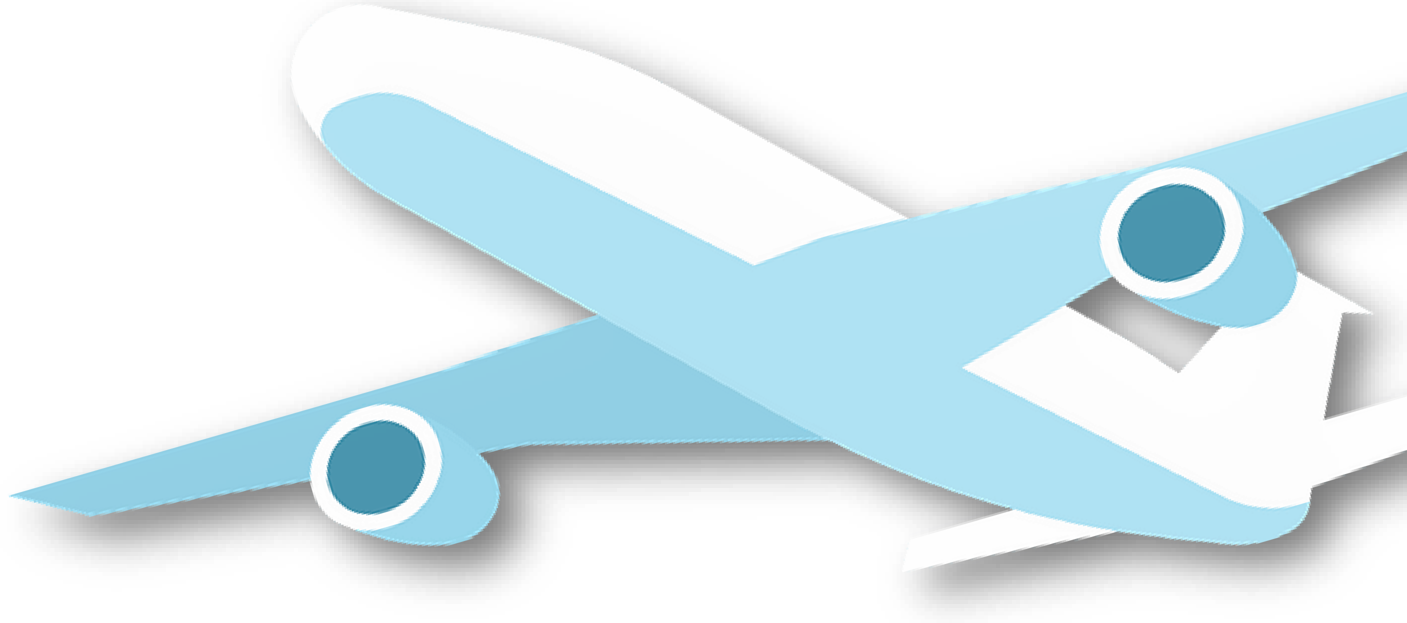
database project

FlightTracker

Gate 1	Arthur Popov
Gate 2	Jakub Samulski
Gate 3	Alina Sukhoverkova



Why this topic?

• Air travel is essential in today's global world
• Millions of flights occur daily
• Reliable systems are needed to track, manage, and analyze flight data

Our Objective

To design a functional, normalized database system that stores, organizes, and queries flight-related data

Departures

TIME	DESTINATION	FLIGHT	GATE	REMARK
09:45	BARCELONA	A4405	D17	ON TIME
09:57	NEW YORK	A1873	D08	ON TIME
10:03	DUBAI	A2960	C05	BOARDING
10:09	SHANGHAI	B0334	D14	DELAYED
10:14	DHAKA	B9700	F20	ON TIME
10:25	LONDON	C1503	F04	BOARDING
10:44	TOKYO	C5702	G11	ON TIME
10:59	SAO PAULO	D2013	G02	DELAYED
11:08	PARIS	D0869	E10	ON TIME
11:16	ISTANBUL	D7310	E06	ON TIME

Main Features:

• Store flight details: time, aircraft, status, airports
• Retrieve airline & aircraft info
• Support detailed search queries (flight number, route, delay status)

Target Users:

• Everyday travelers
• People searching for flights with full info
• Travel apps or price aggregators

