

**UNIVERSIDAD PRIVADA FRANZ TAMAYO**

**FACULTAD DE INGENIERÍA**

**CARRERA DE INGENIERÍA DE SISTEMAS**



**“DEFENSA HITO 3: TAREA FINAL”**

Nombre completo: Glenn Julián Castro duarte

Asignatura: Programación de dispositivos móviles

Carrera: Ing. de sistemas

Docente: Lic. William R. Barra Paredes.

Fecha: 11 de mayo del 2020

Github: <https://github.com/leonjaeger/PDM-2020>

Cochabamba-Bolivia

## **Parte Teórica**

### **1.- Defina que es un componente en React Native y muestre un ejemplo**

Un componente en react es un elemento reutilizable e implementable tanto en cosas pequeñas como en grandes secciones de código implementado. Los componentes permiten separar la interfaz de usuario en piezas independientes, reutilizables y pensar en cada pieza de forma aislada.

Conceptualmente, los componentes son como las funciones de JavaScript. Aceptan entradas arbitrarias (llamadas "props") y devuelven a React elementos que describen lo que debe aparecer en la pantalla.

### **2.Explique como se realiza la navegación entre screens en React Native**

Para lograr una correcta navegación en react native hace falta la integración de los componentes navigator, native y stack, con ello podemos organizar los Screens disponibles en nuestro proyecto a una ruta definida por nosotros en un Main navigator al que podremos consultar mediante el comando navigate y posteriormente la ruta deseada previamente definida.

### **3.Que significa IaaS, PaaS, SaaS.-**

IaaS.- es un método para ofrecer funcionalidades de computación, almacenamiento, redes y de otros tipos a través de Internet. La IaaS permite a las empresas utilizar sistemas de funcionamiento, aplicaciones y almacenamiento basados en la web sin tener que comprar, administrar y brindar soporte a la infraestructura de nube subyacente.

PaaS.- es una oferta de computación en nube que proporciona a los usuarios un entorno de nube en el que pueden desarrollar, gestionar y entregar aplicaciones. Los usuarios, además del almacenamiento y de otros recursos de computación, pueden utilizar un conjunto de herramientas preconstruidas para desarrollar, personalizar y probar sus propias aplicaciones.

SaaS.- es un modelo de distribución de software en el que tanto el software como los datos manejados son centralizados y alojados en un único servidor externo a la empresa.

### **4.-Que es Firebase, Firestore y explique a que se refiere cuando se habla de BaaS**

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.

Firestore: Se podría decir que es la base de datos de firebase, donde son almacenados los datos para ser posteriormente usados en aplicaciones y programas mediante el empleo de consultas en tiempo real.

