**UNIVERSIDAD PRIVADA FRANZ TAMAYO**

**FACULTAD DE INGENIERÍA**

**CARRERA DE INGENIERÍA DE SISTEMAS**

****

**DEFENSA HITO 3 - TAREA FINAL**

ESTUDIANTES: Raquel Terrazas

**PDM**

DOCENTE:  Lic. William R. Barra Paredes

Cochabamba - Bolivia

09/05/2020

<https://github.com/Drakitt/PDM/tree/hito3>

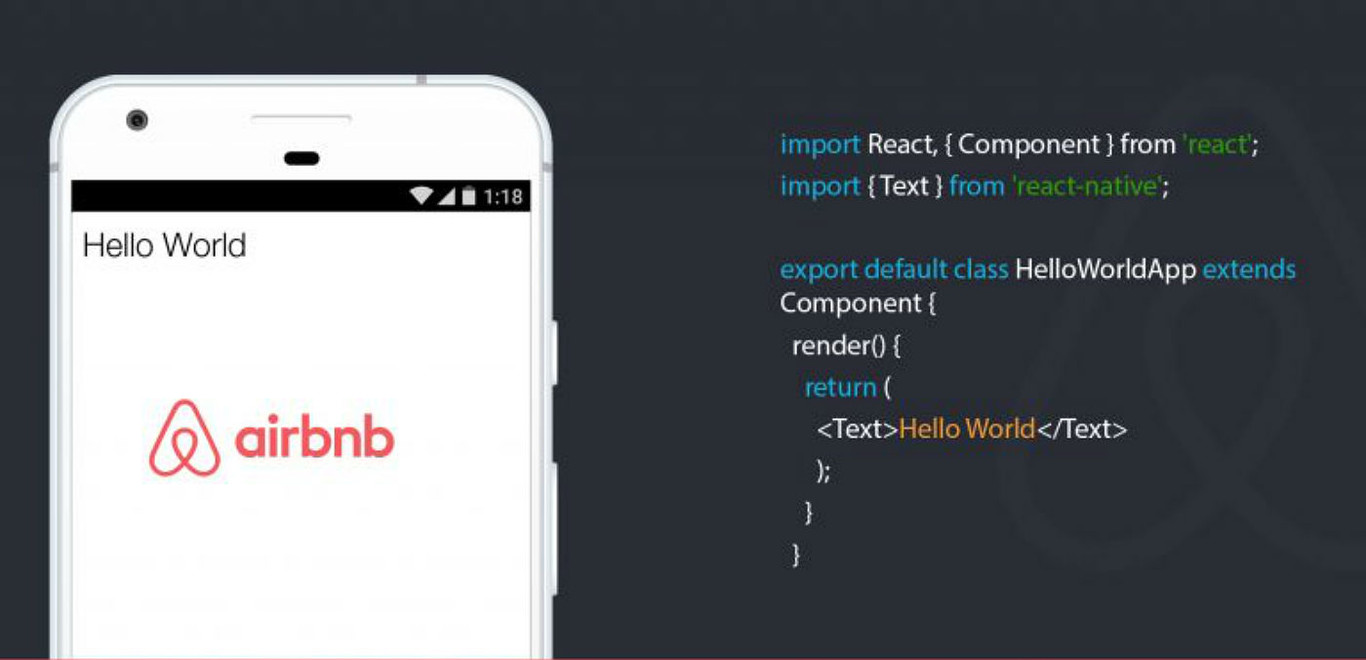
**Parte Teórica.**

1. Preguntas.

Responda de manera breve y clara posible.

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

Se trata de un elemento software visual que tiene su propio estado, recibe unas propiedades e implementa su propia lógica de renderizado.



* **Explique cómo se realiza la navegación entre Screens en React Native.**

La navegación se realiza con react-navigation y react-stack creando un archivo que llama a react-navigation y se crean las pantallas que navegarán dentro de un archivo de navegación.

