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');
```

	cust_id [PK] character varying (20)	cust_firstname character varying (50)	cust_lastname character varying (50)	cust_phone character varying (10)
1	CP101	Ritesh	Patil	9876543210
2	CP102	Sanket	Pawar	9182737465

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

Data Output Messages Notifications

	flight_id [PK] character varying (20)	flight_name character varying (50)	from_city character varying (50)	to_city character varying (50)	cust_id character varying (20)	seat_no character varying (10)

2. Appropriate Interfaces and Classes

a. Flight.java => POJO representing the Flight entity (Flight table in Database)

```
b. @Data
c. @AllArgsConstructor
d. @NoArgsConstructor
e. public class Flight {
f.
g.      private String FlightId;
h.      private String FlightName;
i.      private String FromCity;
j.      private String ToCity;
k.      private String Cust_ID;
l.      private String SeatNo;
m.
n. }
```

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());
    }

}
```

i. JDBCConfig.java => Handle connectivity with the Database. Using DriverManagerDatasource to establish connection with database and JDBCTemplate to perform operations

```
j. @Configuration
k. @ComponentScan("com.infosys.capstone")
l. @PropertySource("database.properties")
m. public class JDBCConfig {
n.
o.      @Autowired
p.      private Environment env;
q.
r.      private final String URL = "url";
s.      private final String USER = "dbuser";
t.      private final String PASS = "dbpass";
u.      private final String DRIVER = "driver";
v.
w.      @Bean
x.      DataSource dataSource() {
y.          DriverManagerDataSource datasource = new
   DriverManagerDataSource();
z.          datasource.setUrl(env.getProperty(URL));
aa.             datasource.setUsername(env.getProperty(USER));
bb.             datasource.setPassword(env.getProperty(PASS));
cc.
        datasource.setDriverClassName(env.getProperty(DRIVER));
dd.             return datasource;
ee.
ff.         }
gg.
hh.         @Bean
ii.         public JdbcTemplate myJdbc(DataSource ds) {
jj.             return new JdbcTemplate(ds);
kk.         }
ll.
mm.     }
```

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);

        }
}
```

3. 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

```
================MENU FOR INFYGO BOOKING===================

1. BOOK A FLIGHT
2. UPDATE CUSTOMER DETAILS FOR A FLIGHT
3. DISPLAY FLIGHTS TO A CITY
4. CANCEL FLIGHT
5. DISPLAY YOUR FLIGHTS
6. EXIT

Enter your choice :
1
Enter flight name :
Air India
Enter Starting city :
Mumbai
Enter Destination city :
London
Enter your Customer Id :
CP101
Enter the seat no :
UP35
Flight booked successfully
```

*FLIGHTS TABLE

SELECT * FROM FLIGHT

Data Output Messages Notifications

	flight_id [PK] character varying (20)	flight_name character varying (50)	from_city character varying (50)	to_city character varying (50)	cust_id character varying (20)	seat_no character varying (10)
1	CP101MLUP35	Air India	Mumbai	London	CP101	UP35

4. 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

```
Flight booked successfully
================MENU FOR INFYGO BOOKING===================

1. BOOK A FLIGHT
2. UPDATE CUSTOMER DETAILS FOR A FLIGHT
3. DISPLAY FLIGHTS TO A CITY
4. CANCEL FLIGHT
5. DISPLAY YOUR FLIGHTS
6. EXIT

Enter your choice :
2
Enter your flight ID :
CP101MLUP35
Customer updated successfully
================MENU FOR INFYGO BOOKING===================
```

*DATABASE TABLE

SELECT * FROM CUSTOMERS

*Phone number is updated here

Data Output Messages Notifications

	cust_id [PK] character varying (20)	cust_firstname character varying (50)	cust_lastname character varying (50)	cust_phone character varying (10)
1	CP102	Sanket	Pawar	9182737465
2	CP101	Ritesh	Patil	9878721033

5. 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

```
Customer updated successfully
================MENU FOR INFYGO BOOKING===================

1. BOOK A FLIGHT
2. UPDATE CUSTOMER DETAILS FOR A FLIGHT
3. DISPLAY FLIGHTS TO A CITY
4. CANCEL FLIGHT
5. DISPLAY YOUR FLIGHTS
6. EXIT

Enter your choice :
3
Enter the destination city
London
Displaying all flights to London
Flight(FlightId=CP101MLUP35, FlightName=Air India, FromCity=Mumbai, ToCity=London, Cust_ID=CP101, SeatNo=UP35)
================MENU FOR INFYGO BOOKING===================
```

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

```
================MENU FOR INFYGO BOOKING===================

1. BOOK A FLIGHT
2. UPDATE CUSTOMER DETAILS FOR A FLIGHT
3. DISPLAY FLIGHTS TO A CITY
4. CANCEL FLIGHT
5. DISPLAY YOUR FLIGHTS
6. EXIT

Enter your choice :
4
Enter the flight ID to cancel :
CP101MLUP35
Flight cancelled successfully
```

*Database table

SELECT * FROM FLIGHTS

**Flight has been deleted

Data Output Messages Notifications

flight_id [PK] character varying (20)	flight_name character varying (50)	from_city character varying (50)	to_city character varying (50)	cust_id character varying (20)	seat_no character varying (10)

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

```
================MENU FOR INFYGO BOOKING===================

1. BOOK A FLIGHT
2. UPDATE CUSTOMER DETAILS FOR A FLIGHT
3. DISPLAY FLIGHTS TO A CITY
4. CANCEL FLIGHT
5. DISPLAY YOUR FLIGHTS
6. EXIT

Enter your choice :
5
Enter your Customer ID:
CP101
Flight(FlightId=CP101MLUP35, FlightName=Air India, FromCity=Mumbai, ToCity=London, Cust_ID=CP101, SeatNo=UP35)
```