List of Files

This document lists all files in the application, including Python scripts, HTML templates, and the database schema, with their purposes.

Python Scripts

- **blog.py**
  - Purpose: Main Flask application script. Defines all routes for public flight search, user authentication (login, registration...), customer features (flight search, purchase, spending...), booking agent features (flight purchase, commission tracking...), and airline staff features (flight management, reports, permissions...). Manages database connections to the MySQL project2_solution database and executes SQL queries for all functionalities.

HTML Templates

- **add_agent.html**
  - Purpose: Form for airline staff (admins) to add booking agents to their airline. Collects the agent's email and displays a list of currently assigned agents.
- **add_airplane.html**
  - Purpose: Form for airline staff (admins) to add new airplanes. Collects airplane ID and seat count, and displays a list of existing airplanes for the airline.
- **add_airport.html**
  - Purpose: Form for airline staff (admins) to add new airports. Collects airport name and city, and displays a list of existing airports.
- **create_flight.html**
  - Purpose: Form for airline staff (admins) to create new flights. Includes fields for flight number, departure/arrival airports, times, price, status, and airplane ID, with dropdowns for available airplanes and airports.
- **grant_permission.html**
  - Purpose: Form for airline staff (admins) to grant Admin or Operator permissions to other staff members. Lists staff members (excluding the current user) and their current permissions.
- **home_AirlineStaff.html**
  - Purpose: Dashboard for logged-in airline staff. Displays upcoming flights, frequent customers, top destinations, booking agent statistics, and provides links to staff-specific actions (e.g., create flight, update status, view reports).

· **home_BookingAgent.html**
  - Purpose: Dashboard for logged-in booking agents. Displays flights purchased on behalf of customers, allows flight search and purchase, and shows commission statistics (ticket count, total, average) and top customers by ticket count and commission.
· **home_Customer.html**
  - Purpose: Dashboard for logged-in customers. Displays purchased flights, allows flight search and purchase, and shows spending statistics (total and monthly breakdown) with a customizable date range.
· **index.html**
  - Purpose: Homepage for unauthenticated users. Provides links to login, register, and public flight information pages.
· **login.html**
  - Purpose: Login page for customers, booking agents, and airline staff. Contains a form to submit username/email, password, and user type (customer, booking agent, or airline staff).
· **public.html**
  - Purpose: Public flight information page accessible to all users. Displays a table of upcoming flights and includes forms to search flights by source/destination city/airport and departure date, or check flight status by flight number and date.
· **register.html**
  - Purpose: Main registration page. Provides links to specific registration pages for customers, booking agents, and airline staff.
· **registerforBookingagent.html**
  - Purpose: Registration form for booking agents. Collects email, password, and booking agent ID, and submits to the /registerAuthBookingAgent route.
· **registerforCustomer.html**
  - Purpose: Registration form for customers. Collects personal details (email, name, address, phone number, passport info, date of birth) and submits to the /registerAuthCustomer route.
· **registerforStaff.html**
  - Purpose: Registration form for airline staff. Collects username, password, first name, last name, date of birth, and airline name, and submits to the /registerAuthStaff route.
· **staff_reports.html**
  - Purpose: Page for airline staff to view ticket sales reports. Displays the total number of tickets sold and a bar chart of monthly ticket counts, with options to filter by custom date ranges or presets (last month, last year).
· **staff_revenue.html**

- Purpose: Page for airline staff to view revenue reports. Shows direct (customer) and indirect (booking agent) revenue for the last month and year, visualized as pie charts.

· **update_status.html**

- Purpose: Form for airline staff (operators) to update the status of flights (e.g., On-time, Delayed). Lists all flights for the airline with options to select a flight and new status.

Other Files

· **Air_database.sql**

- Purpose: Database schema and sample data for the application. Define tables such as customer, booking_agent, airline_staff, flight, ticket, purchases, airplane, airport, permission, and booking_agent_work_for and add some testing data. Used to initialize the MySQL database.

Use Cases and Queries

This document details all use cases in the application, the SQL queries executed, and brief explanations of their functionality. It's organized by user type (Public, Customer, Booking Agent, Airline Staff) to ensure clarity and readability, as required. Each use case corresponds to a route in blog.py and includes the exact queries from the code.

