**SOURCE CODES**

**AirlineDao,java**

package com.dao;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import com.model.Airline;

public class AirlineDao {

private String jdbcURL = "jdbc:mysql://localhost:3306/admin?useSSL=false";

private String jdbcUsername = "admin";

private String jdbcPassword = "admin@12";

// airlineId,airlineName,code, capacity;

public AirlineDao() {

}

protected Connection getConnection() {

Connection connection = null;

try {

Class.forName("com.mysql.jdbc.Driver");

connection = DriverManager.getConnection(jdbcURL, jdbcUsername, jdbcPassword);

System.out.println("Db details " + connection.toString());

} catch (SQLException e) {

// What to do // Debug

e.printStackTrace();

SqlException(e);

} catch (ClassNotFoundException e) {

// What to do // Debug

e.printStackTrace();

}

return connection;

}

public void insert(Airline airline) throws SQLException {

// airlineId,airlineName,code, capacity;

String INSERT\_SQL = "INSERT INTO airlines" + " (airlineName,code,capacity) VALUES "

+ " (?,?,?);";

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(INSERT\_SQL)) {

preparedStatement.setString(1, airline.getAirlineName());

preparedStatement.setString(2, airline.getCode());

preparedStatement.setInt(3, airline.getCapacity());

System.out.println(preparedStatement);

preparedStatement.executeUpdate();

} catch (SQLException e) {

SqlException(e);

}

}

public Airline getByKey(int airlineId) {

String SELECT\_BY = "select airlineId,airlineName,code, capacity from airlines where airlineId =?";

Airline airline = null;

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(SELECT\_BY);) {

preparedStatement.setInt(1, airlineId);

System.out.println(preparedStatement);

ResultSet rs = preparedStatement.executeQuery();

while (rs.next()) {

String airlineName = rs.getString("airlineName");

String code = rs.getString("code");

int capacity = rs.getInt("capacity");

airline = new Airline();

airline.setAirlineId(airlineId);

airline.setAirlineName(airlineName);

airline.setCode(code);

airline.setCapacity(capacity);

System.out.println(airline.toString());

}

} catch (SQLException e) {

SqlException(e);

}

return airline;

}

public List<Airline> listOfAll() {

String SELECT\_ALL = "select \* from airlines";

List<Airline> airlines = new ArrayList<>();

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(SELECT\_ALL);) {

System.out.println(preparedStatement);

ResultSet rs = preparedStatement.executeQuery();

while (rs.next()) {

int airlineId = rs.getInt("airlineId");

String airlineName = rs.getString("airlineName");

String code = rs.getString("code");

int capacity = rs.getInt("capacity");

Airline airline = new Airline();

airline.setAirlineId(airlineId);

airline.setAirlineName(airlineName);

airline.setCode(code);

airline.setCapacity(capacity);

airlines.add( airline);

}

} catch (SQLException e) {

SqlException(e);

}

return airlines;

}

public boolean delete(int airlineId) throws SQLException {

boolean status;

String DELETE\_SQL = "delete from airlines where airlineId = ?;";

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(DELETE\_SQL);) {

preparedStatement.setInt(1, airlineId);

System.out.println(preparedStatement);

status = preparedStatement.executeUpdate() > 0;

}

return status;

}

public boolean update(Airline airline) throws SQLException {

boolean status;

String UPDATE\_SQL = "update airlines set airlineName=?, code=?, capacity=? where airlineId = ?;";

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(UPDATE\_SQL);) {

System.out.println("updated airlines :"+preparedStatement);

preparedStatement.setString(1, airline.getAirlineName());

preparedStatement.setString(2, airline.getCode());

preparedStatement.setInt(3, airline.getCapacity());

preparedStatement.setInt(4, airline.getAirlineId());

System.out.println(preparedStatement);

preparedStatement.executeUpdate();

status = preparedStatement.executeUpdate() > 0;

}

return status;

}

private void SqlException(SQLException ex) {

for (Throwable e : ex) {

if (e instanceof SQLException) {

e.printStackTrace(System.err);

System.err.println("SQLState: " + ((SQLException) e).getSQLState());

System.err.println("Error Code: " + ((SQLException) e).getErrorCode());

System.err.println("Message: " + e.getMessage());

Throwable t = ex.getCause();

while (t != null) {

System.out.println("Cause: " + t);

t = t.getCause();

}

}

}

}

}

**AdminDao.java**

package com.dao;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import com.model.Admin;

import com.model.Customer;

public class AdminDao {

private String jdbcURL = "jdbc:mysql://localhost:3306/admin?useSSL=false";

private String jdbcUsername = "admin";

private String jdbcPassword = "admin@12";

// useName password

public AdminDao() {

}

protected Connection getConnection() {

Connection connection = null;

try {

Class.forName("com.mysql.jdbc.Driver");

connection = DriverManager.getConnection(jdbcURL, jdbcUsername, jdbcPassword);

System.out.println("Db details " + connection.toString());

} catch (SQLException e) {

// What to do // Debug

e.printStackTrace();

SqlException(e);

} catch (ClassNotFoundException e) {

// What to do // Debug

e.printStackTrace();

}

return connection;

}

public void insert(Admin admin) throws SQLException {

// userName password;

String INSERT\_SQL = "INSERT INTO admin" + " (userName,password) VALUES "

+ " (?,?);";

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(INSERT\_SQL)) {

preparedStatement.setString(1, admin.getUserName());

preparedStatement.setString(2, admin.getPassword());

System.out.println(preparedStatement);

preparedStatement.executeUpdate();

} catch (SQLException e) {

SqlException(e);

}

}

public Admin getByKey(String userName) {

String SELECT\_BY = "select password, userName from admin where userName =?";

Admin admin = null;

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(SELECT\_BY);) {

preparedStatement.setString(1, userName);

System.out.println(preparedStatement);

ResultSet rs = preparedStatement.executeQuery();

while (rs.next()) {

String password = rs.getString("password");

String auserName = rs.getString("userName");

admin = new Admin();

admin.setUserName(auserName);

admin.setPassword(password);

System.out.println(admin.toString());

}

} catch (SQLException e) {

SqlException(e);

}

return admin;

}

public List<Admin> listOfAll() {

String SELECT\_ALL = "select \* from admin";

List<Admin> admins = new ArrayList<>();

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(SELECT\_ALL);) {

System.out.println(preparedStatement);

ResultSet rs = preparedStatement.executeQuery();

while (rs.next()) {

String userName = rs.getString("userName");

String password = rs.getString("password");

Admin admin = new Admin(userName,password);

admins.add( admin);

}

} catch (SQLException e) {

SqlException(e);

}

return admins;

}

public boolean delete(String admin) throws SQLException {

boolean status;

String DELETE\_SQL = "delete from admin where userName = ?;";

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(DELETE\_SQL);) {

preparedStatement.setString(1, admin);

System.out.println(preparedStatement);

status = preparedStatement.executeUpdate() > 0;

}

return status;

}

public boolean update(Admin admin) throws SQLException {

boolean status;

String UPDATE\_SQL = "update admin set password=? where userName = ?;";

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(UPDATE\_SQL);) {

System.out.println("updated admin :"+preparedStatement);

preparedStatement.setString(1, admin.getPassword());

preparedStatement.setString(2, admin.getUserName());

