Flight Board Solution Overview

What and why?

Your task is to design a system to implement a Flight Information Display system at an Airport - you know, one of those information boards that show when flights are departing. The goal isn't to have expert domain knowledge (so no points for researching other solutions), but to propose a solution to the problem as described and have discussions around tradeoffs. You can assume any technology stack or implementation that you're comfortable with, just note them down. There are more requirements than you can probably solve for in the time allotted - a complete solution that misses requirements is preferred over an incomplete solution that attempts to boil the ocean.

Functional Overview

Problem Statement

The task is to design a system that meets the following requirements:

1. For this exercise, we'll consider ONLY departing flights (not arrivals as well)
2. There is a flight schedule, which defines when the regularly scheduled flights occur - for example, "Air New Zealand has a flight NZ0128 that flies to Melbourne (MEL) at 6:30am on Monday, Wednesday and Friday"
3. The airlines keep the schedule up to date when they make schedule changes.
4. The flight display has a list of upcoming departures.
5. Each flight has the following properties
    a. An Airline
    b. Flight Number
    c. Destination
    d. Scheduled Departure Time
    e. Estimated Departure Time
    f. Actual Departure Time
    g. Flight Status, which is one of:
        i. On Time
        ii. Check In
        iii. Boarding
        iv. Departed
        v. Cancelled
        vi. Delayed
    h. Departure Gate
        i. A Departure Gate is only assigned to a flight once it transitions to the "Boarding" flight status
6. The big ticker board in the airports will get the information from your system over a web API.
7. The flight information needs to be viewable over the internet (so people can check their flight status before coming to the airport).
8. The internet accessible view of flight information must deal with very large traffic spikes. This is needed for when a lot of people are checking their flight's status (e.g. a storm or other event occurs).
9. Passengers can subscribe to a particular flight and receive push notifications when it's status or details change.
10. Airlines must not be able to update the flight information for other airlines.
11. The interface to update the flight information must not be accessible to the internet

Deliverables

1. A design for how the data should be modeled.

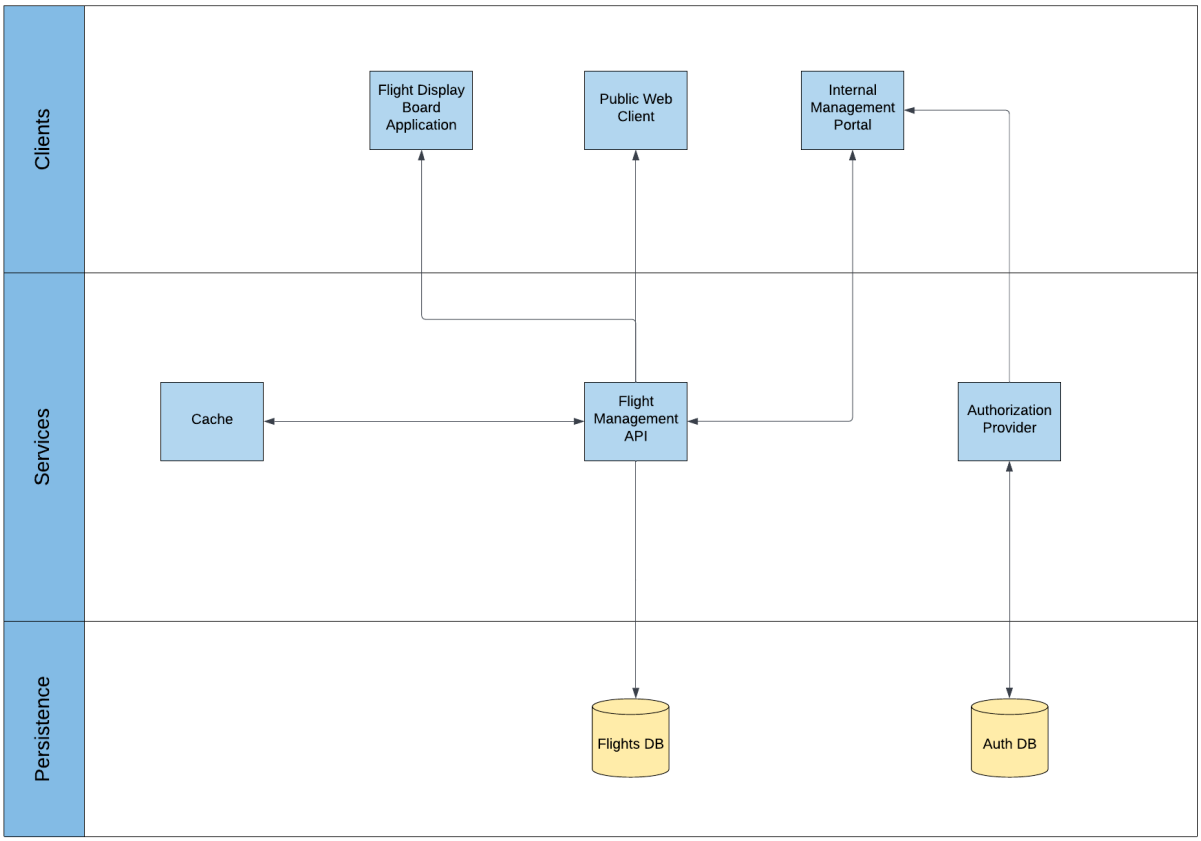
2. A design for how the system would be broken down into components and what each of those components would do, along with technology choice.
3. A description of which requirements are met and not met, any trade-offs considered and any assumptions made.
4. A rough estimate for how long it would take you to implement the system described.