Para aplicar la navegación a un screen se debe poner en las propiedades {navigation} y si se necesita llamar a otro screen solo con navigation.navigate se puede llamar, el screen al que se llama debe tener {navigation}.

* **Que significa IaaS, PaaS y SaaS.**

Distintas formas que puede adoptar la nube

*I****nfraestructure-as-a-Service (IaaS)***

Es el punto donde los desarrolladores empezamos a tocar y desarrollar nuestras propias aplicaciones que se ejecutan en la nube, en este caso con **IaaS** tendremos mucho más control que con PaaS, aunque a cambio de eso tendremos que encargarnos de la gestión de infraestructura.

La principal diferencia es que nosotros nos encargamos de escalar nuestras aplicaciones según nuestras necesidades, además de preparar todo el entorno en las maquinas (aunque existen imágenes de instancias preparadas con las configuraciones más comunes).

***Software-as-a-Service (SaaS)***

Servicio basado en la web como el Webmail de Gmail, los CRM onlines.

En este tipo de servicios nosotros accedemos normalmente a través del navegador sin atender al software. Todo el desarrollo, mantenimiento, actualizaciones, copias de seguridad es responsabilidad del proveedor.

En este caso tenemos poco control, nosotros nos situamos en la parte más arriba de la capa del servicio. Si el servicio se cae es responsabilidad de proveedor hacer que vuelva a funcionar.

***Platform-as-a-Service (PaaS)***

**PaaS** es el punto donde los desarrolladores empezamos a tocar y desarrollar nuestras propias aplicaciones que se ejecutan en la nube. En este caso nuestra única preocupación es la construcción de nuestra aplicación, ya que la infraestructura nos la da la plataforma.

Las soluciones PaaS gestionan automáticamente la escalabilidad usando más recursos si fuera necesario. Los desarrolladores aun así tienen que preocuparse de que sus aplicaciones estén lo mejor optimizadas posibles para consumir menos recursos posibles (número de peticiones, escrituras en disco, espacio requerido, tiempo de proceso, etc..) Pero todo ello sin entrar al nivel de maquinas.

Ejemplos populares son Google App Engine que permite desarrollar aplicaciones en Java o Python desplegándolas en la infraestructura que provee Google, cosa que también hace Heroku con Rails y Django.

* **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***

Cloud Firestore es una base de datos flexible y escalable para la programación en servidores, dispositivos móviles y la Web desde Firebase y Google Cloud Platform. Al igual que Firebase Realtime Database, mantiene tus datos sincronizados entre apps cliente a través de agentes de escucha en tiempo real y ofrece asistencia sin conexión para dispositivos móviles y la Web, por lo que puedes compilar apps con capacidad de respuesta que funcionan sin importar la latencia de la red ni la conectividad a Internet. Cloud Firestore también ofrece una integración sin interrupciones con otros productos de Firebase y Google Cloud Platform, incluido Cloud Functions.

***Backend as a service (BaaS)***

es un modelo para proporcionar a los desarrolladores web y de aplicaciones móviles una forma de vincular estas aplicaciones al almacenamiento en nube (cloud storage), servicios analíticos y/o otras características tales como la gestión de usuarios, la posibilidad de enviar notificaciones push y la integración con servicios de redes sociales Estos servicios se prestan a través de la utilización de kits personalizados de desarrollo de software (SDK) y las interfaces de programación de aplicaciones (API). BaaS es un modelo relativamente reciente en la computación en la nube, donde la mayoría de empresas datan del 2011 o posterior.

