Hello 😊

Based on your expertise, please contribute as much as you can. Don't worry if there are areas where you might not be as knowledgeable—your effort is appreciated!

We have two positions to fill: one for a mid-level developer and another for a junior developer.

If you're more skilled with frontend development find creating REST APIs challenging, use the provided JSON data to display results as described. Conversely, if you're stronger in backend development, focus on that part instead.

If you have any questions, feel free to reach out to me on LinkedIn.

Good luck and have fun on the way.

Jonas

Deadline:

11:00 AM

23-12-2024

If you have problems meeting the deadline, please contact me.

Submission

1. Push your code to a public GitHub repository.
2. Include a README.md file with instructions to run your application locally.
3. Share the repository link

Backend Developer Test: Hotel API

Overview

Your task is to develop a simple REST API that will serve data for a frontend application. The goal is to demonstrate your backend development skills, including how you structure code, implement logic, and handle edge cases. This test focuses on building a C# API using ASP.NET Core to provide hotel data from a JSON file.

Requirements

1. **Application Overview**
   Develop an HTTP API to serve hotel-related data. This API will be consumed by a frontend application. The frontend expects the following functionality:
   - Retrieve a list of hotels.
   - Retrieve details of a specific hotel using its ID.
2. **Endpoints**
   Implement the following endpoints:
   - **GET /api/hotels**: Fetches a list of all hotels.
   - **GET /api/hotels/{id}**: Fetches details of a single hotel by its ID.
     - If the hotel does not exist, return a 404 status code with an appropriate error message.
3. **Data Source**
   - Use the provided hotels.json file as your data source. Do not implement a database.
   - The JSON file contains an array of 10 hotels, with fields such as name, location, rating, image URL, dates of travel, board basis, and room details.
4. **Expected Behavior**
   - Serve hotel data from the JSON file in response to API requests.
   - Return appropriate HTTP status codes for success and failure scenarios.
   - Handle invalid requests gracefully (e.g., missing IDs, malformed inputs).
5. **Error Handling**
   - Ensure the API handles errors such as:

        - Hotel not found (404).
        - Server or data reading errors (500).
    - Include clear and descriptive error messages in the response.

6. **Technical Constraints**
    - Use C# and ASP.NET Core to build the API.
    - Use only the provided JSON file for data storage; no database is required.
7. **Demonstration**
    - Use git for version control. Push your solution to a public GitHub repository.
    - Be prepared to:
        - Run the application locally during the review session.
        - Walk through the code and explain your approach.
        - Answer questions about design decisions, edge cases, and testing.

Additional Notes

- **Code Quality**: Write clean, well-documented, and maintainable code.
- **Performance**: Optimize data handling for efficient processing.
- **Flexibility**: Structure the code in a way that would make it easy to add a database or new features in the future.

JSON File

You'll be provided a hotels.json file with 10 hotel records. Each record includes the following fields:

- id: Unique identifier for the hotel.
- name: Name of the hotel.
- location: Hotel location.
- rating: A float value between 1.0 and 5.0 representing the hotel rating.
- imageUrl: URL for the hotel's image.
- datesOfTravel: An array of strings specifying available dates.
- boardBasis: Description of the meal plan (e.g., "All Inclusive").
- rooms: An array of room objects with roomType and amount.

Example record:

json

Copy code

```
{
 "id": 1,
 "name": "Seaside Paradise",
 "location": "Maldives",
 "rating": 4.9,
 "imageUrl": "https://example.com/images/seaside-paradise.jpg",
 "datesOfTravel": ["2024-01-01", "2024-01-07"],
 "boardBasis": "All Inclusive",
 "rooms": [
  {
   "roomType": "Deluxe Suite",
   "amount": 5
  },
  {
   "roomType": "Family Room",
   "amount": 3
  }
 ]
}
```

Evaluation Criteria

You will be assessed on the following:

1. **Code Quality**: How clean, readable, and maintainable is your code?
2. **API Design**: Are endpoints properly implemented and adhering to REST principles?
3. **Error Handling**: How well do you handle edge cases and errors?
4. **Adherence to Requirements**: Does your API meet the functional specifications?
5. **Presentation**: Can you explain your code, decisions, and approach clearly during the review?

