Challenge Frontend

Deberás desarrollar un cliente para un Blog.
El mismo debe ser armado en React, y consumir los datos de una JSON Placeholder, una API que expone datos ficticios en formato JSON.

Los endpoints que deberás utilizar son:

GET https://jsonplaceholder.typicode.com/posts
Devuelve un listado de posts

GET https://jsonplaceholder.typicode.com/posts/:id
Devuelve el detalle de un post en base al id especificado en el parámetro

POST https://jsonplaceholder.typicode.com/posts
Simula la creación de un nuevo post

PUT/PATCH https://jsonplaceholder.typicode.com/posts/:id
Simula la actualización de un post en base al id especificado en el parámetro

DELETE https://jsonplaceholder.typicode.com/posts/:id
Simula la eliminación de un post en base al id especificado en el parámetro

Requerimientos técnicos

Aprovechando las características de React, deberán crearse las siguientes secciones, y modularizar las mismas en componentes reutilizables.

Formulario de Login

El formulario se deberá renderizar al ingresar a cualquier ruta si el usuario no está autenticado, conteniendo los campos:

- Email.
- Password.
- Botón de “Enviar”.

Al hacer click en “Enviar”, se deberá validar que ambos campos no estén vacíos, y mostrar un mensaje al usuario si lo estuviesen. Caso contrario, se deberá realizar una petición POST a la **siguiente url,** con los campos email y password en el BODY.

Los datos válidos para obtener un token son:

- Email: challenge@alkemy.org
- Password: react

En el caso de obtener un error de la API, se deberá mostrar una alerta, mientras que si es satisfactorio deberá redirigir al Home y almacenar el token obtenido en localStorage.

Las validaciones del formulario deberán realizarse utilizando la librería **Formik**.

Home

Mostrará un listado de posts. En este listado, deberá mostrarse solamente el título de cada uno, y las acciones para ir al detalle del mismo, editarlo o eliminarlo.

Detalle

Deberá recibir el identificador de un post y, en el caso de que exista, mostrar sus datos. Caso contrario, deberá mostrar un mensaje de error.

Formulario de creación

Deberá mostrar un formulario que permita crear un nuevo post. El formulario deberá contener los campos título y contenido, y realizar la validación de los mismos (ambos son obligatorios). Al hacer el submit, debe realizarse la petición al endpoint correspondiente.

Formulario de Edición

Deberá recibir el identificador de un post y mostrar un formulario que permita editarlo. En el caso de que no exista, mostrar un mensaje de error. El formulario deberá contener los campos título y contenido, y realizar la validación de los mismos (ambos son obligatorios). Al hacer el submit, debe realizarse la petición al endpoint correspondiente.

Otras consideraciones

La app deberá contener un encabezado con los links al Home y al Formulario de Edición.
Debe ser responsive, se puede utilizar una plantilla.

La acción de “Eliminar” que contendrán los posts listados en la sección Home deberán realizar la petición al endpoint correspondiente.
La gestión del estado puede realizarse de la forma que prefieran, como así también la lógica de navegación.

Criterios a evaluar

- Diseño responsive, moderno e intuitivo.

- Debe utilizarse Bootstrap para permitir que el proyecto sea responsive, y media queries para los elementos personalizados que se desarrollen.
- Conocimientos básicos de React.
- Validación de formularios utilizando la librería Formik.
- Buenas prácticas de codificación.
- Buenas prácticas para nombre de rutas.
- Código modularizado en componentes reutilizables e independientes.