* **Defina o explique si React es lo mismo que Reac Native. Si son distintos liste cuales son las diferencias.**

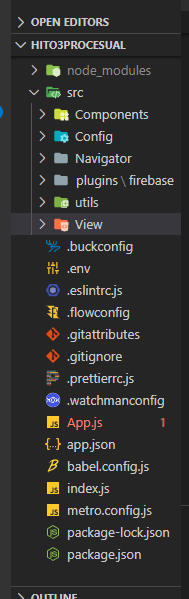
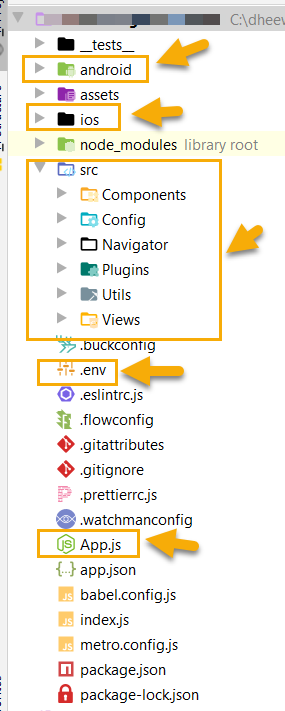
Reactjs en si es una biblioteca de js que nos brinda un entorno de programación por componentes, mientras React-native es un framework para desarrollo en dispositivos móviles.

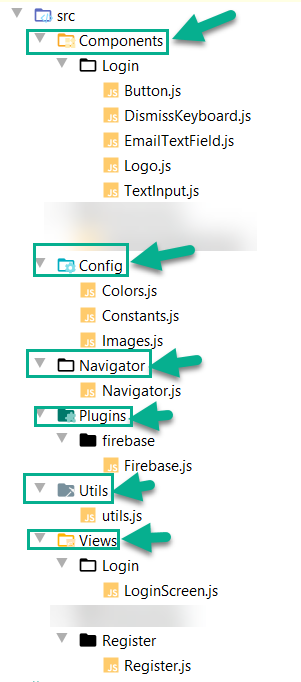
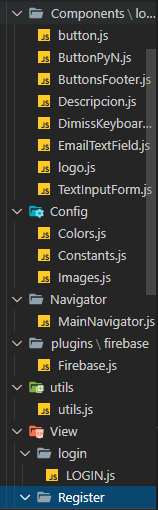
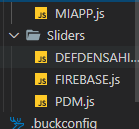
ReactJS está fuertemente basado en componentes. Estos componentes son los elementos que constituyen la interfaz del usuario (un botón, un buscador, etc).

React Native es un **mobile framework** **JavaScript** que usa los componentes de ReactJS para construir **aplicaciones “nativas”** para móviles.

**Parte Práctica.**

Debe de crear un proyecto nuevo desde cero.

* Integrar el Login del proyecto de clase a este nuevo proyecto.
  + Debe de crear un nuevo proyecto en firebase para poder integrar para la autenticación y manejo de base de datos
* Crear una estructura de proyecto similar a lo siguiente.
* 
* Después de tener esa estructura de proyecto el directorio SRC se sugiere administrar de la siguiente forma.

1. Enunciado.

Como objetivo se tiene registrar a nuevos usuarios a la plataforma Firebase utilizando la aplicación desarrolla en React Native.

Para este cometido se tiene un onboarding de 4 VISTAS incluyendo la página de login.

El diseño de la aplicación es la siguiente. (Disponible en: [APP](https://xd.adobe.com/view/46680033-8cec-40f2-48bd-5db3a2bd9a57-3059/))

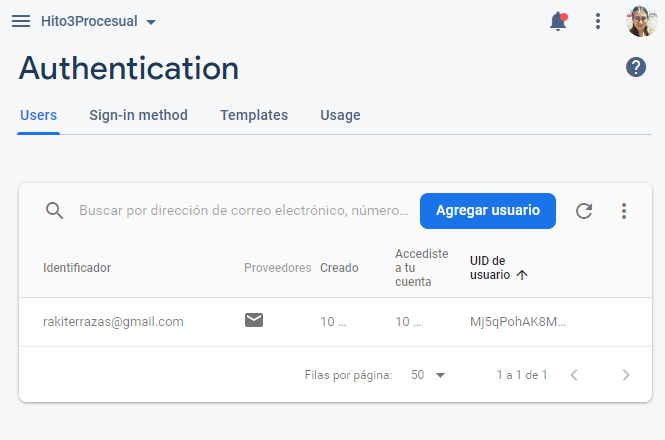
Colores APP: azul: '#428AF8', verde: '#4DCA77', naranja: '#F07243',

