**ACTIVIDAD A REALIZAR CON REACT Y VITE**

**OBJETIVO:** Vincular React con Firebase

**ACTIVIDAD**

PASO 1 CREAR UNA CARPETA LLAMADA CRUD

2. ABRIR EL VISUAL CODE

3. INSTALAR REACT CON VITE

npm create vite@latest react CrudMaqueta

4. navega al Directorio del Proyecto

Cd react

5. **Instalar las dependencias**

**npm install**

**6. Iniciar el servidor de desarrollo**

npm run dev

7. Crear una cta de Firebase (Crud-maqueta)



Dar de alta dos servicios

1. firestores database(crear la base de datos)
2. autentificación
3. firestores database(crear la base de datos)

|  |  |
| --- | --- |
| Interfaz de usuario gráfica, Aplicación, Sitio web  Descripción generada automáticamente |  |
|  | |

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

1. autentificación

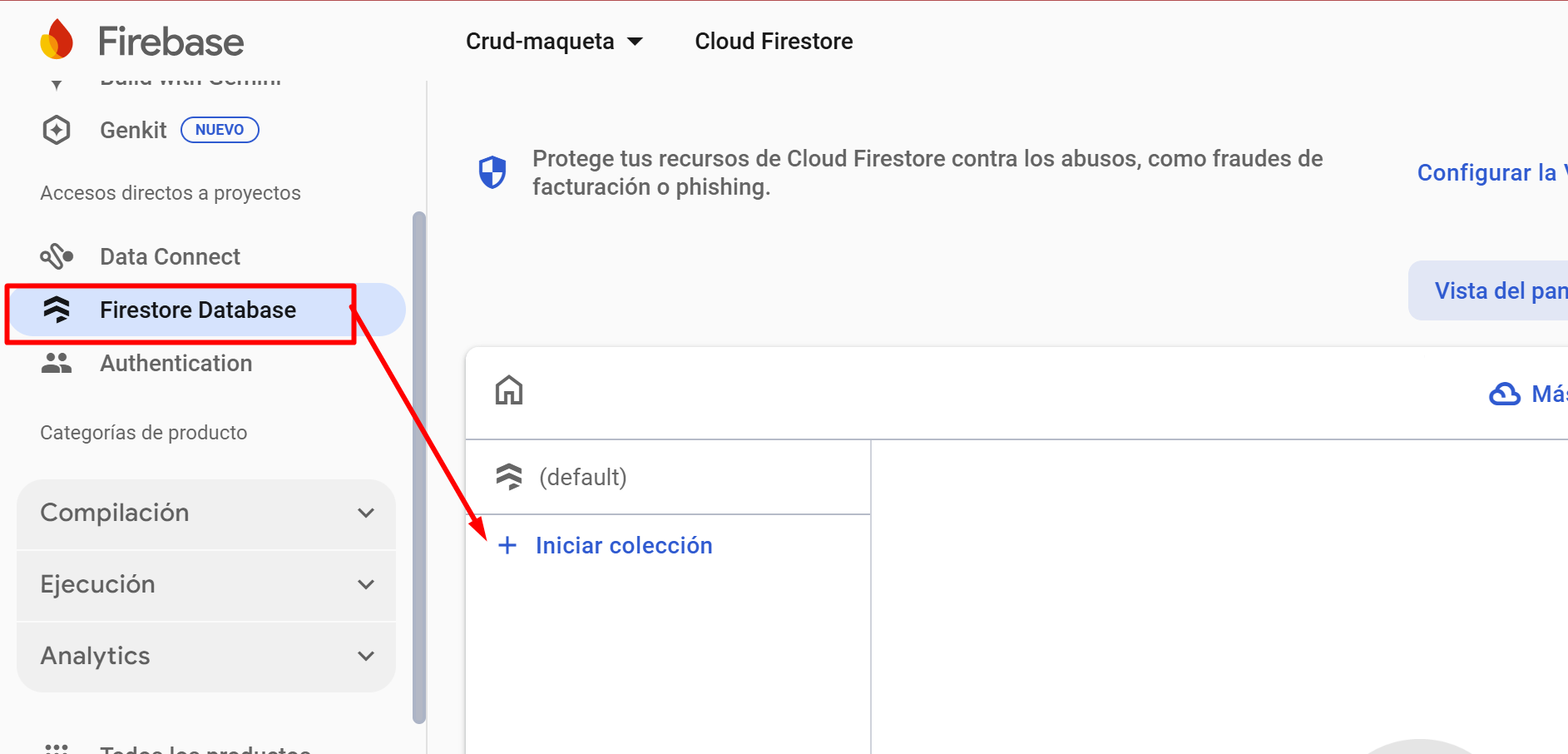
|  |  |
| --- | --- |
| Interfaz de usuario gráfica, Texto, Aplicación  Descripción generada automáticamente |  |

