



## **LAB MID ASSIGNMENT**

**BY:**

KUMAIL SHAH

FA23-BCS-004

# Flight Tracking System (FlightAware Simulation)

## 1. Introduction

This project simulates the FlightAware system, designed to collect, store, and visualize real-time flight data. It mimics aircraft radio signal ingestion using a RESTful FastAPI backend, stores the data in MongoDB, and visualizes active flights on an interactive Folium map.

## 2. System Overview

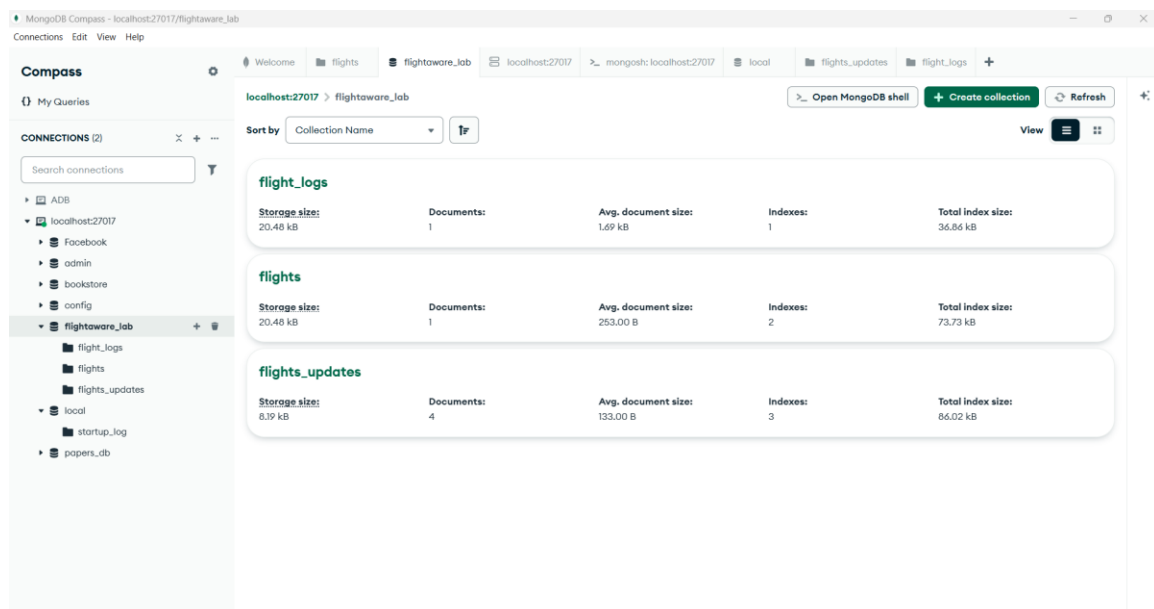
Components of the system include:

- FastAPI Backend — Handles data ingestion and retrieval.
- MongoDB Database — Stores flight and update data.
- Folium Visualization — Displays active flight paths and positions.

## 3. Database Design

MongoDB is used as the backend database. It contains three main collections:

1. flights — Stores master flight details such as flight\_id, origin, destination, and status.
2. flights\_updates — Contains real-time position updates for each flight.
3. flight\_logs — Keeps logs and historical data for auditing and analytics.



localhost:27017 > flightaware\_lab > flight\_logs

Documents 1 Aggregations Schema Indexes 1 Validation

Type a query: { field: 'value' } or [Generate query](#)

ADD DATA EXPORT DATA UPDATE DELETE

```
{
  "_id": "ObjectId('68f7a3ab4823e4cf2ed65e19')",
  "flight_id": "PK306",
  "status": "completed",
  "first_seen": "2025-10-21T15:06:05.235+00:00",
  "last_seen": "2025-10-21T16:01:27.379+00:00",
  "current_position": Object,
  "last_update": "2025-10-21T15:06:27.385861",
  "path": Array (12)
    0: Object
      flight_id: "PK306"
      timestamp: "2025-10-21T15:06:05.235+00:00"
      lat: 34.04555219500415
      lon: 71.53623520095651
      altitude_ft: 10000
      heading: 108
      speed_kts: 460
    1: Object
    2: Object
    3: Object
    4: Object
    5: Object
    6: Object
    7: Object
    8: Object
    9: Object
    10: Object
    11: Object
  "duration_min": 55
}
```

localhost:27017 > flightaware\_lab > flights

Documents 1 Aggregations Schema Indexes 2 Validation

Type a query: { field: 'value' } or [Generate query](#)