Use Cases and Queries

This document outlines all use cases in the flight management application, the SQL queries executed for each, and brief explanations of their functionality. The application is built with Flask (blog.py) and uses a MySQL database (project2_solution). Use cases are grouped by user type: Public, Customer, Booking Agent, and Airline Staff.

Public Use Cases

These features are accessible to all users, logged in or not.

1. **View Upcoming Flights**
   - **Description**: Displays a table of all upcoming flights, showing airline, flight number, departure/arrival airports, times, price, and status.
   - **Query**:
     ```
     SELECT airline_name, flight_num, departure_airport,
     departure_time,
            arrival_airport, arrival_time, price, status
     FROM flight
     WHERE departure_time > NOW()
     ORDER BY departure_time
     ```
   - **Explanation**: Retrieves all flights with a future departure time, sorted by departure time. The results are displayed in a table in public.html.

2. **Search Upcoming Flights**
   - **Description**: Allows users to search for upcoming flights by source city/airport, destination city/airport, departure date, and flight number.
   - **Query**:

     ```
     SELECT f.airline_name, f.flight_num, f.departure_airport,
     f.departure_time, f.arrival_airport, f.arrival_time, f.price,
     f.status
     FROM flight f
     ```

```
JOIN airport a1 ON f.departure_airport = a1.airport_name
JOIN airport a2 ON f.arrival_airport = a2.airport_name
WHERE f.departure_time > NOW()
[AND (a1.airport_city = %s OR a1.airport_name = %s)]
[AND (a2.airport_city = %s OR a2.airport_name = %s)]
[AND DATE(f.departure_time) = %s]
[AND f.flight_num = %s]
ORDER BY f.departure_time
```

- **Explanation**: Dynamically constructs a query to filter flights based on user inputs (source, destination, date, flight number). Joins with the airport table to match city or airport names. Results are shown in public.html. If no flights match, an error message is displayed.

Customer Use Cases

These features are accessible only to logged-in customers.

1. **View and Filter My Flights**
    - **Description**: Displays a table of all upcoming flights booked by the customer, with optional filtering by date range and airports.
    - **Query**:

```
SELECT f.airline_name, f.flight_num, departure_airport,
departure_time,
       arrival_airport, arrival_time, price, status, p.ticket_id
FROM ticket t JOIN flight f JOIN purchases p
WHERE p.ticket_id=t.ticket_id AND t.flight_num=f.flight_num AND
p.customer_email = %s
  AND f.departure_time >= [start_date|current_date]
  [AND additional filters...]
ORDER BY f.departure_time ASC
```

    - **Explanation**:
        - Retrieves flights purchased by the logged-in customer (`session['username']`).
        - Applies dynamic filters:

1. Default: Shows flights departing from today onward.
2. Optional: Date range (`start_date`/`end_date`) and airport filters (partial matches via `LIKE`).

- Results are displayed in `home_Customer.html`.

2. **Search for Available Flights**

- **Description**: Allows customers to search for all future flights (not just their bookings) with filters.
- **Query**:

```
SELECT f.airline_name, f.flight_num, departure_airport,
departure_time,
       arrival_airport, arrival_time, price, status
FROM flight f
WHERE f.departure_time >= %s
  [AND additional filters...]
ORDER BY f.departure_time ASC
```

- **Explanation**:
  - Searches all flights departing after the current date.
  - Supports the same filters as "My Flights" (dates, airports).

3. **Purchase a Flight Ticket**

- **Description**: Handles ticket purchases for selected flights.
- **Key Logic**:
  - Checks flight capacity by comparing `sold_seats` (from `ticket` table) with `total_seats` (from `airplane`).
  - Generates a unique `ticket_id` and inserts records into:
    - `ticket`: Links ticket to flight.