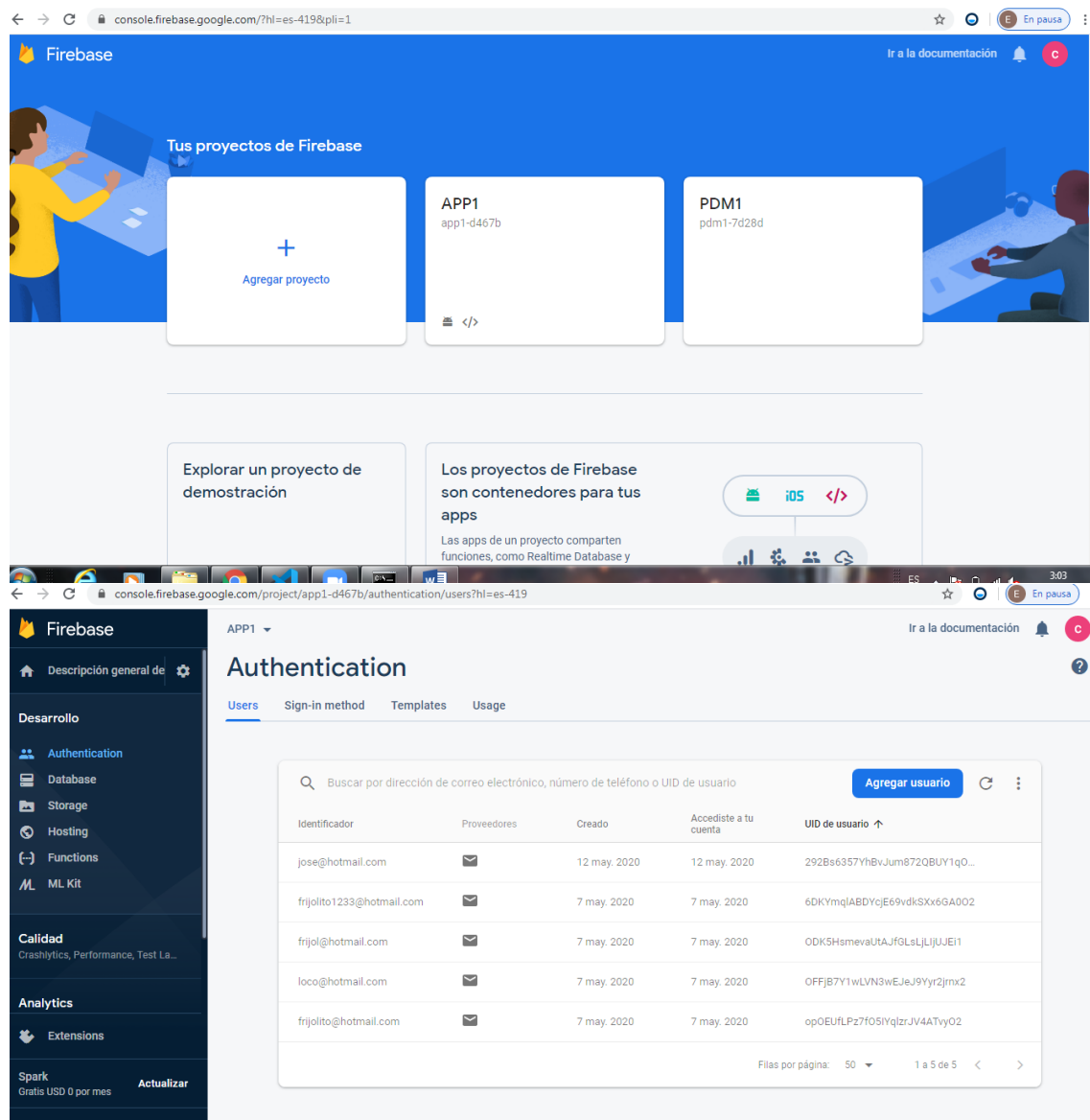
Backend as Service: BaaS es todo un conjunto de utilerías ya implementadas en la nube, en la cual solo nos preocupamos por desarrollar las funciones de negocio o servicios que necesitará nuestra aplicación para funcionar, de tal forma que, en lugar de desplegar aplicaciones, desplegamos funciones que posteriormente serán expuestas como servicios para ser consumidas por la red.

## 5.- Defina o explique si React es lo mismo que react Native. Si son distintos cuales son las diferencias.

No son lo mismo, a pesar que react forme la base para lo que es react native, React es una librería Javascript focalizada en el desarrollo de interfaces de usuario, así se define la propia librería y evidentemente, esa es su principal área de trabajo. Mientras que react Native es un marco móvil que se compila en componente de aplicaciones nativas, lo que le permite crear aplicaciones móviles nativas para diferentes plataformas en JavaScript que le permite usar ReactJS para crear sus componentes.

Parte Practica:

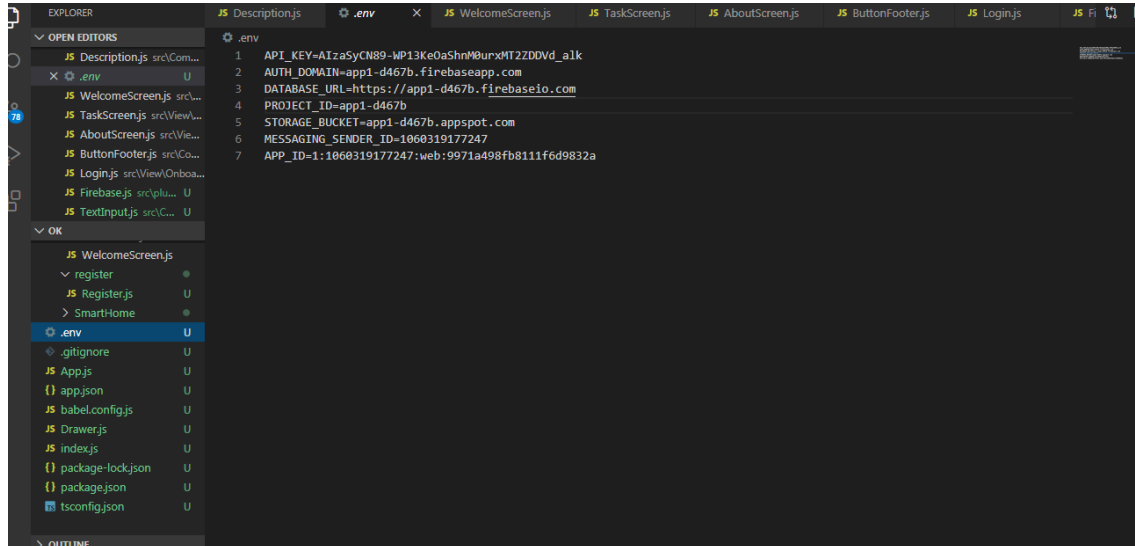
### 1.- Nuevo proyecto en Firebase y Permitir la autenticación de Email