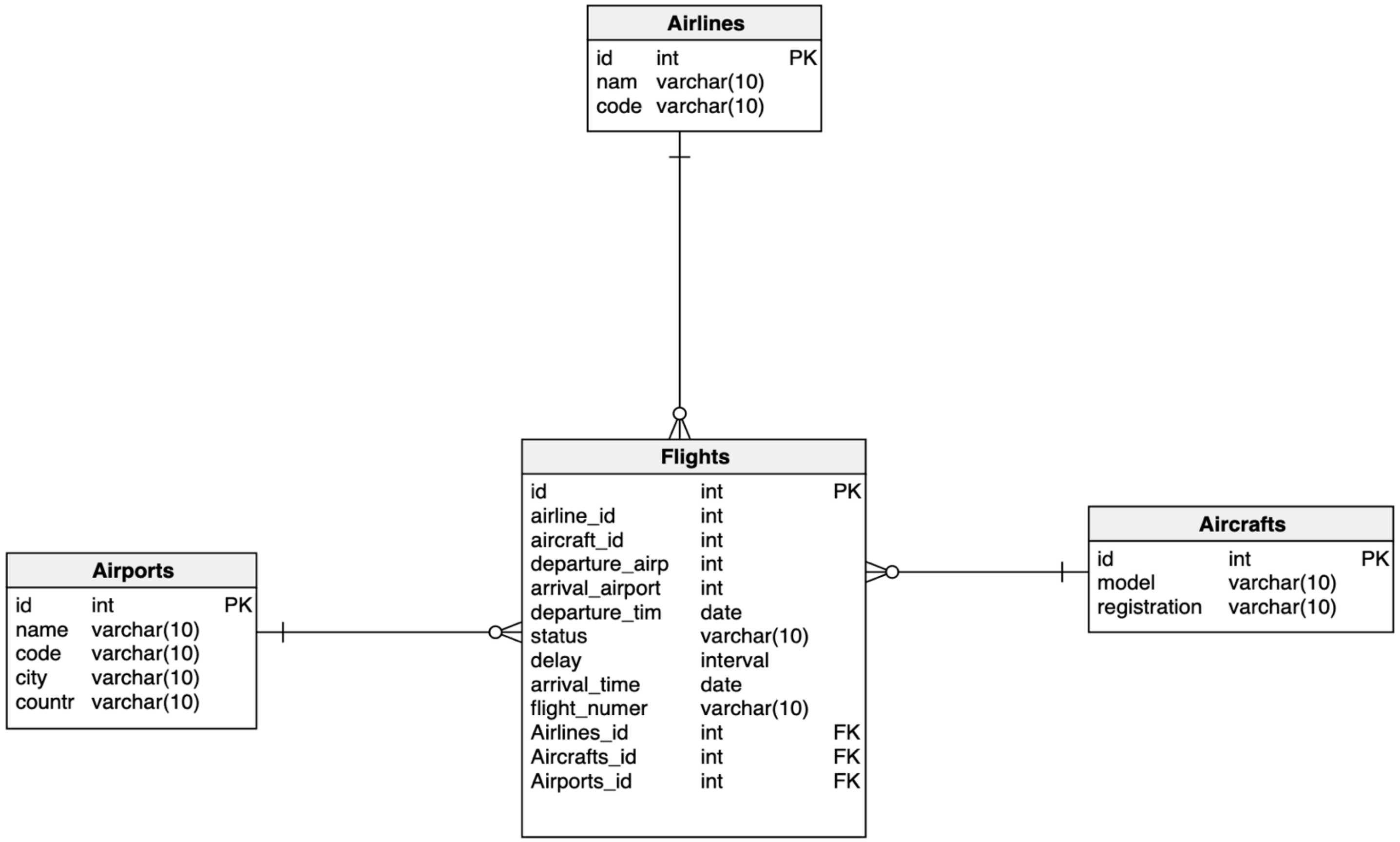
PDM database model

4 main tables:

• Flights is the central entity, referencing Airlines, Aircrafts, and Airports

• Airports are linked twice — as departure and arrival locations

• The schema is normalized and designed to allow flexible querying of flights and related data



Airlines

id	int	PK
nam	varchar(10)	
code	varchar(10)	

Flights

id	int	PK
airline_id	int	
aircraft_id	int	
departure_airp	int	
arrival_airport	int	
departure_tim	date	
status	varchar(10)	
delay	interval	
arrival_time	date	
flight_numer	varchar(10)	
Airlines_id	int	FK
Aircrafts_id	int	FK
Airports_id	int	FK

Airports

id	int	PK
name	varchar(10)	
code	varchar(10)	
city	varchar(10)	
countr	varchar(10)	

Aircrafts

id	int	PK
model	varchar(10)	
registration	varchar(10)	

Stack

FastAPI

React JS

PostgreSQL

Database

Tables

```
CREATE TABLE Flights
(
  id               SERIAL PRIMARY KEY,
  airline_id       INT REFERENCES Airlines (id),
  aircraft_id      INT REFERENCES Aircrafts (id),
  departure_airport INT REFERENCES Airports (id),
  arrival_airport  INT REFERENCES Airports (id),
  departure_time   TIMESTAMP,
  status           VARCHAR(20),
  delay            INTERVAL,
  arrival_time     TIMESTAMP,
  flight_number    VARCHAR(20)
);
```

```
CREATE TABLE Airlines
(
  id   SERIAL PRIMARY KEY,
  nam  VARCHAR(40),
  code VARCHAR(20)
);

CREATE TABLE Aircrafts
(
  id          SERIAL PRIMARY KEY,
  model       VARCHAR(20),
  registration VARCHAR(20)
);

CREATE TABLE Airports
(
  id      SERIAL PRIMARY KEY,
  name    VARCHAR(40),
  code    VARCHAR(20),
  city    VARCHAR(20),
  country VARCHAR(20)
);
```

Sequence

```
CREATE SEQUENCE flight_id_seq
   START WITH 1000
   INCREMENT BY 1
   NO MINVALUE
   NO MAXVALUE
   CACHE 1;

ALTER TABLE Flights ALTER COLUMN id SET DEFAULT nextval('flight_id_seq');
```

Function

```sql
CREATE OR REPLACE FUNCTION get_flight_duration(flight_id INT)
RETURNS INTERVAL
STABLE
AS $$
DECLARE
   duration INTERVAL;
BEGIN
   SELECT (arrival_time - departure_time + delay)
   INTO duration
   FROM Flights
   WHERE id = flight_id;

   RETURN duration;
END;
$$ LANGUAGE plpgsql;
```

View

```
CREATE VIEW flight_view AS
SELECT
   f.id AS flight_id,
   f.flight_number,

   ...

   f.status,
   f.delay,

   get_flight_duration(f.id) AS duration_with_delay
FROM Flights f
JOIN Airlines a ON f.airline_id = a.id
...
```