System.out.println(preparedStatement);

preparedStatement.executeUpdate();

status = preparedStatement.executeUpdate() > 0;

}

return status;

}

private void SqlException(SQLException ex) {

for (Throwable e : ex) {

if (e instanceof SQLException) {

e.printStackTrace(System.err);

System.err.println("SQLState: " + ((SQLException) e).getSQLState());

System.err.println("Error Code: " + ((SQLException) e).getErrorCode());

System.err.println("Message: " + e.getMessage());

Throwable t = ex.getCause();

while (t != null) {

System.out.println("Cause: " + t);

t = t.getCause();

}

}

}

}

}

**CustomerDao.java**

package com.dao;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import com.model.Customer;

public class CustomerDao {

private String jdbcURL = "jdbc:mysql://localhost:3306/admin?useSSL=false";

private String jdbcUsername = "admin";

private String jdbcPassword = "admin@12";

// custId firstName lastName emailId phoneNo password

public CustomerDao() {

}

protected Connection getConnection() {

Connection connection = null;

try {

Class.forName("com.mysql.jdbc.Driver");

connection = DriverManager.getConnection(jdbcURL, jdbcUsername, jdbcPassword);

System.out.println("Db details " + connection.toString());

} catch (SQLException e) {

// What to do // Debug

e.printStackTrace();

SqlException(e);

} catch (ClassNotFoundException e) {

// What to do // Debug

e.printStackTrace();

}

return connection;

}

public void insert(Customer customer) throws SQLException {

// `custId` `firstName` `lastName` `emailId` `password`

String INSERT\_SQL = "INSERT INTO customer" + " (firstName, lastName, emailId, phoneNo, password) VALUES "

+ " (?,?,?,?,?);";

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(INSERT\_SQL)) {

preparedStatement.setString(1, customer.getFirstName());

preparedStatement.setString(2, customer.getLastName());

preparedStatement.setString(3, customer.getEmailId());

preparedStatement.setString(4, customer.getPhoneNo());

preparedStatement.setString(5, customer.getPassword());

System.out.println(preparedStatement);

preparedStatement.executeUpdate();

} catch (SQLException e) {

SqlException(e);

}

}

public Customer getByKey(int custId) {

String SELECT\_BY = "select firstName, lastName, emailId, phoneNo, password from customer where custId =?";

Customer customer = null;

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(SELECT\_BY);) {

preparedStatement.setInt(1, custId);

System.out.println(preparedStatement);

ResultSet rs = preparedStatement.executeQuery();

while (rs.next()) {

//String customerId = rs.getString("custId");

String firstName= rs.getString("firstName");

String lastName = rs.getString("lastName");

String emailId = rs.getString("emailId");

String phoneNo = rs.getString("phoneNo");

String password = rs.getString("password");

customer = new Customer();

customer.setCustId(custId);

customer.setFirstName(firstName);

customer.setLastName(lastName);

customer.setEmailId(emailId);

customer.setPhoneNo(phoneNo);

customer.setPassword(password);

System.out.println(customer.toString());

}

} catch (SQLException e) {

SqlException(e);

}

return customer;

}

public Customer getByKey(String aemailId) {

String SELECT\_BY = "select custId, firstName, lastName, emailId, phoneNo, password from customer where emailId =?";

Customer customer = null;

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(SELECT\_BY);) {

preparedStatement.setString(1, aemailId);

System.out.println(preparedStatement);

ResultSet rs = preparedStatement.executeQuery();

while (rs.next()) {

int custId = rs.getInt("custId");

String firstName= rs.getString("firstName");

String lastName = rs.getString("lastName");

String emailId = rs.getString("emailId");

String phoneNo = rs.getString("phoneNo");

String password = rs.getString("password");

customer = new Customer();

customer.setCustId(custId);

customer.setFirstName(firstName);

customer.setLastName(lastName);

customer.setEmailId(emailId);

customer.setPhoneNo(phoneNo);

customer.setPassword(password);

System.out.println(customer.toString());

}

} catch (SQLException e) {

SqlException(e);

}

return customer;

}

public List<Customer> listOfAll() {

String SELECT\_ALL = "select \* from customer";

List<Customer> customers = new ArrayList<>();

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(SELECT\_ALL);) {

System.out.println(preparedStatement);

ResultSet rs = preparedStatement.executeQuery();

while (rs.next()) {

String custId = rs.getString("custId");

String firstName = rs.getString("firstName");

String lastName = rs.getString("lastName");

String emailId = rs.getString("emailId");

String phoneNo = rs.getString("phoneNo");

String password = rs.getString("password");

Customer customer = new Customer(Integer.parseInt(custId),firstName,lastName,emailId,phoneNo,password);

customers.add( customer);

}

} catch (SQLException e) {

SqlException(e);

}

return customers;

}

public boolean delete(int custId) throws SQLException {

boolean status;

String DELETE\_SQL = "delete from customer where custId = ?;";

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(DELETE\_SQL);) {

preparedStatement.setInt(1, custId);

System.out.println(preparedStatement);

status = preparedStatement.executeUpdate() > 0;

}

return status;

}

public boolean update(Customer customer) throws SQLException {

boolean status;

String UPDATE\_SQL = "update customer set firstName=?, lastName=?, emailId=?,phoneNo=?,password=? where custId = ?;";

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(UPDATE\_SQL);) {

System.out.println("updated customer :"+preparedStatement);

preparedStatement.setString(1, customer.getFirstName());

preparedStatement.setString(2, customer.getLastName());

preparedStatement.setString(3, customer.getEmailId());

preparedStatement.setString(4, customer.getPhoneNo());

preparedStatement.setString(5, customer.getPassword());

preparedStatement.setInt(6, customer.getCustId());

System.out.println(preparedStatement);

preparedStatement.executeUpdate();

status = preparedStatement.executeUpdate() > 0;

}

return status;

}

private void SqlException(SQLException ex) {

for (Throwable e : ex) {

if (e instanceof SQLException) {

e.printStackTrace(System.err);

System.err.println("SQLState: " + ((SQLException) e).getSQLState());

System.err.println("Error Code: " + ((SQLException) e).getErrorCode());

System.err.println("Message: " + e.getMessage());

Throwable t = ex.getCause();

while (t != null) {

System.out.println("Cause: " + t);

t = t.getCause();

}

}

}

}

}

**RouteDao.java**

package com.dao;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.text.DateFormat;

import java.text.SimpleDateFormat;

import java.util.ArrayList;

import java.util.Date;

import java.util.List;

import java.util.Locale;

import com.model.Route;

public class RouteDao {

private String jdbcURL = "jdbc:mysql://localhost:3306/admin?useSSL=false";

private String jdbcUsername = "admin";

private String jdbcPassword = "admin@12";

// routeId, fromCity,toCity, airline,fromDate,toDate,price;

public RouteDao() {

}

protected Connection getConnection() {

Connection connection = null;

try {

Class.forName("com.mysql.jdbc.Driver");

connection = DriverManager.getConnection(jdbcURL, jdbcUsername, jdbcPassword);

System.out.println("Db details " + connection.toString());

} catch (SQLException e) {

// What to do // Debug

e.printStackTrace();

SqlException(e);

} catch (ClassNotFoundException e) {

// What to do // Debug

e.printStackTrace();

}

return connection;

}

public void inser(Route route) throws SQLException {

String INSERT\_SQL = "INSERT INTO routes" + " (fromCity, toCity, airline,fromDate,toDate,price) VALUES "