|  |  |
| --- | --- |
|  |  |

8. Instalar firebase en react

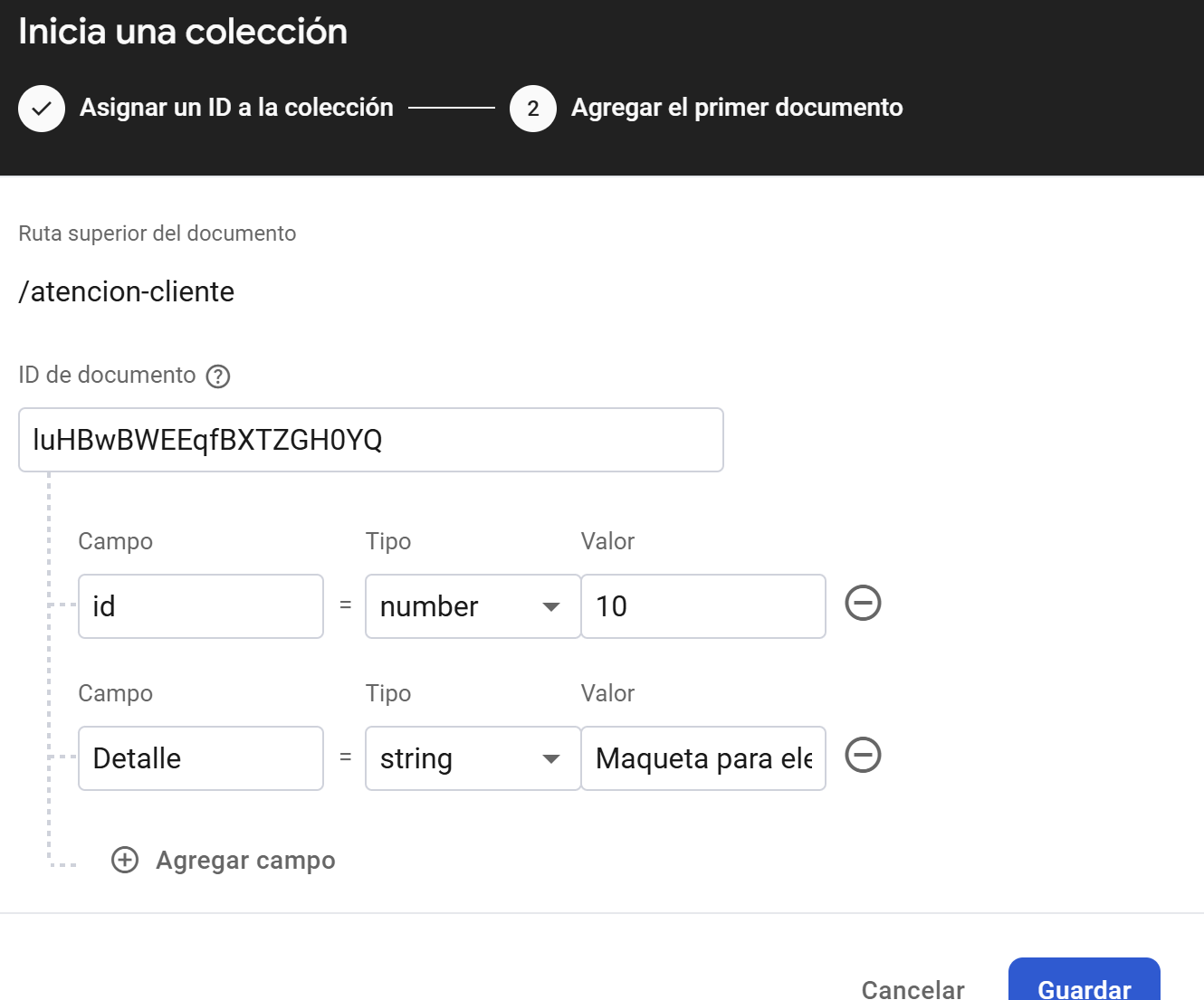
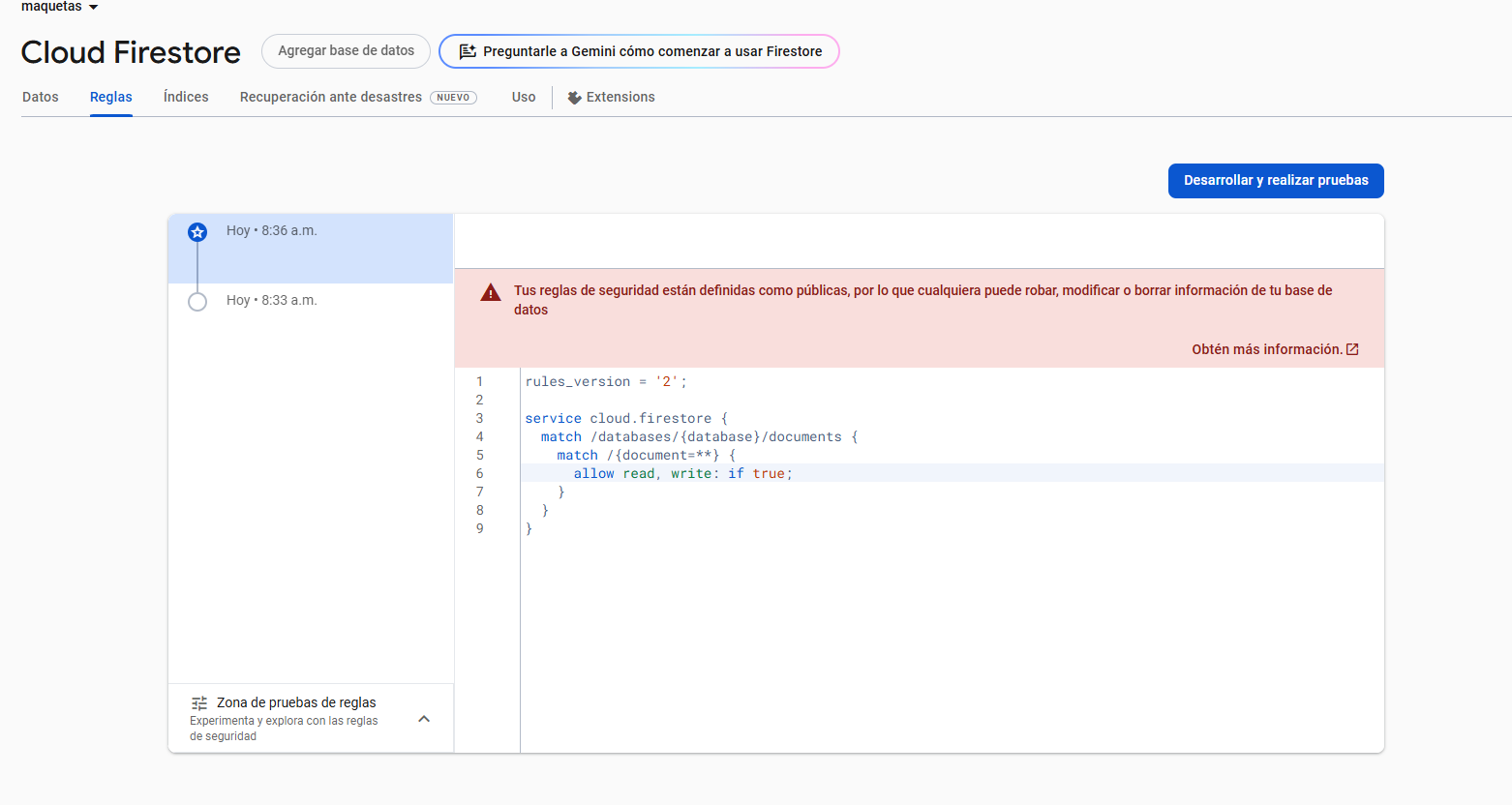
npm i firebase