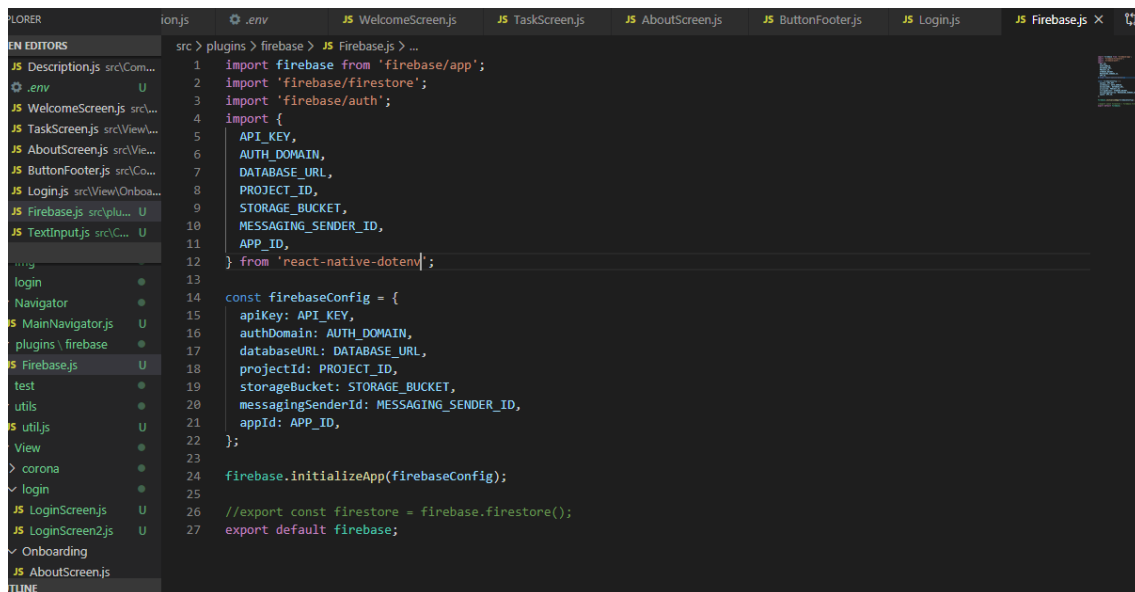
The screenshot shows the Firebase console interface. The top navigation bar includes the Firebase logo, the text 'Tus proyectos de Firebase', and buttons for 'Agregar proyecto', 'APP1', and 'PDM1'. Below this, there are sections for 'Explorar un proyecto de demostración' and 'Los proyectos de Firebase son contenedores para tus apps'. The main content area is titled 'Authentication' and includes a search bar and a table of users.

Identificador	Proveedores	Creado	Accediste a tu cuenta	UID de usuario
jose@hotmail.com	✉	12 may. 2020	12 may. 2020	292Bs6357YhBvJum872QBUI1q0...
frijolito1233@hotmail.com	✉	7 may. 2020	7 may. 2020	6DKYmqIABDYcjE69vdkSXx6GA002
frijol@hotmail.com	✉	7 may. 2020	7 may. 2020	ODK5HsmevaUTAjfgLsLjLjJUE1
loco@hotmail.com	✉	7 may. 2020	7 may. 2020	OFFjB7Y1wLVN3wEJeJ9Yyr2jmx2
frijolito@hotmail.com	✉	7 may. 2020	7 may. 2020	op0EUfLPz7f05IYqlzrJV4ATvy02

2. Configuración de la variables del enviroment .env en react native para determinar la DB que se va usar



Y para su posterior uso se crea el siguiente plugin



### 3. Creación De los componentes y la Screen PDM

Para poder generar los elementos que conformaran las Screens requeridas para el proyecto, se crearan los siguientes componentes:

Button: Es un `TouchableOpacity` que recibirá una variable `onPress` y una variable que definirá el texto del botón, de color transparente

ButtonPN: un par de botones importados desde `Button`, ordenados en forma de fila con la propiedad `row`, a los cuales también se le encadenara las propiedades de titulo y `OnPress`

