1. **BACKEND IN POSTGRES**

CODE :

CREATE TABLE FLIGHT(

FLIGHT\_ID VARCHAR(20) PRIMARY KEY ,

FLIGHT\_NAME VARCHAR(50) NOT NULL,

FROM\_CITY VARCHAR(50),

TO\_CITY VARCHAR(50),

CUST\_ID VARCHAR(20),

SEAT\_NO VARCHAR(10) NOT NULL,

CONSTRAINT FK\_FLIGHT\_CUSTOMER FOREIGN KEY(CUST\_ID) REFERENCES CUSTOMER(CUST\_ID)

);

CREATE TABLE CUSTOMER(

CUST\_ID VARCHAR(20) PRIMARY KEY,

CUST\_FIRSTNAME VARCHAR(50) NOT NULL,

CUST\_LASTNAME VARCHAR(50) NOT NULL,

CUST\_PHONE VARCHAR(10)

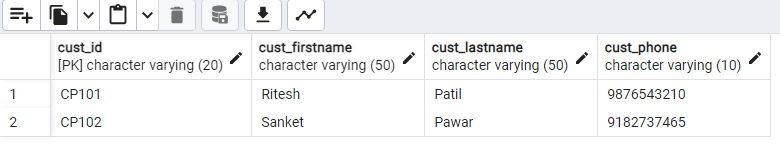
);

\*\*\*\*VALUES FOR CUSTOMER TABLE\*\*\*\*

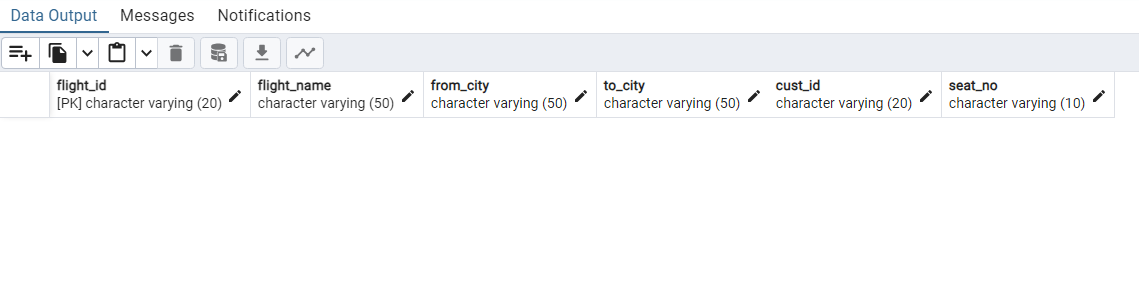
INSERT INTO CUSTOMER VALUES

('CP101', 'Ritesh', 'Patil', '9876543210'),

('CP102', 'Sanket', 'Pawar', '9182737465');



\*\*\*\*FLIGHT TABLE IN EMPTY INITIALLY\*\*\*\*



1. **Appropriate Interfaces and Classes**
2. Flight.java => POJO representing the Flight entity (Flight table in Database)
3. *@Data*
4. *@AllArgsConstructor*
5. *@NoArgsConstructor*
6. public class Flight {
8. private String FlightId;
9. private String FlightName;
10. private String FromCity;
11. private String ToCity;
12. private String Cust\_ID;
13. private String SeatNo;
15. }

b. FlightMapper.java => Implements RowMapper interface to convert a table row into Java object

public class FlightMapper implements RowMapper<Flight>{

*@Override*

public Flight mapRow(ResultSet rs, int rowNum) throws SQLException {

Flight flight = new Flight();

flight.setFlightId(rs.getString("FLIGHT\_ID"));

flight.setFlightName(rs.getString("FLIGHT\_NAME"));

flight.setFromCity(rs.getString("FROM\_CITY"));

flight.setToCity(rs.getString("TO\_CITY"));

flight.setCust\_ID(rs.getString("CUST\_ID"));

flight.setSeatNo(rs.getString("SEAT\_NO"));

return flight;

}

}

c. Customer.java

*@Data*

*@AllArgsConstructor*

*@NoArgsConstructor*

public class Customer {

private String CustId;

private String FirstName;

private String LastName;

private String Phone;

