

UNIVERSIDAD DON BOSCO

ESCUELA DE COMPUTACION
FACULTAD DE INGENIERIA



DESARROLLO DE SOFTWARE PARA IOS MANUAL DE USO TÉCNICO

Autores:

Lovo Juárez, Luis Fernando - #LJ222345
Pérez Ramírez, Miguel Alexander - #PR222602

INDICE

Introducción.....	3
Objetivo General	4
Objetivos específicos	4
Requerimientos.....	5
Requisitos Técnicos	5
Pruebas.	5
Mantenimiento y mejoras continuas.	5
Parte Técnica	6
Node.js:.....	6
React Native:.....	6
Visual Studio Code:	7
Instalación y uso implementado de Node.js	8
Instalación de Git y clonación de repositorio.	9
Instalación y uso implementado de React Native y Expo Dev.	10
Aplicación en ejecución:	13

Introducción

El Salvador es un país con una gran riqueza natural y cultural. Cuenta con una variedad de destinos turísticos que atraen a visitantes de todo el mundo.

Una aplicación de turismo para El Salvador puede ser una herramienta valiosa para los viajeros que visitan el país. Esta aplicación puede ofrecer una variedad de funcionalidades que pueden ayudar a los usuarios a planificar sus viajes, reservar alojamiento y actividades, y encontrar información sobre el destino.

La creación de una aplicación de turismo para El Salvador puede ser una oportunidad para contribuir al desarrollo del sector turístico del país. Una aplicación bien diseñada y ejecutada puede ayudar a atraer a más visitantes a El Salvador y generar ingresos para el país.

Objetivo General

Facilitar una experiencia turística enriquecedora y personalizada para los usuarios, ofreciendo información detallada, recomendaciones y herramientas interactivas que promuevan la exploración, el descubrimiento y el disfrute de destinos turísticos, contribuyendo así al crecimiento del turismo sostenible y a la satisfacción integral de los viajeros.

Objetivos específicos

- **Facilitar información detallada:** Proporcionar datos precisos y actualizados sobre destinos turísticos, incluyendo puntos de interés, actividades, horarios, tarifas, reseñas de usuarios, etc.
- **Personalización:** Ofrecer recomendaciones personalizadas basadas en los intereses, preferencias y comportamientos de los usuarios para mejorar su experiencia turística.
- **Funcionalidades interactivas:** Incorporar herramientas interactivas como mapas interactivos, realidad aumentada, guías de audio, tours virtuales, entre otras, para mejorar la experiencia del usuario durante su viaje.
- **Fomentar el turismo responsable:** Promover prácticas sostenibles y éticas en el turismo, destacando opciones eco-amigables, consejos para reducir el impacto ambiental y cultural, etc.
- **Facilitar la planificación y gestión de viajes:** Permitir a los usuarios planificar itinerarios, reservar alojamientos, adquirir boletos para actividades o eventos, y gestionar su agenda de viaje de manera eficiente.
- **Medición de satisfacción:** Implementar herramientas para recopilar retroalimentación de los usuarios, permitiendo mejorar continuamente la calidad de los servicios ofrecidos.

Requerimientos

Requisitos Técnicos

1. Dispositivo móvil: (Tablet o smartphone)
2. Mínimo 2GB de espacio en almacenamiento interno
3. Sistema operativo: Android o iOS
4. Versión mínima de Android: Marshmallow, 6.0, 6.0.1
5. Versión mínima de iOS: 12.0

Pruebas.

La aplicación se ha desarrollado con una interfaz responsiva, esto quiere decir que se adapta a cualquier dimensión de pantalla de diferentes dispositivos donde se han realizado pruebas de control de calidad para garantizar que sea compatible con la mayoría de los dispositivos. Además, se han desarrollado pruebas de carga de contenido para asegurarse que la información se muestre de manera rápida y completa.

Mantenimiento y mejoras continuas.

El proceso de levantamiento de requerimientos es esencial para garantizar que la transición del sitio web a una aplicación móvil sea exitosa y cumpla con las expectativas de los usuarios. Al hacer actualizaciones programadas y escalamiento continuo que garantiza una mejor experiencia y corrección de errores donde exista poco margen de error.

Parte Técnica

Es importante tener en cuenta que una aplicación exitosa de turismo para El Salvador debería ser precisa, confiable, fácil de usar y estar respaldada por información actualizada y relevante para los viajeros. Además, la colaboración con entidades gubernamentales, proveedores de servicios turísticos y la comunidad local sería fundamental para garantizar la autenticidad y calidad de la información proporcionada.

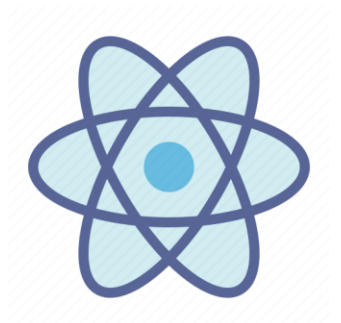


Node.js:

Es un entorno de tiempo de ejecución de JavaScript (de ahí su terminación en “.js” haciendo alusión al lenguaje JavaScript). Este entorno de tiempo de ejecución en tiempo real incluye todo lo que se necesita para ejecutar un programa escrito en JavaScript. También aporta muchos beneficios y soluciona muchísimos problemas.

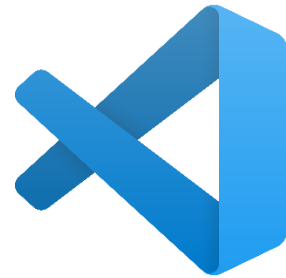
React Native:

React Native es un framework JavaScript para crear aplicaciones nativas. Se usa para desarrollar aplicaciones para Android, iOS, Web y UWP (Windows) y proporciona controles de interfaz de usuario nativa y acceso completo a la plataforma nativa. Trabajar con React Native requiere un conocimiento de los aspectos básicos de JavaScript.



Visual Studio Code:

Es un editor de código fuente ligero pero eficaz que se ejecuta en el escritorio y está disponible para Windows, macOS y Linux. Incluye compatibilidad integrada con JavaScript, TypeScript y Node.js, y cuenta con un amplio ecosistema de extensiones para otros lenguajes y entorno de ejecución (como C++, C#, Java, Python, Go, .NET).



