Front End III

Clase 22: Evaluación final

¡Hola de nuevo! Nos acercamos al final de nuestro recorrido y para cerrar esta materia te proponemos un último desafío que integrará todo lo que aprendimos en el transcurso del cuatrimestre. ¡Tu momento ha llegado, es hora de ponerse a prueba!

Presentación de la actividad

Para esta evaluación final, pondremos en práctica todos los conocimientos que hemos adquirido en este track. Desde la creación y manejo de formularios, Hooks (como useState, useContext, useReducer), estilos, props, peticiones asincrónicas e integración con API, routing, testing y demás apartados fundamentales del ecosistema de React. Para ello, te pediremos que crees una aplicación web ligada a la temática de la salud. Integrando diferentes rutas, formularios de contacto, temas de estado, entre otros. Todo ello consumiendo información de una API (la tendrás disponible dentro del repo base).

Como punto de partida, utilizaremos un template con algunos lineamientos y apartados útiles que les servirán para agilizar su tarea. Pueden acceder al mismo desde el siguiente **link**.

Instrucciones y requisitos de entrega

1) Creación del Context

- Deberán crear un contexto global en su aplicación que ponga a disposición la siguiente información:
    - Tema de colores para toda la app (claro u oscuro).

    - Información traída por la API (pueden utilizar fetch o axios para realizar la llamada).

- Para el manejo y actualización del estado global deberán implementar el Hook **useReducer**.
    - Con dicho Hook, deberán implementar el cambio de tema de colores (con un botón en la app específico para ello), de claro a oscuro y viceversa.

2) Creación de rutas

Utilizando React Router deberán crear las siguientes direcciones:

- **/home**
    - Donde tendrán que renderizar una card por cada dentista devuelto por la API.
- **/dentist/:id**
    - Donde mostrarán en detalle la información de cada dentista (nombre, email, teléfono, website).
    - Cada card renderizada en la ruta **/home** debe ser un link hacia esta dirección.
- **/contacto**
    - Donde deberán implementar un form (con sus validaciones pertinentes) que capture la información del usuario que desea contactar a la empresa. Los campos serán los siguientes:
        - Nombre completo (con longitud mayor a 5).
        - Email (con formato correcto de email).
    - En caso de haber un error, mostrar el siguiente mensaje de error: “Por favor verifique su información nuevamente”.
    - Una vez enviado el formulario, deberán mostrar un mensaje de éxito que contenga el siguiente formato: “Gracias [nombre del usuario], te contactaremos cuando antes vía mail”.
- **/destacados**
    - Renderizar las cards pertinentes a los dentistas destacados. Dicha información deberá estar almacenada dentro del **localStorage** del browser.
- Homogéneo a todas las rutas, deberá mostrarse un navbar (con links de navegación y un botón para cambiar el theme) y footer.

3) Funcionalidad de destacados

- Cada card renderizada debe tener la opción de poder agregarse a destacados.
- Las cards destacadas deben guardarse en la sesión actual del localStorage.
- Manejar dicho comportamiento con el Hook **useReducer**.

4) Estilos

- Sobre la base del **tema** (claro u oscuro) consumido del contexto global, estilar las cuatro rutas de la app (asignar determinados estilos si es “dark” o “light”).

5) Testing

- Realizar por lo menos 5 tests cases que verifiquen alguna funcionalidad de la aplicación (tener en cuenta que no solo se evaluará la cantidad, sino a esta altura también la efectividad o relevancia del test case elegido).

Pasos a seguir

Para ayudar a que puedas organizarte y realizar la entrega a tiempo, te dejamos una propuesta de flujo de trabajo. De todas maneras, puedes organizarte de la manera que creas más conveniente para lograr el objetivo buscado.

Paso 1: Creación de rutas

Importa los componentes pertinentes para envolver la app y crear el enrutado de la página. No importa que aún no tengamos los componentes que irán en cada vista, por el momento podemos dejar solamente un div.

Paso 2: Creación de componentes para cada ruta

Una vez definidas las rutas demandadas por la consigna, pasemos a crear los componentes que se verán en cada URL. Recordemos que para la página **/contacto** deberás implementar un form con sus validaciones; para **/home**, mapear la información de la API y devolver una card por cada dentista; crear un link dinámico para el detalle de cada dentista en **/dentista/:id**; para **/destacados**, consumir la información del localStorage y devolver una card por cada dentista destacado.

Paso 3: Implementación del contexto global

Una vez que ya tenemos toda la estructura de nuestra aplicación, podemos pasar a consumir la API realizando una llamada por fetch o axios. Deberemos guardar dicha información en un contexto global, junto con el theme de la app. Crear los métodos necesarios para el manejo de su comportamiento mediante useReducer.

Paso 4: Consumir el contexto global

Con la información ya en mano, pasamos a “rellenar” los huecos que dejamos en nuestra estructura anterior, suscribiendo nuestros componentes al contexto global (solo aquellos que necesiten información del mismo).

Paso 4: Funcionalidad de destacados

Cada card deberá tener un botón que dispare la acción de guardar, dentro del localStorage, una card que queremos destacar. Deberemos utilizar useReducer para manejar esta lógica.

Paso 6: Estilado de componentes clave

Basándonos en el theme actual, deberíamos pasar determinada variante de colores a los componentes pertenecientes a las rutas (**Home.jsx**, **Favs.jsx**, **Detail.jsx**, **Contact.jsx**). No es obligatorio estilar más apartados, pero dicha opción tendrá peso a la hora de determinar la calificación final.

Paso 7: Testing

Por último, pero no menos importante, deberás realizar las pruebas asociadas a tu aplicación. Para ello deberás crear un nuevo archivo dentro del directorio **/src/test/**. Recordemos que si bien el número de tests cases importa, también su relevancia será decisiva para el cumplimiento de la consigna (no intentes hacer muchos tests fáciles solamente para llegar a la cantidad).

IMPORTANTE

- Para realizar la entrega deberás subir tu código a un repositorio y completar el siguiente formulario.
- Nuevamente, podrás utilizar como guía de progreso los tests que ya incluye el proyecto. Estos servirán luego como ayuda para el profesor a la hora de corregir, pero no determinan la nota final de la evaluación.
- Podrás encontrar más detalles en el README del proyecto base.