}

d. CustomerMapper.java

public class CustomerMapper implements RowMapper<Customer>{

*@Override*

public Customer mapRow(ResultSet rs, int rowNum) throws SQLException {

Customer customer = new Customer();

customer.setCustId(rs.getString("CUST\_ID"));

customer.setFirstName(rs.getString("CUST\_FIRSTNAME"));

customer.setLastName(rs.getString("CUST\_LASTNAME"));

customer.setPhone(rs.getString("CUST\_PHONE"));

return customer;

}

e. CustomerDAO.java

public interface CustomerDAO {

public List<Customer> displayCustomers();

public Customer getCustomerFromFlightID(String Id);

public boolean update(Customer customer);

}

f. CustomerDAOImpl.java

*@Component*

public class CustomerDAOImpl implements CustomerDAO{

*@Autowired*

private JdbcTemplate jdbc;

private final String SQL\_GET\_CUSTOMER\_FROM\_FLIGT = "SELECT CUST\_ID, CUST\_FIRSTNAME, CUST\_LASTNAME, CUST\_PHONE FROM CUSTOMER JOIN FLIGHT USING(CUST\_ID) WHERE FLIGHT\_ID = ?";

private final String SQL\_UPDATE\_CUSTOMER = "UPDATE CUSTOMER SET CUST\_PHONE = ? WHERE CUST\_ID = ?";

*@Override*

public List<Customer> displayCustomers() {

return jdbc.query("SELECT \* FROM CUSTOMER", new CustomerMapper());

}

*@SuppressWarnings*("deprecation")

*@Override*

public Customer getCustomerFromFlightID(String Id) {

return jdbc.~~queryForObject~~(SQL\_GET\_CUSTOMER\_FROM\_FLIGT, new Object[] {Id}, new CustomerMapper());

}

*@Override*

public boolean update(Customer customer) {

return jdbc.update(SQL\_UPDATE\_CUSTOMER, customer.getPhone(), customer.getCustId()) > 0;

}

}

g. FlightDAO.java

public interface FlightDAO {

public boolean bookFlight(Flight flight);

public List<Flight> displayFlightsForDestination(String destination);

public boolean cancelFlight(String id);

public List<Flight> displayFlightForCustomer(String cid);

}

h.FlightDAOImpl.java

*@Component*

public class FlightDAOImpl implements FlightDAO{

*@Autowired*

private JdbcTemplate jdbc;

private final String SQL\_BOOK\_NEW\_FLIGHT = "INSERT INTO FLIGHT VALUES (?, ?, ? , ?, ?, ?)";

private final String SQL\_FLIGHTS\_FOR\_DESTINATION = "SELECT \* FROM FLIGHT WHERE TO\_CITY=?";

private final String SQL\_CANCEL\_FLIGHT = "DELETE FROM FLIGHT WHERE FLIGHT\_ID=?";

private final String SQL\_FIND\_FLIGHT = "SELECT \* FROM FLIGHT WHERE FLIGHT\_ID = ?";

private final String SQL\_FLIGHTS\_FOR\_CUSTOMER = "SELECT \* FROM FLIGHT WHERE CUST\_ID = ?";

*@Override*

public boolean bookFlight(Flight flight) {

return jdbc.update(SQL\_BOOK\_NEW\_FLIGHT, flight.getFlightId(), flight.getFlightName(), flight.getFromCity(), flight.getToCity(), flight.getCust\_ID(), flight.getSeatNo()) > 0;

}

*@Override*

public List<Flight> displayFlightsForDestination(String destination) {

return jdbc.~~query~~(SQL\_FLIGHTS\_FOR\_DESTINATION, new Object[] {destination}, new FlightMapper());

}

*@Override*

public boolean cancelFlight(String id) {

return jdbc.update(SQL\_CANCEL\_FLIGHT, id) > 0;

}

*@Override*

public List<Flight> displayFlightForCustomer(String cid) {

return jdbc.~~query~~(SQL\_FLIGHTS\_FOR\_CUSTOMER, new Object[] {cid}, new FlightMapper());