        - `purchases`: Records customer email and purchase date (no booking agent).
    - Shows error if flight is full or success message otherwise.
- Query:

```
INSERT INTO ticket (ticket_id, airline_name, flight_num)
INSERT INTO purchases (ticket_id, customer_email,
booking_agent_id, purchase_date)
```

4. **Track Spending**
    - **Description**: Displays customer's spending over a customizable time period (default: last 6 months).
    - **Queries**:
        - **Total Spending**:

```
SELECT SUM(f.price) AS total_spent
FROM purchases p
JOIN ticket t ON p.ticket_id = t.ticket_id
JOIN flight f ON t.flight_num = f.flight_num
WHERE p.customer_email = %s
AND p.purchase_date BETWEEN %s AND %s
```

        - **Monthly Breakdown**:

```
SELECT DATE_FORMAT(p.purchase_date, '%Y-%m') AS month, SUM(price)
AS amount
GROUP BY month
ORDER BY month DESC LIMIT 6
```

    - **Features**:
        - Dynamic date range via `start_datess`/`end_datess` parameters.
        - Calculates average monthly spending and max value for chart scaling.
        - Data passed to `home_Customer.html` for visualization.

Booking Agent Use Cases

These features are accessible only to authenticated booking agents.

1. **View and Filter Booked Flights**
    - **Description**: Displays all flights booked by the agent for customers, with optional filtering.
    - **Query**:

```
SELECT f.airline_name, f.flight_num, departure_airport,
departure_time,
       arrival_airport, arrival_time, price, status, p.ticket_id,
p.customer_email
FROM ticket t
JOIN flight f ON t.flight_num=f.flight_num
JOIN purchases p ON p.ticket_id=t.ticket_id
JOIN booking_agent b ON b.booking_agent_id=p.booking_agent_id
WHERE b.email = %s
  [AND f.departure_time BETWEEN start_date AND end_date]
  [AND airport filters...]
ORDER BY f.departure_time ASC
```

    - **Features:**
        - Filters by date range (inclusive of end day via `23:59:59`).
        - Partial airport search using `LIKE` for arrival/departure.
        - Shows customer email and ticket ID for reference.

2. **Search Available Flights**
    - **Description**: Searches flights from airlines the agent works for.
    - **Query**:

```
SELECT f.airline_name, f.flight_num, departure_airport,
departure_time,
       arrival_airport, arrival_time, price, status
FROM flight f
JOIN booking_agent_work_for b ON f.airline_name=b.airline_name
WHERE b.email = %s
  [AND additional filters...]
```

- **Key Difference**:
    - Only shows flights operated by airlines the agent is affiliated with (via `booking_agent_work_for`).

3. **Purchase Tickets for Customers**

- **Workflow**:

1. **Validation Checks**:
    - Verifies customer exists (`SELECT FROM customer`).
    - Checks flight capacity by comparing `sold_seats` (count from `ticket`) vs. `total_seats` (from `airplane`).
2. **Ticket Creation**:
    - Generates unique random `ticket_id` (1000-9999 range).
    - Inserts records into:
        - `ticket`: Links to flight.
        - `purchases`: Associates agent ID (retrieved via session email) and customer email.
3. **Error Handling**:
    - "Flight full" or "Customer not exist" errors displayed via flash messages.

4. **View Commission Reports**

- **Default Period**: Last 30 days (customizable via date inputs).
- **SQL**:

```
SELECT
  COUNT(*) AS ticket_count,
  SUM(flight.price * 0.1) AS total_commission,
  AVG(flight.price * 0.1) AS avg_commission
FROM purchases
```

```
JOIN ticket USING(ticket_id)
JOIN flight ON ticket.flight_num=flight.flight_num
WHERE
  booking_agent_id = %s
  AND purchase_date BETWEEN %s AND %s
```

- **Display**:
    - Total tickets sold, sum of commissions (10% of ticket price), and average per ticket.

5. **Track Top Customers**
    - **Two Ranking Lists**:
    1. **By Tickets (Last 6 Months)**:

```
SELECT customer_email, COUNT(*) AS ticket_count
FROM purchases
WHERE booking_agent_id = %s AND purchase_date >= %s
GROUP BY customer_email
ORDER BY ticket_count DESC LIMIT 5
```

    2. **By Commission (Last Year)**:

```
SELECT customer_email, SUM(flight.price * 0.1) AS total_commission
FROM purchases
JOIN ticket USING(ticket_id)
JOIN flight ON ticket.flight_num=flight.flight_num
WHERE booking_agent_id = %s AND purchase_date >= %s
GROUP BY customer_email
ORDER BY total_commission DESC LIMIT 5
```

    - **Purpose**: Helps agents identify high-value customers for targeted service.