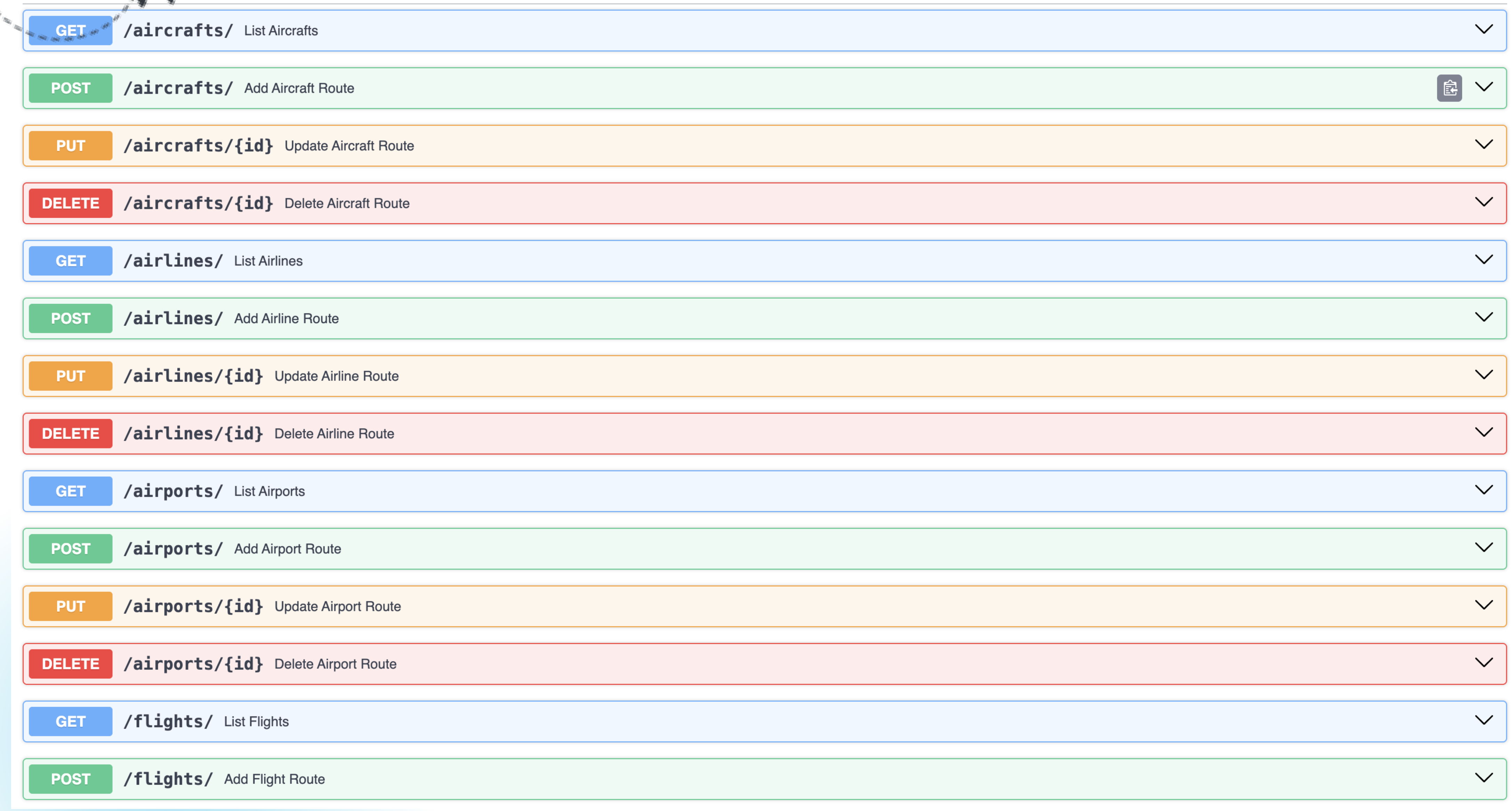
Trigger

```
CREATE OR REPLACE FUNCTION log_flight_status_change()
RETURNS TRIGGER AS $$
BEGIN
   IF OLD.status IS DISTINCT FROM NEW.status THEN
      INSERT INTO FlightStatusLog(flight_id, old_status, new_status)
      VALUES (OLD.id, OLD.status, NEW.status);
   END IF;
   RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trg_flight_status_change
AFTER UPDATE ON Flights
FOR EACH ROW
WHEN (OLD.status IS DISTINCT FROM NEW.status)
EXECUTE FUNCTION log_flight_status_change();
```

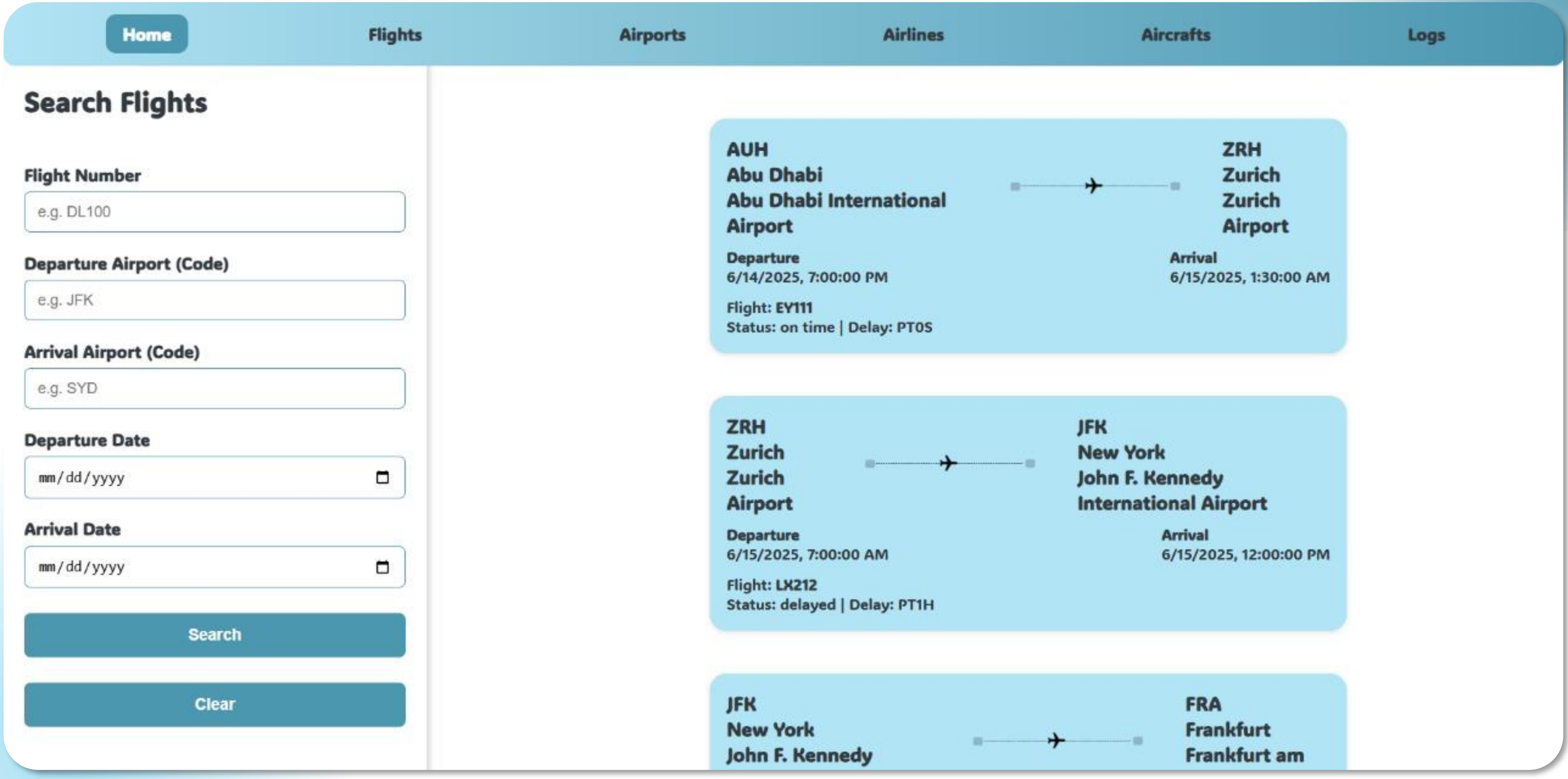
```
CREATE TABLE FlightStatusLog (
   id SERIAL PRIMARY KEY,
   flight_id INT,
   old_status VARCHAR(20),
   new_status VARCHAR(20),
   changed_at TIMESTAMP DEFAULT now()
);
```

