Entrega 3 AYD2 (20%)

UdeA – Juan Pablo Arango

Objetivo:

1. Interiorizar el proceso de desarrollo web usando NextJS, React, Tailwind, Supabase
2. Interiorizar el funcionamiento de la autenticación tanto en el front como en el back
3. Entender el funcionamiento de una app de administración usando funciones básicas de CRUD.

Requisitos del Proyecto:

El proyecto tiene como objetivo desarrollar una aplicación web de administración. La aplicación permitirá a los usuarios gestionar transacciones y movimientos, así como administrar usuarios con diferentes roles. La web se construirá utilizando tecnologías como NextJS, React, y TailwindCSS, y se integrará con una base de datos en Supabase.

Funcionalidades:

- Autenticación de usuarios (Manual, no es necesario realizar o usar una autenticación)
- Página de landing con opción para iniciar sesión
- Sidebar con enlaces a diferentes páginas (Inventarios, Materiales, Usuarios)
- Gestión de transacciones con visualización de movimientos y gráficas
- Gestión de maestros con opción para agregar nuevos
- Gestión de usuarios con roles diferenciados (ADMIN, USER)

Descripción de roles:

- ADMIN: Tiene acceso completo a todas las funcionalidades, incluida la gestión de usuarios y la creación de maestros.
- USER: Tiene acceso a la gestión de transacciones y maestros, pero no puede administrar usuarios ni crear nuevos maestros.

Instrucciones:

- Clonar el código de base desde el repositorio visto en clase, e instalar las dependencias.
- Crear una base de datos en Supabase.

- Crear un archivo .env en la raíz del proyecto, agregar la variable de entorno DATABASE_URL con el string de conexión a la base de datos creada anteriormente y ejecutar el comando npx prisma migrate dev --name migracion-inicial. Este comando creará todas las tablas en la base de datos.

- El proyecto debe contener las siguientes secciones:
    - Una página de landing con un botón para iniciar sesión
    - Una vez se inicia sesión, el sistema muestra un sidebar a la izquierda, que se mantiene fijo y contiene:
    - Información personal: La foto del del usuario que tiene sesión iniciada y su nombre
    - Transacciones: un link para acceder a la página de transacciones. Visible tanto para el rol ADMIN como para el rol USER.
    - Maestros: un link para acceder a la página de Maestros. Visible tanto para el rol ADMIN como para el rol USER.
    - Usuarios: un link para acceder a la página de usuarios. Sólo visible para usuarios con el rol ADMIN.

La página para gestión de Transacciones debe contener:

- Un dropdown para seleccionar el Maestro que se quiere visualizar
- Una tabla para visualizar los movimientos de ese Maestro. La tabla muestra el id del movimiento, la fecha, la cantidad de unidades de la transacción, y la persona que ejecutó el movimiento
- Un botón de “Agregar movimiento”, que muestra un diálogo en el que se puede agregar un movimiento de inventario. Dicho formulario tiene:
    - Un título que muestra el Maestro seleccionado en la tabla
    - El tipo de movimiento (salida o entrada)
    - La cantidad de elementos que salen o entran
    - Un botón para cancelar la entrada. Este botón cierra el diálogo.
    - Un botón para crear el movimiento. Este botón debe crear un movimiento en la base de datos, registrando como responsable del movimiento al usuario que tiene sesión iniciada. Este botón debe mostrar un loading y un mensaje de éxito o error. Si la creación del movimiento es exitosa, el sistema debe cerrar el diálogo y actualizar automáticamente la tabla de movimientos.
- Una gráfica con la evolución de los saldos diarios totales para el Maestro seleccionado.

Reglas adicionales:

- Tanto el rol de USER como el rol de ADMIN pueden acceder a la página de transacciones.

- Tanto el rol de USER como el de ADMIN pueden crear un movimiento de inventario.

La página para gestión de Maestros debe contener:

- Una tabla para visualizar los Maestros del sistema. La tabla muestra el id del material, el nombre del Maestro, el saldo del Maestro y quién creó el Maestro.
- Un botón de “Agregar”, que muestra un diálogo en el que se puede agregar un Maestro. Dicho formulario tiene:
    - Un input para escribir el nombre del Maestro
    - El saldo inicial del Maestro
    - Un botón para cancelar la entrada. Este botón cierra el diálogo.
    - Un botón para crear el movimiento. Este botón debe crear un registro en los Maestros, registrando como responsable del movimiento al usuario que tiene sesión iniciada. Además. Este botón debe mostrar un loading y un mensaje de éxito o error. Si la creación del movimiento es exitosa, el sistema debe cerrar el diálogo y actualizar automáticamente la tabla de maestros.

Reglas adicionales:

- Tanto los usuarios de rol USER como los de rol ADMIN pueden acceder a la página de Maestro
- Sólo los usuarios de rol ADMIN pueden ver el botón de crear Maestro

La página para gestión de usuarios debe contener:

- Una tabla para visualizar los usuarios del sistema. La tabla muestra el id del usuario, la fecha de creación del usuario, el correo del usuario y el rol que tiene asignado.
- Un botón de “Editar usuario”, que muestra un diálogo en el que se puede actualizar el rol del usuario. Dicho formulario tiene:
    - Un texto que muestra el correo del usuario a editar
    - Un dropdown con los roles disponibles en el sistema. Debe aparecer por defecto el rol que tiene el usuario.
    - Un botón para cancelar la entrada. Este botón cierra el diálogo.
    - Un botón para crear el movimiento. Este botón debe actualizar el rol del usuario seleccionado. Este botón debe mostrar un loading y un mensaje de éxito o error. Si la edición del usuario es satisfactoria, el sistema debe cerrar el diálogo y actualizar automáticamente la tabla de usuarios.

Reglas adicionales:

- Solo usuarios con el rol ADMIN pueden acceder a la página de usuarios.

Diseño e Interfaz de Usuario (10%):

- Aplicación de principios de diseño para una interfaz atractiva y funcional
- Se le rebajará al estudiante un 0.05 en la nota por cada error de ortografía. Máximo hasta 1 unidad.

Innovación y Creatividad (20%):

- Elementos únicos en el diseño o funcionalidad.
- Implementación de características adicionales (sin salirse de los requerimientos).

Documentación (10%):

- Documentación del proyecto, incluyendo un README que explique el propósito del proyecto, cómo ejecutarlo y cualquier otra información relevante.
- Documentación interna del código a través de comentarios para explicar la lógica y las decisiones de diseño.

Funcionalidad (60%)

- Funcionalidades según las instrucciones.
- Despliegue funcional.

Entrega:

- Deben crear un repositorio dentro de la organización de la clase de la siguiente manera: **nombreEquipo-Funcionalidad**
- En el README agregar los integrantes del equipo.
- En el README agregar un usuario y un administrador, usuario y contraseña
- Deben crear un enlace de vercel: **nombreEquipo-Funcionalidad.vercel.app**
- Deben agregarme al repositorio en los equipos o como colaborador.
- Se calificará el ultimo commit que se realice en la rama main antes de la hora de finalización del tiempo estipulado para el proyecto
- Presentación grupal del desarrollo durante espacio de clase.

FECHA DE ENTREGA: Jueves 4 DE Diciembre 11:59 PM (04/12/2025 – 23:59).

Gracias éxitos