9. Para practicar en firebase se hará una nueva colección, para entender que ahí se guardaran los datos

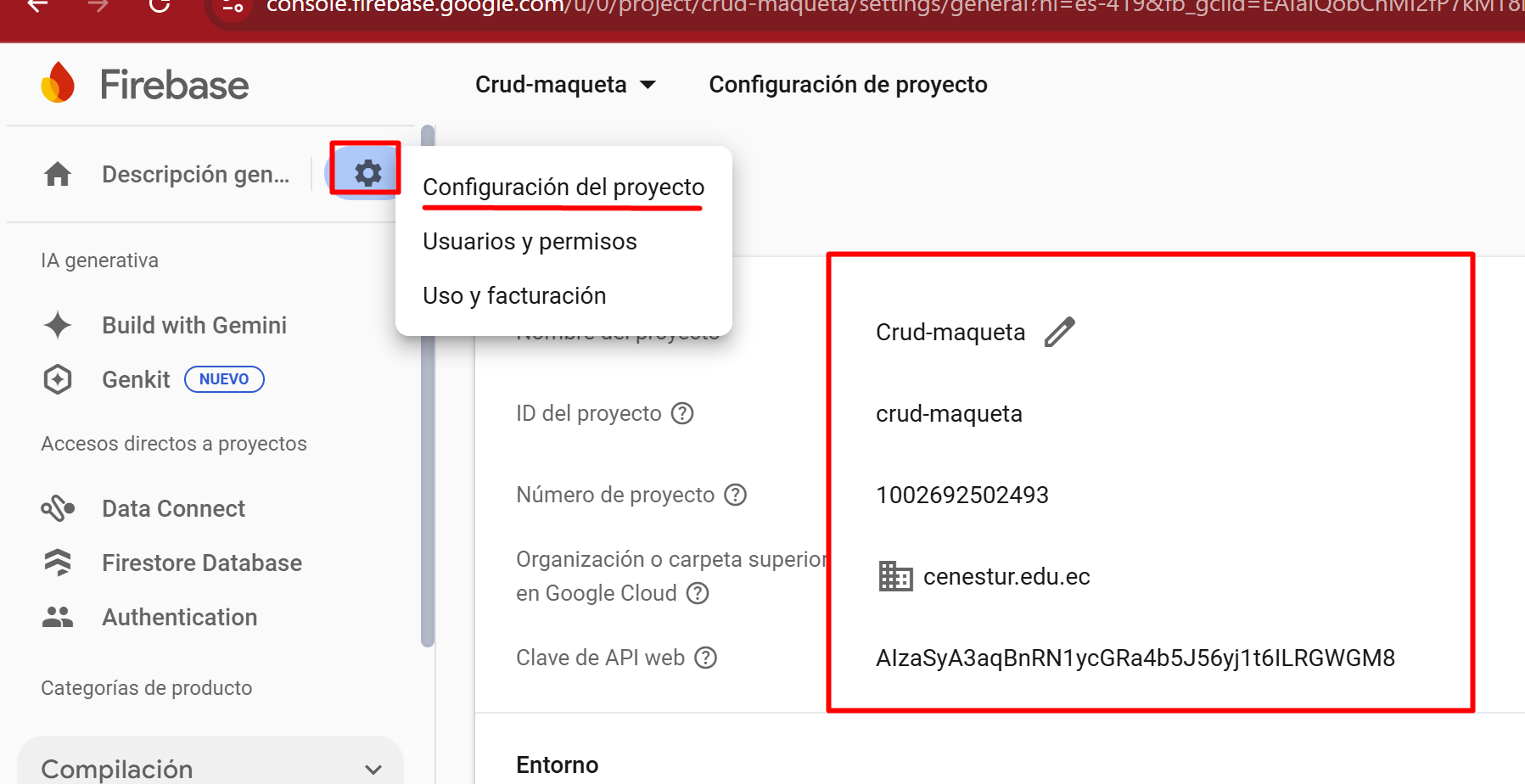
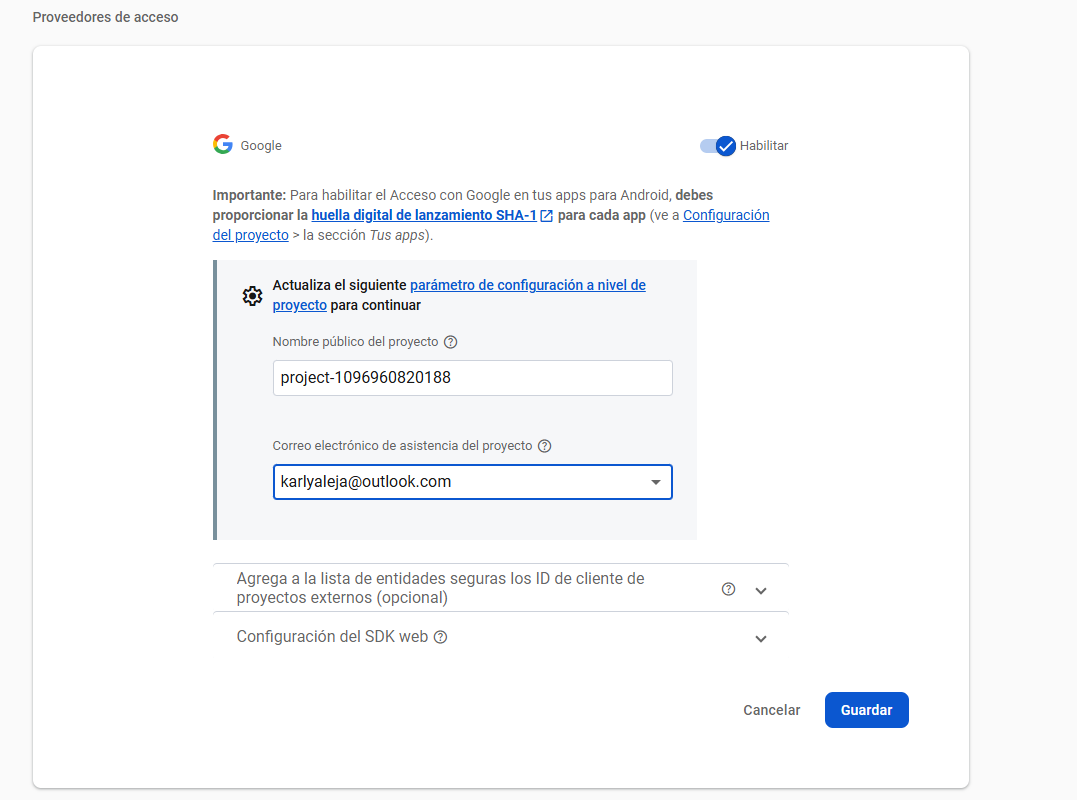