API



GET	`/aircrafts/`	List Aircrafts
POST	`/aircrafts/`	Add Aircraft Route
PUT	`/aircrafts/{id}`	Update Aircraft Route
DELETE	`/aircrafts/{id}`	Delete Aircraft Route
GET	`/airlines/`	List Airlines
POST	`/airlines/`	Add Airline Route
PUT	`/airlines/{id}`	Update Airline Route
DELETE	`/airlines/{id}`	Delete Airline Route
GET	`/airports/`	List Airports
POST	`/airports/`	Add Airport Route
PUT	`/airports/{id}`	Update Airport Route
DELETE	`/airports/{id}`	Delete Airport Route
GET	`/flights/`	List Flights
POST	`/flights/`	Add Flight Route

WEB



Search Flights

Flight Number
e.g. DL100

Departure Airport (Code)
e.g. JFK

Arrival Airport (Code)
e.g. SYD

Departure Date
mm/dd/yyyy

Arrival Date
mm/dd/yyyy

AUH
Abu Dhabi
Abu Dhabi International Airport

ZRH
Zurich
Zurich Airport

Departure
6/14/2025, 7:00:00 PM

Arrival
6/15/2025, 1:30:00 AM

Flight: EY111
Status: on time | Delay: PT0S

ZRH
Zurich
Zurich Airport

JFK
New York
John F. Kennedy International Airport

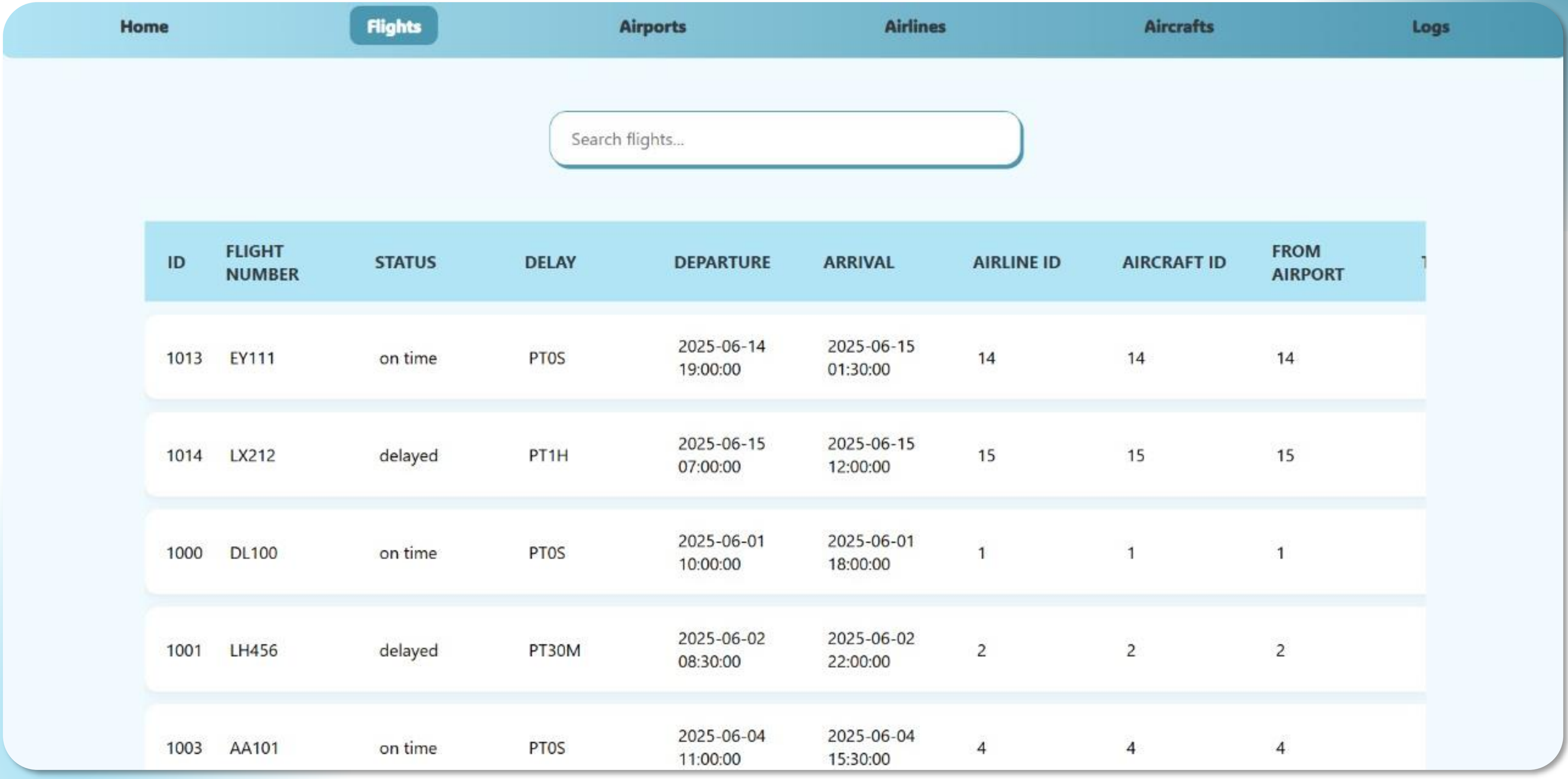
Departure
6/15/2025, 7:00:00 AM

Arrival
6/15/2025, 12:00:00 PM

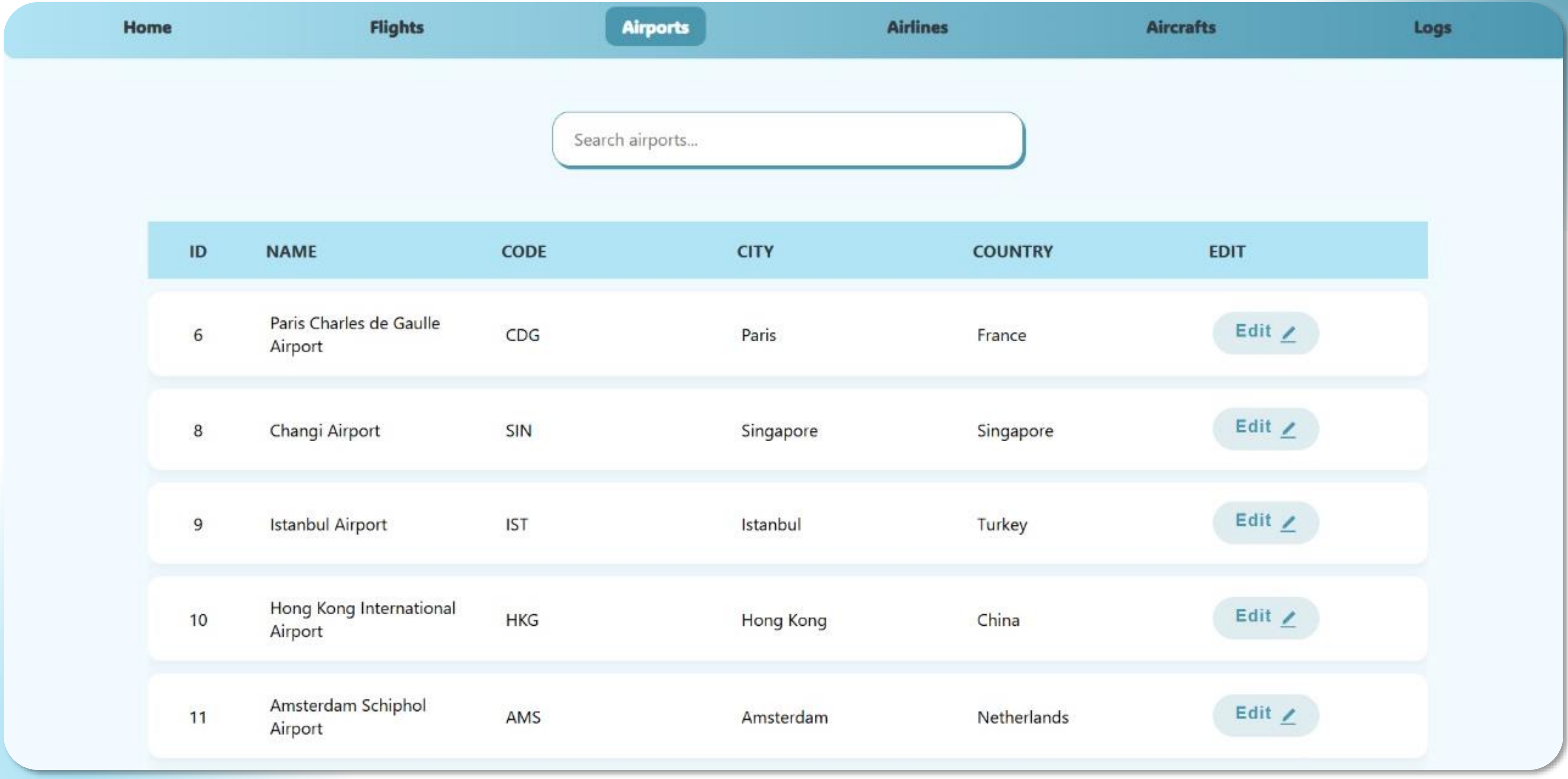
Flight: LX212
Status: delayed | Delay: PT1H

