API Peliculas

Requisitos - instalaciones

- Node < 18+

Creacion con vite

Documentacion de vite

Comando de creacion del proyecto

```
npm create vite@latest
```

- ingresamos el nombre del proyecto

```
Project name:
api-peliculas
```

- seleccionamos react

```
Select a framework:
  Vanilla
  Vue
  React
  Preact
  Lit
  Svelte
  Solid
  Qwik
  Angular
  Others
```

- selecionamos la plantilla (TypeScript)

```
Select a variant:
  TypeScript
  TypeScript + SWC
  JavaScript
  JavaScript + SWC
  React Router v7
  TanStack Router
```

- al final de comando nos indica los pasos a seguir

```
Scaffolding project in /home/acampuzano/test...
```

```
  Done. Now run:

  cd api-peliculas #entrar la carpeta del proyecto
  npm install # instala las dependecias
  npm run dev # corre el proyecto de vite
```

Configuraciones despues de crear el proyecto

- abrimos nuestro proyecto con vscode
- eliminamos contenido por defecto

Archivos Eliminados

- app.css

Modificaciones en el archivo app.tsx

```
function App() {
  return (
    <>
    </>
  )
}
export default App
```

Modificaciones en el archivo main.tsx

```
import { createRoot } from 'react-dom/client'
import './index.css'
import App from './App.tsx'

createRoot(document.getElementById('root')!).render(
    <App />
)
```

Modificaciones en el archivo index.css

```
html,body,#root{
  margin: 0;
  padding: 0;
  min-height: 100vh
}
```

Modificaciones en el archivo index.html

```
<link rel="icon" type="image/svg+xml" href="/icon.png" />
<title>Api Peliculas</title>
```

Instalaciones de nuevas dependecias

Instalar MUI

```
npm install @mui/material @emotion/react @emotion/styled
```

Instalar Fuente

```
npm install @fontsource/roboto
```

Importaciones agrega en el archivo main.tsx

```
import '@fontsource/roboto/300.css';
import '@fontsource/roboto/400.css';
import '@fontsource/roboto/500.css';
import '@fontsource/roboto/700.css';
```

Importaciones agregadas en el archivo app.tsx

```
import { Typography, Stack, TextField, SxProps, Theme } from "@mui/material";
```

Definimos una variable para el manejo de los estilos de los input

```
const stylesInput:SxProps<Theme> ={
  "& .MuiOutlinedInput-root": {
    "& fieldset": {
      borderColor: "#fff",
    },
    "&:hover fieldset": {
      borderColor: "#fff",
    },
    "&.Mui-focused fieldset": {
      borderColor: "#fff",
    },
  },
  "& .MuiInputLabel-root": {
    color: "#fff",
  },
  "& .MuiInputBase-input":{
    color: "#fff",
  }
}
```

Agregamos el siguiente contenido al componente de App

```
const App= () => {
  return (
    <Stack
```

```
      justifyContent="center"
      alignItems="center"
      minHeight="100vh"
      bgcolor="#242e34"
    >
      <Stack bgcolor="#0d253f" gap={2} p={5} borderRadius="16px">
        <Typography
          component="h3"
          variant="h1"
          fontWeight={700}
          color="#fff"
          textAlign="center"
        >
          Login
        </Typography>
        <TextField label="Email" variant="outlined" sx={stylesInput} type="email"/>
        <TextField label="Passowrd" variant="outlined" sx={stylesInput} type="password"/>
      </Stack>
    </Stack>
  );
}
```

Explicación de los componentes utilizados

1. **TextField**
   El componente `TextField` es un campo de entrada de texto proporcionado por Material-UI. Se utiliza para capturar datos del usuario, como texto, correos electrónicos o contraseñas. En este caso, se usaron dos `TextField`:

   - Uno para capturar el correo electrónico del usuario (`type="email"`).
   - Otro para capturar la contraseña del usuario (`type="password"`).

   Además, se personalizó su estilo utilizando la propiedad `sx` con el objeto `stylesInput`.

2. **Stack**
   El componente `Stack` es un contenedor flexible de Material-UI que permite organizar elementos en una dirección específica (horizontal o vertical) con un espaciado uniforme. En este ejemplo:

   - El `Stack` externo centra el contenido en la pantalla con las propiedades `justifyContent="center"` y `alignItems="center"`.
   - El `Stack` interno organiza los elementos (título y campos de entrada) con un fondo personalizado (`bgcolor="#0d253f"`), un espaciado entre elementos (`gap={2}`) y bordes redondeados (`borderRadius="16px"`).

3. **Typography**

El componente `Typography` se utiliza para mostrar texto con estilos predefinidos o personalizados. En este caso, se usó para mostrar el título "Login" con las siguientes propiedades:

- `component="h3"`: Define el elemento HTML como un `<h3>`.
- `variant="h1"`: Aplica los estilos de encabezado de nivel 1.
- `fontWeight={700}`: Define un peso de fuente grueso.
- `color="#fff"`: Cambia el color del texto a blanco.
- `textAlign="center"`: Centra el texto horizontalmente.

Estos componentes son parte de la biblioteca Material-UI, que facilita la creación de interfaces modernas y responsivas en React.