+ " (?, ?, ?,?,?,?);";

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(INSERT\_SQL)) {

preparedStatement.setString(1, route.getFromCity());

preparedStatement.setString(2, route.getToCity());

preparedStatement.setString(3, route.getAirline());

preparedStatement.setDate(4, new java.sql.Date(route.getFromDate().getTime()));

preparedStatement.setDate(5, new java.sql.Date(route.getToDate().getTime()));

preparedStatement.setFloat(6, route.getPrice());

System.out.println(preparedStatement);

preparedStatement.executeUpdate();

} catch (SQLException e) {

SqlException(e);

}

}

public Route getByKey(int arouteId) {

String SELECT\_BY = "select fromCity, toCity, airline,fromDate,toDate,price from routes where routeId =?";

Route route = null;

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(SELECT\_BY);) {

preparedStatement.setInt(1, arouteId);

System.out.println(preparedStatement);

ResultSet rs = preparedStatement.executeQuery();

while (rs.next()) {

//String routeId = rs.getString("routeId");

String fromCity = rs.getString("fromCity");

String toCity = rs.getString("toCity");

String airline = rs.getString("airline");

Date fromDate = rs.getDate("fromDate");

Date toDate = rs.getDate("toDate");

Long price = rs.getLong("price");

route = new Route();

route.setRouteId(arouteId);

route.setFromCity(fromCity);

route.setToCity(toCity);

route.setAirline(airline);

route.setFromDate(fromDate);

route.setToDate(toDate);

route.setPrice(price);

System.out.println(route.toString());

}

} catch (SQLException e) {

SqlException(e);

}

return route;

}

public List<Route> listOfAll() {

String SELECT\_ALL = "select \* from routes";

List<Route> routes = new ArrayList<>();

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(SELECT\_ALL);) {

System.out.println(preparedStatement);

ResultSet rs = preparedStatement.executeQuery();

while (rs.next()) {

String routeId = rs.getString("routeId");

String fromCity = rs.getString("fromCity");

String toCity = rs.getString("toCity");

String airline = rs.getString("airline");

Date fromDate = rs.getDate("fromDate");

Date toDate = rs.getDate("toDate");

Long price = rs.getLong("price");

Route route = new Route(Integer.parseInt(routeId),fromCity,toCity,airline,fromDate, toDate,price);

routes.add( route);

}

} catch (SQLException e) {

SqlException(e);

}

return routes;

}

public List<Route> listOfAllBy(String fCity, String tCity, String air, Date aDate ) {

String SELECT\_ALL\_BY = " select \* from routes ";

boolean someVal = false;

if (!fCity.isEmpty()) {

SELECT\_ALL\_BY = SELECT\_ALL\_BY +" where fromCity ='" + fCity + "'";

System.out.println (SELECT\_ALL\_BY);

someVal = true;

}

if (!tCity.isEmpty()) {

if (someVal == true) {

SELECT\_ALL\_BY = SELECT\_ALL\_BY + " and toCity ='" + tCity + "'";

}else {

SELECT\_ALL\_BY = SELECT\_ALL\_BY +" where toCity ='" + tCity + "'";

someVal = true;

}

}

if (!air.isEmpty()) {

if (someVal == true) {

SELECT\_ALL\_BY = SELECT\_ALL\_BY + " and airline ='" + air + "'";

}else {

SELECT\_ALL\_BY = SELECT\_ALL\_BY +" where airline ='" + air + "'";

someVal = true;

}

}

//if (!aDate) {

//

// TODO

//}

List<Route> routes = new ArrayList<>();

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(SELECT\_ALL\_BY);) {

System.out.println(preparedStatement);

ResultSet rs = preparedStatement.executeQuery();

while (rs.next()) {

String routeId = rs.getString("routeId");

String fromCity = rs.getString("fromCity");

String toCity = rs.getString("toCity");

String airline = rs.getString("airline");

Date fromDate = rs.getDate("fromDate");

Date toDate = rs.getDate("toDate");

Long price = rs.getLong("price");

Route route = new Route(Integer.parseInt(routeId),fromCity,toCity,airline,fromDate, toDate,price);

routes.add( route);

}

} catch (SQLException e) {

SqlException(e);

}

return routes;

}

public boolean delete(int routeId) throws SQLException {

boolean status;

String DELETE\_SQL = "delete from routes where routeId = ?;";

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(DELETE\_SQL);) {

preparedStatement.setInt(1, routeId);

System.out.println(preparedStatement);

status = preparedStatement.executeUpdate() > 0;

}

return status;

}

public boolean update(Route route) throws SQLException {

boolean status;

String UPDATE\_SQL = "update routes set fromCity=?, toCity=?, airline=?,fromDate=?,toDate=?,price=? where routeId = ?;";

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(UPDATE\_SQL);) {

System.out.println("updated route :"+preparedStatement);

preparedStatement.setString(1, route.getFromCity());

preparedStatement.setString(2, route.getToCity());

preparedStatement.setString(3, route.getAirline());

preparedStatement.setDate(4, new java.sql.Date(route.getFromDate().getTime()));

preparedStatement.setDate(5, new java.sql.Date(route.getToDate().getTime()));

preparedStatement.setFloat(6, route.getPrice());

preparedStatement.setInt(7,route.getRouteId());

System.out.println(preparedStatement);

preparedStatement.executeUpdate();

status = preparedStatement.executeUpdate() > 0;

}

return status;

}

private void SqlException(SQLException ex) {

for (Throwable e : ex) {

if (e instanceof SQLException) {

e.printStackTrace(System.err);

System.err.println("SQLState: " + ((SQLException) e).getSQLState());

System.err.println("Error Code: " + ((SQLException) e).getErrorCode());

System.err.println("Message: " + e.getMessage());

Throwable t = ex.getCause();

while (t != null) {

System.out.println("Cause: " + t);

t = t.getCause();

}

}

}

}

}

**PlaceDao.java**

package com.dao;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import com.model.Airline;

import com.model.Place;

public class PlaceDao {

private String jdbcURL = "jdbc:mysql://localhost:3306/admin?useSSL=false";

private String jdbcUsername = "admin";

private String jdbcPassword = "admin@12";

// placeId, sourceCity, destinationCity;

public PlaceDao() {

}

protected Connection getConnection() {

Connection connection = null;

try {

Class.forName("com.mysql.jdbc.Driver");

connection = DriverManager.getConnection(jdbcURL, jdbcUsername, jdbcPassword);

System.out.println("Db details " + connection.toString());

} catch (SQLException e) {

// What to do // Debug

e.printStackTrace();

SqlException(e);

} catch (ClassNotFoundException e) {

// What to do // Debug

e.printStackTrace();

}

return connection;

}

public void insert(Place place) throws SQLException {

// placeId, sourceCity, destinationCity;

String INSERT\_SQL = "INSERT INTO places" + " (sourceCity,destinationCity) VALUES "

+ " (?,?);";

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(INSERT\_SQL)) {

preparedStatement.setString(1, place.getSourceCity());

preparedStatement.setString(2, place.getDestinationCity());

System.out.println(preparedStatement);

preparedStatement.executeUpdate();

} catch (SQLException e) {

SqlException(e);

}

}

public Place getByKey(int placeId) {

String SELECT\_BY = "select placeId,sourceCity,destinationCity from places where placeId =?";