Airline Staff Use Cases

These features require a logged-in airline staff member, with some restricted to Admin or Operator permissions.

1. **Login as Airline Staff**
    - Route: /loginAuth (POST, user_type='airline_staff')
    - Description: Authenticates an airline staff member using their username and password.
    - Queries:
        - Verify credentials:
          ```
          SELECT * FROM airline_staff WHERE username = %s AND password = %s
          ```
        - Get permission:
          ```
          SELECT permission_type FROM permission WHERE username = %s
          ```
        - Get airline:
          ```
          SELECT airline_name FROM airline_staff WHERE username = %s
          ```
    - Explanation: Verifies the staff's credentials, retrieves their permission (Admin/Operator) and airline name, sets session variables (username, user_type, permission, airline_name), and redirects to /home_AirlineStaff. If invalid, shows an error on login.html.

2. **Register as Airline Staff**
    - Description: Registers a new airline staff member with username, password, name, date of birth, and airline name.
    - Queries:
        - Check for existing staff:
          ```
          SELECT * FROM airline_staff WHERE username = %s
          ```
        - Insert new staff:
          ```
          INSERT INTO airline_staff (

             username, password, first_name, last_name, date_of_birth, airline_name
          ```

```
) VALUES (%s, %s, %s, %s, %s, %s)
```

- Explanation: Ensures the username is unique, then inserts the staff's details into the airline_staff table. Redirects to index.html on success or shows an error on registerforStaff.html if the username exists.

3. **View Flights**
   - Description: Displays all upcoming flights (next 30 days) for the staff's airline.
   - Query:

```
SELECT airline_name, flight_num, departure_airport,
departure_time,

       arrival_airport, arrival_time, price, status
FROM flight
WHERE airline_name = %s AND departure_time >= CURDATE()
      AND departure_time <= DATE_ADD(CURDATE(), INTERVAL 30 DAY)
ORDER BY departure_time
```

   - Explanation: Retrieves flights for the staff's airline within the next 30 days, sorted by departure time. Results are displayed in home_AirlineStaff.html.

4. **Search Flights and View Passengers**
   - Description: Allows staff to search for flights with filters (start date, end date, departure/arrival airport) and view passengers for a specific flight.
   - Queries:

     Search flights

```
SELECT airline_name, flight_num, departure_airport,
departure_time,
       arrival_airport, arrival_time, price, status
FROM flight
WHERE airline_name = %s
      [AND departure_time >= %s]
      [AND departure_time <= %s]
      [AND departure_airport = %s]
```

```
      [AND arrival_airport = %s]
ORDER BY departure_time
```

Get passengers:

```
SELECT DISTINCT c.email, c.name
FROM ticket t
JOIN purchases p ON t.ticket_id = p.ticket_id
JOIN customer c ON p.customer_email = c.email
WHERE t.flight_num = %s AND t.airline_name = %s
```

Get airports:

```
SELECT airport_name, airport_city

FROM airport

ORDER BY airport_name
```

- Explanation: Retrieves flights for the airline with optional filters and lists passengers (customer email and name) for a selected flight. Airports are fetched for filter dropdowns. Results are displayed in home_AirlineStaff.html.

5. **Create Flight (Admin)**
   - Description: Allows admins to create a new flight with details (flight number, airports, times, price, status, airplane).
   - Queries:

Get airplanes:

```
SELECT airplane_id FROM airplane WHERE airline_name = %s
```

Get airports:

```
SELECT airport_name FROM airport
```

Get existing flights:

```
SELECT flight_num, departure_time, arrival_time

FROM flight

WHERE airline_name = %s
```

```
ORDER BY departure_time
```

Check for duplicate flight:

```
SELECT * FROM flight WHERE flight_num = %s AND airline_name = %s
```

Insert flight:

```
INSERT INTO flight (

    airline_name, flight_num, departure_airport, departure_time,

    arrival_airport, arrival_time, price, status, airplane_id

      ) VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s)
```

- Explanation: Provides dropdowns for airplanes and airports, lists existing flights, checks for duplicate flight numbers, and inserts a new flight. On success, redirects to /home_AirlineStaff with a success message. If the flight number exists, shows an error on create_flight.html.