Assets APP: [Imagenes para(Screen1, Screen2 y Screen3)](https://drive.google.com/open?id=1QUu9u4zF1Qt7EGEyURLO8SJYNRDPVCce)

|  |
| --- |
|  |

De acuerdo al diseño anterior para dar solución se  sugiere resolver mínimamente los siguientes puntos.

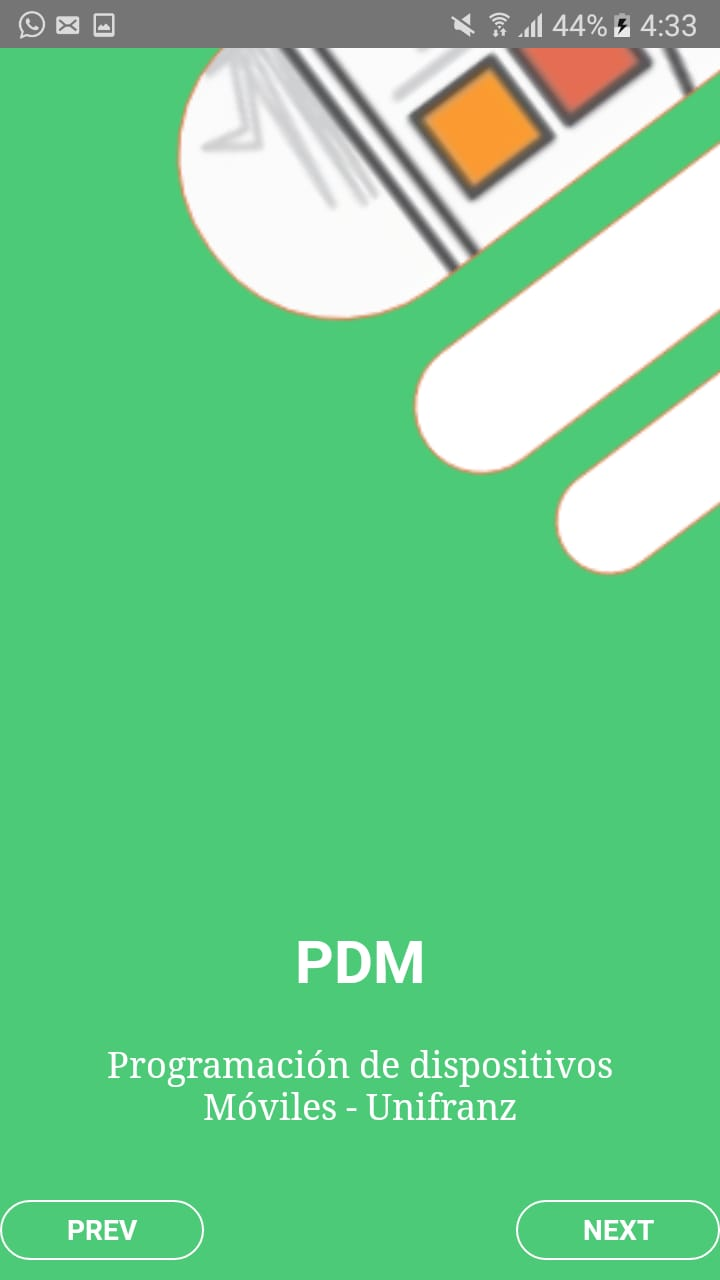
* Creación de un nuevo proyecto en FIREBASE.



* Configuración de React native con Firebase para la integración.

