**Mi Comida Favorita**

**Aplicación React Native con Firebase Authentication y Cloud Firestore**

# Requerimientos Previos

Node.js (versión 14 o superior) npm o yarn

Expo CLI

IDE (Visual Studio Code recomendado)

Cuenta de Firebase

Emulador Android/iOS o dispositivo físico

# Librerías Necesarias

# Dependencias principales

expo-cli

@react-navigation/native

@react-navigation/native-stack

firebase

react-native-elements

expo-constants

# Configuración Inicial

## Crear proyecto Expo

# Crear nuevo proyecto

npx create-expo-app MiComidaFavorita

--

template

blank

cd

MiComidaFavorita

# Instalar dependencias

npm

install @react-navigation/native @react-navigation/native-stack

npm

install firebase

npm

install react-native-elements

npm

install expo-constants

npm

install react-native-safe-area-context

## Configurar Firebase

1. Ir a Firebase Console (<https://console.firebase.google.com/>)
2. Crear nuevo proyecto "MiComidaFavorita"
3. Habilitar Authentication (Email/Password)
4. Crear Cloud Firestore
5. Registrar la aplicación web
6. Copiar configuración de Firebase

## Estructura del Proyecto

MiComidaFavorita/

├──

src/

│

├──

config/

│

│

└──

firebase.js

│

├──

screens/

│

│

├──

LoginScreen.js

│

│

├──

RegisterScreen.js

│

│

└──

HomeScreen.js

│

├──

components/

│

│

└──

ProfileForm.js

│

└──

navigation/

│

└──

AppNavigator.js

├──

App.js

└──

package.json

# Configuración de Firebase

## src/config/firebase.js

import

{

initializeApp

}

from

'firebase/app'

;

import

{

getAuth

}

from

'firebase/auth'

;

import

{

getFirestore

}

from

'firebase/firestore'

;

const

firebaseConfig

=

{

// Pegar configuración de Firebase Console

apiKey

:

"tu-api-key"

,

authDomain

:

"tu-auth-domain"

,

projectId

:

"tu-project-id"

,

storageBucket

:

"tu-storage-bucket"

,

messagingSenderId

:

"tu-sender-id"

,

appId

:

"tu-app-id"

}

;

const

app

=

initializeApp

(

firebaseConfig

)

;

export

const

auth

=

getAuth

(

app

)

;

export

const

db

=

getFirestore

(

app

)

;

# Navegación

## src/navigation/AppNavigator.js

import { NavigationContainer } from '@react-navigation/native'; import { createNativeStackNavigator } from '@react-navigation/native-stack';

import

LoginScreen

from

'../screens/LoginScreen'

;

import

RegisterScreen

from

'../screens/RegisterScreen'

;

import

HomeScreen

from

'../screens/HomeScreen'

;

const

Stack

=

createNativeStackNavigator

()

;

export

default

function

AppNavigator

()

{

return

(

<

NavigationContainer

>

<

Stack

.

Navigator

initialRouteName

=

"Login"

>

<

Stack

.

Screen

name

=

"Login"

component

=

{

LoginScreen

}

options

=

{{

headerShown

:

false

}}

/>

<

Stack

.

Screen

name

=

"Register"

component

=

{

RegisterScreen

}

options

=

{{

headerShown

:

false

}}

/>

<

Stack

.

Screen

name

=

"Home"

component

=

{

HomeScreen

}

options

=

{{

headerShown

:

true

,

title

:

'Mi Perfil'

}}

/>

<

/Stack.Navigator>

<

/NavigationContainer>

);

}

# Pantallas

## src/screens/LoginScreen.js





## src/screens/RegisterScreen.js

import

React

, {

useState

}

from

'react'

;

import

{

View

,

StyleSheet

}

from

'react-native'

;

import

{

Input

,

Button

,

Text

}

from

'react-native-elements'

;

import

{

createUserWithEmailAndPassword

}

from

'firebase/auth'