Frontend Developer Test: Hotel SPA

Overview

Your task is to create a single-page application (SPA) using React and TypeScript. The application will fetch and display hotel information provided by a backend API. This exercise is designed to assess your ability to build a basic functional frontend application, focusing on the use of React, TypeScript, and simple design principles.

What You Need to Do

You will build a small React application that:

1. Fetches a list of hotels from a backend API and displays it.
2. Shows detailed information about a single hotel when selected.

Even if you're not experienced with every part of this task, focus on writing clean and functional code and keeping your solution simple.

What We're Looking For

- **Clean Code**: Easy-to-read and well-structured code.
- **Functionality**: The app should work as described.
- **Error Handling**: Your app should handle things like loading states or missing data gracefully.
- **Basic UI Design**: A simple, user-friendly design is sufficient. It doesn't have to be fancy.

Requirements

1. Routing

- Use **React Router** (or a similar tool) for navigation.
- Create two main pages:
  - **Hotels List Screen**: Available at /hotels
  - **Hotel Detail Screen**: Available at /hotels/:id

  - Any other URL should redirect the user to /hotels.

2. Hotels List Screen (/hotels)

- Fetch a list of hotels from the API endpoint: GET /api/hotels.
- Display the following details for each hotel:
  - Hotel Name
  - Location
  - Rating
  - Image
  - Dates of Travel
  - Board Basis
  - Number of Rooms and Room Types
- Show a **loading spinner or message** while the data is being fetched.
- Create a simple card or list layout to display the hotels.
- Clicking on a hotel should take the user to its detail screen.

3. Hotel Detail Screen (/hotels/:id)

- Fetch detailed hotel information from the API endpoint: GET /api/hotels/{id}.
- Display all the available details for the selected hotel (e.g., name, rating, location, image, etc.).
- Show a **loading spinner or message** while the data is being fetched.
- If the hotel doesn't exist (e.g., invalid ID), show an error message like "Hotel Not Found."

4. Redirects

- If a user tries to access any route that doesn't exist, redirect them to /hotels.

How to Approach This

If You're a Beginner:

- Focus on getting the main functionality working:
  1. Displaying the list of hotels.
  2. Navigating to the detail page for a specific hotel.
  3. Showing loading indicators while fetching data.
- Keep the UI design simple and focus on functionality first.
- Use built-in React state (useState, useEffect) for data management.

- Use the browser’s console to debug and test your code.

If You’re More Experienced:

- Consider breaking your app into reusable components (e.g., a HotelCard for list items).
- Consider using TypeScript (not required) to define interfaces for API responses (e.g., Hotel, Room).
- Structure your code to handle future scalability (e.g., a separate services folder for API calls).
- Add extra features like error boundaries or better navigation.

Tools You Can Use

For a Simple Approach:

- **React**: Use useState and useEffect to manage state and lifecycle events.
- **React Router**: For navigation between the list and detail screens.
- **CSS**: For simple styling, you can use plain CSS or CSS Modules.
- **Fetch API**: To make API calls.

For an Enhanced Approach:

- **Axios**: An alternative to the Fetch API for cleaner HTTP requests.
- **TypeScript**: Use TypeScript interfaces to type your data.
- **CSS Frameworks**: Use a library like Tailwind or Bootstrap to style the UI quickly.