}

}

1. JDBCConfig.java => Handle connectivity with the Database. Using DriverManagerDatasource to establish connection with database and JDBCTemplate to perform operations
2. *@Configuration*
3. *@ComponentScan*("com.infosys.capstone")
4. *@PropertySource*("database.properties")
5. public class JDBCConfig {
6. *@Autowired*
7. private Environment env;
8. private final String URL = "url";
9. private final String USER = "dbuser";
10. private final String PASS = "dbpass";
11. private final String DRIVER = "driver";
12. *@Bean*
13. DataSource dataSource() {
14. DriverManagerDataSource datasource = new DriverManagerDataSource();
15. datasource.setUrl(env.getProperty(URL));
16. datasource.setUsername(env.getProperty(USER));
17. datasource.setPassword(env.getProperty(PASS));
18. datasource.setDriverClassName(env.getProperty(DRIVER));
19. return datasource;
20. }
21. *@Bean*
22. public JdbcTemplate myJdbc(DataSource ds) {
23. return new JdbcTemplate(ds);
24. }
25. }

j. CapstoneApplication.java (MAIN CLASS)

*@SpringBootApplication*

public class CapstoneApplication implements CommandLineRunner {

public static void main(String[] args) {

SpringApplication.*run*(CapstoneApplication.class, args);

}

*@Override*

public void run(String... args) throws Exception {

ApplicationContext context = new AnnotationConfigApplicationContext(JDBCConfig.class);

// Access the Data Access Beans

CustomerDAO customer = context.getBean(CustomerDAOImpl.class);

FlightDAO flight = context.getBean(FlightDAOImpl.class);

Scanner sc = new Scanner(System.***in***);

Integer choice = 0;

do {

System.***out***.println("================MENU FOR INFYGO BOOKING===================");

System.***out***.println("\n1. BOOK A FLIGHT");

System.***out***.println("2. UPDATE CUSTOMER DETAILS FOR A FLIGHT ");

System.***out***.println("3. DISPLAY FLIGHTS TO A CITY");

System.***out***.println("4. CANCEL FLIGHT");

System.***out***.println("5. DISPLAY YOUR FLIGHTS");

System.***out***.println("6. EXIT");

System.***out***.println("\nEnter your choice :");

choice = sc.nextInt();

sc.nextLine();

switch (choice) {

case 1:

System.***out***.println("Enter flight name : ");

String flightName = sc.nextLine();

System.***out***.println("Enter Starting city : ");

String source = sc.nextLine();

System.***out***.println("Enter Destination city : ");

String destination = sc.nextLine();

System.***out***.println("Enter your Customer Id : ");

String CustId = sc.nextLine();

System.***out***.println("Enter the seat no : ");

String seatNo = sc.nextLine();

//System generated Flight ID

String FID = CustId + source.charAt(0) + destination.charAt(0) + seatNo;

Flight myFlight = new Flight(FID, flightName, source, destination, CustId, seatNo);

if (flight.bookFlight(myFlight)) {

System.***out***.println("Flight booked successfully");

}

break;

case 2:

Customer myCustomer = new Customer();

System.***out***.println("Enter your flight ID : ");

String FlightId = sc.nextLine();

try {

myCustomer = customer.getCustomerFromFlightID(FlightId);

myCustomer.setPhone("9878721033");

if (customer.update(myCustomer)) {

System.***out***.println("Customer updated successfully");

}

} catch (DataAccessException e) {

System.***out***.println(e);

}

break;

case 3:

System.***out***.println("Enter the destination city");

String city = sc.nextLine();

System.***out***.println("Displaying all flights to " + city);

for (Flight f : flight.displayFlightsForDestination(city)) {

System.***out***.println(f);

}

break;

case 4:

System.***out***.println("Enter the flight ID to cancel : ");

String id = sc.nextLine();

try {

if(flight.cancelFlight(id)) {

System.***out***.println("Flight cancelled successfully");

}

} catch (DataAccessException e) {

System.***out***.println(e);

}

break;

case 5:

System.***out***.println("Enter your Customer ID: ");

String CID = sc.nextLine();

try {

for(Flight fl : flight.displayFlightForCustomer(CID)) {

System.***out***.println(fl);

}

}catch (DataAccessException e) {

System.***out***.println(e);

}

break;

}

} while (choice != 6);

}

}

1. **BOOK A NEW FLIGHT**

\*bookFlight() method

*@Override*

public boolean bookFlight(Flight flight) {

return jdbc.update(SQL\_BOOK\_NEW\_FLIGHT, flight.getFlightId(), flight.getFlightName(), flight.getFromCity(), flight.getToCity(), flight.getCust\_ID(), flight.getSeatNo()) > 0;

