sinistar

Test Technique

(English below)

Objectif

Le principal objectif de ce test technique est d'évaluer la maîtrise du candidat en React, Material UI, TypeScript, développement d'algorithmes et intégration d'API tierces (Google Maps JS SDK). Le candidat devra démontrer sa capacité à construire une application web conviviale qui trie et affiche dynamiquement les options d'hébergement temporaires en fonction de critères définis par l'utilisateur.

Aperçu du test

Développer une application React qui aide les utilisateurs à trouver leur logement idéal en triant une liste d'options en fonction de différents critères. L'application doit utiliser Material UI comme principal framework d'interface utilisateur et doit être écrite en TypeScript. Elle comprend un algorithme de tri qui évalue les logements en fonction de la distance, du taux de réponse de l'hôte, du score des avis et de la flexibilité d'extension.

Exigences

1. Configuration de l'application React

- Initialiser une nouvelle application React en utilisant Create React App avec le modèle TypeScript.
- Installer et configurer Material UI pour styliser l'application.

2. Intégration de Material UI

- Utiliser les composants Material UI pour créer une interface utilisateur propre et moderne.

3. Utilisation de TypeScript

- Utiliser TypeScript pour tous les composants et utilitaires afin de garantir la sécurité des types et la fiabilité du code.

4. Algorithme de tri des logements

- Développer un algorithme de tri qui évalue une liste de logements (fournie sous forme de fichier JSON) en fonction d'un système de notation pondéré. Ce système devrait permettre d'assigner un poids entre 0 et 100 pour chaque critère et de calculer un score, sur 100, pour chaque propriété.

- Les pondérations des critères de tri (distance, taux de réponse de l'hôte, score des avis, flexibilité d'extension) doivent être ajustables via l'interface utilisateur.
- Mettre en œuvre une fonctionnalité d'ajustement dynamique des pondérations pour permettre aux utilisateurs de prioriser différents critères en fonction de leurs préférences.

5. Composants de l'interface utilisateur

- Un champ de saisie pour entrer une adresse, avec l'intégration de l'API JavaScript de Google Maps pour la saisie automatique des adresses.
- Des contrôles UI pour ajuster le poids de chaque critère de tri.
- Un bouton pour déclencher le re-tri des logements en fonction des poids mis à jour.
- Une liste des logements affichant les détails clés (nom, adresse, note de l'hôte).
- Une carte affichant les emplacements des logements listés, en utilisant l'API JavaScript de Google Maps.

6. Considérations supplémentaires

- Écrire du code propre et maintenable en suivant les meilleures pratiques.

Livrables

- Code source de l'application React terminée (sans node_modules).
- Un fichier README avec des instructions sur la façon de configurer et d'exécuter l'application, y compris toutes les clés d'API ou configurations nécessaires.
- Une brève explication de la logique de l'algorithme de tri et du raisonnement derrière les poids choisis pour les critères de tri.

Critères d'évaluation

- Correction et efficacité de l'algorithme de tri.
- Utilisation efficace de React et Material UI.
- Qualité du code, y compris sa lisibilité, sa structure et son respect des bonnes pratiques TypeScript.
- Conception et expérience utilisateur de l'application.
- Intégration et utilisation correctes de l'API JavaScript de Google Maps.

Outils

Utiliser cette clé API pour Google Maps SDK:
AIzaSyCvTXZStHgV1E4eFBzXXXOeVyKBLHfq2uU

sinistar

Technical Test

Objective

The primary goal of this technical test is to assess the candidate's proficiency in React, Material UI, TypeScript, algorithm development, and integration of third-party APIs (Google Maps JS SDK). The candidate will demonstrate their ability to build an user-friendly web application that dynamically sorts and displays accommodation options based on user-defined criteria.

Test overview

Develop a React application that helps users find their ideal accommodation by sorting a list of options based on various criteria. The application should utilize Material UI as the main UI framework and be written in TypeScript. It will include a sorting algorithm that evaluates accommodations based on distance, host response rate, review score, and extension flexibility.

Requirements

1. React Application Setup

- Initialize a new React application using Create React App with TypeScript template.
- Install and configure Material UI to style the application.

2. Material UI Integration

- Use Material UI components to create a clean and modern UI.
- Ensure the application is accessible.

3. TypeScript Usage

- Utilize TypeScript for all components and utilities to ensure type safety and code reliability.

4. Accommodation Sorting Algorithm

- Develop a sorting algorithm that evaluates a list of accommodations (provided as a JSON file) based on a weighted scoring system. This system should allow for assigning a weight between 0 and 100 for each criterion and calculate a score, out of 100, for each property.
- The weights for the sorting criteria (distance, host response rate, review score, extension flexibility) should be adjustable through the UI.
- Implement dynamic weight adjustment functionality to allow users to prioritize different criteria according to their preferences.

5. User Interface Components

- An input field for entering an address, integrating Google Maps JavaScript API for address autocomplete.
- UI controls for adjusting the weight of each sorting criterion.
- A button to trigger the re-sorting of accommodations based on the updated weights.
- A map displaying the locations of the listed accommodations, utilizing Google Maps JavaScript API.
- A list view of the accommodations showing key details (e.g., name, address, host rating).

6. Additional Considerations

- Write clean and maintainable code following best practices.

Deliverables

- Source code of the completed React application (without node_modules).
- A README file with instructions on how to set up and run the application, including any necessary API keys or configurations.
- A brief explanation of the sorting algorithm's logic and the rationale behind the chosen weights for the sorting criteria.

Evaluation Criteria

- Correctness and efficiency of the sorting algorithm.
- Effective use of React and Material UI.
- Code quality, including readability, structure, and adherence to TypeScript best practices.
- Design and user experience of the application.
- Proper integration and use of the Google Maps JavaScript API.

Tools

Use this API key for Google Maps SDK: AIzaSyCvTXZStHgV1E4eFBzXXXOeVyKBLHfq2uU