Guía para Nuevos Desarrolladores – Proyecto "Limpieza de Aulas"

1. Introducción:

El sistema "Limpieza de Aulas" es una aplicación web desarrollada para gestionar el estado de limpieza de las aulas en la EPET N°20. Permite registrar usuarios, iniciar sesión y actualizar el estado (limpio/sucio) de cada aula.

2. Requisitos previos:

Lista de lo que se necesita para poder ejecutar el proyecto:

-Node.js y npm instalados

-Visual Studio Code o cualquier editor de código

-Git (para control de versiones)

-Navegador (Google Chrome, Edge, etc.)

3. Instalación

Pasos para clonar y ejecutar el proyecto:

-Clonar el repositorio

git clone <https://github.com/Maiaagonzalez/proyectogrupal>

-Se entrar al proyecto

cd Maiaagonzalez/proyctogrupal

git pull se usa para bajar lo que otros subieron.

git status muestra el estado actual del directorio de trabajo y del área de preparación en Git

git add. se utiliza para agregar archivos nuevos o modificados al área de preparación

git commit -m"la pagina" utiliza el comando git commit para guardar la instantánea

git push para subir

-Se Instalan las dependencias

npm install

-Iniciar el servidor

npm start

4. Estructura del proyecto

Explicar el mapa de carpetas:

/proyectogrupal

│

├── /.firebase/ #configuración y datos locales de Firebase

├── /.gitignore #Archivos a ignorar por Git

├── /.firebaserc #Configuración del proyecto Firebase

├── /firebase.json #Archivo de configuración principal de Firebase

│

├── /node\_modules/ #Librerías instaladas con npm (React, Firebase, etc.)

|

├── /public/ #Archivos públicos (visibles directamente en el navegador)

│ ├── index.html #Archivo HTML principal

│ ├── favicon.ico #Ícono de la app

│ ├── manifest.json #Configuración PWA

│ └── imágenes y logos

│

├── /src/ #Código fuente principal de la aplicación

│ ├── /components/ #Componentes React (formularios, listas, etc.)

│ ├── /dataconnect-generated/ #Archivos autogenerados por Firebase Data Connect

│ ├── /pages/ #Páginas de navegación (Inicio, Login, Registro, etc.)

│ ├── /docs/ #documentación de la página

│ ├── /providers/ #rutas y autenticación

│ ├── firebaseConfig.js #Inicialización y configuración de Firebase

│ ├── index.css #Imágenes y estilos del proyecto

│ ├── App.js #Componente principal de React

│ └── index.js #Punto de entrada de la app

5. Tecnologías usadas:

Frontend: React.js, HTML, CSS

Backend: Node.js,

Base de datos: firebaserc

Documentacion: readme.md

Control de versiones: Git/GitHub

6. Guía de desarrollo-ejemplo de una pagina nueva:

import React, { useState, useEffect } from 'react';

import { BrowserRouter as Router, Routes, Route, Link } from 'react-router-dom';

import { Button } from '@/components/ui/button';

import { Card, CardContent, CardHeader, CardTitle } from '@/components/ui/card';

// Página principal del sistema Limpieza de Aulas

function Home() {

return (

<div className="p-6 text-center space-y-4">

<h1 className="text-3xl font-bold">Sistema de Limpieza de Aulas</h1>

<p>Bienvenido al sistema. Desde aquí podés acceder a la guía de desarrollo o gestionar aulas.</p>

<div className="flex justify-center gap-4">

<Link to="/guia"><Button>Guía de desarrollo</Button></Link>

<Link to="/aulas"><Button variant="outline">Gestión de Aulas</Button></Link>

</div>

</div>

);

}

// Componente de gestión de aulas

function GestionAulas() {

const [aulas, setAulas] = useState([

{ id: 1, nombre: 'Aula 1', estado: 'Limpio' },

{ id: 2, nombre: 'Aula 2', estado: 'Sucio' }

]);

const [nuevaAula, setNuevaAula] = useState('');

const [estado, setEstado] = useState('Limpio');

const agregarAula = () => {

if (!nuevaAula.trim()) return;

const id = aulas.length + 1;

setAulas([...aulas, { id, nombre: nuevaAula, estado }]);

setNuevaAula('');