JFK
New York
John F. Kennedy

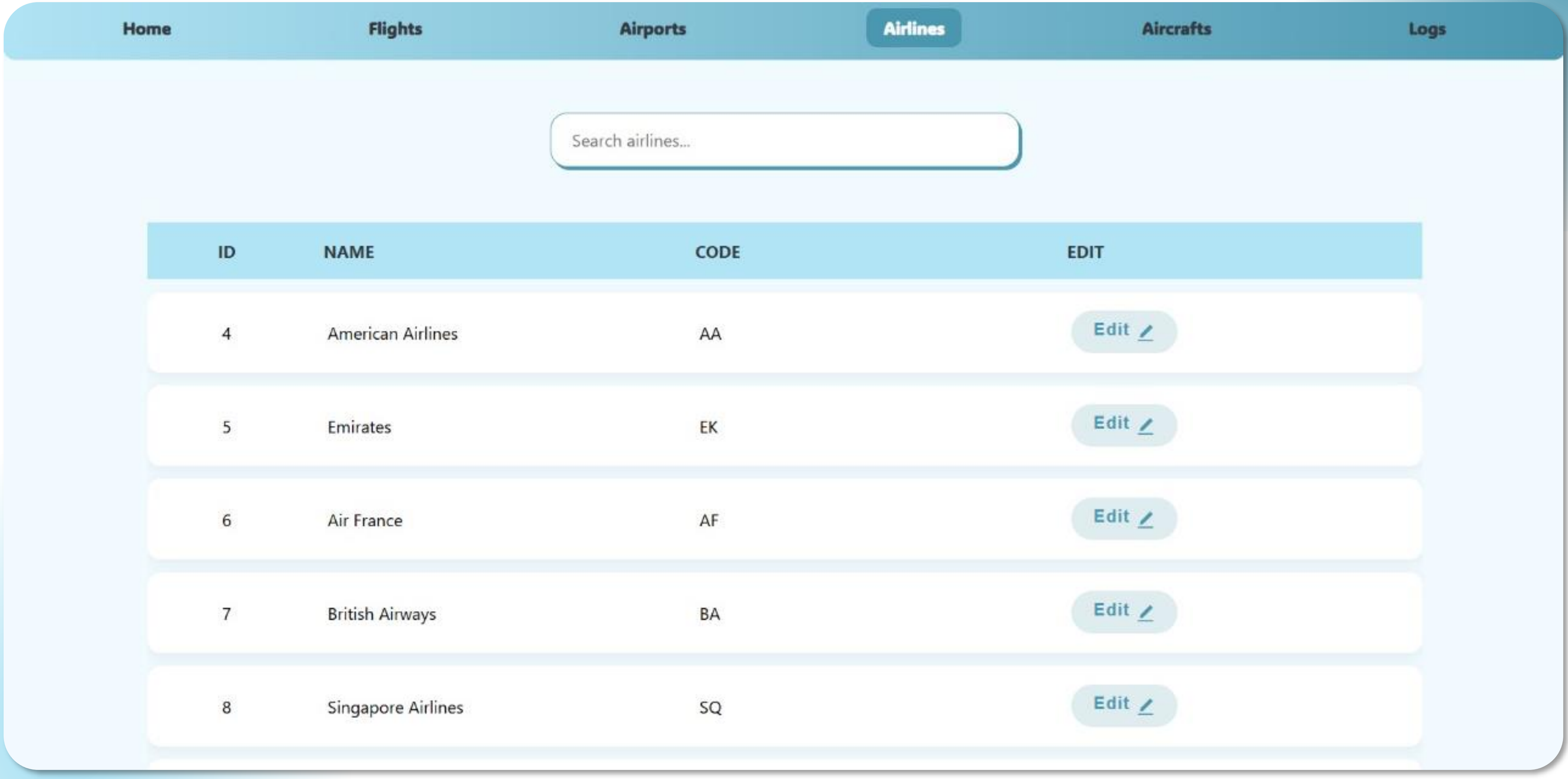
FRA
Frankfurt
Frankfurt am



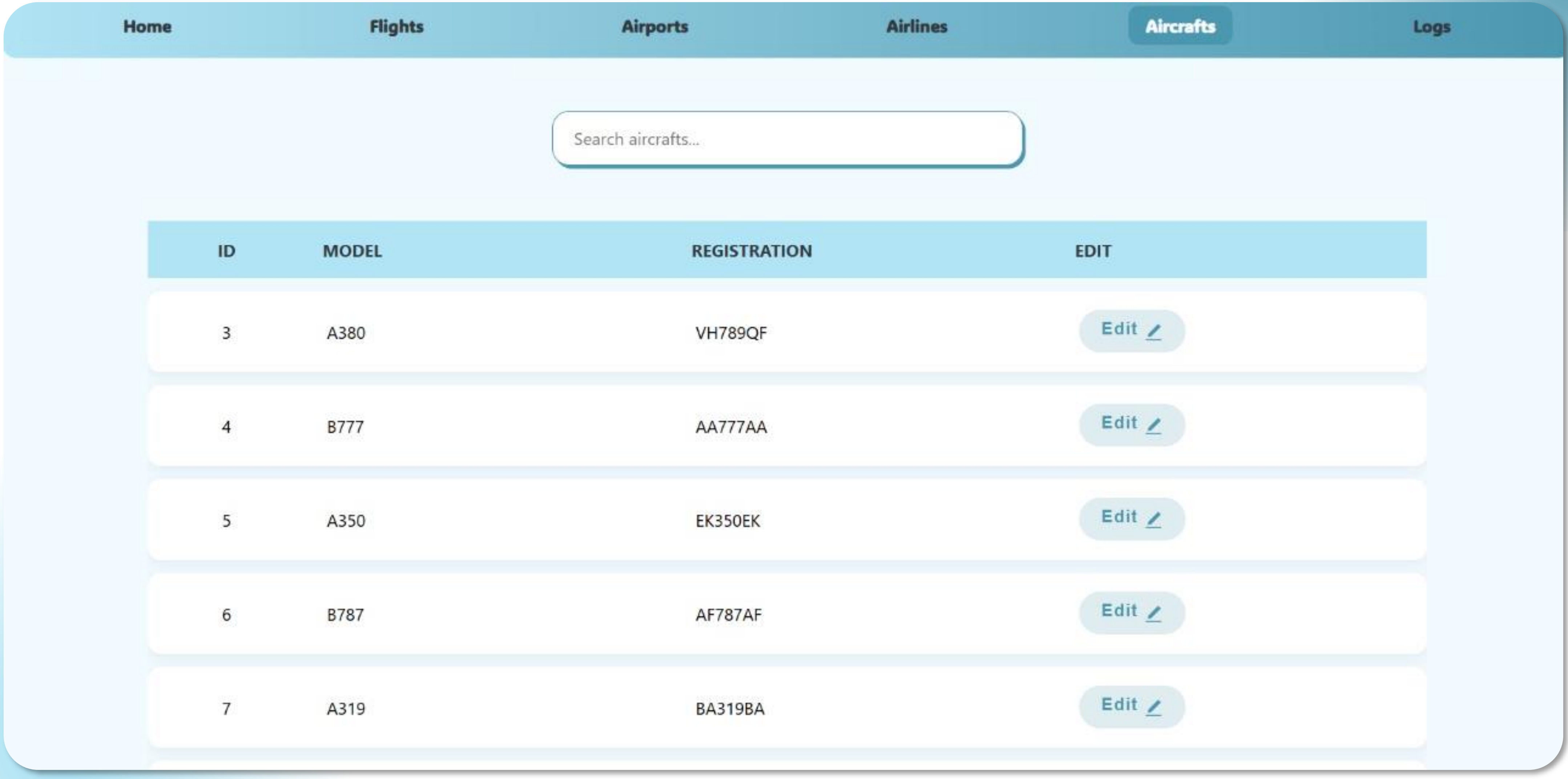
ID	FLIGHT NUMBER	STATUS	DELAY	DEPARTURE	ARRIVAL	AIRLINE ID	AIRCRAFT ID	FROM AIRPORT
1013	EY111	on time	PT0S	2025-06-14 19:00:00	2025-06-15 01:30:00	14	14	14
1014	LX212	delayed	PT1H	2025-06-15 07:00:00	2025-06-15 12:00:00	15	15	15
1000	DL100	on time	PT0S	2025-06-01 10:00:00	2025-06-01 18:00:00	1	1	1
1001	LH456	delayed	PT30M	2025-06-02 08:30:00	2025-06-02 22:00:00	2	2	2
1003	AA101	on time	PT0S	2025-06-04 11:00:00	2025-06-04 15:30:00	4	4	4