Firebase:

Es una plataforma ubicada en la nube, integrada con Google Cloud Platform, que usa un conjunto de herramientas para la creación y sincronización de proyectos que serán dotados de alta calidad, haciendo posible el crecimiento del número de usuarios y dando resultado también a la obtención de una mayor monetización.



Firebase

Instalación y uso implementado de Node.js

Visite el sitio web oficial de Node.js y descargue el instalador para su sistema operativo.



El uso de node.js lo vemos reflejado a la hora de entrar a nuestro Símbolo del sistema y ejecutar el comando **“node --version”**:

```
C:\WINDOWS\system32\cmd. X + v
Microsoft Windows [Versión 10.0.22621.2715]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\carlo>node --version
v18.18.0

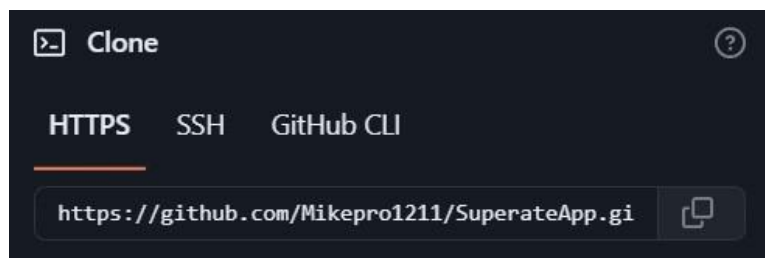
C:\Users\carlo>
```


Instalación de Git y clonación de repositorio.

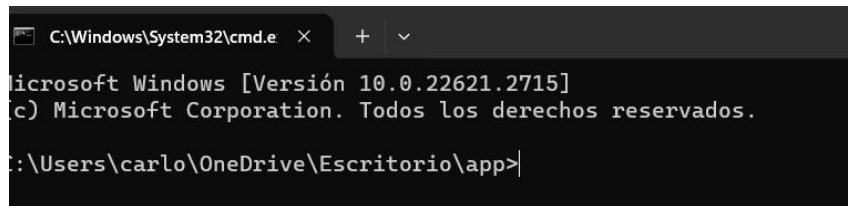
```
bolajiyadeji@Bolajis-MacBook: ~/Desktop/nextjs-blog
→ nextjs-blog git:(passwordless-auth) git remote show origin
* remote origin
Fetch URL: git@github.com:BolajiAyodeji/nextjs-blog.git
Push URL: git@github.com:BolajiAyodeji/nextjs-blog.git
HEAD branch: master
Remote branches:
  dev          tracked
  master       tracked
  passwordless-auth tracked
  staging      tracked
Local branches configured for 'git pull':
  master          merges with remote master
  passwordless-auth merges with remote passwordless-auth
Local refs configured for 'git push':
  master          pushes to master          (up to date)
  passwordless-auth pushes to passwordless-auth (up to date)
→ nextjs-blog git:(passwordless-auth) █
```

Posteriormente clonamos nuestro repositorio para acceder a nuestro proyecto.

Repositorio: <https://github.com/Mikepro1211/SuperateApp.git>

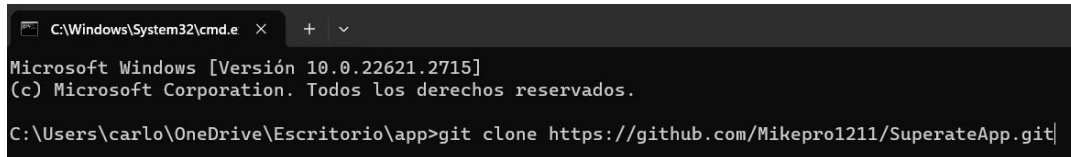


Creamos una nueva carpeta donde estará alojado nuestro proyecto, abrimos y ejecutamos el siguiente comando desde la consola de Windows (CMD):



```
C:\Windows\System32\cmd.e X + v
Microsoft Windows [Versión 10.0.22621.2715]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\carlo\OneDrive\Escritorio\app>
```



```
C:\Windows\System32\cmd.e X + v
Microsoft Windows [Versión 10.0.22621.2715]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\carlo\OneDrive\Escritorio\app>git clone https://github.com/Mikepro1211/SuperateApp.git
```

Ahora que tenemos nuestro proyecto clonado, lo que nos interesa a nosotros de node.js es el comando “npm” el cual es el manejador de paquetes de Node js, con él podemos instalar dependencias a nuestro proyecto o instalar programas globalmente en nuestro sistema.

Instalación y uso implementado de React Native y Expo Dev.

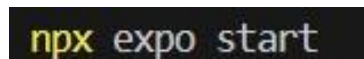
Para correr la aplicación es necesario mantener previamente instalado el Node.js, teniendo en cuenta eso, en su sitio web <https://reactnative.dev/docs/environment-setup> encontraremos lo necesario para realizar su instalación lo cual se base en tener a la mano los códigos de instalación según tu sistema operativo (en nuestro caso, Windows).

Accedemos a la carpeta donde tenemos nuestro proyecto instalado y ejecutamos el siguiente comando:



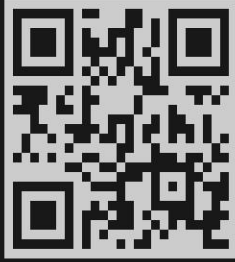
```
npm install
```

Con este comando instalaremos todos los paquetes necesarios para poder empezar a ejecutar la aplicación. Posteriormente hacemos uso del comando “**npx expo start**” para poder correr nuestra aplicación y escanear el código QR para visualizarla en nuestro dispositivo.



```
npx expo start
```