Place place = null;

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(SELECT\_BY);) {

preparedStatement.setInt(1, placeId);

System.out.println(preparedStatement);

ResultSet rs = preparedStatement.executeQuery();

while (rs.next()) {

String sourceCity = rs.getString("sourceCity");

String destinationCity = rs.getString("destinationCity");

place = new Place();

place.setPlaceId(placeId);

place.setSourceCity(sourceCity);

place.setDestinationCity(destinationCity);

System.out.println(place.toString());

}

} catch (SQLException e) {

SqlException(e);

}

return place;

}

public List<Place> listOfAll() {

String SELECT\_ALL = "select \* from places";

List<Place> places = new ArrayList<>();

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(SELECT\_ALL);) {

System.out.println(preparedStatement);

ResultSet rs = preparedStatement.executeQuery();

while (rs.next()) {

int placeId = rs.getInt("placeId");

String sourceCity = rs.getString("sourceCity");

String destinationCity = rs.getString("destinationCity");

Place place = new Place();

place.setPlaceId(placeId);

place.setSourceCity(sourceCity);

place.setDestinationCity(destinationCity);

places.add( place);

}

} catch (SQLException e) {

SqlException(e);

}

return places;

}

public boolean delete(int placeId) throws SQLException {

boolean status;

String DELETE\_SQL = "delete from places where placeId = ?;";

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(DELETE\_SQL);) {

preparedStatement.setInt(1, placeId);

System.out.println(preparedStatement);

status = preparedStatement.executeUpdate() > 0;

}

return status;

}

public boolean update(Place place) throws SQLException {

boolean status;

String UPDATE\_SQL = "update places set sourceCity=?, destinationCity=? where placeId = ?;";

try (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(UPDATE\_SQL);) {

System.out.println("updated places :"+preparedStatement);

preparedStatement.setString(1, place.getSourceCity());

preparedStatement.setString(2, place.getDestinationCity());

preparedStatement.setInt(3, place.getPlaceId());

System.out.println(preparedStatement);

preparedStatement.executeUpdate();

status = preparedStatement.executeUpdate() > 0;

}

return status;

}

private void SqlException(SQLException ex) {

for (Throwable e : ex) {

if (e instanceof SQLException) {

e.printStackTrace(System.err);

System.err.println("SQLState: " + ((SQLException) e).getSQLState());

System.err.println("Error Code: " + ((SQLException) e).getErrorCode());

System.err.println("Message: " + e.getMessage());

Throwable t = ex.getCause();

while (t != null) {

System.out.println("Cause: " + t);

t = t.getCause();

}

}

}

}

}

**MainController.java**

package com.controller;

import java.io.IOException;

import java.sql.SQLException;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.List;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletConfig;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import com.dao.AdminDao;

import com.dao.AirlineDao;

import com.dao.CustomerDao;

import com.dao.PlaceDao;

import com.dao.RouteDao;

import com.model.Admin;

import com.model.Airline;

import com.model.Customer;

import com.model.Place;

import com.model.Route;

@WebServlet("/")

public class MainController extends HttpServlet {

private static final long serialVersionUID = 1L;

private RouteDao routeDao;

private CustomerDao customerDao;

private AdminDao adminDao;

private AirlineDao airlineDao;

private PlaceDao placeDao;

//Lets initialize our database object

public void init(ServletConfig config) throws ServletException {

routeDao = new RouteDao();

customerDao = new CustomerDao();

adminDao = new AdminDao();

placeDao = new PlaceDao();

airlineDao = new AirlineDao();

if (routeDao == null) {

System.out.println ("Error : Database issues - make sure you run the SQL setup");

}

}

//

// Main Handler routine for handling all events

//

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

String action = request.getServletPath();

System.out.println("RoteManagement=>doGet action:" + action);

try {

switch (action) {

case "/admin":

formtoCall(request, response,"validate-admin-form.jsp");

break;

// validate admin password.

case "/admin-process":

adminRoute(request, response);

break;

case "/changePwd":

formtoCall(request, response,"change-password-form.jsp");

break;

// change admin password.

case "/password-process":

passwordProcess(request, response);

break;

// Register Customer before dummypayment

case "/payment-process":

paymentProcess(request, response);

break;

// Insert airline Route

case "/insert":

insertRoute(request, response);

break;

// Delete airline Route

case "/delete":

deleteRoute(request, response);

break;

case "/edit":

editForm(request,response);

break;

// update airline Route

case "/update":

updateRoute(request, response);

break;

// search airline Routes

case "/search":

listRoutesBy(request, response,"index.jsp");

break;

// Final message to customer after payment

case "/success":

successProcess(request, response);

break;

// When customer decided to book a ticket

case "/book":

int routeId = Integer.parseInt(request.getParameter("routeId"));

Route aRoute = routeDao.getByKey(routeId);

System.out.println(aRoute.toString());

request.setAttribute("route", aRoute);

formtoCall(request, response,"register-user-form.jsp");

break;

case "/admin-back":

adminRefesh(request,response);

break;

//Catch all, route back to home page

default:

listRoutes(request, response,"index.jsp");

break;

}

} catch (SQLException ex) {

throw new ServletException(ex);

} catch (Exception e) {

e.printStackTrace();

}

}

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

// TODO Auto-generated method stub

doGet(request, response);

}

//

// general method to call any jsp page

//

private void formtoCall(HttpServletRequest request, HttpServletResponse response,String jspName)

throws ServletException, IOException {

RequestDispatcher dispatcher = request.getRequestDispatcher(jspName);

dispatcher.forward(request, response);

//response.sendRedirect(jspName);

}

//

// method to validate admin credential and display error

//

private void adminRoute(HttpServletRequest request, HttpServletResponse response)

throws SQLException, IOException, Exception {

// not a normal path

if(request.getParameter("userName")==null ||

request.getParameter("userName").equals("")) {

System.out.println ("Shouldnt access directly coming to admin-process page - redirect to home");

List<Route> listRoutes = routeDao.listOfAll();

request.setAttribute("routes", listRoutes);

formtoCall(request,response,"index.jsp");

}

String userName =request.getParameter("userName");

String password =request.getParameter("password");

Admin admin = adminDao.getByKey(userName);

if(password.equals(admin.getPassword())){

request.setAttribute("title", "Admin - Add Route");

adminRefesh(request,response);

// List<Airline> listAirlines = airlineDao.listOfAll();

// List<Place> listPlaces = placeDao.listOfAll();

//

// System.out.println(admin.toString());

// System.out.println ("userName:"+ userName +" password:"+ password );

//

// List<Route> listRoutes = routeDao.listOfAll();

// request.setAttribute("routes", listRoutes);

//

// request.setAttribute("airlines", listAirlines);

// request.setAttribute("places", listPlaces);

//

// formtoCall(request, response,"add-route-form.jsp");

}

//should we invalidate the session ?

request.setAttribute("error", "Please Check - Invalid user or password");

formtoCall(request, response,"validate-admin-form.jsp");

}

private void adminRefesh(HttpServletRequest request, HttpServletResponse response)

throws SQLException, IOException, Exception {

List<Airline> listAirlines = airlineDao.listOfAll();

List<Place> listPlaces = placeDao.listOfAll();

