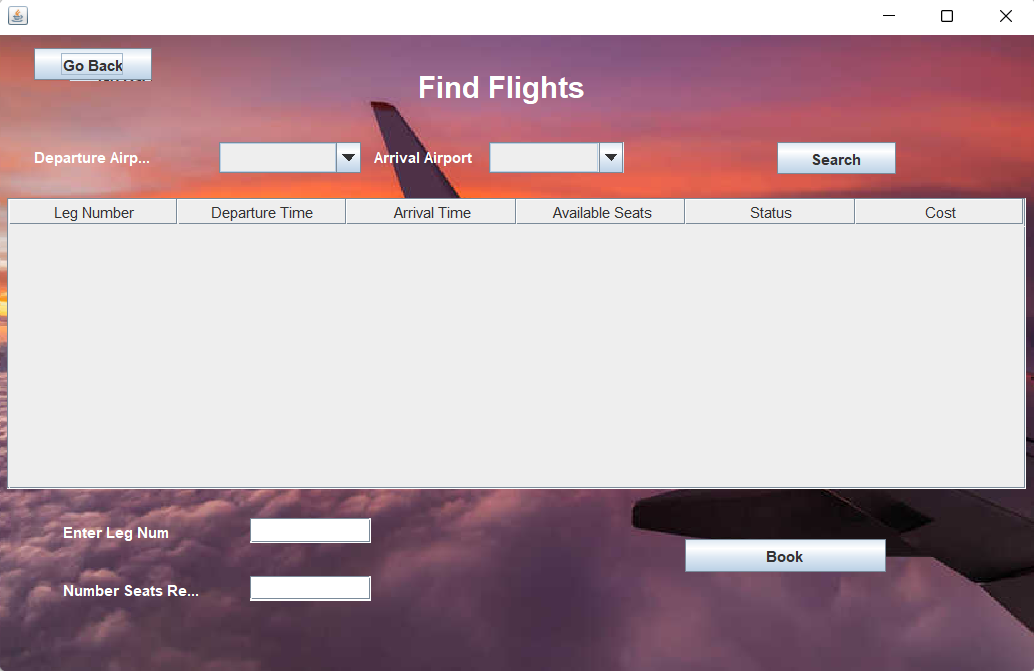
* Customer Dashboard



* Customer Details



* Flight Details



* Ticket Details