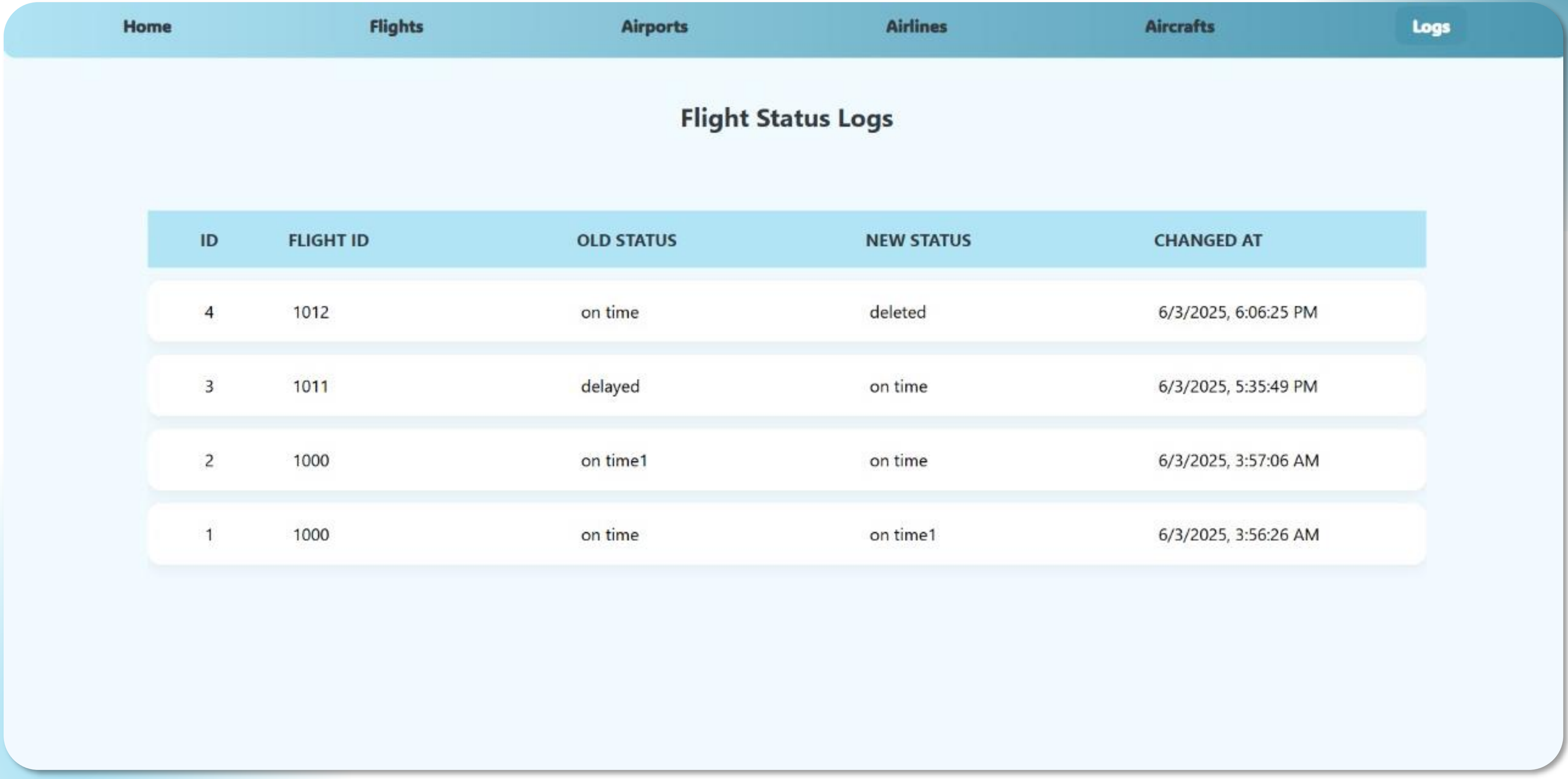
ID	NAME	CODE	CITY	COUNTRY	EDIT
6	Paris Charles de Gaulle Airport	CDG	Paris	France	Edit
8	Changi Airport	SIN	Singapore	Singapore	Edit
9	Istanbul Airport	IST	Istanbul	Turkey	Edit
10	Hong Kong International Airport	HKG	Hong Kong	China	Edit
11	Amsterdam Schiphol Airport	AMS	Amsterdam	Netherlands	Edit



ID	NAME	CODE	EDIT
4	American Airlines	AA	Edit
5	Emirates	EK	Edit
6	Air France	AF	Edit
7	British Airways	BA	Edit
8	Singapore Airlines	SQ	Edit



ID	MODEL	REGISTRATION	EDIT
3	A380	VH789QF	Edit
4	B777	AA777AA	Edit
5	A350	EK350EK	Edit
6	B787	AF787AF	Edit
7	A319	BA319BA	Edit