Estimates

User story / feature (in the order of importance)	Estimate, man hours	Notes
Implement Flight Board service with the flights data model and data access	4	
Implement public API that will serve flight information over the Internet	4	
Implement Flight Board UI that with a view that will display flight information	2	
Package application by using Docker Compose, including a SQL Server instance	4	
Implement flight management API that will be hosted internally (no authentication and authorization)	2	
Implement flight management UI that won't have public access (no authentication and authorization)	8	
Add registration, authentication and authorization to the flight management UI and API	4	
Implement		
Introduce Caching service and add it to the public API (invalidate cache on flight updates)	4	
Add subscriptions and push notifications to the public client	8	
Add isolated tests and e2e tests to the UI	4	

Solution Architecture

The diagram below outlines the main application components and will constitute the future solution.



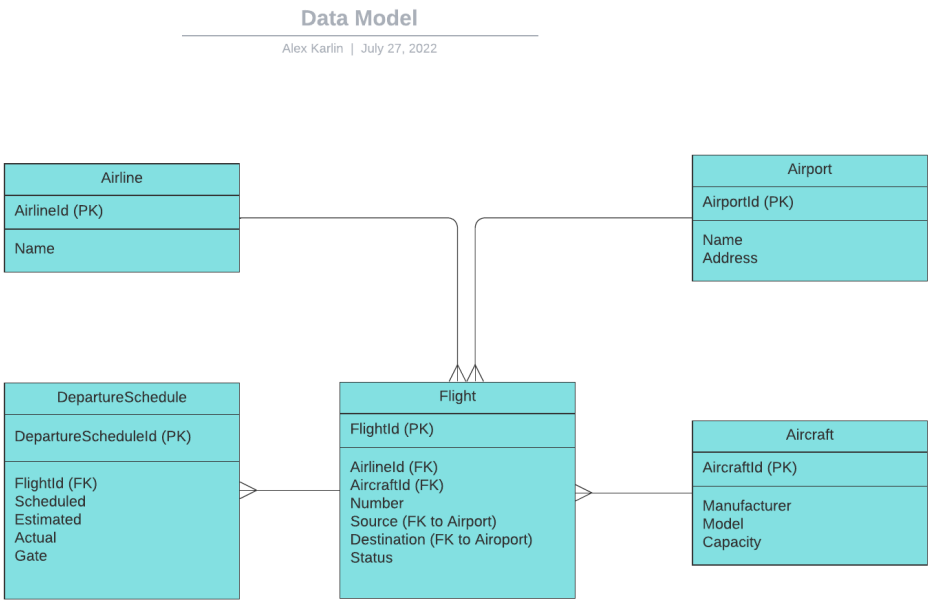
- Internal management portal will be used by authenticated airline workers to keep the schedule up to date. This will be an Intranet application
- Flight display board application will be used show a list of upcoming departures
- Public web client will be used to view flight information over the Internet

- And external cache will be used to deal with large traffic spikes
- Web Push Notifications will be used to be notified about flight details changes
- Authentication and authorization will be used to manage access rights to the airline data, e.g. airlines must not be able to update the flight information for other airlines

Web Push Notifications

TBD - can add considerable scope of work

Data Model



Technical Constraints

- Flight display board application will require knowledge of specific hardware and will be *OUT OF SCOPE*
- Authorization Server will take a while to implement and to save time we may have to physically host it with the API and use some predefined users / roles to start with
- Web Push Notifications piece has the most unknowns at this point and potentially will add considerable amount of work so we will leave this part until the end (*potential OUT OF SCOPE*)

Deployment Architecture

- Docker Compose will be used to simplify deployments during development, feature testing and production

Test Strategy and Approach

Key functional areas to focus on

- Public web client
  - View flight information
  - Flight information get updated on browser refresh when there were changes
  - Subscribe for flight push notifications
  - Receive push notifications for specific flights
- Internal flight management portal
  - Authenticate
  - View flight information specific to the user's airline
  - Don't see flights from other airlines

- Edit flights for user’s airline
- Cannot edit flights for other airlines (parameter tempering)

Non functional areas to focus on

- Performance of the applications should meet the expectations (*TODO:* define the average response time indicator)
- applications be responsive to the devices

Environments and testing scope

- Testing (manual and automation) will be performed on user stories