ADD DATA EXPORT DATA UPDATE DELETE

```
{
  "_id": "ObjectId('68f7a0cc264b32990e2e6228')",
  "flight_id": "PK300",
  "status": "enroute",
  "first_seen": "2025-10-21T15:03:40.295+00:00",
  "last_seen": "2025-10-21T15:59:02.452+00:00",
  "current_position": Object,
  "last_update": "2025-10-21T15:04:02.457478"
}
```

```
{
  "_id": "ObjectId('68f7a0e4264b32990e2e6235')",
  "flight_id": "PK301",
  "status": "enroute",
  "first_seen": "2025-10-21T15:04:04.463+00:00",
  "last_seen": "2025-10-21T15:59:26.590+00:00",
  "current_position": Object,
  "last_update": "2025-10-21T15:04:26.598337"
}
```

```
{
  "_id": "ObjectId('68f7a0fc264b32990e2e6242')",
  "flight_id": "PK302",
  "status": "enroute",
  "first_seen": "2025-10-21T15:04:28.604+00:00",
  "last_seen": "2025-10-21T15:59:50.751+00:00",
  "current_position": Object,
  "last_update": "2025-10-21T15:04:50.759073"
}
```

```
{
  "_id": "ObjectId('68f7a114264b32990e2e624f')",
  "flight_id": "PK303"
}
```

localhost:27017 > flightaware\_lab > flights\_updates

Documents 4 Aggregations Schema Indexes 3 Validation

Type a query: { field: 'value' } or [Generate query](#)

ADD DATA EXPORT DATA UPDATE DELETE

```
{
  "_id": "ObjectId('68f7a0cc264b32990e2e6227')",
  "flight_id": "PK300",
  "timestamp": "2025-10-21T15:03:40.295+00:00",
  "lat": 33.65260279456022
  "lon": 73.09091440716712
  "altitude_ft": 10000
  "heading": 210
  "speed_kts": 387
}
```

```
{
  "_id": "ObjectId('68f7a0ce264b32990e2e6229')",
  "flight_id": "PK300",
  "timestamp": "2025-10-21T15:08:42.322+00:00",
  "lat": 32.880848694339
  "lon": 72.49143584276564
  "altitude_ft": 12272
  "heading": 79
  "speed_kts": 401
}
```

```
{
  "_id": "ObjectId('68f7a0d0264b32990e2e622a')",
  "flight_id": "PK300",
  "timestamp": "2025-10-21T15:13:44.341+00:00",
  "lat": 32.116019694180174
  "lon": 71.92811873852546
  "altitude_ft": 14545
  "heading": 301
  "speed_kts": 342
}
```

## 4. API Endpoints

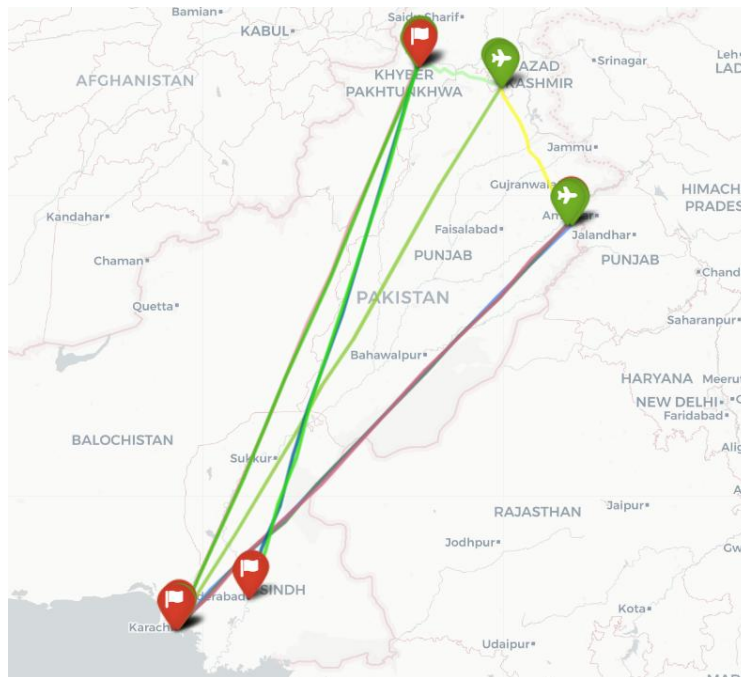
Endpoint	Method	Description
/api/ingest	POST	Ingests new flight telemetry data.
/api/active	GET	Returns all currently active flights.
/api/track/{flight_id}	GET	Retrieves the flight path and updates.
/api/complete/{flight_id}	GET	Marks flight as completed and archives it.