Escaneamos el código QR de nuestro proyecto:



```
> Metro waiting on exp://192.168.0.9:8081
> Scan the QR code above with Expo Go (Android) or the Camera app (iOS)

> Using Expo Go
> Press s | switch to development build

> Press a | open Android
> Press w | open web

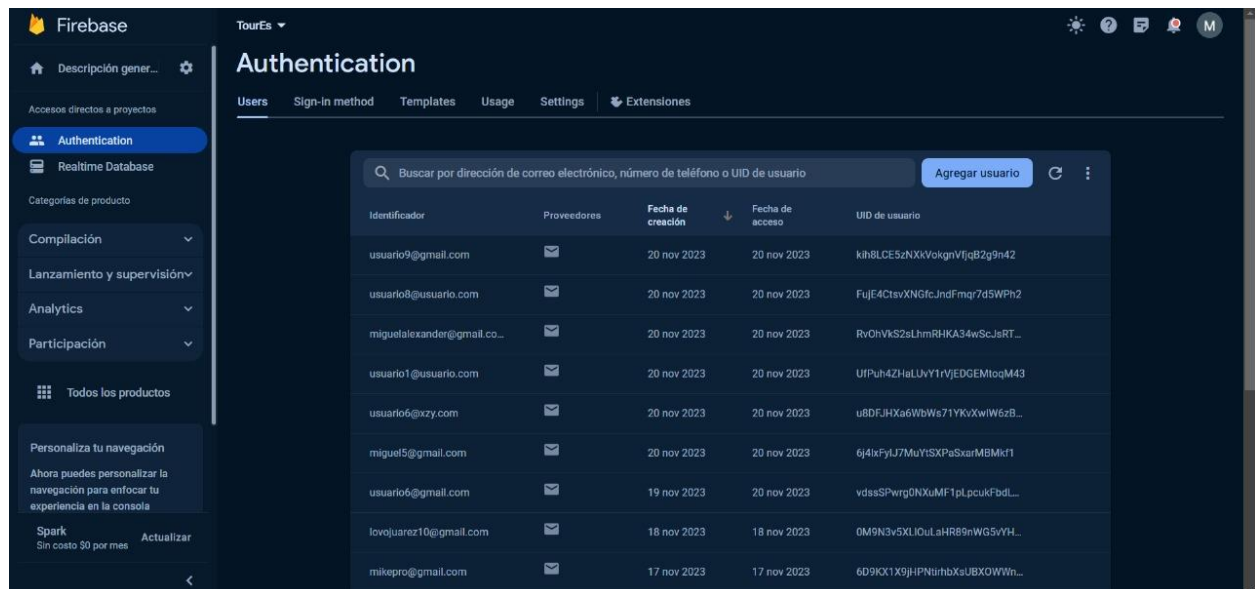
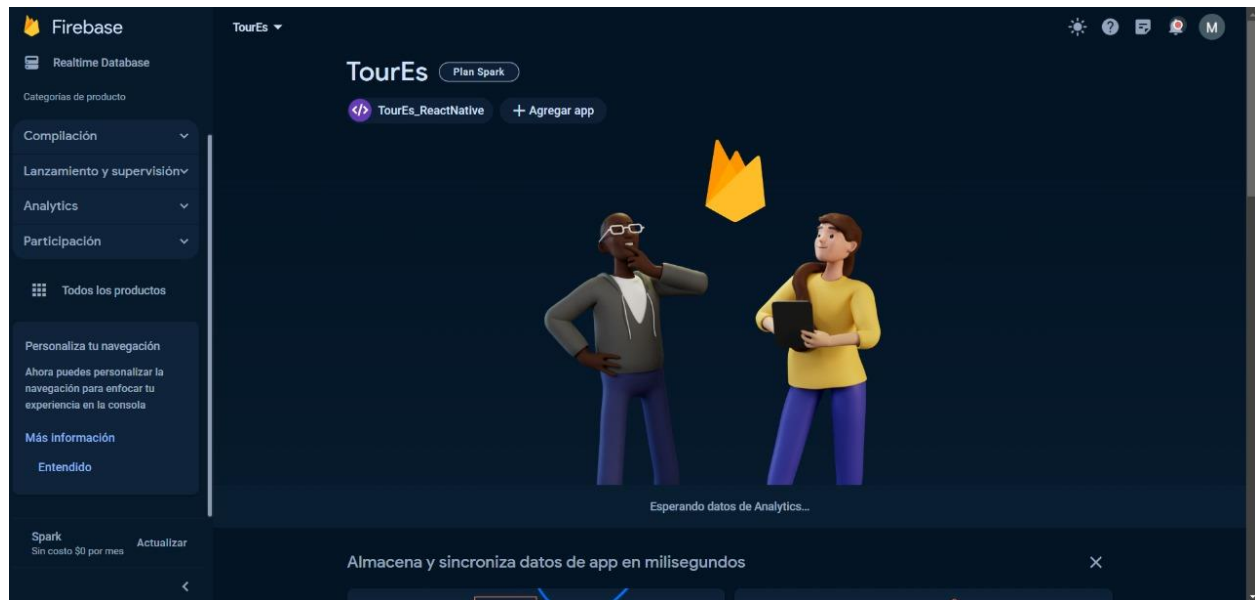
> Press j | open debugger
> Press r | reload app
> Press m | toggle menu
> Press o | open project code in your editor

> Press ? | show all commands

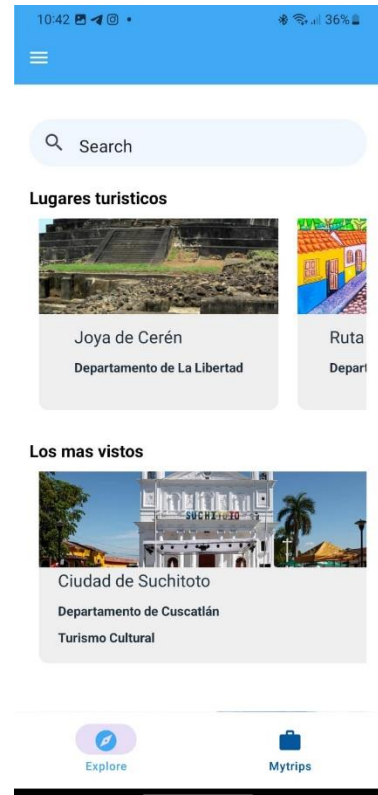
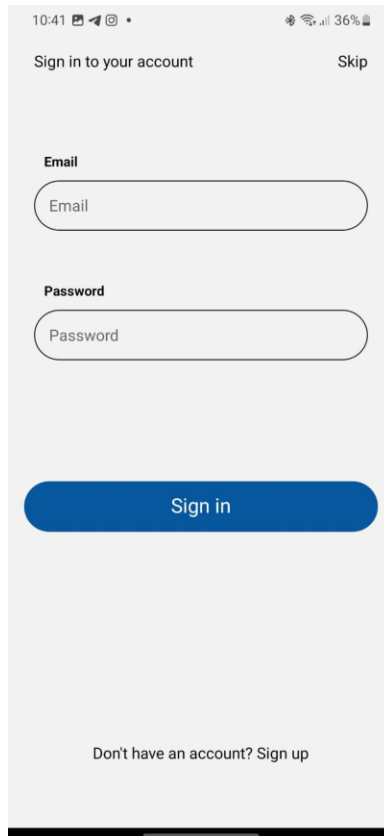
Logs for your project will appear below. Press Ctrl+C to exit.
█
```