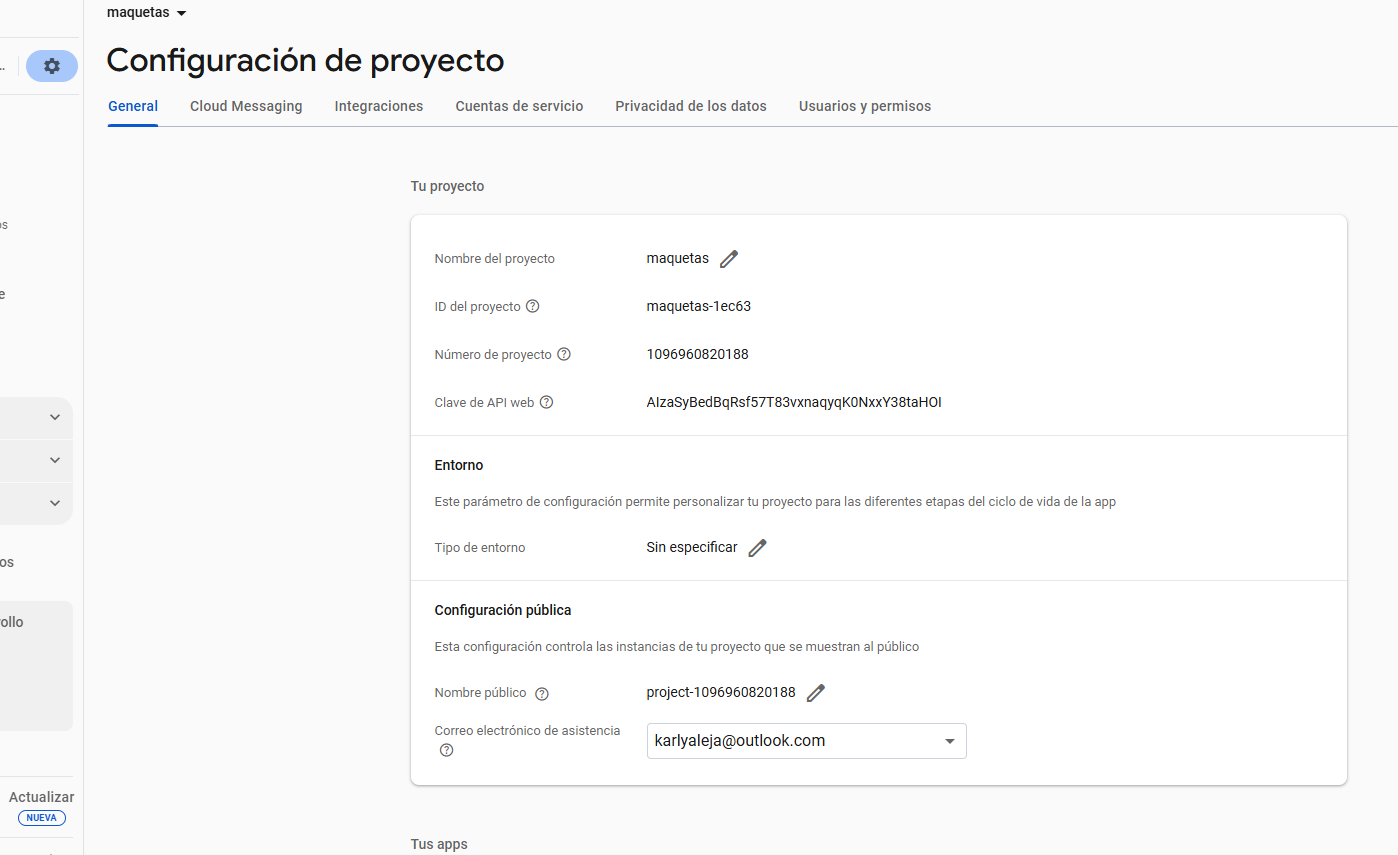
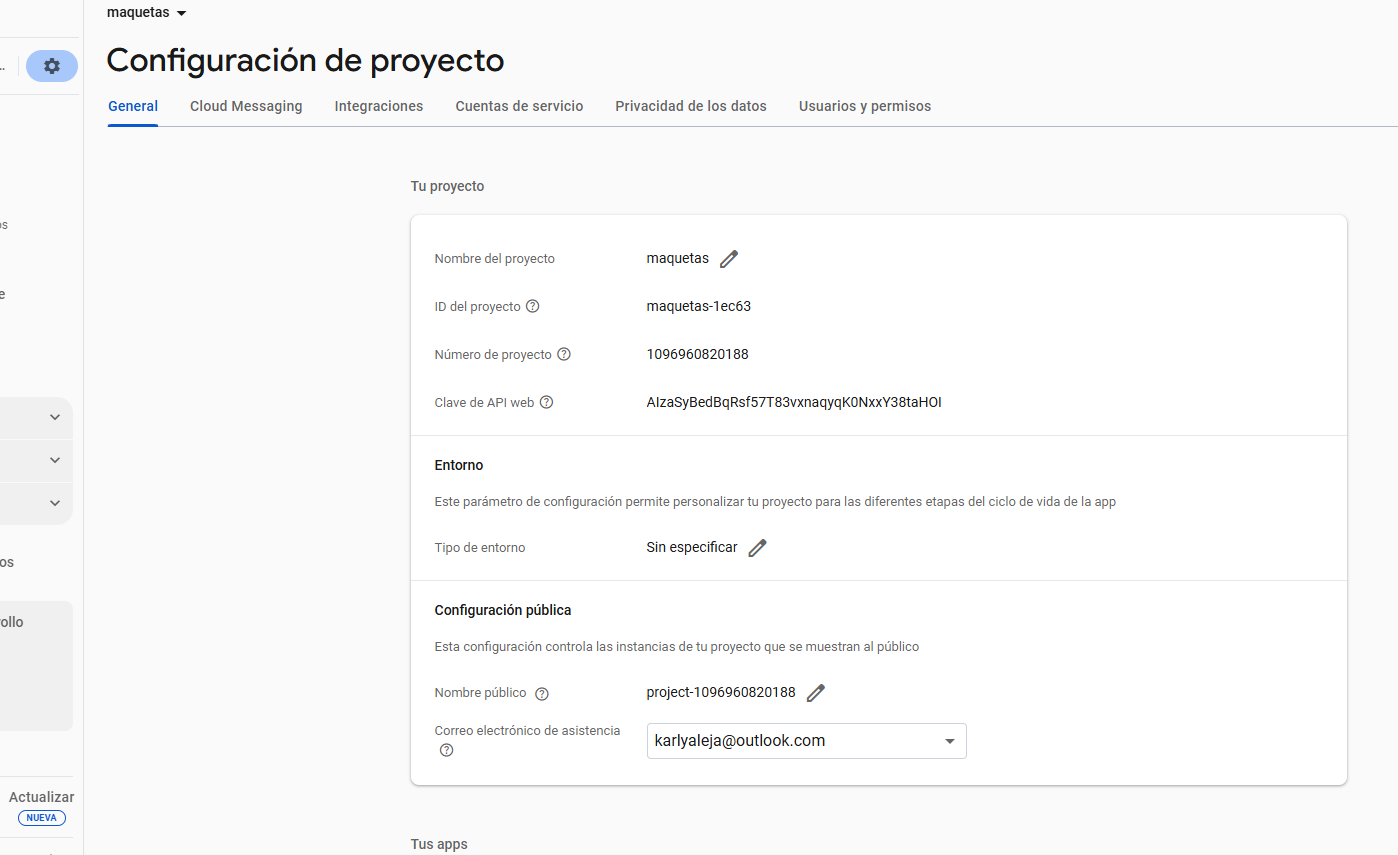
Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