Primero se descarga con npm install @react-native-community/masked-view, react-navigation/native

* import firebase from 'firebase/app';
* import 'firebase/firestore';
* import 'firebase/auth';
* import 'firebase/storage';
* import{
* API\_KEY,
* AUTH\_DOMAIN,
* DATABASE\_URL,
* PROJECT\_ID,
* STORAGE\_BUCKET,
* MESSAGING\_SENDER\_ID,
* APP\_ID,
* } from 'react-native-dotenv';
* const firebaseConfig = {
* apiKey: API\_KEY,
* authDomain: AUTH\_DOMAIN,
* databaseURL: DATABASE\_URL,
* projectId: PROJECT\_ID,
* storageBucket: STORAGE\_BUCKET,
* messagingSenderId: MESSAGING\_SENDER\_ID,
* appId: APP\_ID,
* measurementId: "G-RT541JB9TN"
* };
* firebase.initializeApp(firebaseConfig);
* //export const firestore = firebase.firestore();
* export default firebase;
* Creación de el componente PDM (1ra  vista)



import React ,{ useState } from 'react';

import { StyleSheet, View, TextInput, Text, Image } from 'react-native';

import PropTypes from 'prop-types';

import ButtonLogin from '../../Components/login/button';

import Colors from '../../Config/Colors';

import Constants from '../../Config/Constants';

import Images from '../../Config/Images';

import ButtonFooter from '../../Components/login/ButtonsFooter';

import Descripcion from '../../Components/login/Descripcion';

const Pdm = ({ navigation }) => {

  const [isLoading, setIsLoading] = useState(false);

  const \_onPressP = () => {

  };

  const \_onPressN = () => {

    navigation.navigate('DefensaHito3');

  };

  return (

    <View style={styles.container}>

      <Descripcion

      source={Images.IMG1}

      style={styles.desc}

      textoTit={'PDM'}

      textoDesc={'Programación de dispositivos Móviles - Unifranz' }>

      </Descripcion>

      <ButtonFooter

        isLoading={isLoading}

        onPressP={\_onPressP}

        onPressN={\_onPressN}>

      </ButtonFooter>

    </View>

  );

};