Base de datos

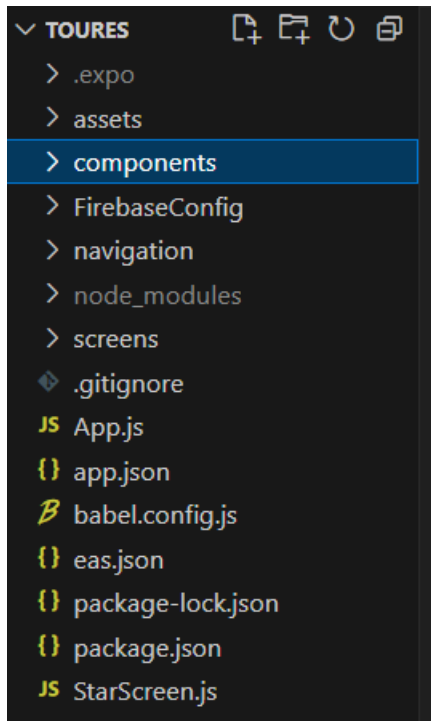
En este caso tenemos a firebase como nuestra base de datos donde esta conectada con el login y donde podemos ver los diferentes usuarios que están ingresados en nuestra aplicación, además de esto podemos tener un registro de estos



Aplicación en ejecución:



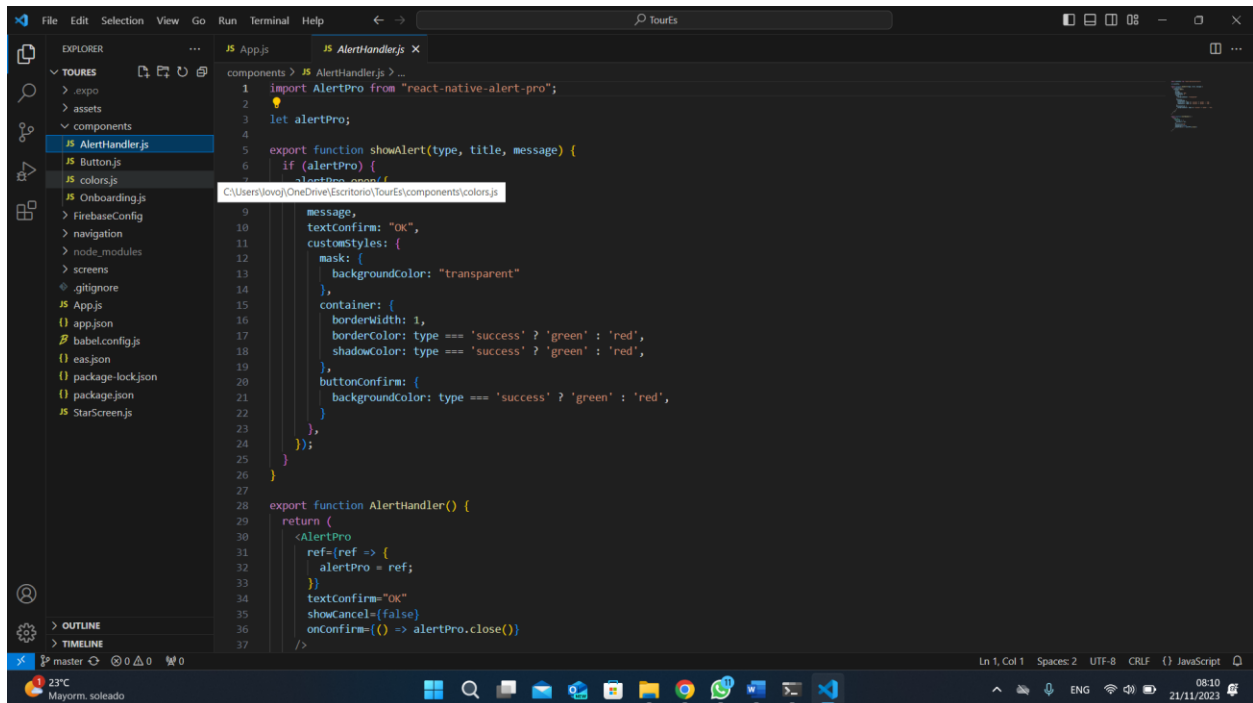
Código:



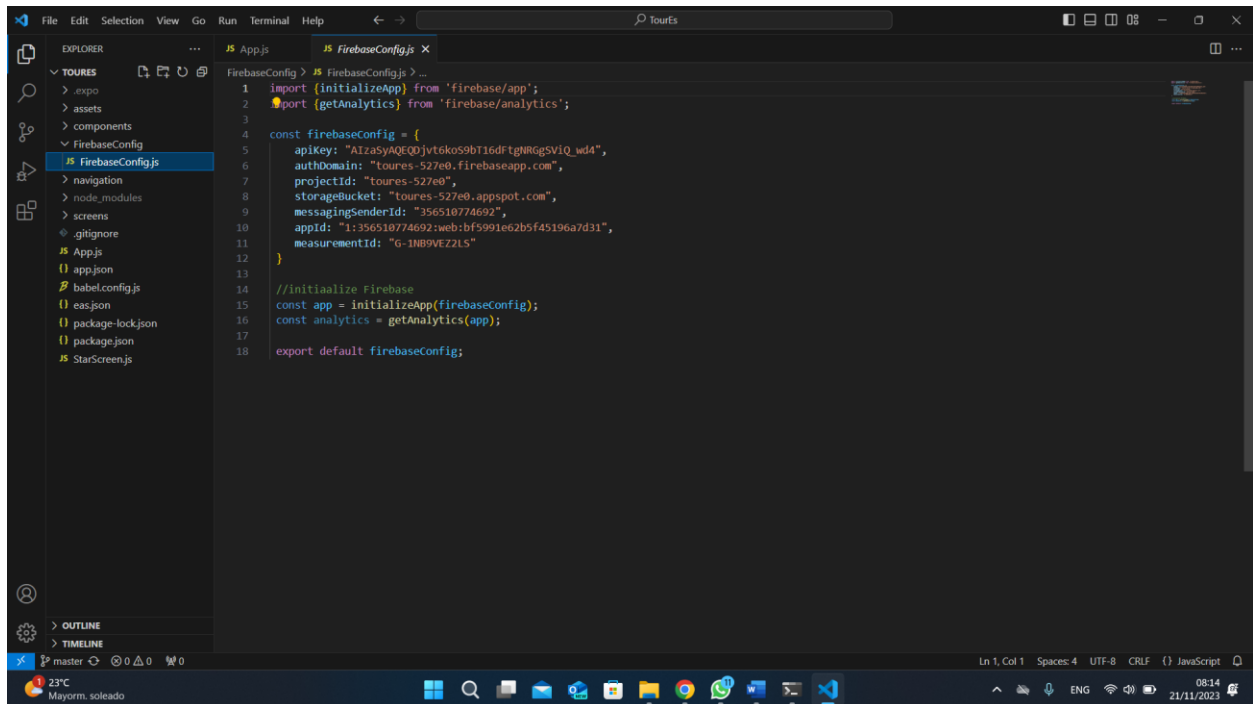
Dentro del proyecto obtendremos diferentes carpetas:

- **Components**
- **FirebaseConfig**
- **Navigation**
- **screens**

Components: tenemos diferentes componentes donde nosotros damos diferentes usos

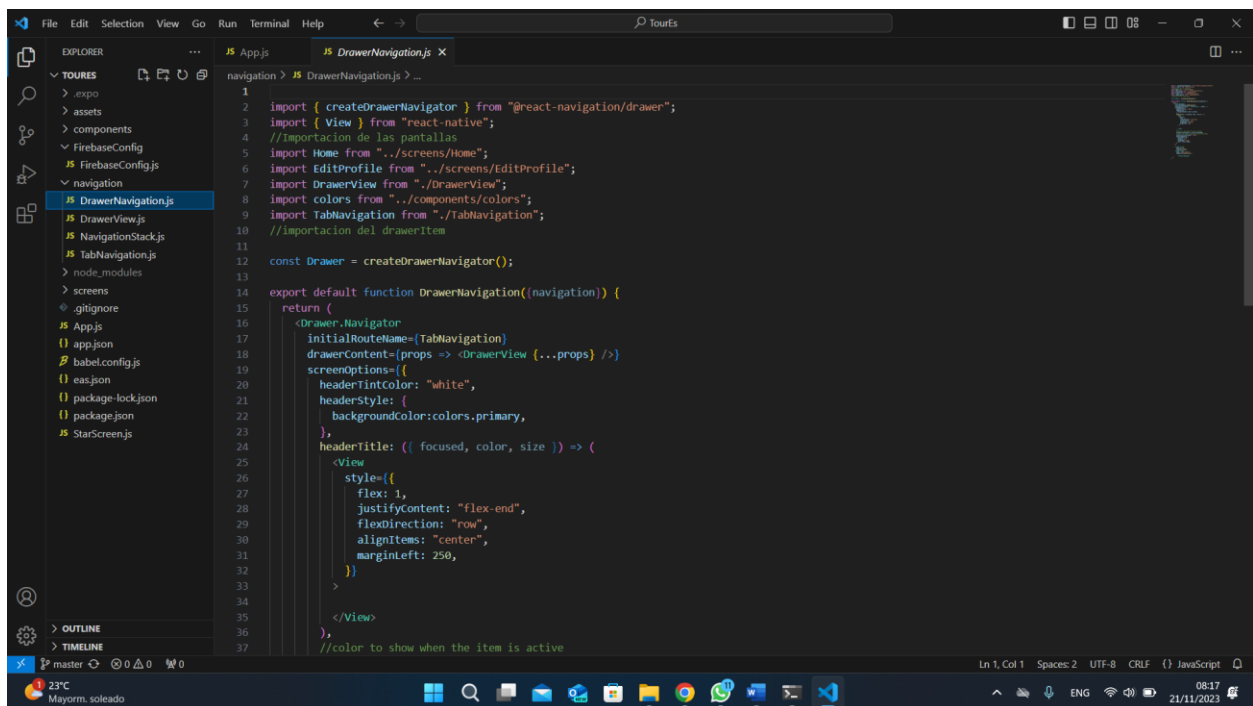


FirebaseConfig: tenemos las configuraciones usadas para firebase

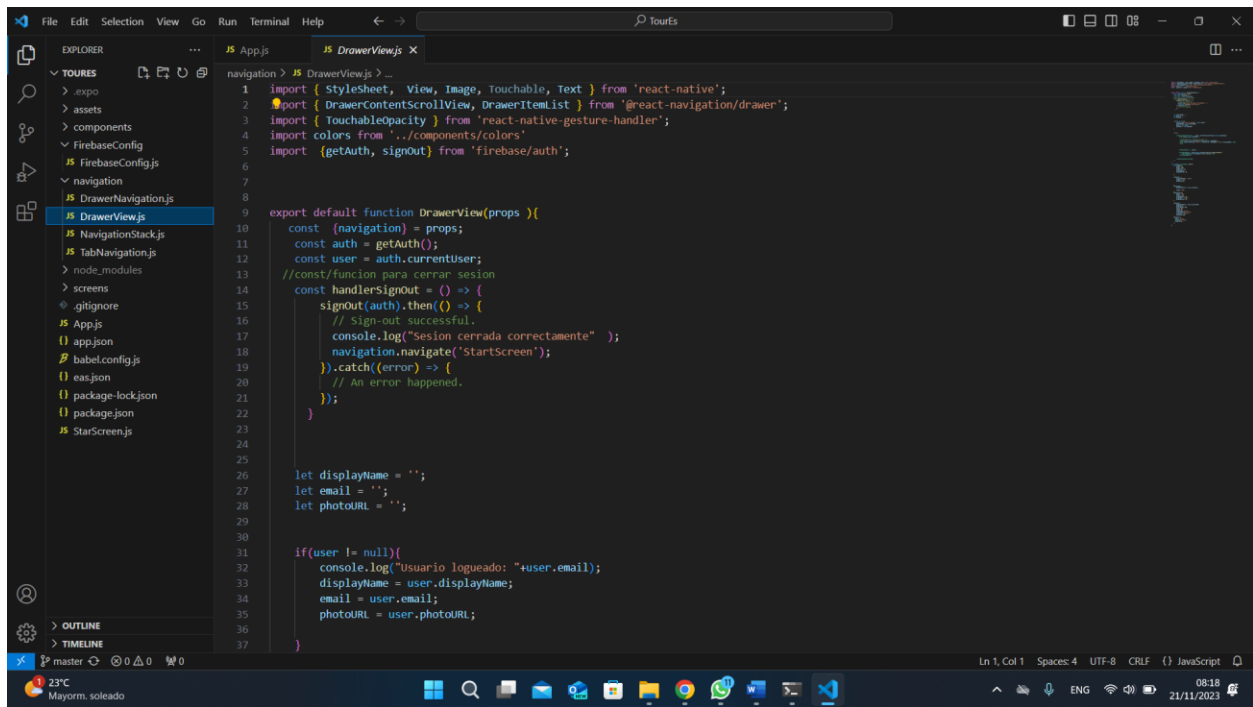


```
1 import { initializeApp } from 'firebase/app';
2 import { getAnalytics } from 'firebase/analytics';
3
4 const firebaseConfig = {
5   apiKey: "AIzaSyAQEQJy6k0S9b11dFtgmGgSVlQ_wd4",
6   authDomain: "toures-527e0.firebaseio.com",
7   projectId: "toures-527e0",
8   storageBucket: "toures-527e0.appspot.com",
9   messagingSenderId: "356510774692",
10  appId: "1:356510774692:web:bf5991e62b5f45196a7d31",
11  measurementId: "G-1N89VEZ2LS"
12 }
13
14 //initialize Firebase
15 const app = initializeApp(firebaseConfig);
16 const analytics = getAnalytics(app);
17
18 export default firebaseConfig;
```

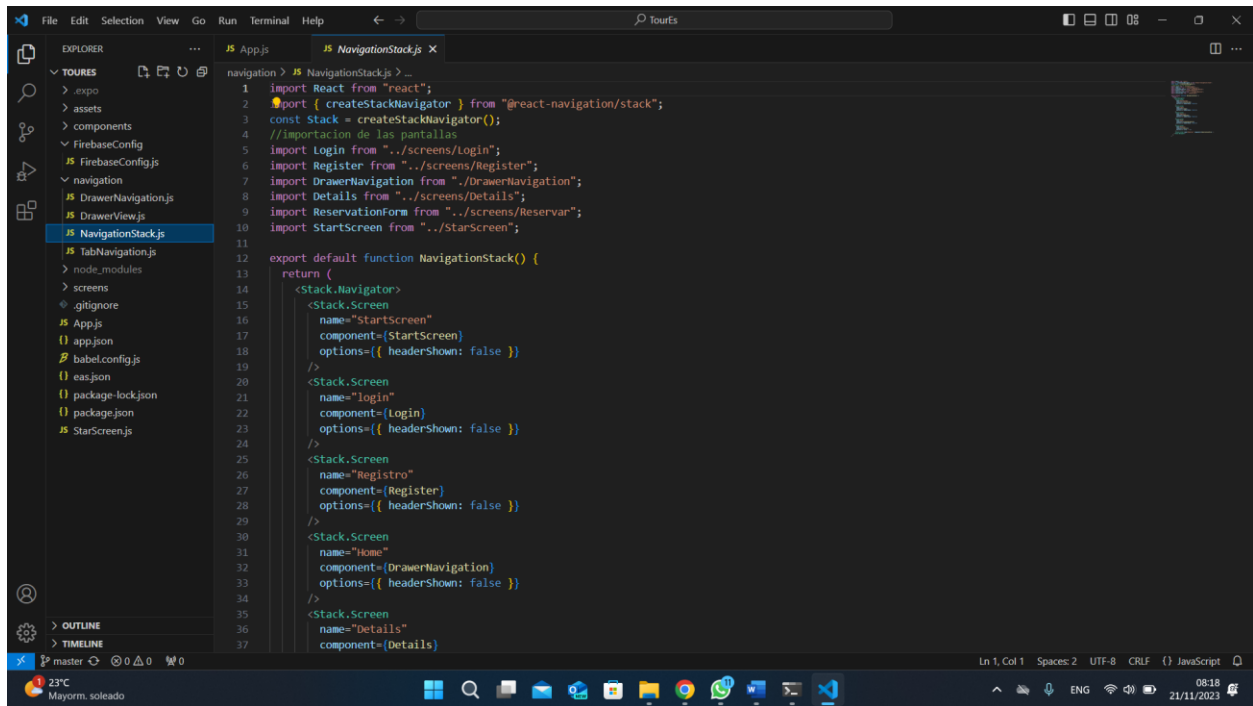
Navigation: tenemos la forma de navegación usando el drawer



```
1 import { createDrawerNavigator } from '@react-navigation/drawer';
2 import { View } from 'react-native';
3 //importacion de las pantallas
4 import Home from '../screens/home';
5 import EditProfile from '../screens/EditProfile';
6 import DrawerView from '../DrawerView';
7 import colors from '../components/colors';
8 import TabNavigation from './TabNavigation';
9 //importacion del drawerItem
10
11 const Drawer = createDrawerNavigator();
12
13 export default function DrawerNavigation((navigation)) {
14   return (
15     <Drawer.Navigator
16       initialRouteName=TabNavigation
17       drawerContent={props => <DrawerView {...props} />}
18       screenOptions={{
19         headerTintColor: "white",
20         headerStyle: {
21           backgroundColor: colors.primary,
22         },
23         headerTitle: ({ focused, color, size }) => (
24           <View
25             style={{
26               flex: 1,
27               justifyContent: "flex-end",
28               flexDirection: "row",
29               alignItems: "center",
30               marginLeft: 250,
31             }}
32           />
33         ),
34       </View>
35     ),
36   );
37   //color to show when the item is active
```

