PES UNIVERSITY, Bengaluru
Department of Computer Science and Engineering
B. Tech (CSE) – 5th Semester – Aug-Dec 2024

DBMS

MINIPROJECT REPORT

TITLE: AIRLINE MANAGEMENT SYSTEM

PES1UG22CS363	**MUJASEEM D**
PES1UG22CS368	**N SWETHA**

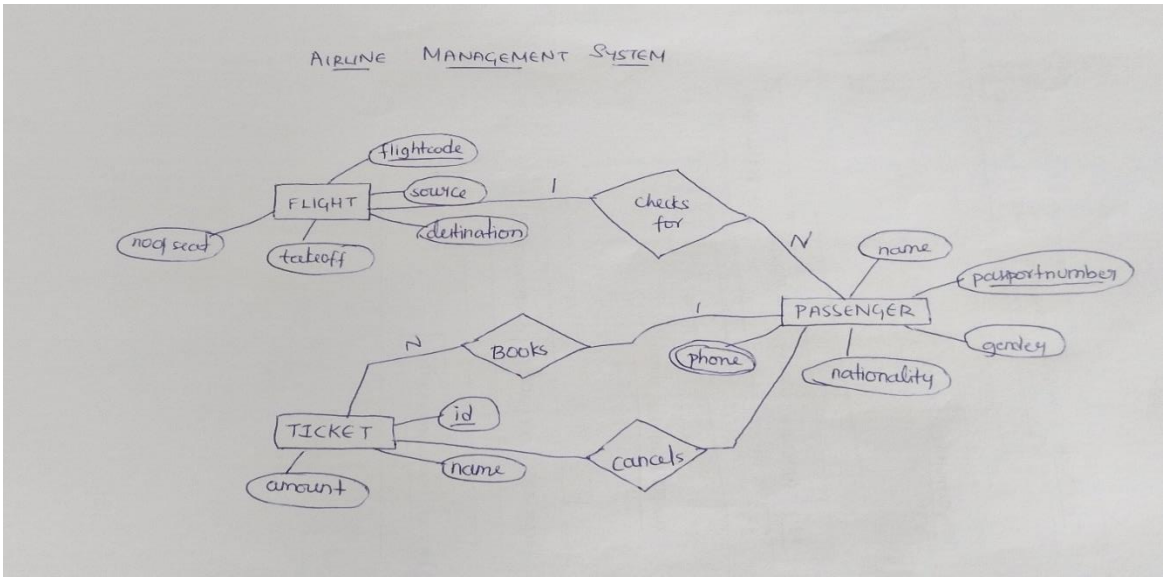
Description:

The Airline Management System is a robust database-driven application designed to manage key operations of an airline efficiently. It incorporates the management of users, flights, ticket booking and ticket cancellation ensuring a seamless experience for passengers and admin.