List<Route> listRoutes = routeDao.listOfAll();

request.setAttribute("routes", listRoutes);

request.setAttribute("airlines", listAirlines);

request.setAttribute("places", listPlaces);

formtoCall(request, response,"add-route-form.jsp");

}

//

// Validate and reset admin password

//

private void passwordProcess(HttpServletRequest request, HttpServletResponse response)

throws SQLException, IOException, Exception {

String userName =request.getParameter("userName");

String oPassword =request.getParameter("oldPassword");

String nPassword =request.getParameter("newPassword");

Admin admin = adminDao.getByKey(userName);

if (admin !=null && oPassword.equals(admin.getPassword())){

admin.setPassword(nPassword);

adminDao.update(admin);

request.setAttribute("info", "Password changed successfully - relogin");

formtoCall(request, response,"validate-admin-form.jsp");

System.out.println(admin.toString());

}

//should we invalidate the session ?

request.setAttribute("error", "Please Check - Invalid user or password");

formtoCall(request, response,"validate-admin-form.jsp");

}

//

// Customer Registration and process to dummypayment

//

private void paymentProcess(HttpServletRequest request, HttpServletResponse response)

throws SQLException, IOException, Exception {

Customer customer = null ;

String firstName=request.getParameter("firstName");

String lastName =request.getParameter("lastName");

String emailId =request.getParameter("emailId");

String phoneNo =request.getParameter("phoneNo");

String password =request.getParameter("password");

// avoid duplicate creation of customer if e-mail is already registered.

customer = customerDao.getByKey(emailId);

if (customer == null) {

customer = new Customer();

customer.setFirstName(firstName);

customer.setLastName(lastName);

customer.setEmailId(emailId);

customer.setPhoneNo(phoneNo);

customer.setPassword(password);

customerDao.insert(customer);

}

System.out.println(customer.toString());

Route route = new Route();

int routeId = Integer.parseInt(request.getParameter("routeId"));

String fromCity = request.getParameter("fromCity");

String toCity = request.getParameter("toCity");

String airline = request.getParameter("airline");

Long price = Long.parseLong(request.getParameter("price"));

route.setRouteId(routeId);

route.setFromCity(fromCity);

route.setToCity(toCity);

route.setAirline(airline);

route.setPrice(price);

System.out.println(route.toString());

request.setAttribute("route", route);

request.setAttribute("customer", customer);

formtoCall(request, response,"dummypayment.jsp");

}

//

// success-process(request, response,"success.jsp");

//

private void successProcess(HttpServletRequest request, HttpServletResponse response)

throws SQLException, IOException, Exception {

Customer customer = null;

Route route = null;

String emailId =request.getParameter("emailId");

// avoid duplicate creation of customer if e-mail is already registered.

customer = customerDao.getByKey(emailId);

if (customer == null) {

System.out.println( "Email id is empty - problem: " + emailId);

}

System.out.println(customer.toString());

int routeId = Integer.parseInt(request.getParameter("routeId"));

route = routeDao.getByKey(routeId);

if (route == null) {

System.out.println( "Route id is empty - problem: " + routeId);

}

System.out.println(route.toString());

request.setAttribute("route", route);

request.setAttribute("customer", customer);

formtoCall(request, response,"success.jsp");

}

//

// List all the routes

//

private void listRoutes(HttpServletRequest request, HttpServletResponse response,String jspToList)

throws SQLException, IOException, ServletException {

System.out.println ("in List Route");

if (routeDao == null ){

System.out.println ("routeDao is null");

routeDao = new RouteDao();

}

List<Route> listRoutes = routeDao.listOfAll();

request.setAttribute("routes", listRoutes);

RequestDispatcher dispatcher = request.getRequestDispatcher(jspToList);

dispatcher.forward(request, response);

//response.sendRedirect("user-list.jsp");

}

private void listRoutesBy(HttpServletRequest request, HttpServletResponse response,String jspToList)

throws SQLException, IOException, ServletException, ParseException {

SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd");

String fromCity = request.getParameter("fromCity");

String toCity = request.getParameter("toCity");

String airline = request.getParameter("airline");

Date fromDate = null;

System.out.println (request.getParameter("travelDate"));

if (!request.getParameter("travelDate").isEmpty() ) {

fromDate = dateFormat.parse(request.getParameter("travelDate"));

}

if (routeDao == null ){

System.out.println ("routeDao is null");

routeDao = new RouteDao();

}

List<Route> listRoutes = routeDao.listOfAllBy(fromCity,toCity,airline,fromDate);

request.setAttribute("routes", listRoutes);

if (listRoutes.size() == 0) {

request.setAttribute("warn", "No data found , admin to create or just press search");

}

RequestDispatcher dispatcher = request.getRequestDispatcher(jspToList);

dispatcher.forward(request, response);

//response.sendRedirect("user-list.jsp");

}

//

// Admin main edit route form

//

private void editForm(HttpServletRequest request, HttpServletResponse response)

throws SQLException, ServletException, IOException {

int routeId = Integer.parseInt(request.getParameter("routeId"));

if (routeDao == null ){

System.out.println ("routeDao is null");

routeDao = new RouteDao();

}

Route aRoute = routeDao.getByKey(routeId);

System.out.println(aRoute.toString());

RequestDispatcher dispatcher = request.getRequestDispatcher("update-route-form.jsp");

request.setAttribute("route", aRoute);

dispatcher.forward(request, response);

}

//

// insertRoute

//

private void insertRoute(HttpServletRequest request, HttpServletResponse response)

throws SQLException, IOException, Exception {

// not a normal path - lets take one parm to check

if(request.getParameter("price")==null ||

request.getParameter("price").equals("")) {

System.out.println ("Shouldnt access directly coming to admin-process page - redirect to home");

List<Route> listRoutes = routeDao.listOfAll();

request.setAttribute("routes", listRoutes);

formtoCall(request,response,"index.jsp");

}

SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd");

// String fromCity = request.getParameter("fromCity");

// String toCity = request.getParameter("toCity");

Date fromDate = dateFormat.parse(request.getParameter("fromDate"));

Date toDate = dateFormat.parse(request.getParameter("toDate"));

Long price = Long.parseLong(request.getParameter("price"));

Airline airline = airlineDao.getByKey(Integer.parseInt(request.getParameter("airlineId")));

String code = airline.getCode();

Place place = placeDao.getByKey(Integer.parseInt(request.getParameter("placeId")));

String sCity = place.getSourceCity();

String tCity = place.getDestinationCity();

Route art = new Route();

// art.setFromCity(fromCity);

// art.setToCity(toCity);

art.setFromCity(sCity);

art.setToCity(tCity);

art.setAirline(code);

art.setFromDate(fromDate);

art.setToDate(toDate);

art.setPrice(price);

System.out.println(art.toString());

System.out.println(place.toString());

System.out.println(airline.toString());

// Route aroute = new Route(fromCity,toCity,airline,capacity,fromDate, toDate,price);

//routeDao.insertUser(aroute);

if (routeDao == null ){

System.out.println ("routeDao is null");

routeDao = new RouteDao();

}

routeDao.inser(art);

List<Route> listRoutes = routeDao.listOfAll();

request.setAttribute("routes", listRoutes);

List<Airline> listAirlines = airlineDao.listOfAll();

request.setAttribute("airlines", listAirlines);

List<Place> listPlaces = placeDao.listOfAll();

request.setAttribute("places", listPlaces);

formtoCall(request, response,"add-route-form.jsp");