}

\*Inside Main : Taking inputs from user and generating a Booking ID and then inserting using

bookFlight() method

System.***out***.println("Enter flight name : ");

String flightName = sc.nextLine();

System.***out***.println("Enter Starting city : ");

String source = sc.nextLine();

System.***out***.println("Enter Destination city : ");

String destination = sc.nextLine();

System.***out***.println("Enter your Customer Id : ");

String CustId = sc.nextLine();

System.***out***.println("Enter the seat no : ");

String seatNo = sc.nextLine();

//System generated Flight ID

String FID = CustId + source.charAt(0) + destination.charAt(0) + seatNo;

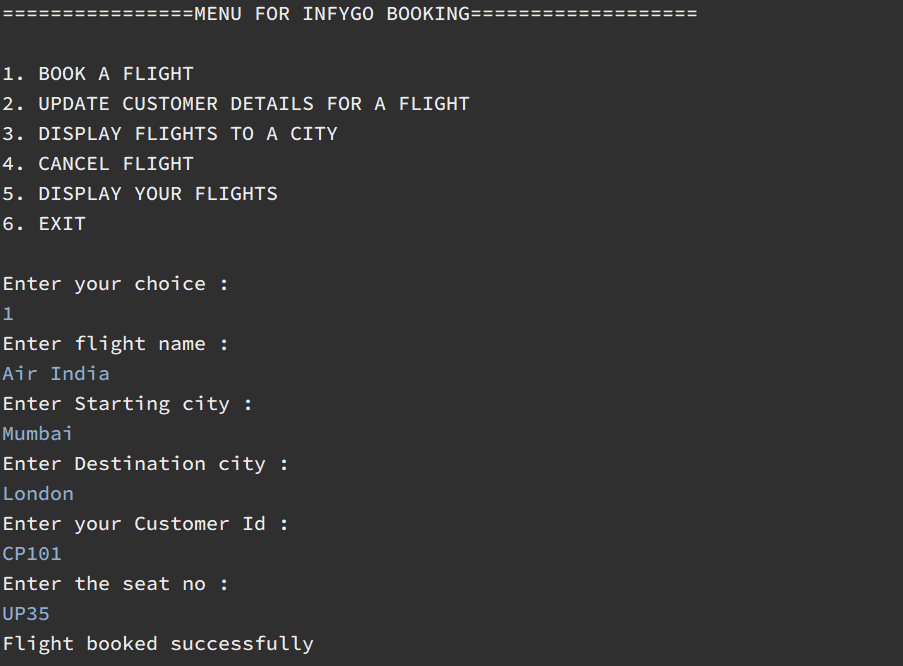
Flight myFlight = new Flight(FID, flightName, source, destination, CustId, seatNo);

if (flight.bookFlight(myFlight)) {

System.***out***.println("Flight booked successfully");

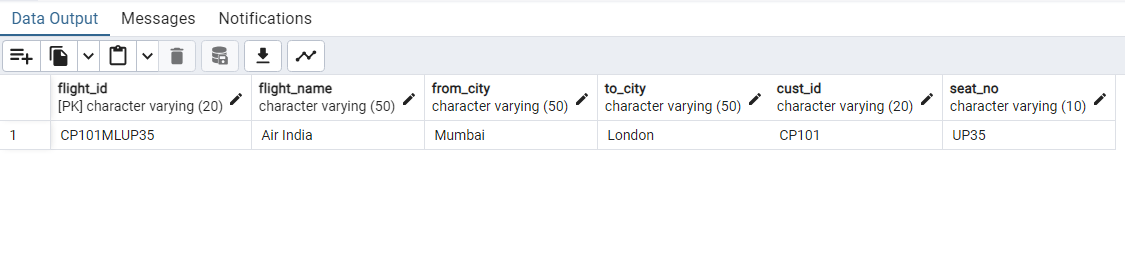
}

\*OUTPUT



\*FLIGHTS TABLE

SELECT \* FROM FLIGHT



1. **Update customer details for a flight**

\*Inside update() method => we use update() method of JdbcTemplate to update customer details and Inner join query to join customer and flight table using foreign key cust\_id

public boolean update(Customer customer) {

return jdbc.update(SQL\_UPDATE\_CUSTOMER, customer.getPhone(), customer.getCustId()) > 0;