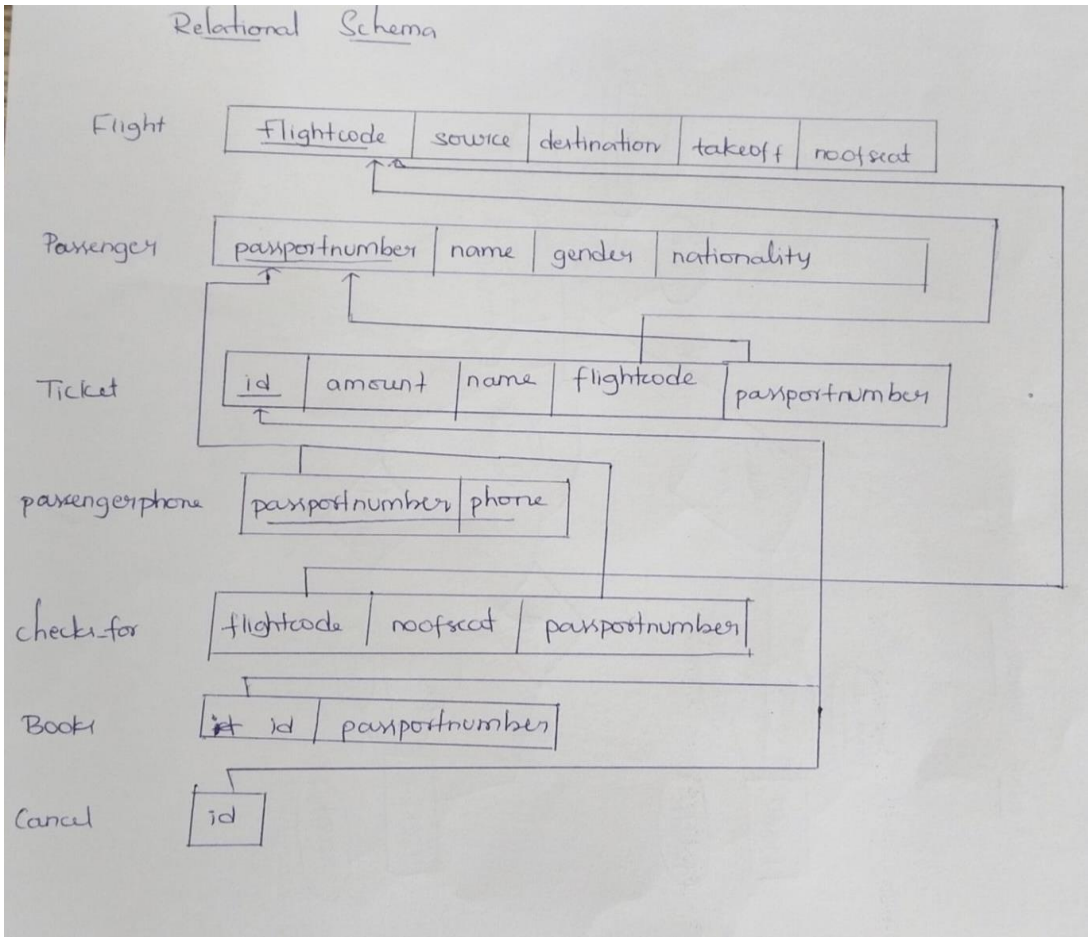
Core Features:

- **User Management**
- **Flight Management**
- **Ticket Management**

ER DIAGRAM:



RELATIONAL SCHEMA:



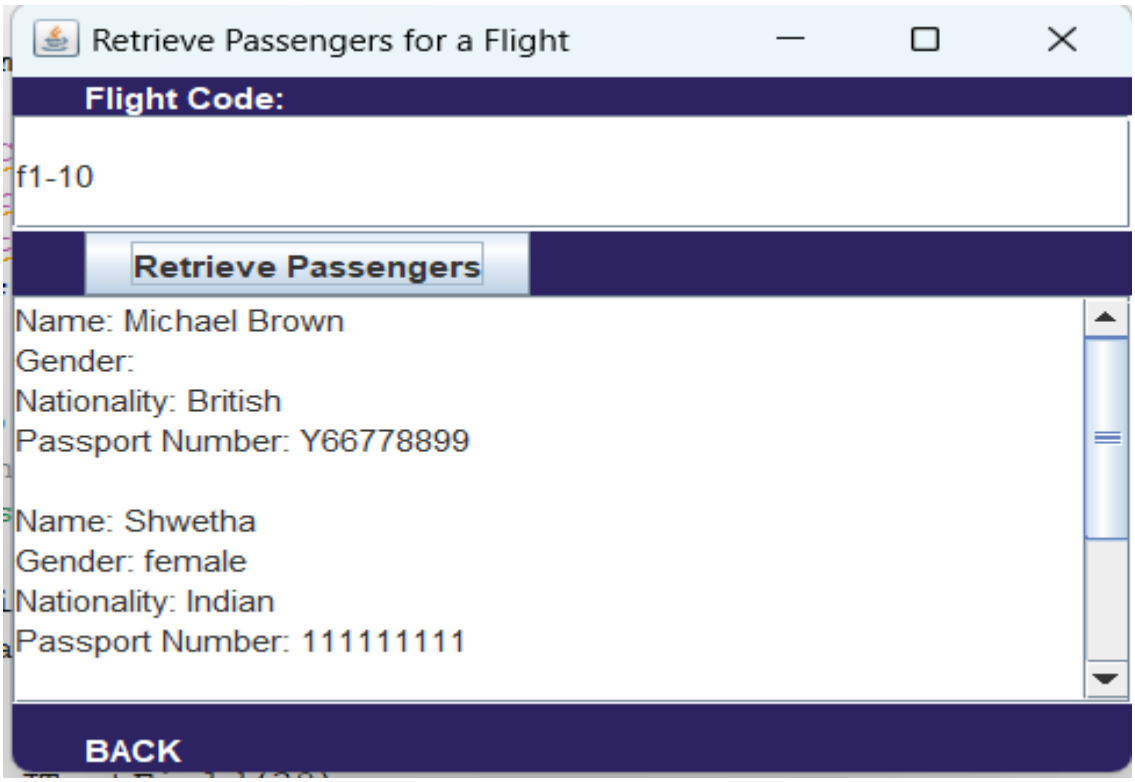
JOIN:

1.It retrieves the passenger details (name, gender, nationality, and passport number) for all passengers booked on a specified flight.

Query:

```
SELECT p.name, p.gender, p.nationality, p.passportnumber
FROM ticket_booking t
JOIN managepassenger p ON t.passportnumber = p.passportnumber
WHERE t.flightcode = flightCode;
```

Output:



NESTED Query:

Query: This query retrieves all records from the ticket_booking table for flights that have 'Bengaluru' as their source city in the manageflight table**.**

```
SELECT *
FROM
   ticket_booking
WHERE
   flightcode IN (
      SELECT flightcode
      FROM manageflight
      WHERE source = 'Bengaluru'
   );
```

Output:

id	name	flightcode	gender	passportnumber	amount	nationality
220	Surya	f1-202	Male	Y66778899	5000	Indian

AGGREGATE FUNCTIONS:

1. **Highest Ticket Price**

   This query retrieves the highest ticket price (amount) for each flight by selecting the flightcode and the maximum amount from the ticket_booking table, grouped by flightcode

   Query:
   ```
   SELECT flightcode, MAX(amount) AS highest_ticket_price
   FROM ticket_booking
   GROUP BY flightcode;
   ```

   Output:

flightcode	highest_ticket_price
f1-10	5000

2. **Total Revenue**

   This query calculates the total revenue from the ticket_booking table for flight code "f1-10" by summing the amount values, casting them as decimal to ensure precise calculation.

   Query:
   ```
   SELECT SUM(CAST(amount AS DECIMAL(10,2))) as totalRevenue
   FROM ticket_booking
   WHERE flightcode = "f1-10";
   ```

   Output:

totalRevenue
6000.00

PROCEDURES:

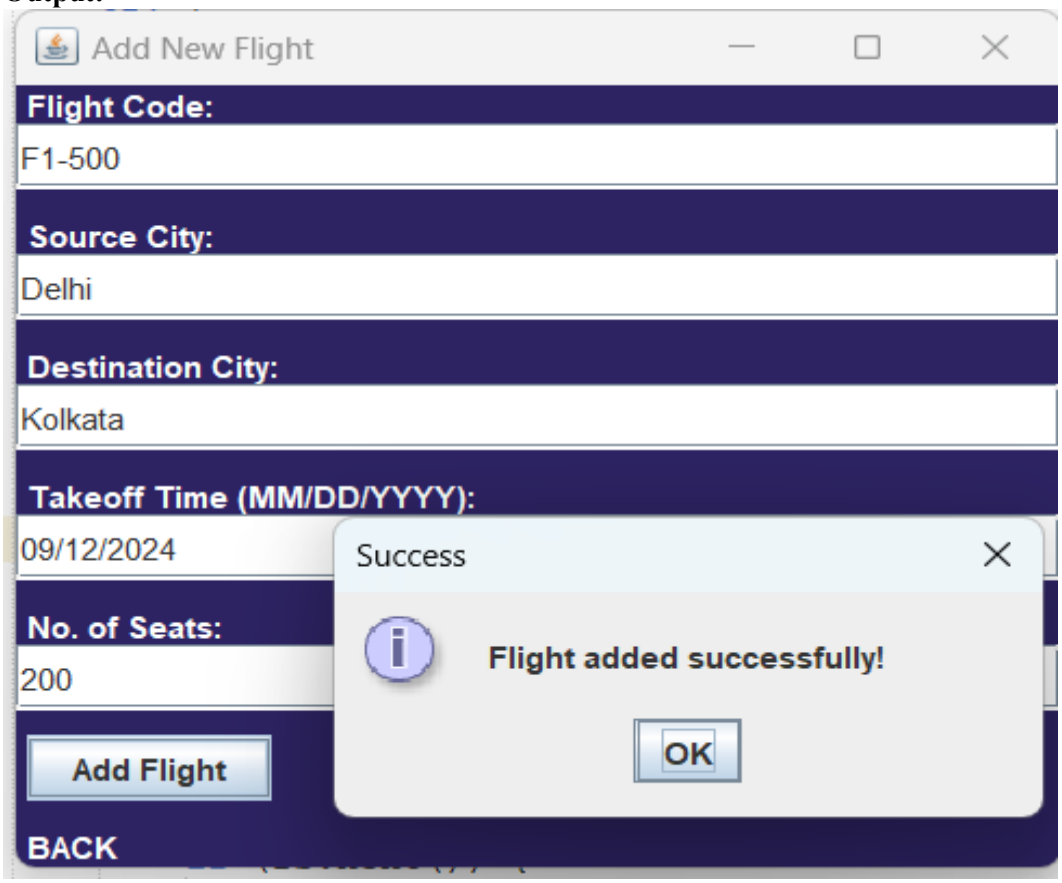
1. **Add New Flight:** This procedure inserts a new flight record into the manageflight table using the provided flight details and returns a success message.

