IDEAS | PEOPLE | TRUST

Solutions Development

Developer Technical Test

December 2023

BDO

Contents

1. Developer Technical Test

1.1. Overview

We have put this scenario together to see how you might deal with a real-world coding task. **At your interview we would like you to present your solution (via screen share) to the tasks**, as you would do in a sprint review, and show us anything of note in your code. There are no right or wrong answers, the aim is to see the approaches and patterns you use to solve the problem and for us to discuss the pros and cons during the interview. We will expect you to present for 20-30 minutes with questions along the way from us.

Make sure you read through everything here before you get started.

Spend as much time as you would like to on the scenario but no more than 2.5 hours on the front end and 2.5 hours on the back end.

1.2. Scenario

Application Name: Weather Dashboard

Objective: Create a dashboard that displays weather information for a user-specified location.

1.3. Front-End (ReactJS)

1.3.1. User Interface

- Create a search bar for entering a location.
- Create a display area showing weather details: temperature, humidity, wind speed, and a weather icon.
- Create a component to set a default location.
- BONUS: Make the design responsive for desktop and mobile viewing.

1.3.2. Functional Requirements

- Update the weather display based upon user input.
- Allow users to set and change the default location, which is shown on application load.
- Ensure there is error handling for invalid city names or data fetching issues.

1.3.3. Technical Requirements

- Use functional components and React hooks.
- Implement state management effectively.
- Include unit tests for components.

1.4. Back-End (C#)

1.4.1. API Integration

- Create a RESTful API to interact with the front-end.
- Integrate with the OpenWeatherMap API for fetching weather data. If you have problems with the public API either select an alternative or create an endpoint that returns some weather data.

1.4.2. Functional Requirements

- API endpoint to search for weather by city name.
- API endpoint for users to set a default location.
- BONUS: Implement caching to optimize API calls.

1.4.3. Technical Requirements

- Use ASP.NET Core for the back-end.
- Follow best practices for error handling and logging.
- Include suitable unit tests for this functionality.

1.5. Documentation

1.5.1. Provide a README file

- Setup and installation instructions.
- Brief documentation of the API endpoints.
- Any necessary API keys or credentials (if applicable).

1.6. Submission Guidelines

- Submit the complete source code via a GitHub repository.
- Ensure the code is well-commented and follows clean code principles.
- Include a brief reflection on the chosen architecture and any challenges faced.

2. Evaluation Criteria

2.1. Code quality

Clarity, maintainability, and adherence to standard practices.

2.2. Functionality

How well the application works and meets the requirements.

2.3. Problem Solving

Effectiveness in handling edge cases and potential errors.

2.4. Design and Architecture

Structure of the application and separation of concerns.

2.5. Testing

Adequacy and coverage of unit tests.

FOR MORE INFORMATION:

JIM MCCORMICK

+44 (0)20 3833 3098
jim.mccormick@bdo.co.uk

www.bdo.co.uk