const styles = StyleSheet.create({

  container: {

    flex: 3,

    paddingBottom:10,

    backgroundColor: Colors.green,

    justifyContent: 'space-between',

},

  text: {

    color: Colors.white,

    textAlign: 'center',

    fontWeight: 'bold',

    height: 20,

  },

  desc:{

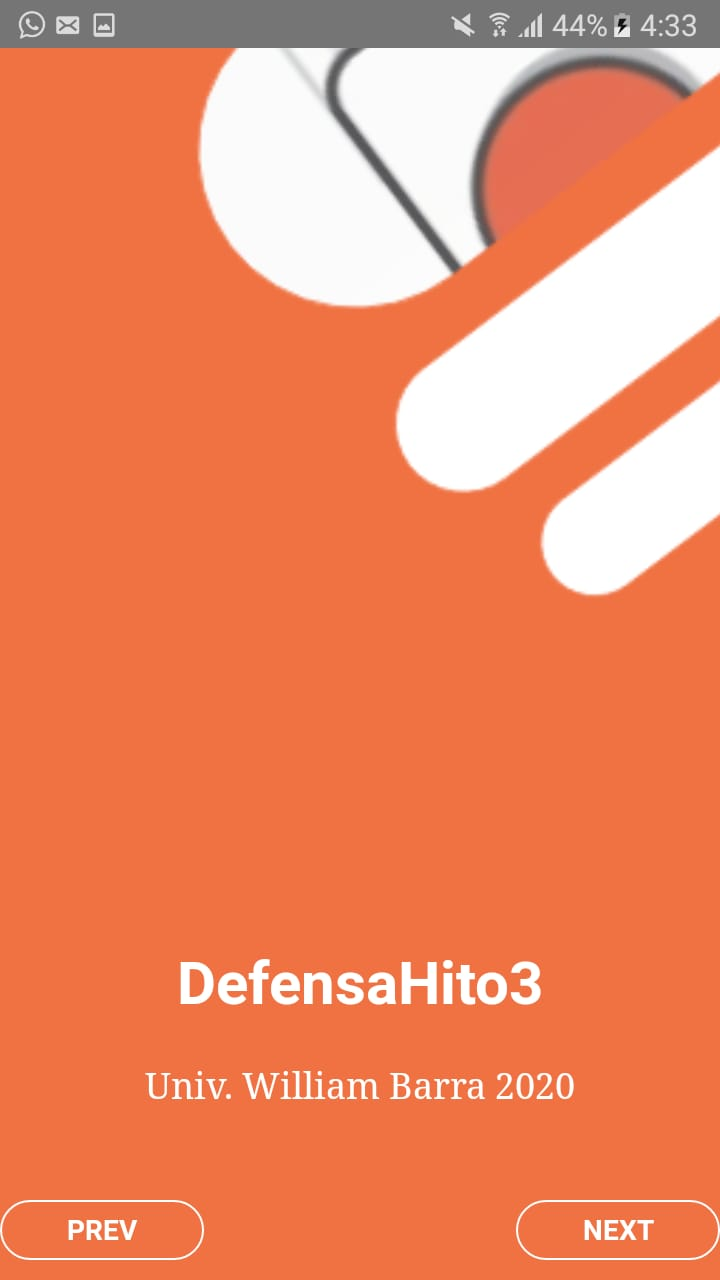
    textAlign:'center',

  }

});

export default Pdm;

* Creación del componente DEFENSAHITO3 (2da Vista)



import React ,{ useState } from 'react';

import { StyleSheet, View, TextInput, Text, Image } from 'react-native';

import PropTypes from 'prop-types';

import ButtonLogin from '../../Components/login/button';

import Colors from '../../Config/Colors';

import Constants from '../../Config/Constants';

import Images from '../../Config/Images';

import ButtonFooter from '../../Components/login/ButtonsFooter';

import Descripcion from '../../Components/login/Descripcion';

const DefensaHito3 = ({ navigation }) => {

  const [isLoading, setIsLoading] = useState(false);

  const \_onPressP = () => {

    navigation.navigate('Pdm');

  };

  const \_onPressN = () => {

    navigation.navigate('Firebase');

  };

  return (

    <View style={styles.container}>

      <Descripcion

      source={Images.IMG2}

      style={styles.desc}

      textoTit={'DefensaHito3'}

      textoDesc={'Univ. William Barra 2020' }>

      </Descripcion>

      <ButtonFooter

        isLoading={isLoading}

        onPressP={\_onPressP}

        onPressN={\_onPressN}>

      </ButtonFooter>

    </View>

  );

};