   SQL Query:
   ```
   DELIMITER $$
   CREATE DEFINER=`` PROCEDURE `AddNewFlight`(
      IN flightcode VARCHAR(6),
      IN source VARCHAR(30),
      IN destination VARCHAR(30),
      IN takeoff VARCHAR(30),
      IN noofseat BIGINT
   )
   BEGIN
      -- Insert the new flight into the manageflight table
   ```

```
    INSERT INTO manageflight (flightcode, source, destination, takeoff, noofseat)
    VALUES (flightcode, source, destination, takeoff, noofseat);

    -- Return a success message
    SELECT 'Flight added successfully!' AS message;
END$$
DELIMITER ;
```

Output:



2. **Calculate Total Revenue:** This procedure calculates and returns the total revenue for a specified flight based on ticket bookings, or a message if no bookings are found.

SQL Query:

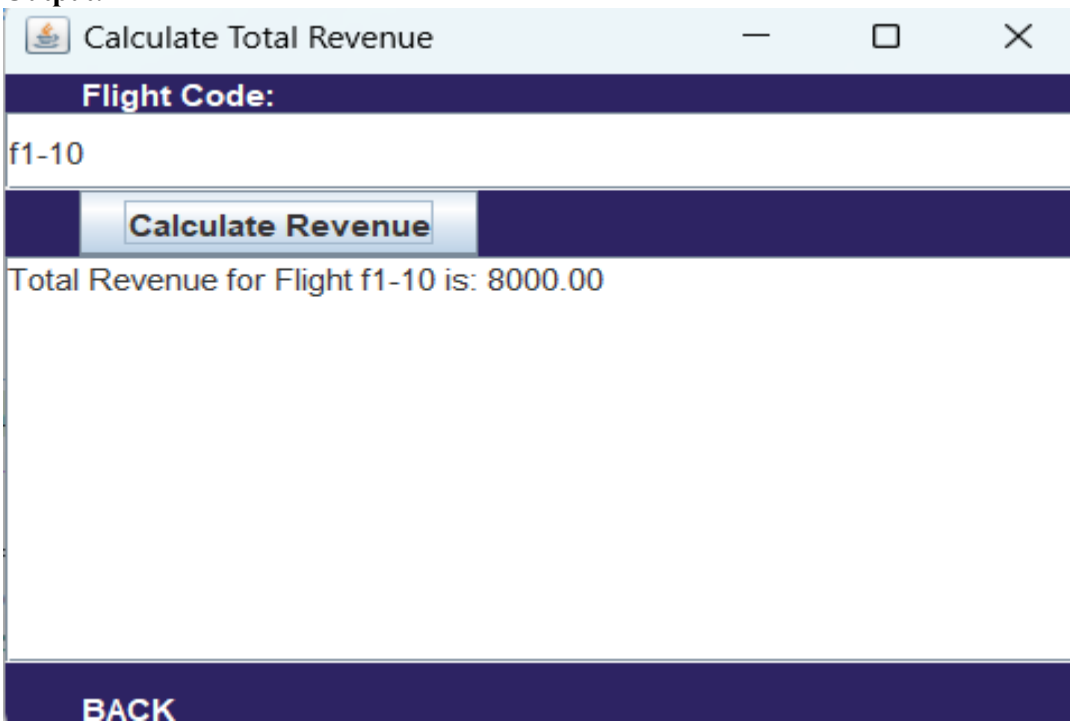
```
DELIMITER $$

CREATE DEFINER=`` PROCEDURE `CalculateTotalRevenue`(
    IN flightCode VARCHAR(6)
)
BEGIN
    DECLARE totalRevenue DECIMAL(10,2);

    -- Calculate the total revenue by summing up the amount from the ticket_booking table
    SELECT SUM(CAST(amount AS DECIMAL(10,2))) INTO totalRevenue
    FROM ticket_booking
    WHERE flightcode = flightCode;
```

```
    -- Return the total revenue
    IF totalRevenue IS NOT NULL THEN
        SELECT CONCAT('Total Revenue for Flight ', flightCode, ' is: ', totalRevenue) AS
message;
    ELSE
        SELECT 'No bookings found for this flight.' AS message;
    END IF;
END$$
DELIMITER ;
```