6. **Update Flight Status (Operator)**
   - Description: Allows operators to update the status of a flight (e.g., On-time, Delayed).
   - Queries:

     Get flights:

```
SELECT flight_num, departure_time, arrival_time, price, status,

       airline_name, airplane_id, departure_airport,
arrival_airport

FROM flight

WHERE airline_name = %s

ORDER BY departure_time
```

     Validate flight:

```
SELECT * FROM flight
```

```
WHERE flight_num = %s AND airline_name = %s
```

Update status:

```
UPDATE flight

SET status = %s

WHERE flight_num = %s AND airline_name = %s
```

- Explanation: Lists all flights for the airline, validates the selected flight, and updates its status. On success, redirects to /staff_update_status with a success message. If the flight doesn't exist, it shows an error on update_status.html. Requires Operator permission.

Add Airplane (Admin)

- Description: Allows admins to add a new airplane with an ID and seat count.
- Queries:
  - Get existing airplanes:
    ```
    SELECT airplane_id, seats FROM airplane WHERE airline_name = %s
    ```
  - Check for duplicate airplane:
    ```
    SELECT * FROM airplane WHERE airplane_id = %s AND airline_name = %s
    ```

Insert airplane:

```
INSERT INTO airplane (airline_name, airplane_id, seats)

VALUES (%s, %s, %s)
```

- Explanation: Displays existing airplanes, checks for duplicate airplane IDs, and inserts a new airplane. On success, redisplays add_airplane.html with a success message. If the ID exists, shows an error.

Add Airport (Admin)

- Description: Allows admins to add a new airport with a name and city.
- Queries:

- Get existing airports:
  ```
  SELECT airport_name, airport_city FROM airport
  ```

- Check for duplicate airport:
  ```
  SELECT * FROM airport WHERE airport_name = %s
  ```

Insert airport:

```
INSERT INTO airport (airport_name, airport_city)

VALUES (%s, %s)
```

- Explanation: Displays existing airports, checks for duplicate airport names, and inserts a new airport. On success, redisplays add_airport.html with a success message. If the airport exists, shows an error.

View Booking Agents

- Route: /staff_view_agents (GET)
- Description: Displays the top 5 booking agents by ticket sales (last month and year) and commission (last year).
- Queries:

  Top agents by tickets (last month):

  ```
  SELECT b.email, b.booking_agent_id, COUNT(t.ticket_id) as
  ticket_count

  FROM ticket t

  JOIN purchases p ON t.ticket_id = p.ticket_id

  JOIN booking_agent b ON p.booking_agent_id = b.booking_agent_id

  JOIN flight f ON t.flight_num = f.flight_num AND t.airline_name =
  f.airline_name

  WHERE f.airline_name = %s AND p.purchase_date >=
  DATE_SUB(CURDATE(), INTERVAL 1 MONTH)

  GROUP BY b.booking_agent_id

  ORDER BY ticket_count DESC
  ```

```
LIMIT 5
```

Top agents by tickets (last year):

```
SELECT b.email, b.booking_agent_id, COUNT(t.ticket_id) as
ticket_count

FROM ticket t

JOIN purchases p ON t.ticket_id = p.ticket_id

JOIN booking_agent b ON p.booking_agent_id = b.booking_agent_id

JOIN flight f ON t.flight_num = f.flight_num AND t.airline_name =
f.airline_name

WHERE f.airline_name = %s AND p.purchase_date >=
DATE_SUB(CURDATE(), INTERVAL 1 YEAR)

GROUP BY b.booking_agent_id

ORDER BY ticket_count DESC

LIMIT 5
```