}

\*Inside Main class

Customer myCustomer = new Customer();

System.***out***.println("Enter your flight ID : ");

String FlightId = sc.nextLine();

try {

myCustomer = customer.getCustomerFromFlightID(FlightId);

myCustomer.setPhone("9878721033");

if (customer.update(myCustomer)) {

System.***out***.println("Customer updated successfully");

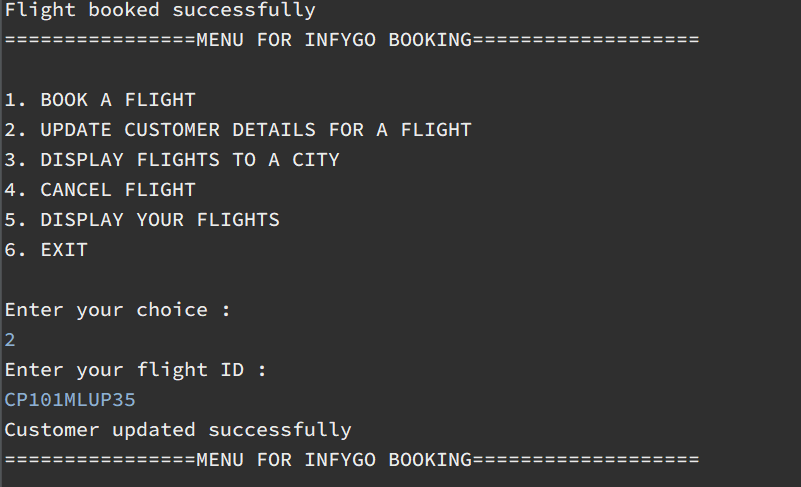
}

} catch (DataAccessException e) {

System.***out***.println(e);

}

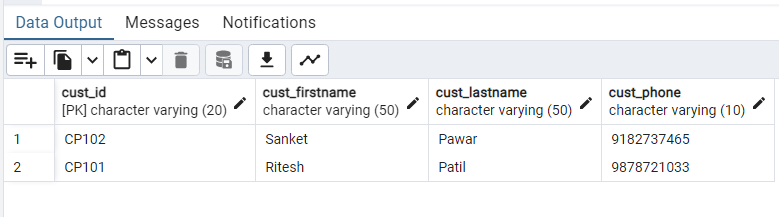
\*OUTPUT



\*DATABASE TABLE

SELECT \* FROM CUSTOMERS

\*Phone number is updated here



1. **Display flights for particular destination**

\*Inside displayFlightForDestination() method => we use query method and pass destination city as an argument

*@Override*

public List<Flight> displayFlightsForDestination(String destination) {

return jdbc.~~query~~(SQL\_FLIGHTS\_FOR\_DESTINATION, new Object[] {destination}, new FlightMapper());

}

\*Inside Main Class

System.***out***.println("Enter the destination city");

String city = sc.nextLine();

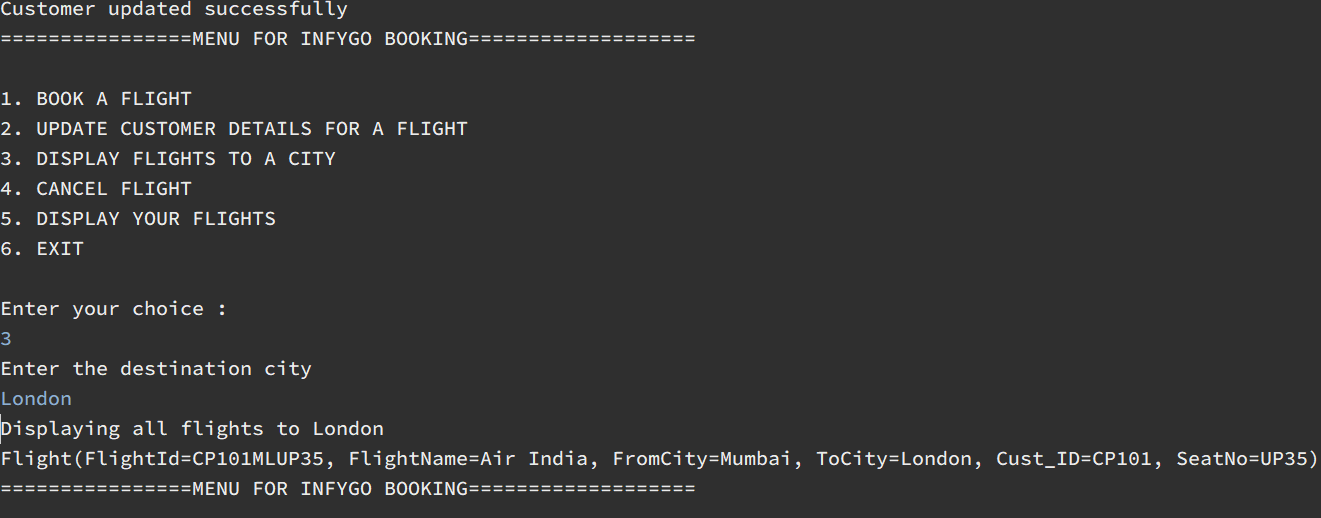
System.***out***.println("Displaying all flights to " + city);