Output:

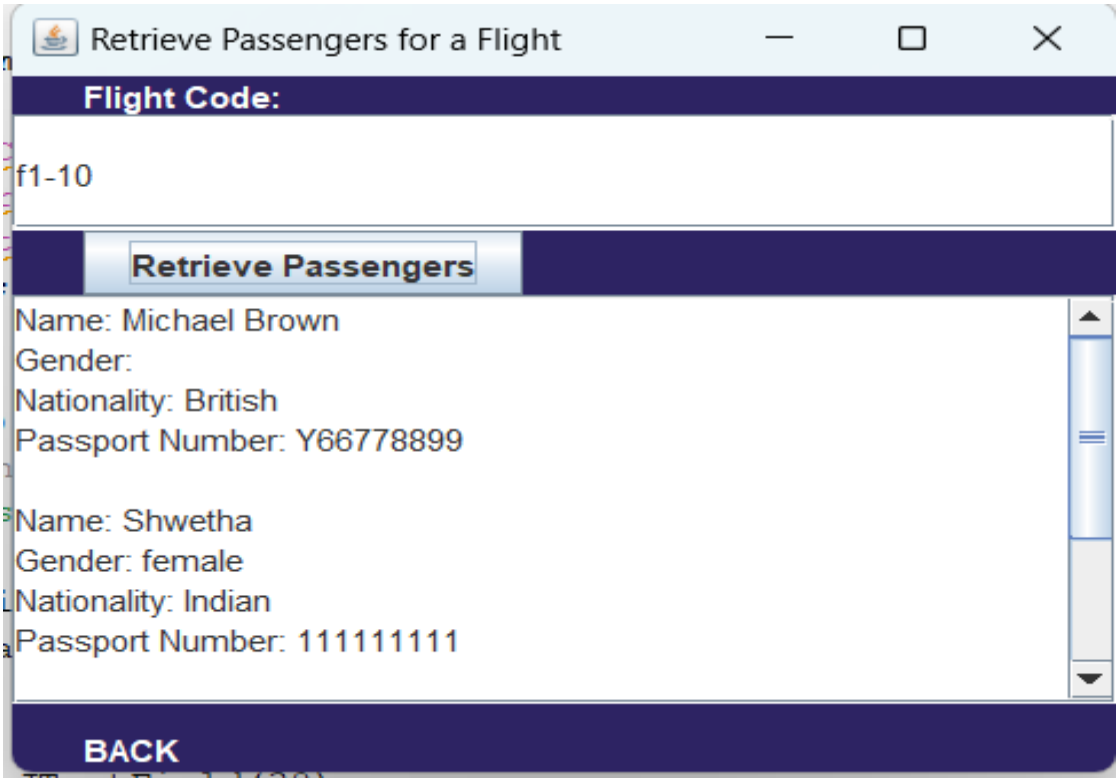


3. **Get all passengers for a particular flight:** This procedure retrieves the passenger details (name, gender, nationality, and passport number) for all passengers booked on a specified flight.