## 5. Data Flow

1. Flight data is received via POST /api/ingest.
2. Data is validated and stored in MongoDB.
3. Active flights can be queried using GET /api/active.
4. Folium uses these endpoints to visualize positions on a live map.

## 6. Visualization

Folium is used to create interactive maps showing flight paths. Each active flight is represented as a polyline, and its latest position is shown with a red marker. The start position is green. Multiple flights are plotted simultaneously.



## 8. Implementation Details

- Framework: FastAPI
- Database: MongoDB
- Visualization: Folium (Python)
- Deployment: Localhost with Uvicorn server
- Tools: MongoDB Compass, cURL/Postman for API testing

### Active:

Curl

```
curl -X 'GET' \
  'http://127.0.0.1:5000/api/active' \
  -H 'accept: application/json'
```

Request URL

http://127.0.0.1:5000/api/active

Server response

Code	Details
200	<div>Response body</div> <div><pre>{   "active_flights": [     {       "flight_id": "PK300",       "status": "enroute",       "first_seen": "2025-10-21T15:03:40.295000",       "last_seen": "2025-10-21T15:59:02.452000",       "current_position": {         "lat": 24.902128678955552,         "lon": 67.03355818233992,         "altitude_ft": 35000,         "heading": 143       },       "last_update": "2025-10-21T15:04:02.457478"     },     {       "flight_id": "PK301",       "status": "enroute",       "first_seen": "2025-10-21T15:04:04.463000",       "last_seen": "2025-10-21T15:59:26.590000",       "current_position": {         "lat": 24.838959288349404,         "lon": 67.03661789940418,         "altitude_ft": 35000,         "heading": 119       },       "last_update": "2025-10-21T15:04:26.598337"     }   ] }</pre></div>

### Track:

GET /:

Parameters

Name

flight\_id \*

string

(path)

Curl

```
curl -X 'GET' \
  'http://127.0.0.1:5000/api/track/PK308' \
  -H 'accept: application/json'
```

Request URL

http://127.0.0.1:5000/api/track/PK308

Server response

Code	Details
200	<div>Response body</div> <div><pre>{   "flight_id": "PK308",   "path": [     {       "flight_id": "PK308",       "timestamp": "2025-10-21T15:06:53.537000",       "lat": 34.8625807605171,       "lon": 71.54694957237547,       "altitude_ft": 10000,       "heading": 79,       "speed_kts": 337     },     {       "flight_id": "PK308",       "timestamp": "2025-10-21T15:11:55.550000",       "lat": 33.237845635819895,       "lon": 71.26465916101074,       "altitude_ft": 12272,       "heading": 189,       "speed_kts": 309     },     {       "flight_id": "PK308",       "timestamp": "2025-10-21T15:16:57.566000",       "lat": 32.444728852320004,       "lon": 70.99586179378648,       "altitude_ft": 14545, </pre></div>

Complete:

POST

/api/complete/{flight\_id}

Complete Flight

Parameters

Name	Description
<div><div>flight_id</div><div>* required</div></div> <div>string</div> <div>(path)</div>	<div>PK309</div>

Ingest:

POST

/api/ingest

Ingest Update

Parameters

No parameters

Request body

required

Edit Value

Schema

```
{
  "flight_id": "string",
  "timestamp": "2025-10-21T15:57:39.626Z",
  "lat": 0,
  "lon": 0,
  "altitude_ft": 0,
  "heading": 0,
  "speed_kts": 0
}
```

Execute

## 9. Conclusion

The Flight Tracking System demonstrates real-time data ingestion, storage, and visualization using modern Python frameworks. It provides a strong foundation for building scalable, production-grade aviation analytics or tracking systems.