JSON Data

```
[
 {
  "id": 1,
  "name": "Seaside Paradise",
  "location": "Maldives",
  "rating": 4.9,
  "imageUrl": "https://example.com/images/seaside-paradise.jpg",
  "datesOfTravel": ["2024-01-01", "2024-01-07"],
  "boardBasis": "All Inclusive",
  "rooms": [
   {
     "roomType": "Deluxe Suite",
     "amount": 5
   },
   {
     "roomType": "Family Room",
     "amount": 3
   }
  ]
 },
 {
  "id": 2,
  "name": "Mountain Retreat",
  "location": "Swiss Alps",
  "rating": 4.7,
  "imageUrl": "https://example.com/images/mountain-retreat.jpg",
  "datesOfTravel": ["2024-02-15", "2024-02-22"],
  "boardBasis": "Bed & Breakfast",
  "rooms": [
   {
     "roomType": "Standard Room",
     "amount": 10
   }
  ]
 },
 {
  "id": 3,
  "name": "Urban Oasis",
  "location": "New York City, USA",
  "rating": 4.5,
  "imageUrl": "https://example.com/images/urban-oasis.jpg",
  "datesOfTravel": ["2024-03-10", "2024-03-17"],
  "boardBasis": "Room Only",
  "rooms": [
   {
    "roomType": "Luxury Suite",
    "amount": 2
   },
   {
    "roomType": "Standard Room",
    "amount": 20
   }
  ]
 },
 {
  "id": 4,
  "name": "Desert Dream",
  "location": "Dubai, UAE",
  "rating": 4.8,
  "imageUrl": "https://example.com/images/desert-dream.jpg",
  "datesOfTravel": ["2024-04-01", "2024-04-10"],
  "boardBasis": "Half Board",
  "rooms": [
   {
    "roomType": "Luxury Villa",
    "amount": 10
   },
   {
      "roomType": "Family Room",
      "amount": 5
    }
  ]
},
{
  "id": 5,
  "name": "Tropical Escape",
  "location": "Bali, Indonesia",
  "rating": 4.6,
  "imageUrl": "https://example.com/images/tropical-escape.jpg",
  "datesOfTravel": ["2024-05-01", "2024-05-10"],
  "boardBasis": "All Inclusive",
  "rooms": [
    {
      "roomType": "Beach Villa",
      "amount": 8
    },
    {
      "roomType": "Garden Room",
      "amount": 12
    }
  ]
},
{
  "id": 6,
  "name": "Historic Haven",
  "location": "Rome, Italy",
  "rating": 4.4,
  "imageUrl": "https://example.com/images/historic-haven.jpg",
  "datesOfTravel": ["2024-06-01", "2024-06-07"],
    "boardBasis": "Breakfast Included",
    "rooms": [
      {
        "roomType": "Classic Room",
        "amount": 15
      },
      {
        "roomType": "Luxury Suite",
        "amount": 5
      }
    ]
  },
  {
    "id": 7,
    "name": "Safari Lodge",
    "location": "Serengeti, Tanzania",
    "rating": 4.9,
    "imageUrl": "https://example.com/images/safari-lodge.jpg",
    "datesOfTravel": ["2024-07-10", "2024-07-20"],
    "boardBasis": "Full Board",
    "rooms": [
      {
        "roomType": "Luxury Tent",
        "amount": 10
      },
      {
        "roomType": "Family Suite",
        "amount": 3
      }
    ]
  },
{
 "id": 8,
 "name": "Ocean Breeze",
 "location": "Gold Coast, Australia",
 "rating": 4.3,
 "imageUrl": "https://example.com/images/ocean-breeze.jpg",
 "datesOfTravel": ["2024-08-01", "2024-08-10"],
 "boardBasis": "Self Catering",
 "rooms": [
  {
   "roomType": "Ocean View Suite",
   "amount": 6
  },
  {
   "roomType": "Family Room",
   "amount": 8
  }
 ]
},
{
 "id": 9,
 "name": "Rainforest Retreat",
 "location": "Costa Rica",
 "rating": 4.7,
 "imageUrl": "https://example.com/images/rainforest-retreat.jpg",
 "datesOfTravel": ["2024-09-01", "2024-09-15"],
 "boardBasis": "All Inclusive",
 "rooms": [
  {
   "roomType": "Jungle Bungalow",
   "amount": 7
```

```
    },
    {
      "roomType": "Standard Room",
      "amount": 10
    }
  ]
 },
 {
  "id": 10,
  "name": "Island Bliss",
  "location": "Hawaii, USA",
  "rating": 4.8,
  "imageUrl": "https://example.com/images/island-bliss.jpg",
  "datesOfTravel": ["2024-10-01", "2024-10-10"],
  "boardBasis": "Full Board",
  "rooms": [
    {
      "roomType": "Beachfront Suite",
      "amount": 12
    },
    {
      "roomType": "Standard Room",
      "amount": 15
    }
  ]
 }
]
```