SQL Query:

```
DELIMITER $$
CREATE DEFINER=`` PROCEDURE `GetAllPassengersForFlight`(
    IN flightCode VARCHAR(6)
)
BEGIN
    -- Retrieve the passenger details for the given flight code
    SELECT p.name, p.gender, p.nationality, p.passportnumber
    FROM ticket_booking t
    JOIN managepassenger p ON t.passportnumber = p.passportnumber
    WHERE t.flightcode = flightCode;
END$$
DELIMITER ;
```

Output:

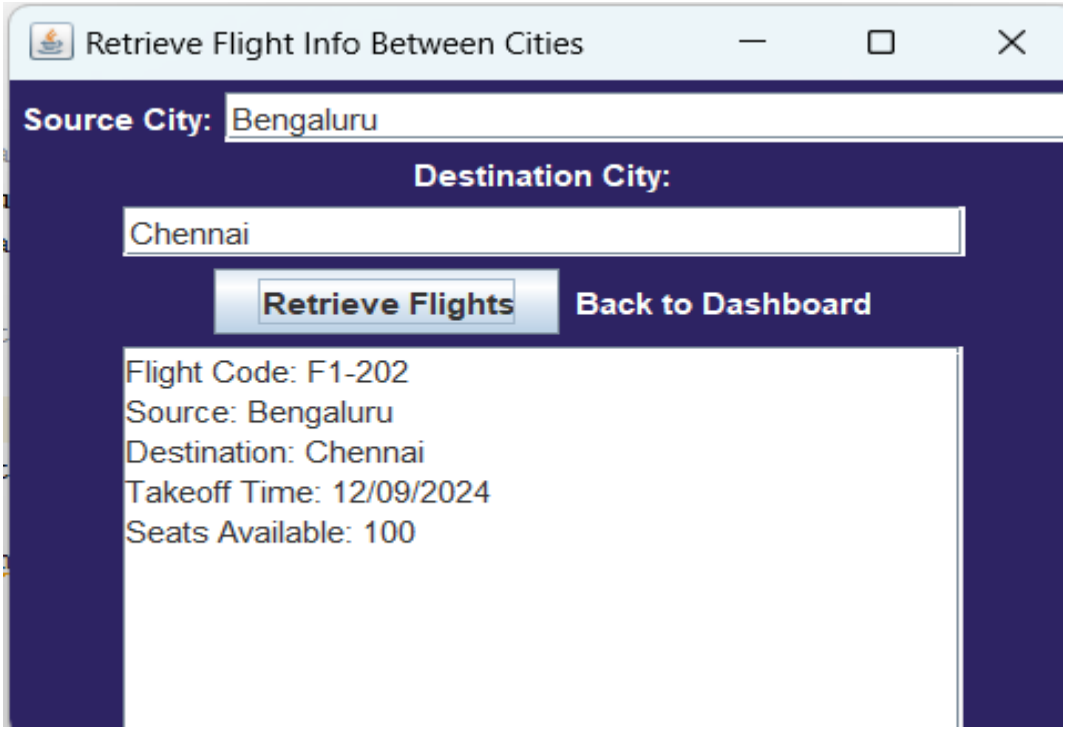


4. **Retrieve Flight info between cities:** This procedure retrieves flight information (flight code, source, destination, takeoff time, and number of seats) for flights between the specified source and destination cities.

   SQL Query:

```
DELIMITER $$
CREATE DEFINER=`` PROCEDURE `RetrieveFlightInfoBetweenCities`(
   IN source_city VARCHAR(30),
   IN destination_city VARCHAR(30)
)
BEGIN
   -- Retrieve flight information between the specified source and destination
   SELECT flightcode, source, destination, takeoff, noofseat
   FROM manageflight
   WHERE source = source_city AND destination = destination_city;
END$$
DELIMITER ;
```

   Output:



TRIGGERS:

1.**Seat Count:** This trigger keeps track of the number of seats available per flight.