//response.sendRedirect("list");

}

//

// Update Route

//

private void updateRoute(HttpServletRequest request, HttpServletResponse response)

throws SQLException, IOException, Exception {

// not a normal path - lets take one parm to check

if(request.getParameter("price")==null ||

request.getParameter("price").equals("")) {

System.out.println ("Shouldnt access directly coming to admin-process page - redirect to home");

List<Route> listRoutes = routeDao.listOfAll();

request.setAttribute("routes", listRoutes);

formtoCall(request,response,"index.jsp");

}

System.out.println (request.getParameter("routeId"));

int routeid = Integer.parseInt(request.getParameter("routeId"));

SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd");

String fromCity = request.getParameter("fromCity");

String toCity = request.getParameter("toCity");

String airline = request.getParameter("airline");

Date fromDate = dateFormat.parse(request.getParameter("fromDate"));

Date toDate = dateFormat.parse(request.getParameter("toDate"));

Long price = Long.parseLong(request.getParameter("price"));

Route art = new Route();

art.setRouteId(routeid);

art.setFromCity(fromCity);

art.setToCity(toCity);

art.setAirline(airline);

art.setFromDate(fromDate);

art.setToDate(toDate);

art.setPrice(price);

System.out.println(art.toString());

//Route aroute = new Route(routeid,fromCity,toCity,airline,capacity,fromDate, toDate,price);

routeDao.update(art);

List<Route> listRoutes = routeDao.listOfAll();

request.setAttribute("routes", listRoutes);

formtoCall(request, response,"add-route-form.jsp");

}

//

// Delete route

//

private void deleteRoute(HttpServletRequest request, HttpServletResponse response)

throws Exception {

int routeId = Integer.parseInt(request.getParameter("routeId"));

if (routeDao == null ){

System.out.println ("routeDao is null");

routeDao = new RouteDao();

}

routeDao.delete(routeId);

adminRefesh(request,response);

//response.sendRedirect("list");

}

}

**Airline.java**

package com.model;

public class Airline {

private int airlineId;

private String airlineName;

private String code;

private int capacity;

public Airline() {

}

public Airline(int airlineId, String airlineName, String code, int capacity) {

super();

this.airlineId = airlineId;

this.airlineName = airlineName;

this.code = code;

this.capacity = capacity;

}

public Airline(String airlineName, String code, int capacity) {

super();

this.airlineName = airlineName;

this.code = code;

this.capacity = capacity;

}

public int getAirlineId() {

return airlineId;

}

public void setAirlineId(int airlineId) {

this.airlineId = airlineId;

}

public String getAirlineName() {

return airlineName;

}

public void setAirlineName(String airlineName) {

this.airlineName = airlineName;

}

public String getCode() {

return code;

}

public void setCode(String code) {

this.code = code;

}

public int getCapacity() {

return capacity;

}

public void setCapacity(int capacity) {

this.capacity = capacity;

}

@Override

public String toString() {

return "Airlines [airlineId=" + airlineId + ", airlineName=" + airlineName + ", code=" + code + ", capacity="

+ capacity + "]";

}

}

**Route,java**

package com.model;

public class Airline {

private int airlineId;

private String airlineName;

private String code;

private int capacity;

public Airline() {

}

public Airline(int airlineId, String airlineName, String code, int capacity) {

super();

this.airlineId = airlineId;

this.airlineName = airlineName;

this.code = code;

this.capacity = capacity;

}

public Airline(String airlineName, String code, int capacity) {

super();

this.airlineName = airlineName;

this.code = code;

this.capacity = capacity;

}

public int getAirlineId() {

return airlineId;

}

public void setAirlineId(int airlineId) {

this.airlineId = airlineId;

}

public String getAirlineName() {

return airlineName;

}

public void setAirlineName(String airlineName) {

this.airlineName = airlineName;

}

public String getCode() {

return code;

}

public void setCode(String code) {

this.code = code;

}

public int getCapacity() {

return capacity;

}

public void setCapacity(int capacity) {

this.capacity = capacity;

}

@Override

public String toString() {

return "Airlines [airlineId=" + airlineId + ", airlineName=" + airlineName + ", code=" + code + ", capacity="

+ capacity + "]";

}

}

**Place.java**

package com.model;

public class Place {

private int placeId;

private String sourceCity;

private String destinationCity;

public Place() {

}

public Place(int placeId, String sourceCity, String destinationCity) {

super();

this.placeId = placeId;

this.sourceCity = sourceCity;

this.destinationCity = destinationCity;

}

public Place(String sourceCity, String destinationCity) {

super();

this.sourceCity = sourceCity;

this.destinationCity = destinationCity;

}

public int getPlaceId() {

return placeId;

}

public void setPlaceId(int placeId) {

this.placeId = placeId;

}

public String getSourceCity() {

return sourceCity;

}

public void setSourceCity(String sourceCity) {

this.sourceCity = sourceCity;

}

public String getDestinationCity() {

return destinationCity;

}

public void setDestinationCity(String destinationCity) {

this.destinationCity = destinationCity;

}

@Override

public String toString() {

return "Places [placeId=" + placeId + ", sourceCity=" + sourceCity + ", destinationCity=" + destinationCity

+ "]";

}

}

**Admin.java**

package com.model;

public class Admin {

private String userName;

private String password;

public Admin() {

}

public Admin(String userName, String password) {

super();

this.userName = userName;

this.password = password;

}

public Admin(String password) {

super();

this.password = password;

}

public String getUserName() {

return userName;

}

public void setUserName(String userName) {

this.userName = userName;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

@Override

public String toString() {

return "Admin [userName=" + userName + ", password=" + password + "]";

}

}

**Customer.java**

package com.model;

public class Customer {

private int custId;

private String firstName;

private String lastName;

private String emailId;

private String phoneNo;

private String password;

public Customer() {

}

public Customer(int custId, String firstName, String lastName, String emailId, String phoneNo, String password) {

super();

this.custId = custId;

this.firstName = firstName;

this.lastName = lastName;

this.emailId = emailId;

this.phoneNo = phoneNo;

this.password = password;

}

public Customer(String firstName, String lastName, String emailId, String phoneNo, String password) {

super();

this.firstName = firstName;

this.lastName = lastName;

this.emailId = emailId;

this.phoneNo = phoneNo;

this.password = password;

}

public int getCustId() {

return custId;

}

public void setCustId(int custId) {

this.custId = custId;

}

public String getFirstName() {

return firstName;

}

public void setFirstName(String firstName) {

this.firstName = firstName;

}

public String getLastName() {

return lastName;

}

public void setLastName(String lastName) {

this.lastName = lastName;

}

public String getEmailId() {

return emailId;

}

public void setEmailId(String emailId) {

this.emailId = emailId;

}

public String getPhoneNo() {

return phoneNo;

}

public void setPhoneNo(String phoneNo) {

this.phoneNo = phoneNo;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

@Override

public String toString() {

return "Customer [custId=" + custId + ", firstName=" + firstName + ", lastName=" + lastName + ", emailId="

+ emailId + ", phoneNo=" + phoneNo + ", password=" + password + "]";

}

}

**Pom.xml**

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>final2</groupId>

<artifactId>final2</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>war</packaging>

