Software Engineer Take-Home Assignment: Flight Search & Aggregation System

Overview

Build a flight search and aggregation system that combines flight data from multiple airline APIs, processes and filters results, and returns optimized search results to users. The system should handle real-world challenges like varying data formats, caching, price changes, and performance optimization.

Requirements

Core Functionality

1. **Aggregate Flight Data from Multiple Sources**
   - Fetch flight data from multiple airline/provider APIs (mock APIs provided)
   - Normalize data into a common format
   - Handle different response structures and data formats
2. **Search & Filter Capabilities**
   - Search by origin, destination, and date
   - Filter by: price range, number of stops, departure/arrival time, airlines, duration
   - Sort by: price (lowest/highest), duration (shortest/longest), departure time, arrival time
3. **Price Comparison & Ranking**
   - Compare prices across providers for the same flight
   - Calculate total trip duration including layovers
   - Rank results based on "best value" (combination of price and convenience)
4. **Handle Data Inconsistencies**
   - Deal with different time formats and time zones
   - Handle missing optional fields
   - Validate flight data (e.g., arrival time after departure time)

Technical Requirements

- Clean, production-ready code with proper error handling
- Clear separation of concerns (providers, aggregation, filtering, caching)
- API Performance (Complexity, etc.)
- Documentation (README with setup and usage instructions)

Mock Data

Search Request Structure (JSON)

```
{

  "origin": "CGK",

  "destination": "DPS",

  "departureDate": "2025-12-15",

  "returnDate": null,

  "passengers": 1,

  "cabinClass": "economy"

}
```

Provider's List (Attached):

- Garuda Indonesia API Response (garuda_indonesia_search_response.json)
- Lion Air API Response (lion_air_search_response.json)
- Batik Air API Response (batik_air_search_response.json)
- AirAsia API Response (airasia _search_response.json)

Mock Provider APIs

You should simulate these providers as separate functions/services that return the mock data above. To simulate real-world conditions:

- **Garuda Indonesia**: Fast response (50-100ms delay)
- **Lion Air**: Medium response (100-200ms delay)
- **Batik Air**: Slower response (200-400ms delay)
- **AirAsia**: Fast but occasionally fails (90% success rate, 50-150ms delay)

Expected Output Format

After aggregation and normalization, your system should return results in a unified format as attached.

Constraints

- You can use any libraries/packages you prefer
- Focus on backend logic; no UI required
- Mock the API calls (no actual HTTP requests needed)

Bonus Points (Optional)

- Implement "best value" scoring algorithm (price + convenience)
- Support for round-trip searches
- Handle timezone conversions properly (WIB, WITA, WIT)
- Add rate limiting for provider APIs
- Implement retry logic with exponential backoff for failed providers
- Support for currency display (IDR formatting with thousands separator)
- Parallel provider queries with timeout handling
- Support for multi-city searches

Submission

Please provide:

1. Source code (GitHub repository or ZIP file)
2. README with setup and run instructions
3. Brief explanation of your design choices

Good luck! If you have any questions about the requirements, feel free to ask.