MID-TERM CHALLENGES

CASO SAFETEEFY

Safeteefy.io es un startup que busca ofrecer a la comunidad global una plataforma por medio de la cual, los ciudadanos de una ciudad reporten emergencias, con el fin de que las autoridades tomen conocimiento rápidamente y realicen las acciones correspondientes según el caso.

Dentro de la filosofía de Safeteefy.io, se busca empoderar a los ciudadanos, por lo cual usa el concepto de ***Guardian*** para referirse a un ciudadano que reporta emergencias. Por otro lado, a cada reporte de emergencia el equipo de Safeteefy.io lo referencia como ***Urgency***.

Mientras un equipo se encuentra abocado a la creación de un RESTful API que brinde soporte a las operaciones de Safeteefy.io, usted se incorpora al equipo que desarrolla una aplicación web ***SafeteeCall*** (con nombre de proyecto ***safetee-call***), que brindará soporte al personal del call-center. Este será un canal secundario de atención, pues el proyecto incluye aplicaciones con las que los guardians interactuarán directamente.

La información que se preserva de los ***Guardians*** incluye:

id (number), obligatorio, autogenerado,
username (string), obligatorio, máximo 30 caracteres,
email (string), obligatorio, máximo 30 caracteres,
firstName (string), máximo 60 caracteres,
lastName (string), máximo 60 caracteres,
gender (string), obligatorio,
address (string).

La información de los ***Urgencies*** incluye **id** (number, autogenerado), **title** (string, obligatorio), **summary** (string), **latitude** (double, obligatorio), **longitude** (double, obligatorio), **reportedAt** (Date, auto-poblado por el sistema).

Los ***Urgencies*** están asociados de forma exclusiva a un solo ***Guardian***.

Business Rules:

- Un Guardian no puede crear dos Urgencies con el mismo título desde la misma geolocalización (latitude, longitude) en la misma fecha de reporte.
- Un Guardian debe ser tener valor en firstName y lastName para poder registrar un reporte.

Backend features:

Durante la etapa de desarrollo, le asignan trabajar el módulo de SafetyCall que interactúa en específico sobre dos endpoints del backend:

```
/api/guardians
/api/guardians/{guardianId}/urgencies
```

Guardians Endpoint (/api/guardians)
Implementa las operaciones en el RESTful API para poder obtener todos, agregar, obtener un Guardian por Id, actualizar un Guardian existente y eliminar un Guardian (esto último implica eliminar todos los Urgencies creados por el Guardian).

Urgencies Endpoint (/api/guardians/{guardianId}/urgencies)
Implementa las operaciones sobre los Urgencies, como obtener todos los Urgencies asociados a un Guardian, obtener un Urgency específico por Id, así como la creación, actualización o eliminación de Urgencies.

Sólo puede crear Urgencies para un Guardian existente. No se puede crear Urgencies de forma independiente, pues un Urgency es dependiente y pertenece a un Guardian. En caso se elimine un Guardian, ello implicará la eliminación en cascada de sus Urgencies asociados.
Para independizar su avance y no depender del equipo de backend, le indican que debe simular la existencia del api con json-server.

CHALLENGES

Debe crear una aplicación que contemple Responsive Web Design y que presente un navigation bar con el nombre de branding **SafeteeCall by Safeteefy.io** y las opciones **Home** y **Urgencies**.

La vista **Home** (con path /home) debe mostrar una tabla de datos con la información de los guardians actualmente registrados, permitiendo recorrer la lista. Cada fila de la tabla de datos tiene a la derecha una zona de actions, en la cual hay un botón **Set Current**, al hacer click en el botón se establece como Current Guardian al Guardian seleccionado. Cuando se selecciona un Current Guardian, su nombre debe aparecer en la parte superior derecha del navigation bar.

La vista **Urgencies** (con path /guardians/{guardianId}/urgencies) debe presentar una tabla de datos con la información de las urgencies, debiendo permitir que se realice las operaciones CRUD sobre la información de Urgencies para el Current Guardian (el cual fue seleccionado en la vista Home) cuyo nombre debe estar visible en la parte superior. Esto quiere decir que debe permitirse agregar, modificar, consultar y eliminar Urgencies. La información de la tabla debe soportar además paginación, ofreciendo al usuario la opción de visualizar 5, 10 o 15 elementos por página en la tabla de datos. La vista debe incluir la posibilidad de hacer search y sorting sobre las columnas mostradas. Dado que otro equipo se encargará de la integración con Maps, en esta aplicación para el CRUD se considerará el ingreso manual de latitude y longitude.

Restricciones técnicas: El equipo requiere que la interfaz de usuario esté basada en Material Design y los componentes de **Vuetify**, mientras que para la comunicación con el backend debe apoyarse en **axios** (https://github.com/axios/axios). La aplicación debe soportar in-app navigation y utilizar **VueRouter** para el manejo de routing en la aplicación. Debe usar **Vuex** para implementar el State Management de Guardians. La interfaz de usuario debe mostrar los textos en **inglés**.

Para su cliente no solo importa el cumplimiento de las características funcionales, sino el diseño de interfaz de usuario, así como la estructura del proyecto, aplicación de convenciones de nomenclatura de objetos de programación en inglés, convenciones de nomenclatura de Vue.js, organización y eficiencia del código, cumplimiento de restricciones técnicas. Igualmente, el equipo de Safeteefy.io toma en cuenta la aplicación de patrones de diseño.