;

|  |
| --- |
| import { auth } from '../config/firebase';  export default function RegisterScreen({ navigation }) { const [email, setEmail] = useState('');  const [password, setPassword] = useState('');  const [error, setError] = useState('');  const handleRegister = async () => {  try {  const userCredential = await createUserWithEmailAndPassword(auth, email, password);  navigation.replace('Home');  } catch (error) {  setError('Error al registrarse: ' + error.message);  }  };  return (  <View style={styles.container}> <Text h3 style={styles.title}>Registro</Text>  <Input  placeholder="Email"  value={email}  onChangeText={setEmail}  autoCapitalize="none"  />  <Input  placeholder="Contraseña"  value={password} onChangeText={setPassword}  secureTextEntry  />  {error ? <Text style={styles.error}>{error}</Text> : null}  <Button  title="Registrarse"  onPress={handleRegister}  containerStyle={styles.button}  />  <Button  title="Volver al Login"  type="outline" onPress={() => navigation.navigate('Login')}  containerStyle={styles.button}  />  </View>  ); }  const styles = StyleSheet.create({  container: {  flex: 1,  padding: 20,  justifyContent: 'center', |

},

title

:

{

textAlign

:

'center'

,

marginBottom

:

30

,

},

button

:

{

marginVertical

:

10

,

},

error

:

{

color

:

'red'

,

textAlign

:

'center'

,

marginBottom

:

10

,

},

})

;

## src/screens/HomeScreen.js



|  |
| --- |
| await setDoc(doc(db, 'usuarios', auth.currentUser.uid), profile);  alert('Perfil actualizado exitosamente');  } catch (error) {  console.error('Error al actualizar perfil:', error);  alert('Error al actualizar perfil');  }  };  const handleSignOut = async () => {  try {  await signOut(auth);  navigation.replace('Login');  } catch (error) {  console.error('Error al cerrar sesión:', error);  }  };  return (  <View style={styles.container}> <Text h4 style={styles.title}>Mi Perfil</Text>  <Input  placeholder="Nombre"  value={profile.nombre} onChangeText={(text) => setProfile({...profile, nombre: text})}  />  <Input  placeholder="Apellido"  value={profile.apellido} onChangeText={(text) => setProfile({...profile, apellido: text})}  />  <Input  placeholder="Comida Favorita"  value={profile.comidaFavorita} onChangeText={(text) => setProfile({...profile, comidaFavorita: text})}  />  <Button  title="Actualizar Perfil"  onPress={handleUpdate}  containerStyle={styles.button}  />  <Button  title="Cerrar Sesión"  type="outline"  onPress={handleSignOut}  containerStyle={styles.button}  />  </View>  ); }  const styles = StyleSheet.create({  container: { |

flex

:

1

,

padding

:

20

,

},

title

:

{

textAlign

:

'center'

,

marginBottom

:

30

,

},

button

:

{

marginVertical

:

10

,

},

})

;

# Componente Principal

## App.js

import

React

from

'react'

;

import

{

SafeAreaProvider

}

from

'react-native-safe-area-context'

;

import

AppNavigator

from

'./src/navigation/AppNavigator'

;

export

default

function

App

()

{

return

(

<

SafeAreaProvider

>

<

AppNavigator

/>

<

/SafeAreaProvider>

);

}

# Pruebas y Ejecución

1. Asegurarse de que todas las dependencias estén instaladas:

npm

install

1. Iniciar la aplicación:

npx expo

start

1. Probar en emulador o dispositivo físico:

Presionar 'a' para Android

Presionar 'i' para iOS

Escanear código QR con Expo Go en dispositivo físico

# Funcionalidades Implementadas

Registro de usuarios con email y contraseña

Inicio de sesión

Persistencia de sesión

Formulario de perfil con datos personales

Almacenamiento en Firestore

Cerrar sesión

# Mejoras Posibles

1. Validación de formularios
2. Manejo de errores más detallado
3. Indicadores de carga
4. Diseño más elaborado
5. Foto de perfil
6. Implementar más proveedores de autenticación

# Debugging Común

1. **Error: "Firebase App not initialized"** Verificar firebaseConfig en firebase.js

Asegurar que las credenciales sean correctas

1. **Error: "Authentication failed"**

Verificar que Authentication esté habilitado en Firebase Console

Comprobar que el método email/password esté activado

1. **Error: "Permission denied" en Firestore** Revisar reglas de seguridad en Firestore

Verificar que el usuario esté autenticado

# Reglas de Seguridad Firestore

rules\_version

=

'2'

;

service

cloud

.

firestore

{

match

/

databases

/

{

database

}

/

documents

{

match

/

usuarios

/

{

userId

}

{

allow

read

,

write

:

if

request

.

auth

!=

null

&&

request

.

auth

.

uid

==

userId

;

}

}

}