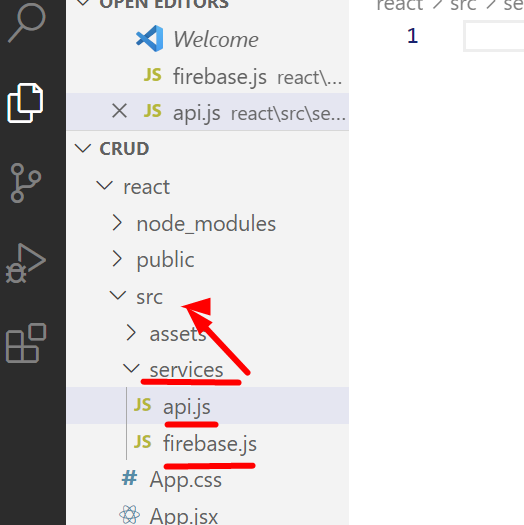
 

Obtener las credenciales

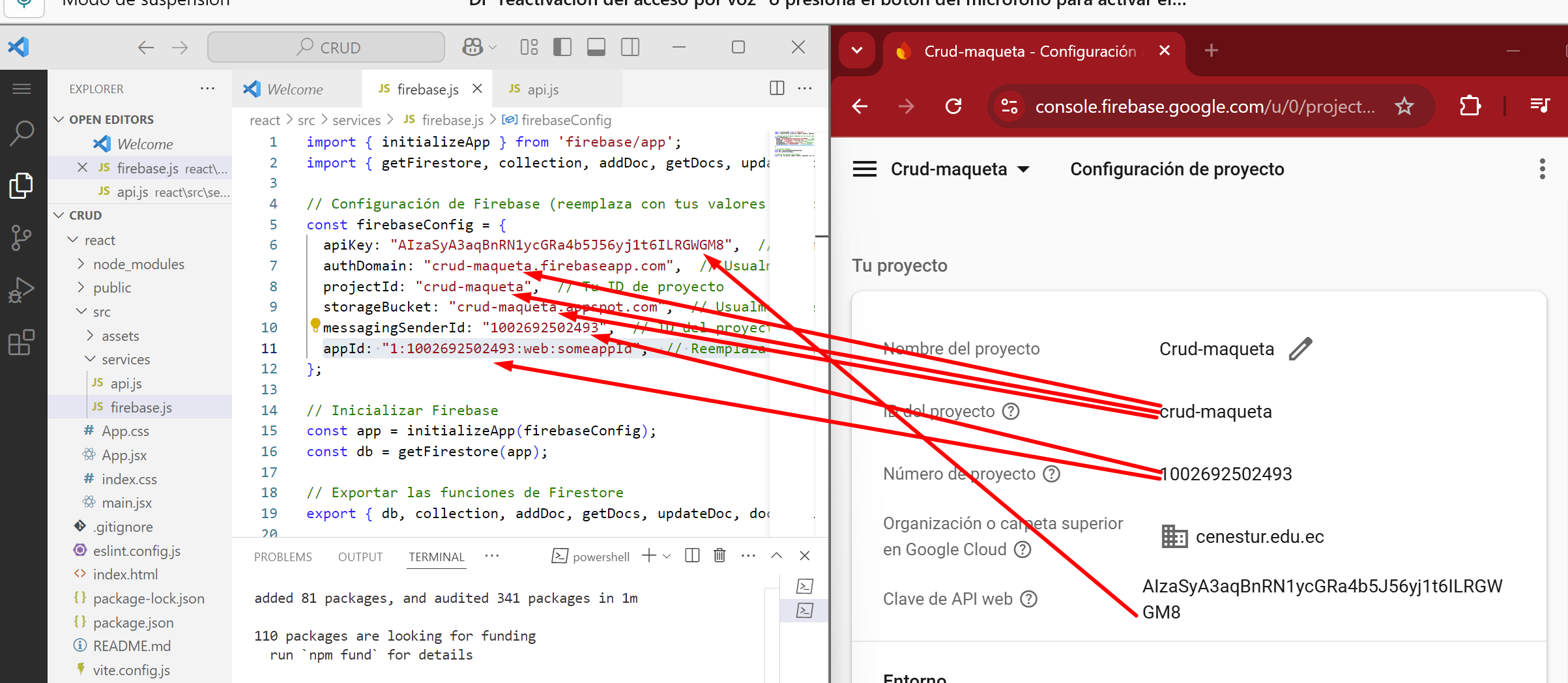
 

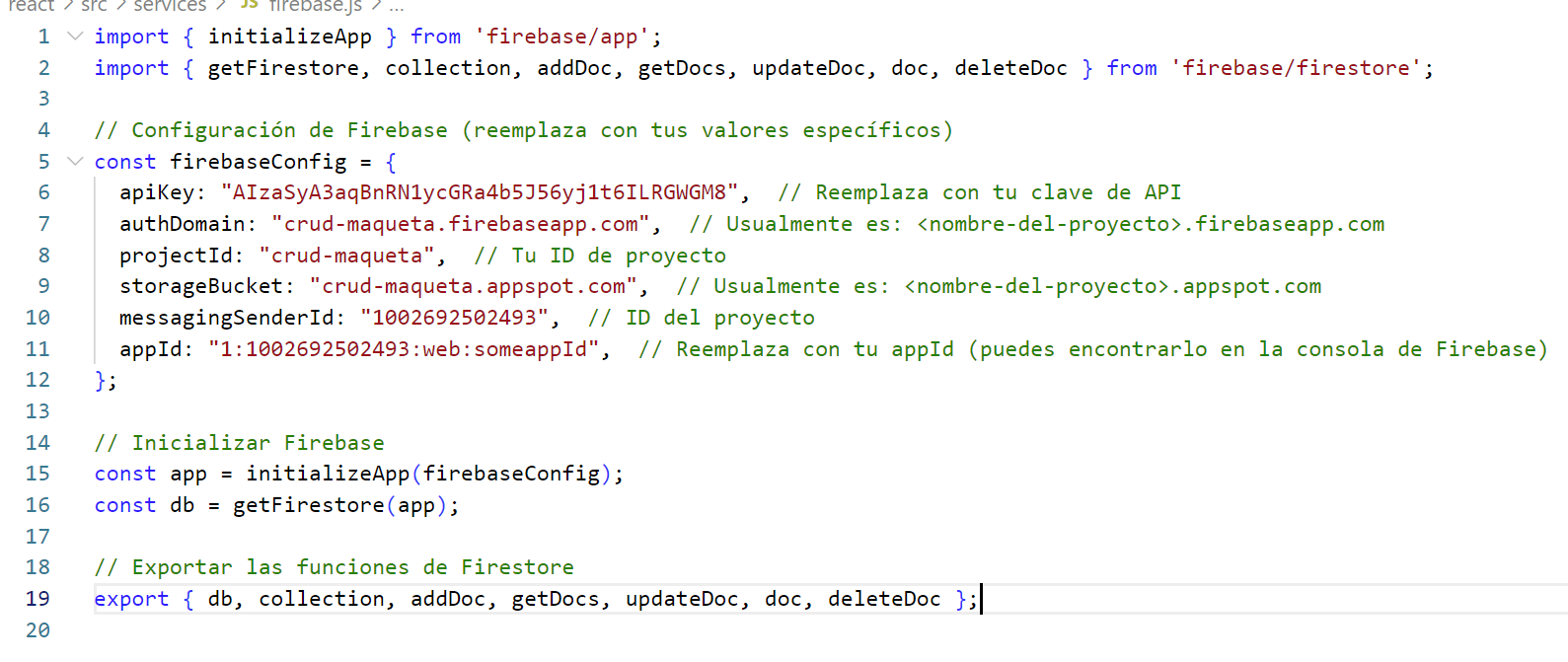


Visual code crear una carpeta con dos archivos



En firebase.js





Crear un formulario donde ingrese el código del proyecto, descripción del proyecto, precio y fecha de entrega.