for (Flight f : flight.displayFlightsForDestination(city)) {

System.***out***.println(f);

}

\*OUTPUT



6**. Cancel a flight**

\*Inside cancelFlight() method

*@Override*

public boolean cancelFlight(String id) {

return jdbc.update(SQL\_CANCEL\_FLIGHT, id) > 0;

}

\*Inside Main class

System.***out***.println("Enter the flight ID to cancel : ");

String id = sc.nextLine();

try {

if(flight.cancelFlight(id)) {

System.***out***.println("Flight cancelled successfully");

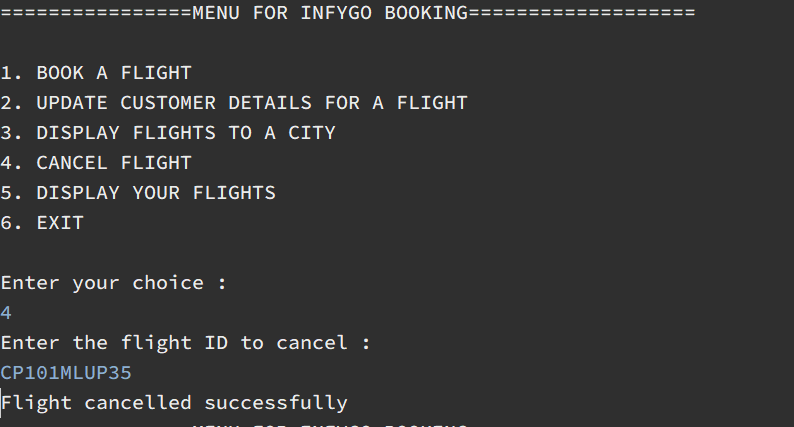
}

} catch (DataAccessException e) {

System.***out***.println(e);

}

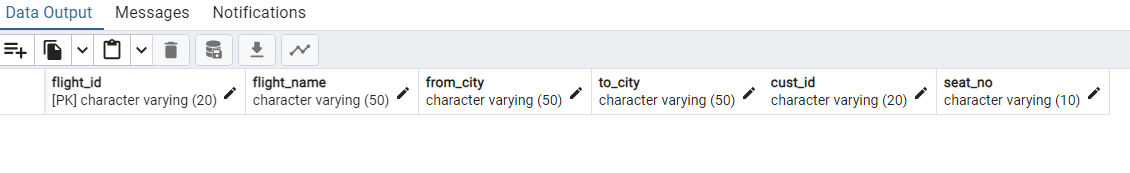
\*\*OUTPUT



\*Database table

SELECT \* FROM FLIGHTS

\*\*Flight has been deleted



7**.Display flight for a customer**

\*Inside displayFlightForCustomer() method => we use query method and pass Customer id as argument

public List<Flight> displayFlightForCustomer(String cid) {

return jdbc.~~query~~(SQL\_FLIGHTS\_FOR\_CUSTOMER, new Object[] {cid}, new FlightMapper());

}

\*Inside main class

System.***out***.println("Enter your Customer ID: ");

String CID = sc.nextLine();

try {

for(Flight fl : flight.displayFlightForCustomer(CID)) {

System.***out***.println(fl);

}

}catch (DataAccessException e) {

System.***out***.println(e);

}

\*OUTPUT

