<u>CAA Coding Challenge</u>

This assessment is designed to evaluate your ability to follow instructions and consider the overall impact of your code. The goal is not to trick you, but to gain insight into your approach to software development.

You have the flexibility to work at your own pace, but please be aware that **<u>the deadline is "23rd August 2024, 5 PM NZST",</u>** and extensions will not be provided. You can search for solutions, refer to documentation, but make sure you fully understand any code you didn't originally create before incorporating it.

Keep in mind that your submission will be evaluated based on the expectations for the role you've applied for, so it's important to clearly demonstrate the key skills and objectives required.

Task Name: Flight Information API

Task Description:

You are tasked with creating a RESTful API for managing flight information using C# and .NET Core 8. The API should follow best practices and industry standards, implementing CRUD (Create, Read, Update, Delete) operations for flights. Additionally, you should write automated test cases to ensure the API's functionality.

Task Requirements:

1. Create a Flight model with the following properties:

   - Id (int)

   - FlightNumber (string)

   - Airline (string)

   - DepartureAirport (string)

   - ArrivalAirport (string)

   - DepartureTime (DateTime)

   - ArrivalTime (DateTime)

   - Status (enum: Scheduled, Delayed, Cancelled, InAir, Landed)

2. Implement the following API endpoints:

   - GET /api/flights: Retrieve all flights

- GET /api/flights/{id}: Retrieve a specific flight by ID
- POST /api/flights: Create a new flight
- PUT /api/flights/{id}: Update an existing flight
- DELETE /api/flights/{id}: Delete a flight
- GET /api/flights/search: Search flights by various criteria (e.g., airline, departure/arrival airport, date range)

3. Use Entity Framework Core for data persistence with an in-memory database for simplicity.
4. Implement proper error handling and return appropriate HTTP status codes.
5. Implement best practices for RESTful API design.
6. Apply the SOLID principles in your code.
7. Implement logging using ILogger or Nlog.
8. Implement input validation using data annotations or a validation library of your choice.
9. Write automated test cases using xUnit or NUnit, Mock dependencies where appropriate to isolate tests (e.g., using Moq).
10. Add Swagger/OpenAPI documentation.

Deliverables:

- A .NET Core 8 Web API project implementing the Flight Information API.
- Provide your solution as a GitHub repository with clear instructions on how to run the API and execute the tests.
- A suite of automated tests validating the API's functionality and reliability.
- API documentation accessible via Swagger UI or similar.

Evaluation Criteria

- ✓ Code organization and structure
- ✓ Proper use of C# and .NET Core 8 features
- ✓ Adherence to RESTful principles
- ✓ Implementation of required features
- ✓ Quality and coverage of unit tests
- ✓ Error handling and input validation

- ✓ Use of design patterns and SOLID principles
- ✓ Code readability and commenting