};

const cambiarEstado = (id) => {

setAulas(

aulas.map((aula) =>

aula.id === id ? { ...aula, estado: aula.estado === 'Limpio' ? 'Sucio' : 'Limpio' } : aula

)

);

};

useEffect(() => {

console.log('Aulas actualizadas:', aulas);

}, [aulas]);

return (

<div className="p-6 space-y-4">

<h2 className="text-2xl font-bold">Gestión de Aulas</h2>

<div className="flex items-center gap-2">

<input

type="text"

placeholder="Nombre del aula"

value={nuevaAula}

onChange={(e) => setNuevaAula(e.target.value)}

className="border p-2 rounded w-1/2"

/>

<select

value={estado}

onChange={(e) => setEstado(e.target.value)}

className="border p-2 rounded"

>

<option value="Limpio">Limpio</option>

<option value="Sucio">Sucio</option>

</select>

<Button onClick={agregarAula}>Agregar Aula</Button>

</div>

<ul className="mt-4 space-y-2">

{aulas.map((aula) => (

<li key={aula.id} className="flex justify-between bg-gray-100 p-2 rounded">

<span>{aula.nombre} - Estado: {aula.estado}</span>

<Button variant="outline" onClick={() => cambiarEstado(aula.id)}>

Cambiar estado

</Button>

</li>

))}

</ul>

<Link to="/"><Button variant="link">Volver al inicio</Button></Link>

</div>

);

}

// Página de guía de desarrollo

function GuiaDesarrollo() {

return (

<div className="p-6 space-y-6">

<Card className="shadow-xl">

<CardHeader>

<CardTitle>6. Guía de desarrollo</CardTitle>

</CardHeader>

<CardContent className="space-y-4">

<section>

<h2 className="font-semibold text-lg">Cómo crear una nueva página o componente</h2>

<p>Crear un archivo nuevo dentro de <code>/src/components</code> o <code>/src/pages</code>. Luego importarlo y usarlo dentro de App.jsx.</p>

</section>

<section>

<h2 className="font-semibold text-lg">Cómo agregar una nueva aula o modificar el estado</h2>

<p>Podés agregar un componente que use <code>useState</code> para manejar las aulas y un botón que cambie su estado de Limpio a Sucio.</p>

</section>

<section>

<h2 className="font-semibold text-lg">Buenas prácticas</h2>

<ul className="list-disc ml-6">

<li>Usar nombres descriptivos para variables y componentes.</li>

<li>Organizar carpetas según funcionalidad.</li>

<li>Reutilizar componentes cuando sea posible.</li>

<li>Usar clases de Tailwind para estilos consistentes.</li>

</ul>

</section>

<section>

<h2 className="font-semibold text-lg">Uso de useState, useEffect y props (React)</h2>

<p>

<code>useState</code> permite manejar estados locales, como el listado de aulas o el texto de entrada.<br />

<code>useEffect</code> se utiliza para ejecutar código cuando cambia un estado o al montar el componente.<br />

Las <code>props</code> sirven para pasar datos entre componentes.

</p>

</section>

<Link to="/"><Button variant="link">Volver al inicio</Button></Link>

</CardContent>

</Card>

</div>

);

}

// Aplicación principal con rutas

export default function App() {

return (

<Router>

<Routes>

<Route path="/" element={<Home />} />

<Route path="/aulas" element={<GestionAulas />} />

<Route path="/guia" element={<GuiaDesarrollo />} />

</Routes>

</Router>

);

}

7. Créditos / Equipo de desarrollo

Proyecto desarrollado por los alumnos de la EPET N°20

Curso: 6°1

Integrantes: Calfin Abril, Garrido Molena, Molina Alan, Cesari Lautaro, Gonzalez Maia.