## Descripción del Problema

Se desea crear una Single Page Application (SPA) dedicada a la gestión de una tienda de cosméticos. Este proyecto incluye la implementación de funcionalidades clave como la autenticación de usuarios, gestión de rutas protegidas y persistencia de sesión utilizando tecnologías modernas JS, HTML5 y JavaScript. Además, se deberá simular una base de datos utilizando `json-server` para realizar operaciones CRUD, asegurando la consistencia y la integridad de los datos. Esta SPA proporcionará una experiencia completa tanto para los administradores, que podrán gestionar productos y pedidos, como para los visitantes, que podrán explorar y comprar productos.

## Funcionalidades Generales

Sistema de Autenticación:

- Registro de usuarios (rol administrador, rol visitante).

- Login de usuarios.

- Protección de rutas mediante un guardián en `Router.js`.

Persistencia de Sesión:

- Uso del Local Storage para mantener la sesión iniciada.

Consistencia de Datos:

- Uso de `json-server` para simular una base de datos y manejar operaciones CRUD.

Estructura del Proyecto:

- Generar una organización a nivel estructural de los archivos, para generar una separación de responsabilidades y que sea más sencillo navegar en el proyecto. Generar las carpetas que se consideren necesarias, como `assets`, `components`, `utilities`, `services`, entre otros.

Sistema de Administración:

- Apartado para usuario administrador que permita la gestión completa de productos (Crear, Actualizar, Eliminar, Obtener).

Vista General Usuario Visitante:

- Los usuarios visitantes tendrán una vista en la que solo se les permitirá ver los productos disponibles. Se les debe bloquear el acceso a las acciones de Crear, Editar y Eliminar.

## Tipos de Usuarios

Usuario Administrador:

- En el archivo `db.json`, debe existir un usuario con el rol de administrador por defecto. Este usuario podrá iniciar sesión con este rol.

- El usuario administrador tendrá las capacidades de editar y eliminar productos.

Usuario Visitante:

- Los visitantes podrán explorar productos y realizar compras.

- En la misma vista, los visitantes podrán visualizar sus pedidos.

## Lógica de Rutas

Si el usuario no está autenticado e intenta acceder a una de las siguientes rutas, deberá ser redirigido a una página personalizada `not-found.js`.  
Si el usuario ya se encuentra autenticado e intenta acceder a la ruta `/login` o `/register`, deberá ser redirigido a la ruta raíz del dashboard `/dashboard`.

## Persistencia

Estructura del Archivo `db.json`:

Productos: Con información preestablecida bajo la siguiente interfaz:

- `id`: number

- `name`: string

- `description`: string

- `price`: number

- `stock`: number

- `category`: string

Roles: Con dos valores fijos: `Usuario Visitante` y `Usuario Administrador`. Debe cumplir la siguiente interfaz:

- `id`: number

- `role`: string

Usuarios: Inicialmente vacío, excepto por el usuario administrador. Debe cumplir la siguiente interfaz:

- `id`: number

- `name`: string

- `email`: string

- `password`: string

- `role`: string

Pedidos: Inicialmente vacío. Debe cumplir la siguiente interfaz:

- `id`: number

- `userId`: number

- `products`: array of product ids

- `total`: number

## LocalStorage

## Vistas

Vista de Registro - Path: `/register`:

En la vista de registro, se le pedirá al usuario que proporcione la siguiente información:

- Nombre

- Correo electrónico

- Fecha de nacimiento

- Contraseña

Criterios de Aceptación:

- Se debe verificar que el correo electrónico sea válido desde la etiqueta HTML y desde JavaScript, utilizando una función llamada `emailValidator()` que retorna un booleano. Esta función debe evaluar que incluya un arroba `@` y un punto `.`.

Vista de Login - Path: `/login`:

En la vista de login se le pedirá al usuario la siguiente información:

- Correo electrónico

- Contraseña

Criterios de Aceptación:

- Si el correo electrónico y contraseña son correctos, se debe ingresar a un dashboard de layout propio que tendrá un navbar fijo con los siguientes ítems en el menú: `Productos`, `Pedidos`.

Vista de Home - Path: `/dashboard`:

La vista de HOME tendrá las siguientes opciones comunes para ambos roles:

- Sección superior: Con título “Productos actuales”, el cual debe mostrar todos los productos en una tarjeta cada uno, con información completa (la disposición de los elementos queda a tu criterio), a excepción del id.

Para rol Administrador:

- Para cada fila de la sección común “Todos los productos”, solo al administrador se le habilitarán dos botones: `editar` y `eliminar`.

- Si el usuario administrador presiona `editar`, lo debe redirigir a la vista de edición del producto en cuestión.

- Si el usuario administrador presiona `eliminar`, una alerta de tipo confirmación debe dispararse y preguntar si realmente desea eliminar el producto. Acto seguido, se eliminará del archivo `db.json`.

- Tendrá, justo debajo de la vista de todos los productos, un botón “crear producto” que debe redirigir a la vista de “crear productos”.

Para rol Visitante:

- Para cada fila de la sección común “Todos los productos”, solo al usuario visitante se le habilitará un botón “comprar” en cada fila de productos.

- Si el usuario da click en el botón “comprar”, un pop-up de tipo confirm aparecerá y le informará al usuario si realmente quiere comprar el producto.

Vista de Crear Productos - Path: `/dashboard/products/create`:

Esta vista solo debe estar habilitada para el rol “administrador”.

Una vez dentro, el administrador verá los campos a rellenar del formulario:

- `Product Name`: Nombre del producto

- `Description`: Descripción del producto

- `Price`: Precio del producto

- `Stock`: Stock del producto

- `Category`: Categoría del producto

Vista de Editar Productos - Path: `/dashboard/products/edit`:

Esta vista solo debe estar habilitada para el rol “administrador”.

Una vez dentro, el administrador verá los campos del formulario llenos por defecto, con base en el id.

- Solo se pueden editar los siguientes campos:

- `Descripción`

- `Precio`

- `Stock`

- `Categoría`