Descripción: contendrá la imagen y el texto de cada Screen junto con un logo que cambiara según el Screen

onboardingFactory: Es el componente donde se importan y empleando todos los de mas componentes antes mencionados y se les asigna valores a los componentes como el Onpress de los botones, las imágenes de logo, los textos de la descripción, los títulos de los botones prev y next. Todo para obtener un resultado similar a este:



```

JS OnboardingFactory.js x JS Description.js JS WelcomeScreen.js JS TaskScreen.js JS AboutScreen.js JS ButtonFooter.js JS Login.js
src > Components > Onboarding > JS OnboardingFactory.js > OnboardingFactory > render
22 | console.log('Prev')
23 | };
24 |
25 | export default class OnboardingFactory extends Component{
26 |   constructor(props){
27 |     super(props);
28 |   }
29 |   render(){
30 |     return(
31 |       <View style={this.props.stilo} >
32 |         <View style={{marginTop:100}}>
33 |           <Description
34 |             source={this.props.source}
35 |             text1={this.props.text1}
36 |             text2={this.props.text2}
37 |             text3={this.props.text3}
38 |           />
39 |           <ButtonFooter
40 |             style={{marginTop:100}}
41 |             Next={this.props.Next}
42 |             onPressNext={this.props.onPressNext}
43 |             Prev={this.props.Prev}
44 |             onPressPrev={this.props.onPressPrev}
45 |           />
46 |         </View>
47 |       </View>
48 |     );
49 |   }
50 | }
51 | const stylesLoginScreen = StyleSheet.create({
52 |   container: {

```

#### 4.-Creacion del Screen DEFENSAHITO3

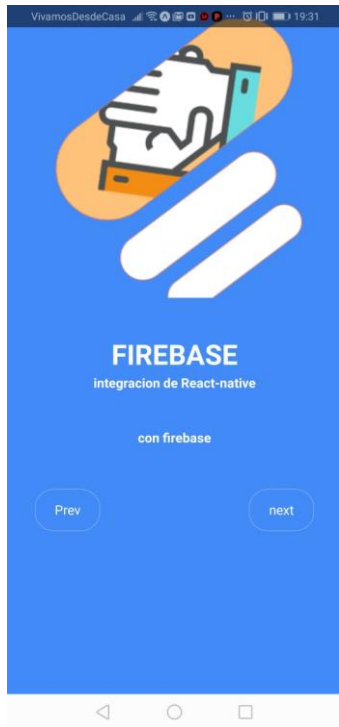
Similar al creen anteriormente creado, se debe modificar el valor que reciben los valores que recibe el componente como también las rutas a las cuales se dirigirá la función navigate.



```
src > View > Onboarding > JS AboutScreenjs > AboutScreen
1  import React, {useState} from 'react';
2  import {
3    StyleSheet,
4    View,
5    SafeAreaView,
6    KeyboardAvoidingView,
7    Alert,
8  } from 'react-native';
9  import Images from '../Config/images';
10 import Colors from '../Config/Colors';
11 import OnboardingFactory from '../Components/Onboarding/OnboardingFactory';
12 const AboutScreen = ({navigation}) => {
13
14   return(
15     <OnboardingFactory
16       stilo={stylesloginScreen.container}
17       source={Images.SCREEN2}
18       Next={'next'}
19       Prev={['Prev']}
20       text1={'Defensa Hito 3'}
21       text2={'Univ:Glenn Julian Castro Duarte'}
22       text3={'gestion 2020'}
23       onPressNext={()=> navigation.navigate('TaskScreen')}
24       onPressPrev={()=> navigation.navigate('WelcomeScreen')}
25     ></OnboardingFactory>
26
27   );
28
29 };
30
31
32 const stylesloginScreen = StyleSheet.create({
33   container: {
```

## 5.- Creación del Screen FireBase

Se seguirán los pasos antes mencionados para las pantallas previas haciendo también cambios en la función `navigate` para que esta lleve al login