Before Booking :

flightcode	source	destination	takeoff	noofseat
f1-10	Mumbai	Hyderabad	08/11/2022	99
f1-101	Chennai	Bhopal	09/22/2000	501
f1-200	Singapore	Bengaluru	08/22/2005	499
F1-202	Bengaluru	Chennai	12/09/2024	100
F1-500	Delhi	Kolkata	09/12/2024	200
f1-900	CHintamani	Bengaluru	09/27/2000	50

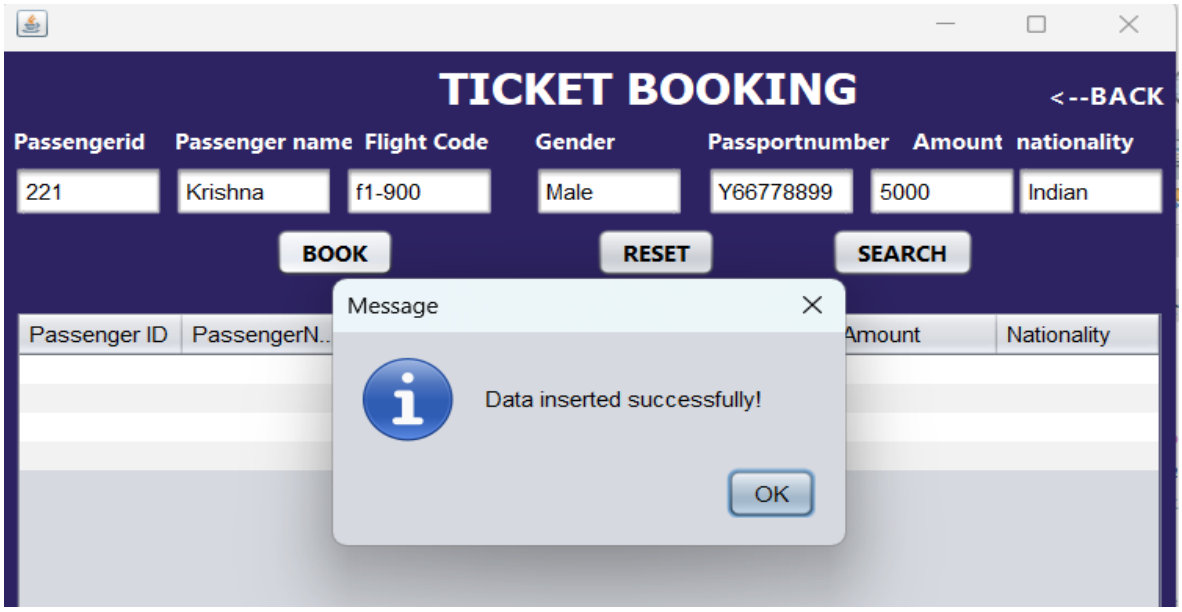
Query:

```
CREATE TRIGGER `after_ticket_insert`
AFTER INSERT ON `ticket_booking`
 FOR EACH ROW BEGIN
    UPDATE manageflight
    SET noofseat = noofseat - 1
```

WHERE flightcode = NEW.flightcode;

END

After Booking:



flightcode	source	destination	takeoff	noofseat
f1-10	Mumbai	Hyderabad	08/11/2022	99
f1-101	Chennai	Bhopal	09/22/2000	501
f1-200	Singapore	Bengaluru	08/22/2005	499
F1-202	Bengaluru	Chennai	12/09/2024	99
F1-500	Delhi	Kolkata	09/12/2024	200
f1-900	CHintamani	Bengaluru	09/27/2000	49

2.Preventing Duplicate Flight entries:

Before:

flightcode	source	destination	takeoff	noofseat
f1-10	Mumbai	Hyderabad	08/11/2022	99
f1-101	Chennai	Bhopal	09/22/2000	501
f1-200	Singapore	Bengaluru	08/22/2005	499
F1-202	Bengaluru	Chennai	12/09/2024	99
F1-500	Delhi	Kolkata	09/12/2024	200
f1-900	CHintamani	Bengaluru	09/27/2000	49

Query:

```
CREATE TRIGGER `prevent_duplicate_flight_code`
BEFORE INSERT ON `manageflight`
 FOR EACH ROW BEGIN
   -- Check if the flight code already exists
   IF EXISTS (SELECT 1 FROM manageflight WHERE flightcode = NEW.flightcode) THEN
      SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Flight code already exists!';
   END IF;
END
```

After :