Top agents by commission:

```
SELECT b.email, b.booking_agent_id, SUM(f.price) * 0.1 as
commission

FROM ticket t

JOIN purchases p ON t.ticket_id = p.ticket_id

JOIN booking_agent b ON p.booking_agent_id = b.booking_agent_id

JOIN flight f ON t.flight_num = f.flight_num AND t.airline_name =
f.airline_name

WHERE f.airline_name = %s AND p.purchase_date >=
DATE_SUB(CURDATE(), INTERVAL 1 YEAR)

GROUP BY b.booking_agent_id

ORDER BY commission DESC
```

```
LIMIT 5
```

- Explanation: Retrieves the top booking agents for the airline based on ticket sales and commission earned. Results are displayed in home_AirlineStaff.html.

View Frequent Customer

- Route: /staff_view_customers (GET)
- Description: Displays the most frequent customer (by ticket count in the last year) and their purchased flights.
- Queries:

  Get frequent customer:

```
SELECT c.email, c.name, COUNT(t.ticket_id) as ticket_count

FROM ticket t

JOIN purchases p ON t.ticket_id = p.ticket_id

JOIN customer c ON p.customer_email = c.email

JOIN flight f ON t.flight_num = f.flight_num AND t.airline_name =
f.airline_name

WHERE f.airline_name = %s AND p.purchase_date >=
DATE_SUB(CURDATE(), INTERVAL 1 YEAR)

GROUP BY c.email

ORDER BY ticket_count DESC

LIMIT 1
```

  Get customer's flights:

```
SELECT f.airline_name, f.flight_num, f.departure_airport,
f.departure_time,

       f.arrival_airport, f.arrival_time, f.price, f.status

FROM ticket t

JOIN purchases p ON t.ticket_id = p.ticket_id
```

```
JOIN flight f ON t.flight_num = f.flight_num AND t.airline_name =
f.airline_name

WHERE p.customer_email = %s AND f.airline_name = %s
```

- Explanation: Identifies the customer with the most tickets purchased in the past year and lists their flights for the airline. Results are displayed in home_AirlineStaff.html.

View Reports

- Route: /staff_reports (GET/POST)
- Description: Displays ticket sales reports, including total tickets sold and monthly ticket counts, with customizable date ranges or presets (last month, last year).
- Queries:

  Total tickets:
  ```
  SELECT COUNT(t.ticket_id) as total_tickets

  FROM ticket t

  JOIN purchases p ON t.ticket_id = p.ticket_id

  JOIN flight f ON t.flight_num = f.flight_num AND t.airline_name =
  f.airline_name

  WHERE f.airline_name = %s AND p.purchase_date BETWEEN %s AND %s
  ```

  Monthly ticket counts:
  ```
  SELECT DATE_FORMAT(p.purchase_date, '%Y-%m') as month,
  COUNT(t.ticket_id) as ticket_count

  FROM ticket t

  JOIN purchases p ON t.ticket_id = p.ticket_id

  JOIN flight f ON t.flight_num = f.flight_num AND t.airline_name =
  f.airline_name

  WHERE f.airline_name = %s AND p.purchase_date BETWEEN %s AND %s

  GROUP BY DATE_FORMAT(p.purchase_date, '%Y-%m')
  ```

```
ORDER BY month
```

- Explanation: Calculates the total number of tickets sold and monthly aggregates for the specified date range. Results are displayed in staff_reports.html as a bar chart.

View Revenue

- Route: /staff_revenue (GET)
- Description: Displays direct (customer-purchased) and indirect (agent-purchased) revenue for the last month and last year.
- Query:

```
SELECT

    SUM(CASE WHEN p.booking_agent_id IS NULL THEN f.price ELSE 0
END) as direct_revenue,

    SUM(CASE WHEN p.booking_agent_id IS NOT NULL THEN f.price ELSE
0 END) as indirect_revenue

FROM ticket t

JOIN purchases p ON t.ticket_id = p.ticket_id

JOIN flight f ON t.flight_num = f.flight_num AND t.airline_name =
f.airline_name

WHERE f.airline_name = %s AND p.purchase_date BETWEEN %s AND %s
```

- Explanation: Calculates revenue from direct (no agent) and indirect (via agent) ticket sales for two time periods (last month and last year). Results are visualized as pie charts in staff_revenue.html.