**Información para prueba en api.jsx**

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

import { db, collection, addDoc, getDocs } from './firebase';  // Importa correctamente las funciones

const TestFirestore = () => {

  const [data, setData] = useState([]);

  const [loading, setLoading] = useState(true);

  const [error, setError] = useState(null);

  // Estado para los campos del formulario

  const [detalle, setDetalle] = useState('');

  const [precio, setPrecio] = useState('');

  // Función para obtener datos de Firestore

  const fetchData = async () => {

    try {

      const querySnapshot = await getDocs(collection(db, "productos"));

      const productosData = [];

      querySnapshot.forEach((doc) => {

        productosData.push({ id: doc.id, ...doc.data() });

      });

      setData(productosData);

      setLoading(false);

    } catch (err) {

      setError('Error al obtener los datos: ' + err.message);

      setLoading(false);

    }

  };

  // Función para guardar los datos en Firestore

  const saveData = async () => {

    if (detalle === '' || precio === '') {

      alert('Por favor, completa todos los campos.');

      return;

    }

    try {

      await addDoc(collection(db, "productos"), {

        detalle: detalle,

        precio: parseFloat(precio)  // Asegúrate de convertir a número el precio

      });

      alert('Producto guardado con éxito');

      fetchData(); // Vuelve a obtener los datos después de guardar

      setDetalle(''); // Limpiar los campos del formulario

      setPrecio('');

    } catch (err) {

      console.error('Error al guardar el producto: ', err.message);

    }

  };

  // Cargar los datos al montar el componente

  useEffect(() => {

    fetchData();

  }, []);

  if (loading) {

    return <div>Cargando...</div>;

  }

  if (error) {

    return <div>{error}</div>;

  }

  return (

    <div>

      <h2>Datos de Firestore:</h2>

      <ul>

        {data.map((producto) => (

          <li key={producto.id}>

            {producto.detalle} - ${producto.precio}

          </li>

        ))}

      </ul>

      {/\* Formulario para agregar nuevo producto \*/}

      <div>

        <h3>Agregar nuevo producto:</h3>

        <form onSubmit={(e) => e.preventDefault()}>

          <input

            type="text"

            placeholder="Detalle del producto"

            value={detalle}

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

          />

          <input

            type="number"

            placeholder="Precio del producto"

            value={precio}

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

          />

          <button onClick={saveData}>Guardar Producto</button>

        </form>

      </div>

    </div>

  );

};

export default TestFirestore;

**firebase.js**

import { initializeApp } from 'firebase/app';

import { getFirestore, collection, addDoc, getDocs, updateDoc, doc, deleteDoc } from 'firebase/firestore';

// Configuración de Firebase (reemplaza con tus valores específicos)

const firebaseConfig = {

  apiKey: "AIzaSyA3aqBnRN1ycGRa4b5J56yj1t6ILRGWGM8",  // Reemplaza con tu clave de API

  authDomain: "crud-maqueta.firebaseapp.com",  // Usualmente es: <nombre-del-proyecto>.firebaseapp.com

  projectId: "crud-maqueta",  // Tu ID de proyecto

  storageBucket: "crud-maqueta.appspot.com",  // Usualmente es: <nombre-del-proyecto>.appspot.com

  messagingSenderId: "1002692502493",  // ID del proyecto

  appId: "1:1002692502493:web:someappId",  // Reemplaza con tu appId (puedes encontrarlo en la consola de Firebase)

};

// Inicializar Firebase

const app = initializeApp(firebaseConfig);

const db = getFirestore(app);

// Exportar las funciones de Firestore

export { db, collection, addDoc, getDocs, updateDoc, doc, deleteDoc };

**app.jsx**

import { useState } from 'react';

import reactLogo from './assets/react.svg';

import viteLogo from '/vite.svg';

import TestFirestore from './services/api';

import './App.css';

function App() {

  const [count, setCount] = useState(0);

  return (

    <div className="App">

      <h1>Prueba de conexión con Firebase Firestore</h1>

      <TestFirestore />

    </div>

  );

}

export default App;

**Actividad a realizar**

1. Guardar los datos en una colección llamada maquetas y agregar 6 campos

* idMaqueta
* DetalledeMaqueta
* Cantidad
* Precio
* Propietario
* FechaRetiro

1. Mejora el diseño de usuario
2. Descarga la información de firebase
3. APLICAR EL CRUD COMPLETO