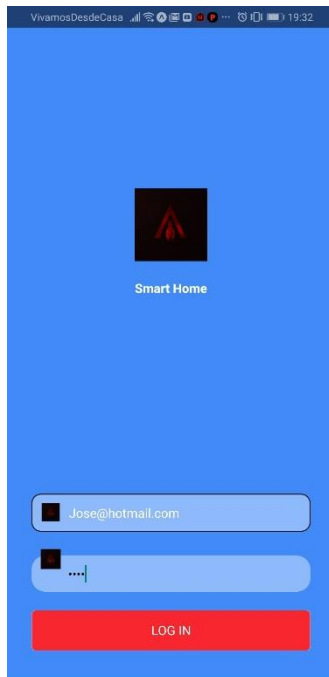


```
src > View > Onboarding > JS TaskScreens > TaskScreen
6   KeyboardAvoidingView,
7   Alert,
8 } from 'react-native';
9
10  import Images from '../../Config/images';
11  import Colors from '../../Config/Colors';
12  import OnboardingFactory from '../../Components/Onboarding/OnboardingFactory';
13  const TaskScreen = ({navigation}) => {
14
15    return(
16      <OnboardingFactory
17        stilo={stylesLoginScreen.container}
18        source={Images.SCREEN3}
19        Next={'next'}
20        Prev={'Prev'}
21        text1={'FIREBASE'}
22        text2={'integracion de React-native'}
23        text3={'con firebase'}
24        onPressPrev={()=> navigation.navigate('AboutScreen')}
25        onPressNext={()=> navigation.navigate('login')}
26      ></OnboardingFactory>
27    );
28  );
29
30  );
31
32  };
33  const stylesLoginScreen = StyleSheet.create({
34    container: {
35      flex: 1,
36      backgroundColor: Colors.blue,
```

## 6.- Creación y validación de login con Firebase



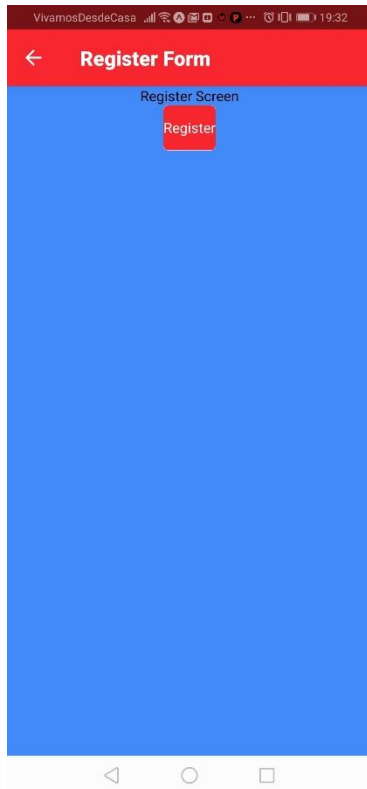
Haciendo uso de componentes previamente diseñados como ser Button, Logo, EmailTextField y el textInput, se creara una función para verificar mediante el plugin añadido que servirá para realizar la autenticación y la validación de campos incompletos.



```
src > View > Onboarding > JS Login.js > stylesLoginScreen > container
73  * @name loginApp
74  * @param {string} email
75  * @param {string} password
76  */
77  const loginApp = (email, password) => {
78    try {
79      setIsLoading(true);
80      FirebasePlugin.auth()
81        .signInWithEmailAndPassword(email, password)
82        .then(user => {
83          setIsLoading(false);
84          navigation.navigate('register');
85        })
86        .catch(error => {
87          FirebasePlugin.auth()
88            .createUserWithEmailAndPassword(email, password)
89            .then(user => {
90              setIsLoading(false);
91              navigation.navigate('register');
92            })
93            .catch(error => {
94              setIsLoading(false);
95              Alert.alert('Invalid Values', error.message);
96            });
97        });
98    } catch (error) {
99      setIsLoading(true);
100      Alert.alert('Firebase Error', error.message);
101    }
102  };
103
104
105
```

## 7.- Creación del Screen Register

Una vez concluida la validación de Firebase, se mostrara un componente vacío con un simple botón, pero que solo será visible para un usuario logueado.



```
JS ButtonFooter.js JS Login.js JS Firebase.js JS Button.js JS TextInput.js JS Register.js ...register JS Register
src > View > Onboarding > JS Register.js > ...
1  import React from 'react';
2  import {StyleSheet, View, Text} from 'react-native';
3
4  import Constants from '../../Config/constants';
5  import Colors from '../../Config/Colors';
6  import ButtonLogin from '../../Components/login/Button';
7
8  const RegisterScreen = ({navigation}) => {
9    const onPress = () => {
10      console.log('register');
11    };
12
13    return (
14      <View style={styles.container}>
15        <Text>Registro Completado</Text>
16        <ButtonLogin onPress={onPress} titleButton={Constants.STRING.REGISTER} />
17      </View>
18    );
19  };
20
21  const styles = StyleSheet.create({
22    container: {
23      flex: 1,
24      backgroundColor: Colors.blue,
25      alignItems: 'center',
26    },
27    text: {
28      color: Colors.white,
29      textAlign: 'center',
30      fontWeight: 'bold',
31      height: 20,
32    },
33  });
```