<properties>

<maven.compiler.source>11</maven.compiler.source>

<maven.compiler.target>11</maven.compiler.target>

</properties>

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>3.8.1</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>javax.servlet-api</artifactId>

<version>3.1.0</version>

<scope>provided</scope>

</dependency>

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>javax.servlet-api</artifactId>

<version>3.1.0</version>

<scope>provided</scope>

</dependency>

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>jstl</artifactId>

<version>1.2</version>

</dependency>

<dependency>

<groupId>taglibs</groupId>

<artifactId>standard</artifactId>

<version>1.1.2</version>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.27</version>

<!-- <version>8.0.16</version> -->

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.8.1</version>

<configuration>

<release>11</release>

</configuration>

</plugin>

<plugin>

<artifactId>maven-war-plugin</artifactId>

<version>3.2.3</version>

</plugin>

</plugins>

</build>

</project>

**JAVA SERVER PAGES**

**Commonstyles.jsp**

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

<%@ taglib prefix = "c" uri = "http://java.sun.com/jsp/jstl/core" %>

<%@page isELIgnored="false" %>

<!DOCTYPE *html*>

<html>

<style>

    h1  {text-align: center;}

    h2  {text-align: center;}

    h3  {text-align: right; STYLE="background-color:red;"}

    h4  {text-align: center;STYLE="color:Tomato;"}

    p   {text-align: center;}

    div {text-align: center;}

    form{text-align: center;}

    table {

        font-family: arial, sans-serif;

        border-collapse: collapse;

        width: 100%;

    }

    td, th {

        border: 1px solid #dddddd;

        text-align: left;

        padding: 8px;

    }

    tr:nth-child(even) {

    background-color: #dddddd;

    }

</style>

<title><c:out *value*="${title}">Flyaway Home</c:out></title>

<h3 *STYLE*="background-color:red;"> <c:out *value*="${error}"></c:out> </h3>

<h3 *STYLE*="background-color:lightblue;"> <c:out *value*="${info}"></c:out> </h3>

<h3 *STYLE*="background-color:yellow;"> <c:out *value*="${warn}"></c:out> </h3>

<body>

 <body *style*="background-color:lightblue;">

</body>

</html>

**validate-admin-form.jsp**

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

<!DOCTYPE *html*>

<html>

<head>

<meta *charset*="ISO-8859-1">

<%@include file="./commonstyles.jsp" %>

</head>

<body>

<h1>Flyaway - Your well deserved vacation a click away</h1>

<h2>Admin Maintanance module</h2>

    <div>

        <form *action*="${pageContext.request.contextPath }/admin-process" *method* ="post">

            User Name<input *type*="text" *id*="userName"*name*="userName" *placeholder*="admin" *required*><br>

            Password &nbsp;&nbsp;<input *type*="password" *id*="password" *name*="password" *placeholder*="admin" *required*><br><br>

            <button *type*="submit">Login</button> &nbsp;

            <a *href*="${pageContext.request.contextPath}/changePwd">Change Password </a> &nbsp;&nbsp;

            <a *href*="${pageContext.request.contextPath}/">Cancel</a> <br>

        </form>

    </div>

</body>

</html>

**add-route-form.jsp**

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

<!DOCTYPE *html*>

<html>

<head>

<meta *charset*="ISO-8859-1">

<%@include file="./commonstyles.jsp" %>

</head>

<body>

<h1>Flyaway - Your well deserved vacation a click away</h1>

<h2>Admin Maintenance module</h2>

<div>

        <form *action*="insert" *method* ="post" >

            <select *name*="placeId">

                <c:forEach *items*="${places}" *var*="place">

                    <option *value*="${place.placeId}">${place.sourceCity}</option>

                </c:forEach>

            </select>

             <select *name*="placeId">

                <c:forEach *items*="${places}" *var*="place">

                    <option *value*="${place.placeId}">${place.destinationCity}</option>

                </c:forEach>

            </select>

*<!--        <input type="text" id="fromCity"name="fromCity" placeholder="From City" required>*

*<input type="text" id="toCity"name="toCity" placeholder="To City" required>*

*-->*

            <select *name*="airlineId">

                <c:forEach *items*="${airlines}" *var*="airline">

                    <option *value*="${airline.airlineId}">${airline.code}</option>

                </c:forEach>

            </select>

*<!-- <input type="text" id="airline"name="airline" placeholder="Airline" required> -->*

            <input *type*="date" *id*="fromDate"*name*="fromDate" *placeholder*="From Date" *required*>

            <input *type*="date" *id*="toDate"*name*="toDate" *placeholder*="To Date" *required*>

            <input *type*="text" *id*="price" *name*="price" *placeholder*="price" *required*><br>

            <button *type*="submit">Add</button> &nbsp;&nbsp;&nbsp;

<%--            <a *href*="${pageContext.request.contextPath}/airlines">Airlines</a> &nbsp;

            <a *href*="${pageContext.request.contextPath}/places">Places</a>  --%>

            <a *href*="${pageContext.request.contextPath}/">Home</a> <br>

        </form>

        <div>

        <form *action*="" *method*="post" >

        <table>

          <tr>

            <th>Route Id</th>

            <th>From City</th>

            <th>To City</th>

            <th>Airline</th>

            <th>From Date </th>

            <th>To Date </th>

            <th>Price</th>

            <th>Action</th>

          </tr>

          <c:forEach *items*="${routes}" *var*="route">

            <tr>

              <td>${route.routeId}</td>

              <td>${route.fromCity}</td>

              <td>${route.toCity}</td>

              <td>${route.airline}</td>

              <td>${route.fromDate}</td>

              <td>${route.toDate}</td>

              <td>&#8377;${route.price}</td>

              <td><a *href*="edit?routeId=<c:out   value='${route.routeId}'/>">Edit</a></td>

              <td><a *href*="delete?routeId=<c:out value='${route.routeId}'/>">Delete</a></td>

            </tr>

          </c:forEach>

        </table>

        </form>

        </div>

    </div>

</body>

</html>

**Success.jsp**

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

<!DOCTYPE *html*>

<html>

<head>

<meta *charset*="ISO-8859-1">

<%@include file="./commonstyles.jsp" %>

</head>

<body>

<h1>Flyaway - Your well deserved vacation a click away</h1>

<h2>Registration</h2>

    <div>

        <form>

            <h4 *style*="color:Tomato;"> Congratulations Booking confirmed  </h4><br><br>

            <h4> Hello ${customer.firstName} ${customer.lastName}  </h4><br>

            <h4 *style*="color:Tomato;"> Your Flight Details <br>

                From ${routes.fromCity}<br>

                  To ${routes.toCity}<br>

                  by ${routes.airline}v

                  cost ${routes .price}<br><br>

             </h4>

            <a *href*="${pageContext.request.contextPath}/">Home</a> <br>

        </form>

    </div>

</body>

</html>

**change-password-form.jsp**

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

<!DOCTYPE *html*>

<html>

<head>

<meta *charset*="ISO-8859-1">

<%@include file="./commonstyles.jsp" %>

</head>

<body>

<h1>Flyaway - Your well deserved vacation a click away</h1>

<h2>Admin Maintanance module</h2>

    <div>

        <form *action*="${pageContext.request.contextPath }/password-process" *method* ="post">

            UserId &nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp

            <input *type*="text" *id*="userName"*name*="userName" *placeholder*="User ID" *required*><br>