View Top Destinations

- Route: /staff_top_destinations (GET)
- Description: Displays the top 3 destination cities by ticket count for the last 3 months and last year.
- Queries:

  Last 3 months:

```
SELECT a.airport_city, COUNT(t.ticket_id) as ticket_count
```

```
FROM ticket t

JOIN purchases p ON t.ticket_id = p.ticket_id

JOIN flight f ON t.flight_num = f.flight_num AND t.airline_name =
f.airline_name

JOIN airport a ON f.arrival_airport = a.airport_name

WHERE f.airline_name = %s AND p.purchase_date >=
DATE_SUB(CURDATE(), INTERVAL 3 MONTH)

GROUP BY a.airport_city

ORDER BY ticket_count DESC

LIMIT 3
```

Last year:

```
SELECT a.airport_city, COUNT(t.ticket_id) as ticket_count

FROM ticket t

JOIN purchases p ON t.ticket_id = p.ticket_id

JOIN flight f ON t.flight_num = f.flight_num AND t.airline_name =
f.airline_name

JOIN airport a ON f.arrival_airport = a.airport_name

WHERE f.airline_name = %s AND p.purchase_date >=
DATE_SUB(CURDATE(), INTERVAL 1 YEAR)

GROUP BY a.airport_city

ORDER BY ticket_count DESC

LIMIT 3
```

- Explanation: Identifies the most popular destination cities based on ticket sales for two time periods. Results are displayed in home_AirlineStaff.html.

Grant Permission (Admin)

- Route: /staff_grant_permission (GET/POST)
- Description: Allows admins to grant Admin or Operator permissions to other staff members in their airline.
- Queries:
    - Check admin permission:
      ```
      SELECT permission_type FROM permission WHERE username = %s
      AND permission_type = 'Admin'
      ```
    - Get staff list:
      ```
      SELECT username, first_name, last_name,

      (SELECT permission_type FROM permission p WHERE p.username =
      s.username) as permission_type

      FROM airline_staff s

      WHERE airline_name = %s AND username != %s
      ```
    - Verify staff:
      ```
      SELECT username FROM airline_staff WHERE username = %s AND
      airline_name = %s
      ```
    - Remove existing permission:
      ```
      DELETE FROM permission WHERE username = %s
      ```
    - Insert new permission:
      ```
      INSERT INTO permission (username, permission_type)

      VALUES (%s, %s)
      ```
- Explanation: Verifies the user is an admin, lists other staff members in the airline with their current permissions, validates the selected staff, removes any existing permission, and assigns a new permission (Admin, Operator, or none). On success, redirects to /home_AirlineStaff with a success message.

Add Booking Agent (Admin)

- Route: /staff_add_agent (GET/POST)
- Description: Allows admins to add a booking agent to their airline by specifying the agent's email.

- Queries:
  - Check admin permission:
    ```
    SELECT permission_type FROM permission WHERE username = %s
    AND permission_type = 'Admin'
    ```
  - Get current agents:
    ```
    SELECT ba.email, ba.booking_agent_id

    FROM booking_agent ba

    JOIN booking_agent_work_for wf ON ba.email = wf.email

    WHERE wf.airline_name = %s
    ```
  - Check agent existence:
    ```
    SELECT email FROM booking_agent WHERE email = %s
    ```
  - Check if already added:
    ```
    SELECT email FROM booking_agent_work_for WHERE email = %s
    AND airline_name = %s
    ```
  - Add agent:
    ```
    INSERT INTO booking_agent_work_for (email, airline_name)

    VALUES (%s, %s)
    ```

- Explanation: Verifies the user is an admin, displays current agents, validates the agent's email (must exist in booking_agent and not already be linked to the airline), and adds the agent to booking_agent_work_for. On success, redisplays add_agent.html with a success message.

Team Work Plan

Qingyue Zhu

- Code functions for customer
- Code functions for booking agent
- Ui design for all home page

Yutong Liu

- Code functions for public search
- Code functions for airline staff
- Ui design for airline staff function page