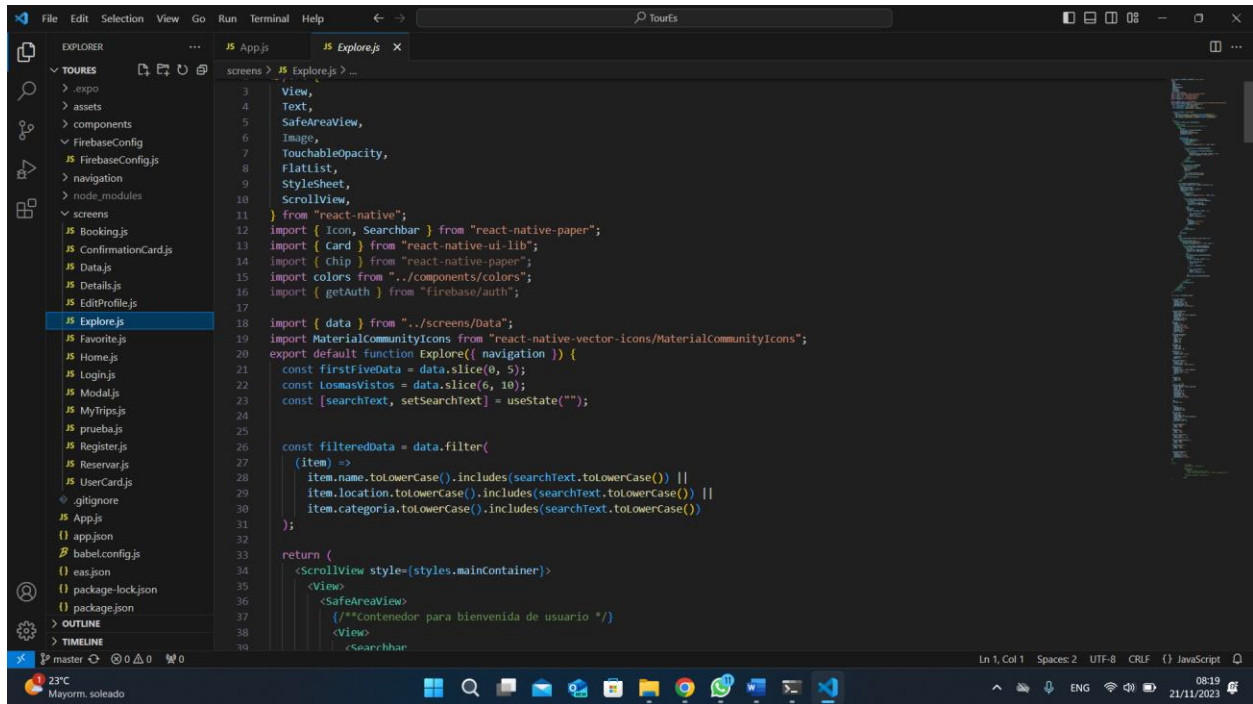


```
1 import { StyleSheet, View, Image, Touchable, Text } from 'react-native';
2 import { DrawerContentScrollView, DrawerItemList } from '@react-navigation/drawer';
3 import { TouchableOpacity } from 'react-native-gesture-handler';
4 import colors from '../components/colors';
5 import { getAuth, signOut } from 'firebase/auth';
6
7
8
9 export default function DrawerView(props) {
10   const { navigation } = props;
11   const auth = getAuth();
12   const user = auth.currentUser;
13   //const/function para cerrar sesion
14   const handlersignout = () => {
15     signOut(auth).then(() => {
16       // Sign-out successful.
17       console.log("Sesion cerrada correctamente" );
18       navigation.navigate("StartScreen");
19     }).catch((error) => {
20       // An error happened.
21     });
22   };
23
24   let displayName = '';
25   let email = '';
26   let photoURL = '';
27
28   if(user != null){
29     console.log("Usuario logueado: "+user.email);
30     displayName = user.displayName;
31     email = user.email;
32     photoURL = user.photoURL;
33   }
34 }
```