            Current Password <input *type*="password" *id*="oldPassword"*name*="oldPassword" *placeholder*="Current Password" *required*><br>

            New Password &nbsp&nbsp&nbsp&nbsp<input *type*="password" *id*="newPassword"*name*="newPassword" *placeholder*="New Password" *required*><br>

            <button *type*="submit">Change</button> &nbsp;

            <a *href*="${pageContext.request.contextPath}/">Cancel</a> <br>

        </form>

    </div>

</body>

</html>

**Dummypayment.jsp**

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

<!DOCTYPE *html*>

<html>

<head>

<meta *charset*="ISO-8859-1">

<%@include file="./commonstyles.jsp" %>

</head>

<body>

<h1>FlyAway - Your well deserved vacation a click away</h1>

<h2>Registration</h2>

    <div>

        <form *action*="success" *method* ="post">

            <h4> Hello ${customer.firstName} ${customer.lastName}  </h4><br>

            <h4 *style*="color:Tomato;"> Your Flight Details <br>

                From ${route.fromCity}

                  To ${route.toCity}

                  by ${route.airline}

                  cost ${route.price}<br><br>

             </h4>

*<!--  pass key values for final confirmation screen -->*

            <input *type*="hidden" *id*="emailId"   *name*="emailId"  *value*="${customer.emailId}">

            <input *type*="hidden" *id*="routeId"   *name*="routeId"  *value*="${route.routeId}">

            <h4> Payment Details </h4>

            Full Name   <input *type*="text" *id*="fullName"*name*="fullName" *placeholder*="Full Name" *required*><br>

            Card Number <input *type*="text" *id*="cardNumber"*name*="cardNumber" *placeholder*="1234-456-789" *required*><br>

            Expiry Date <input *type*="text" *id*="expiryDate"*name*="expiryDate" *placeholder*="mm-yyyy" *required*><br>

            CVV number  <input *type*="number" *id*="cvvNo"*name*="cvvNo" *placeholder*="123" *required*><br>

            <button *type*="submit"> Pay </button> &nbsp;

            <a *href*="${pageContext.request.contextPath}/">Cancel</a> <br>

        </form>

    </div>

</body>

</html>

**Index.jsp**

<html>

<%@include file="./commonstyles.jsp" %>

<body>

    <h1>Flyaway - Your well deserved vacation a click away</h1>

    <div>

        <a *href*="${pageContext.request.contextPath}/admin">Admin</a>

    </div>

    <br>

    <br>

    <div>

        <form *action*="search" *method* ="post">

            <input *type*="text" *id*="fromCity"*name*="fromCity"     *placeholder*="From City">

            <input *type*="text" *id*="toCity"  *name*="toCity"       *placeholder*="To City">

            <input *type*="text" *id*="airline" *name*="airline"      *placeholder*="Airline">

            <input *type*="date" *id*="travelDate"*name*="travelDate" *placeholder*="Travel Date">

            <button *type*="submit">Search</button> &nbsp;

        </form>

    </div>

    <div>

        <form *action*="" *method*="post" >

        <table>

          <tr>

            <th>Route Id</th>

            <th>From City</th>

            <th>To City</th>

            <th>Airline</th>

              <th>Price</th>

            <th>Action</th>

          </tr>

          <c:forEach *items*="${routes}" *var*="route">

            <tr>

              <td>${route.routeId}</td>

              <td>${route.fromCity}</td>

              <td>${route.toCity}</td>

              <td>${route.airline}</td>

              <td>&#8377;${route.price}</td>

              <td><a *href*="book?routeId=<c:out   value='${route.routeId}'/>">Book</a></td>

            </tr>

          </c:forEach>

        </table>

        </form>

    </div>

<body>

</html>

**update-route-form.jsp**

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

<!DOCTYPE *html*>

<html>

<head>

<meta *charset*="ISO-8859-1">

<%@include file="./commonstyles.jsp" %>

</head>

<body>

<h1>Flyaway - Your well deserved vacation a click away</h1>

<h2>Admin Maintanance module</h2>

        <form *action*="update" *method*="post">

            Route Id :

            <input *type*="text" *id*="routeId" *name*="routeId"  *value*="${route.routeId}"  *readonly*><br>

            <input *type*="text" *id*="fromCity"*name*="fromCity" *value*="${route.fromCity}"*placeholder*="From City" *required*><br>

            <input *type*="text" *id*="toCity"  *name*="toCity"   *value*="${route.toCity}"  *placeholder*="To City"   *required*><br>

            <input *type*="text" *id*="airline" *name*="airline"  *value*="${route.airline}" *placeholder*="Airline"   *required*><br>

            <input *type*="date" *id*="fromDate"*name*="fromDate" *value*="${route.fromDate}"*placeholder*="From Date" *required*><br>

            <input *type*="date" *id*="toDate"  *name*="toDate"   *value*="${route.toDate}"  *placeholder*="To Date" *required*><br>

            <input *type*="text" *id*="airline" *name*="price"    *value*="${route.price}"   *placeholder*="Price" *required*><br><br>

            <button *type*="submit">Update</button> &nbsp;

            <a *href*="${pageContext.request.contextPath}/admin-back">Back</a> <br>

        </form>

</body>

</html>

**register-user-form.jsp**

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

<!DOCTYPE *html*>

<html>

<head>

<meta *charset*="ISO-8859-1">

<%@include file="./commonstyles.jsp" %>

</head>

<body>

<h1>Flyaway - Your well deserved vacation a click away</h1>

<h2>Registration</h2>

    <div>

        <form *action*="${pageContext.request.contextPath }/payment-process" *method* ="post">

            <input *type*="hidden" *id*="routeId"*name*="routeId" *value*="${route.routeId}" *STYLE*="color:lightblue;" >

            <input *type*="hidden" *id*="fromCity"*name*="fromCity" *value*="${route.fromCity}" *STYLE*="color:lightblue;">

            <input *type*="hidden" *id*="toCity"    *name*="toCity" *value*="${route.toCity}" *STYLE*="color:lightblue;" >

            <input *type*="hidden" *id*="airline" *name*="airline" *value*="${route.airline}" *STYLE*="color:lightblue;" >

            <input *type*="hidden" *id*="price" *name*="price" *value*="${route.price}" *STYLE*="color:lightblue;"><br><br>

            <h4 *style*="color:Tomato;"> Your Flight Details <br>

                From ${route.fromCity}

                  To ${route.toCity}

                  by ${route.airline}

                  cost ${route.price}<br><br>

             </h4>

            <h4> To Book Your Flight - Please register below </h4>

            First Name<input *type*="text" *id*="firstName"*name*="firstName" *placeholder*="First Name" *required*><br>

            Last Name <input *type*="text" *id*="lastName"*name*="lastName" *placeholder*="Last Name" *required*><br>

            e-mail Id <input *type*="email" *id*="emailId"*name*="emailId" *placeholder*="example@gmail.com" *required*><br>

            Phone No  <input *type*="text" *id*="phoneNo"*name*="phoneNo" *placeholder*="Phone No" *required*><br>

            Password  <input *type*="password" *id*="password" *name*="password" *placeholder*="password" *required*><br><br>

            <button *type*="submit">Register & Pay </button> &nbsp;

            <a *href*="${pageContext.request.contextPath}/">Cancel</a> <br>

        </form>

    </div>

</body>

</html>