TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE TLÁHUAC

“LA ESENCIA DE LA GRANDEZA RADICA EN LAS RAICES”

INGENIERÍA EN SISTEMAS COMPUTACIONALES.

MATERIA:

DESARROLLO FRONTEND

TITULO:

EXAMEN UNIDAD 4

GRUPO:

8S2

P R E S E N T A:

MENDOZA ORTEGA MARÍA FERNANDA.

PROFESOR:

AQUINO SEGURA ROLDAN.

Tláhuac, Cd. Mx., 28 de Noviembre 2024.

Explicación del código App.jsx

Este archivo define el componente principal de la aplicación, que incluye la lógica de captura de datos del formulario, el cálculo del RFC, y la presentación del resultado.

1. Importaciones:
   Se importan React, el hook useState para manejar estados, Bootstrap para estilos básicos, y un archivo CSS. También se importa el componente RFC, encargado de mostrar el RFC generado.

2. Estados:
   Se utilizan estados (useState) para capturar y manejar los datos del formulario: apellido paterno, apellido materno, nombre, fecha de nacimiento y el RFC calculado.
   Los useState permiten que un componente guarde datos que pueden cambiar con el tiempo, como valores de formularios.

3. Función calcularRFC:
   Contiene la lógica para generar el RFC.
   - Toma la fecha de nacimiento y la divide en año, mes y día utilizando split("-").
   - Extrae las letras necesarias de los apellidos y nombres:
     - La primera letra del apellido paterno.
     - La primera vocal interna del apellido paterno, identificada mediante un recorrido con la función buscaVocal.
     - La primera letra del apellido materno.
     - La primera letra del nombre.
   - Finalmente, concatena estas letras con la fecha en el formato deseado (año, mes y día) para construir el RFC.

4. Manejo del formulario (handleSubmit):
   Cuando se envía el formulario:
   - La función evita la recarga de la página con e.preventDefault().
   - Llama a calcularRFC() para obtener el RFC.
   - Actualiza el estado del RFC con setRfc(resultadoRFC), haciendo que React renderice nuevamente la vista con el nuevo valor.

5. Visualización:
   El componente retorna una estructura de formulario estilizada con Bootstrap.
   - El formulario captura los datos necesarios con campos de entrada (input) y los actualiza con las funciones de estado (onChange).
   - Al presionar el botón "Calcular RFC", se genera el RFC y, si está calculado, se pasa como prop al componente RFC para su visualización.

rfc_calculadora > src > App.jsx > ...

```jsx
import React, { useState } from "react";
import "bootstrap/dist/css/bootstrap.min.css";
import "./index.css";
import { RFC } from "./RFC";

export const App = () => {
  // Estados para los datos del formulario y el RFC
  const [paterno, setPaterno] = useState("");
  const [materno, setMaterno] = useState("");
  const [nombre, setNombre] = useState("");
  const [fechaNacimiento, setFechaNacimiento] = useState("");
  const [rfc, setRfc] = useState("");

  // Función para calcular el RFC
  const calcularRFC = () => {
    const [anio, mes, dia] = fechaNacimiento.split("-");
    const buscaVocal = (cadena) => {
      for (let i = 1; i < cadena.length; i++) {
        if ("AEIOU".includes(cadena[i].toUpperCase())) return cadena[i].toUpperCase();
      }
      return "";
    };
    const letra1 = paterno[0]?.toUpperCase() || "";
    const letra2 = buscaVocal(paterno);
    const letra3 = materno[0]?.toUpperCase() || "";
    const letra4 = nombre[0]?.toUpperCase() || "";

    return `${letra1}${letra2}${letra3}${letra4}${anio.slice(2)}${mes}${dia}`;
  };

  // Manejo del envío del formulario
  const handleSubmit = (e) => {
    e.preventDefault();
    const resultadoRFC = calcularRFC(); // Calcular el RFC
    setRfc(resultadoRFC); // Actualizar el estado del RFC
  };

  return (
    <div className="container d-flex justify-content-center align-items-center custom-container">
```

rfc_calculadora > src > App.jsx > ...

```
6   export const App = () => {
38    return (
39      <div className="container d-flex justify-content-center align-items-center custom-container">
40        <div className="row justify-content-center align-items-center w-100">
41          <div className="col-md-6 col-lg-4">
42            <div className="card custom-card">
43              <div className="card-body p-4">
44                <h2 className="text-center text-primary mb-4">Calculadora de RFC</h2>
45                <form onSubmit={handleSubmit}>
46                  <div className="mb-3">
47                    <label htmlFor="paterno" className="form-label">Apellido Paterno</label>
48                    <input
49                      type="text"
50                      className="form-control"
51                      id="paterno"
52                      value={paterno}
53                      onChange={(e) => setPaterno(e.target.value)}
54                      required
55                    />
56                  </div>
57                  <div className="mb-3">
58                    <label htmlFor="materno" className="form-label">Apellido Materno</label>
59                    <input
60                      type="text"
61                      className="form-control"
62                      id="materno"
63                      value={materno}
64                      onChange={(e) => setMaterno(e.target.value)}
65                      required
66                    />
67                  </div>
68                  <div className="mb-3">
69                    <label htmlFor="nombre" className="form-label">Nombre</label>
70                    <input
71                      type="text"
72                      className="form-control"
73                      id="nombre"
74                      value={nombre}
75                      onChange={(e) => setNombre(e.target.value)}
```

Explicación del código main.jsx

Este archivo es el punto inicial de la aplicación React, donde se establece la conexión entre React y el DOM del navegador.

Creación de ReactDOM:

Se utiliza ReactDOM.createRoot para vincular la aplicación al elemento HTML con el identificador root (generalmente definido en el archivo index.html).

Renderización del componente App:

El componente principal App se incluye dentro de React.StrictMode, una herramienta que ayuda a detectar errores comunes, como el uso de funciones desactualizadas.

Este archivo garantiza que la aplicación React se cargue y funcione correctamente en el navegador.

rfc_calculadora > src > main.jsx

```jsx
import React from 'react';
import ReactDOM from 'react-dom/client';
import { App } from './App';

ReactDOM.createRoot(document.getElementById('root')).render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
```

Explicación del código RFC.jsx

El componente RFC se encarga de mostrar el RFC que se pasa como prop desde el componente App.

1. Uso de props:

Se utiliza la desestructuración ({ rfc }) para recibir el RFC como un argumento. Esto permite que el componente sea flexible y reutilizable, ya que solo necesita recibir un valor para funcionar.

2. Estructura visual:

Contiene un encabezado (h4) y un párrafo (p) que presentan el RFC calculado. Estos elementos tienen clases de Bootstrap estilizar el contenido.

rfc_calculadora > src > RFC.jsx > ...

```jsx
import React from "react";

export const RFC = ({ rfc }) => {
  return (
    <div className="mt-4">
      <h4 className="text-center">RFC:</h4>
      <p className="text-center">{rfc}</p>
    </div>
  );
};
```