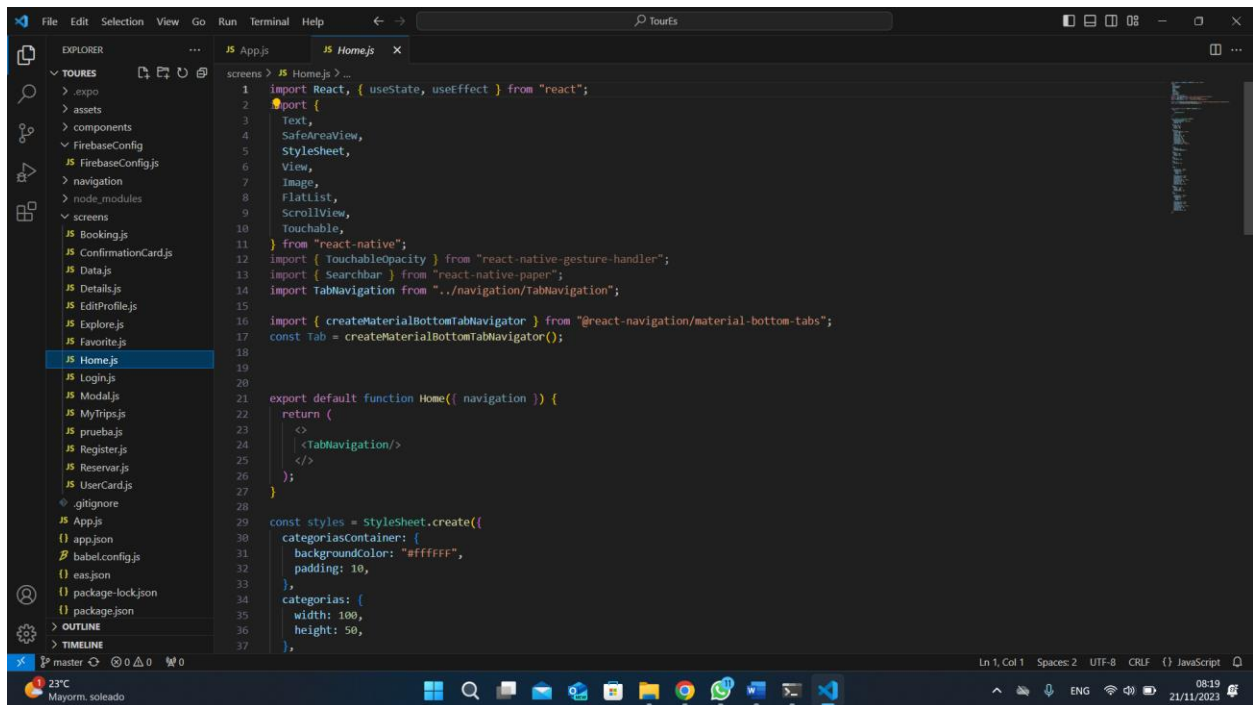
```
1 import React from "react";
2 import { createStackNavigator } from "@react-navigation/stack";
3 const Stack = createStackNavigator();
4 //importacion de las pantallas
5 import Login from "../screens/Login";
6 import Register from "../screens/Register";
7 import DrawerNavigation from "../DrawerNavigation";
8 import Details from "../screens/Details";
9 import ReservationForm from "../screens/Reservar";
10 import StartScreen from "../StartScreen";
11
12 export default function NavigationStack() {
13   return (
14     <Stack.Navigator>
15       <Stack.Screen
16         name="StartScreen"
17         component={StartScreen}
18         options={{ headerShown: false }}
19       />
20       <Stack.Screen
21         name="Login"
22         component={Login}
23         options={{ headerShown: false }}
24       />
25       <Stack.Screen
26         name="Registro"
27         component={Register}
28         options={{ headerShown: false }}
29       />
30       <Stack.Screen
31         name="Home"
32         component={DrawerNavigation}
33         options={{ headerShown: false }}
34       />
35       <Stack.Screen
36         name="Details"
37         component={Details}
38       />
39     </Stack.Navigator>
40   );
41 }
```


Screens: tenemos las diferentes pantallas de navegación



The screenshot shows the VS Code editor with the Explorer sidebar on the left. The 'screens' directory is expanded, and 'Explore.js' is selected. The main editor displays the code for 'Explore.js', which is a React Native screen component. It imports various libraries like 'react-native', 'react-native-paper', 'react-native-vector-icons', and 'firebase/auth'. It also imports data from '../screens/Data'. The code defines an 'Explore' function that takes 'navigation' as a prop and returns a JSX element. The JSX element consists of a 'ScrollView' containing a 'SafeAreaView' and a 'View' with a 'Searchbar' component. The 'Searchbar' is used to filter a list of items based on search text.

```
1 View,
2 Text,
3 SafeAreaView,
4 Image,
5 TouchableOpacity,
6 FlatList,
7 StyleSheet,
8 ScrollView,
9 } from "react-native";
10 import { Icon, Searchbar } from "react-native-paper";
11 import { Card } from "react-native-ui-lib";
12 import { Chip } from "react-native-paper";
13 import colors from "../components/colors";
14 import { getAuth } from "firebase/auth";
15
16 import { data } from "../screens/Data";
17 import MaterialCommunityIcons from "react-native-vector-icons/MaterialCommunityIcons";
18 export default function Explore({ navigation }) {
19   const firstFiveData = data.slice(0, 5);
20   const losmasVistos = data.slice(6, 10);
21   const [searchText, setSearchText] = useState("");
22
23   const filteredData = data.filter(
24     (item) =>
25     item.name.toLowerCase().includes(searchText.toLowerCase()) ||
26     item.location.toLowerCase().includes(searchText.toLowerCase()) ||
27     item.categoria.toLowerCase().includes(searchText.toLowerCase())
28   );
29
30   return (
31     <ScrollView style={styles.mainContainer}>
32       <View>
33         <SafeAreaView>
34           {/*Contenedor para bienvenida de usuario */}
35           <View>
36             <Searchbar
```



The screenshot shows the VS Code editor with the Explorer sidebar on the left. The 'screens' directory is expanded, and 'Home.js' is selected. The main editor displays the code for 'Home.js', which is a React Native screen component. It imports 'react', 'react-native', 'react-native-gesture-handler', 'react-native-paper', and '@react-navigation/material-bottom-tabs'. It also imports 'createMaterialBottomTabNavigator' from '@react-navigation/material-bottom-tabs'. The code defines a 'Home' function that takes 'navigation' as a prop and returns a JSX element. The JSX element consists of a 'TabNavigation' component. The 'TabNavigation' is used to create a material bottom tab navigator with a single tab labeled 'Home'. The 'Home' tab is styled with a white background color, a padding of 10, and a width of 100.

```
1 import React, { useState, useEffect } from "react";
2 import {
3   Text,
4   SafeAreaView,
5   StyleSheet,
6   View,
7   Image,
8   FlatList,
9   ScrollView,
10  Touchable,
11 } from "react-native";
12 import { TouchableOpacity } from "react-native-gesture-handler";
13 import { Searchbar } from "react-native-paper";
14 import TabNavigation from "../navigation/TabNavigation";
15
16 import { createMaterialBottomTabNavigator } from "@react-navigation/material-bottom-tabs";
17 const Tab = createMaterialBottomTabNavigator();
18
19 export default function Home({ navigation }) {
20   return (
21     <TabNavigation>
22     </TabNavigation>
23   );
24 }
25
26 const styles = StyleSheet.create({
27   categoriasContainer: {
28     backgroundColor: "#ffffff",
29     padding: 10,
30   },
31   categorias: {
32     width: 100,
33     height: 50,
34   },
35 });
```

Appjs: tenemos el archivo principal