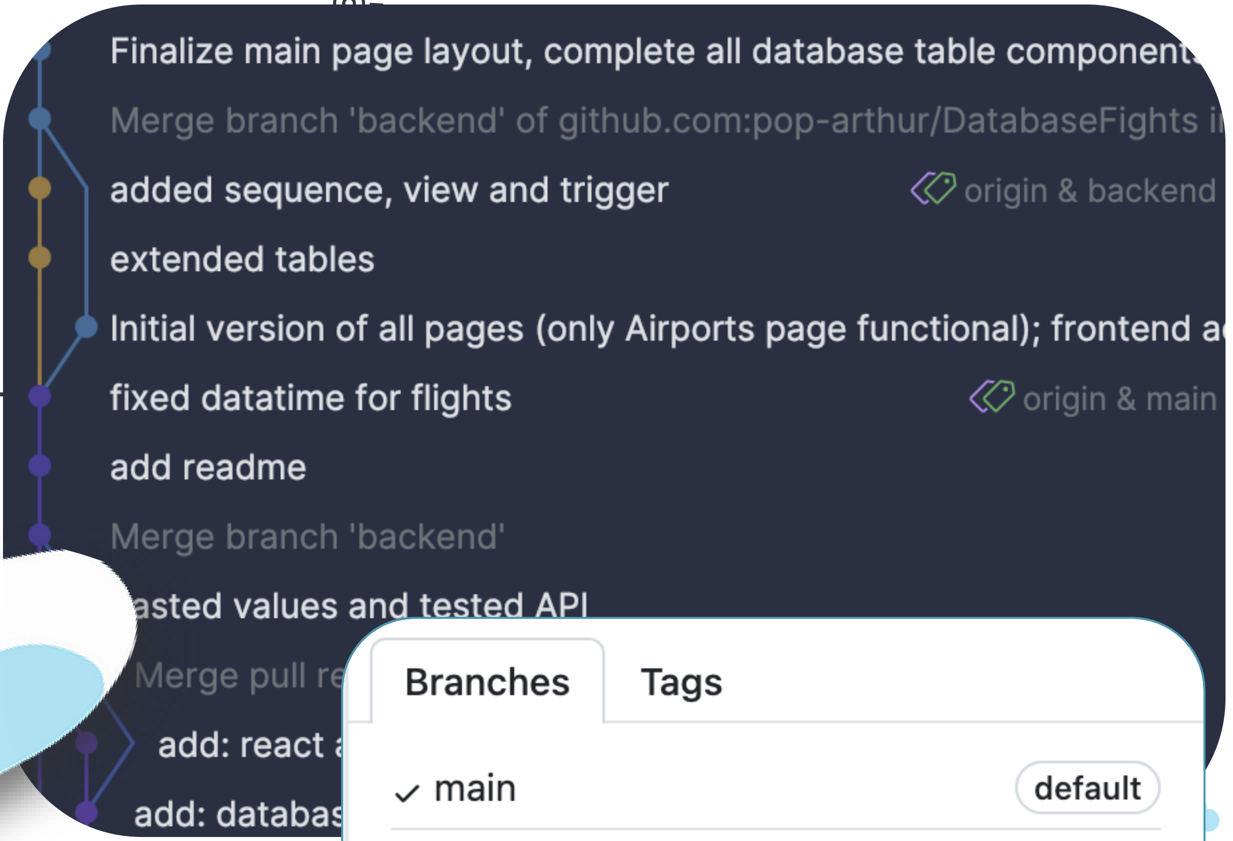


Flight Status Logs

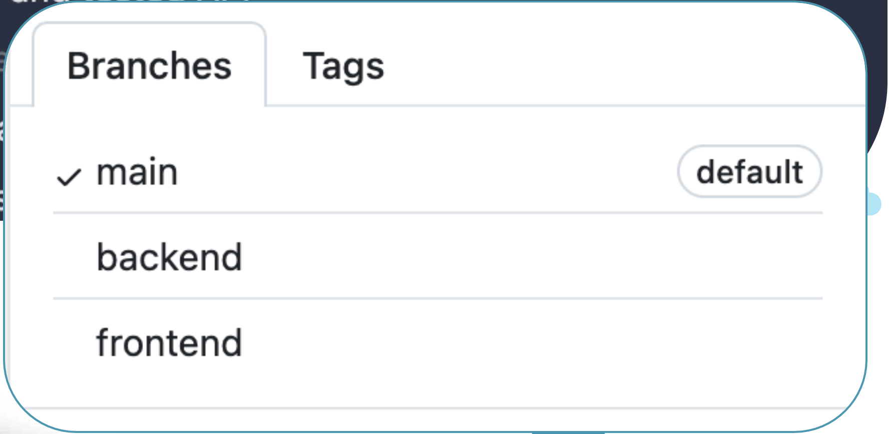
ID	FLIGHT ID	OLD STATUS	NEW STATUS	CHANGED AT
4	1012	on time	deleted	6/3/2025, 6:06:25 PM
3	1011	delayed	on time	6/3/2025, 5:35:49 PM
2	1000	on time1	on time	6/3/2025, 3:57:06 AM
1	1000	on time	on time1	6/3/2025, 3:56:26 AM

Github

https://github.com/pop-arthur/DatabaseFights



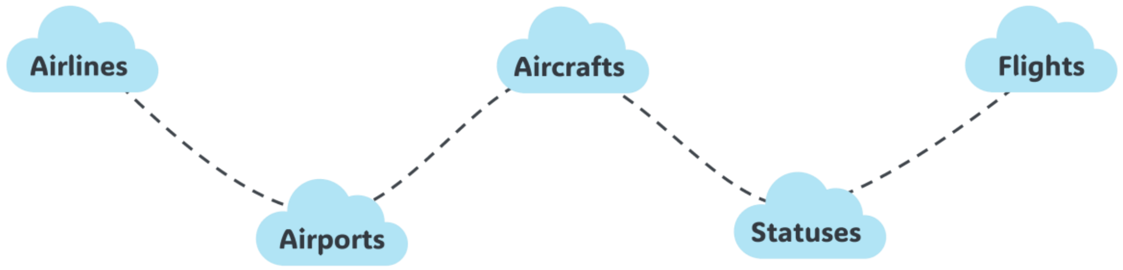
Finalize main page layout, complete all database table component
Merge branch 'backend' of github.com:pop-arthur/DatabaseFights i
added sequence, view and trigger origin & backend
extended tables
Initial version of all pages (only Airports page functional); frontend a
fixed datatime for flights origin & main
add readme
Merge branch 'backend'
asted values and tested API
Merge pull re
add: react
add: databas

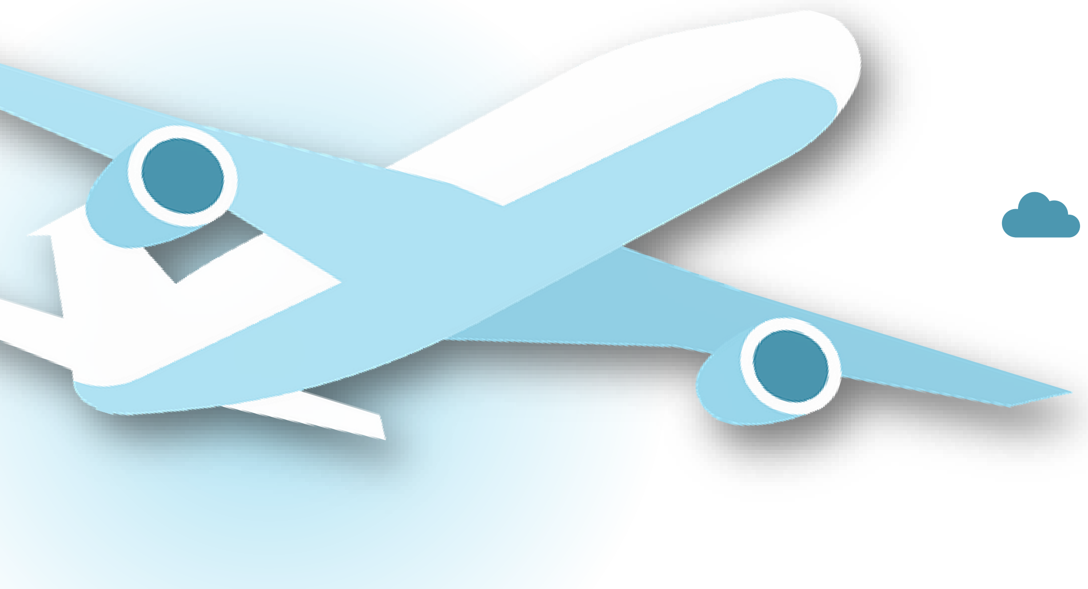


Branches Tags

✓ main default
backend
frontend

conclusion





- FlightTracker is a clean, well-structured database that connects flights, airlines, airports, aircrafts, and statuses

- Database models show clear relationships and normalized data — ready for real queries and simple app integration

- We're excited to turn this into a working flight info tool!

goodbye

Thank you!!

We are ready for your questions