const styles = StyleSheet.create({

  container: {

    flex: 3,

    paddingBottom:10,

    backgroundColor: Colors.orange,

    justifyContent: 'space-between',

},

  text: {

    color: Colors.white,

    textAlign: 'center',

    fontWeight: 'bold',

    height: 20,

  },

  desc:{

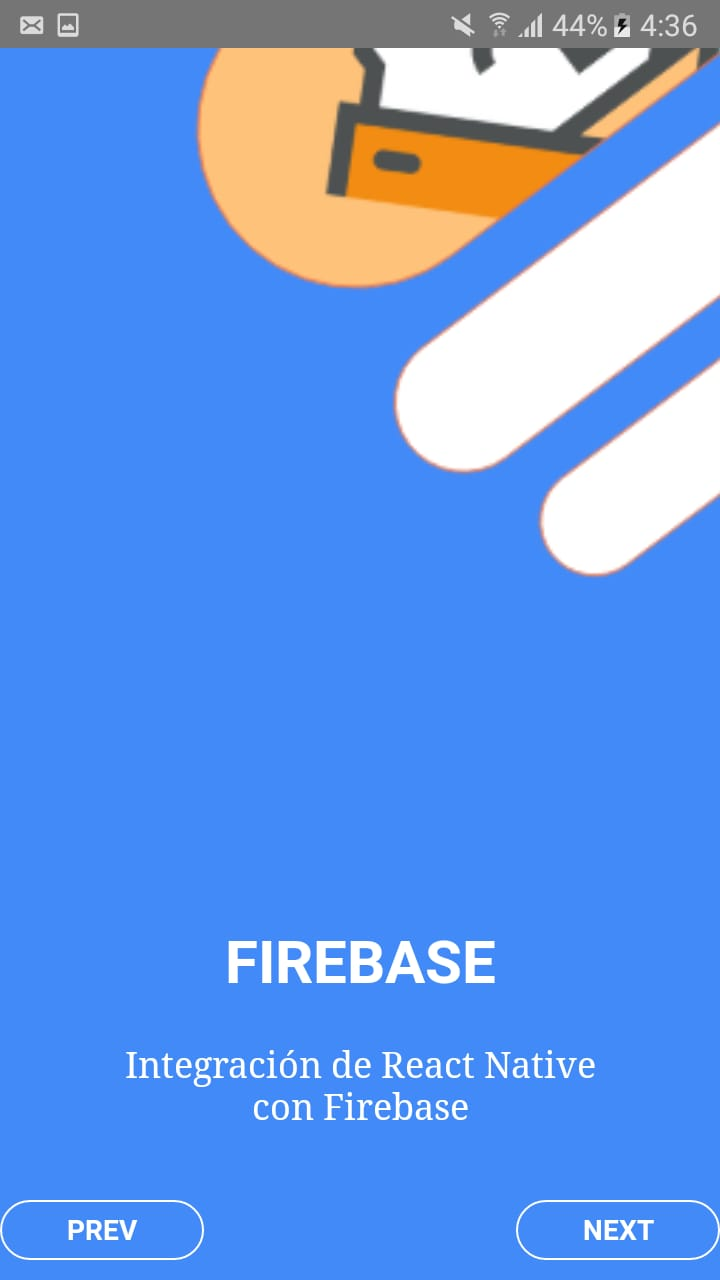
    textAlign:'center',

  }

});

export default DefensaHito3;

* Creación del componente FIREBASE (3ra Vista)



import React ,{ useState } from 'react';

import { StyleSheet, View, TextInput, Text, Image } from 'react-native';

import PropTypes from 'prop-types';

import ButtonLogin from '../../Components/login/button';

import Colors from '../../Config/Colors';

import Constants from '../../Config/Constants';

import Images from '../../Config/Images';

import ButtonFooter from '../../Components/login/ButtonsFooter';

import Descripcion from '../../Components/login/Descripcion';

const Firebase = ({ navigation }) => {

  const [isLoading, setIsLoading] = useState(false);

  const \_onPressP = () => {

    navigation.navigate('DefensaHito3');

  };

  const \_onPressN = () => {

    navigation.navigate('Login');

  };

  return (

    <View style={styles.container}>

      <Descripcion

      source={Images.IMG3}

      style={styles.desc}

      textoTit={'FIREBASE'}

      textoDesc={'Integración de React Native con Firebase' }>

      </Descripcion>

      <ButtonFooter

        isLoading={isLoading}

        onPressP={\_onPressP}

        onPressN={\_onPressN}>

      </ButtonFooter>

    </View>

  );

};

const styles = StyleSheet.create({

  container: {

    flex: 3,

    paddingBottom:10,

    backgroundColor: Colors.bluee,

    justifyContent: 'space-between',

},

  text: {

    color: Colors.white,

    textAlign: 'center',

    fontWeight: 'bold',

    height: 20,

  },

  desc:{

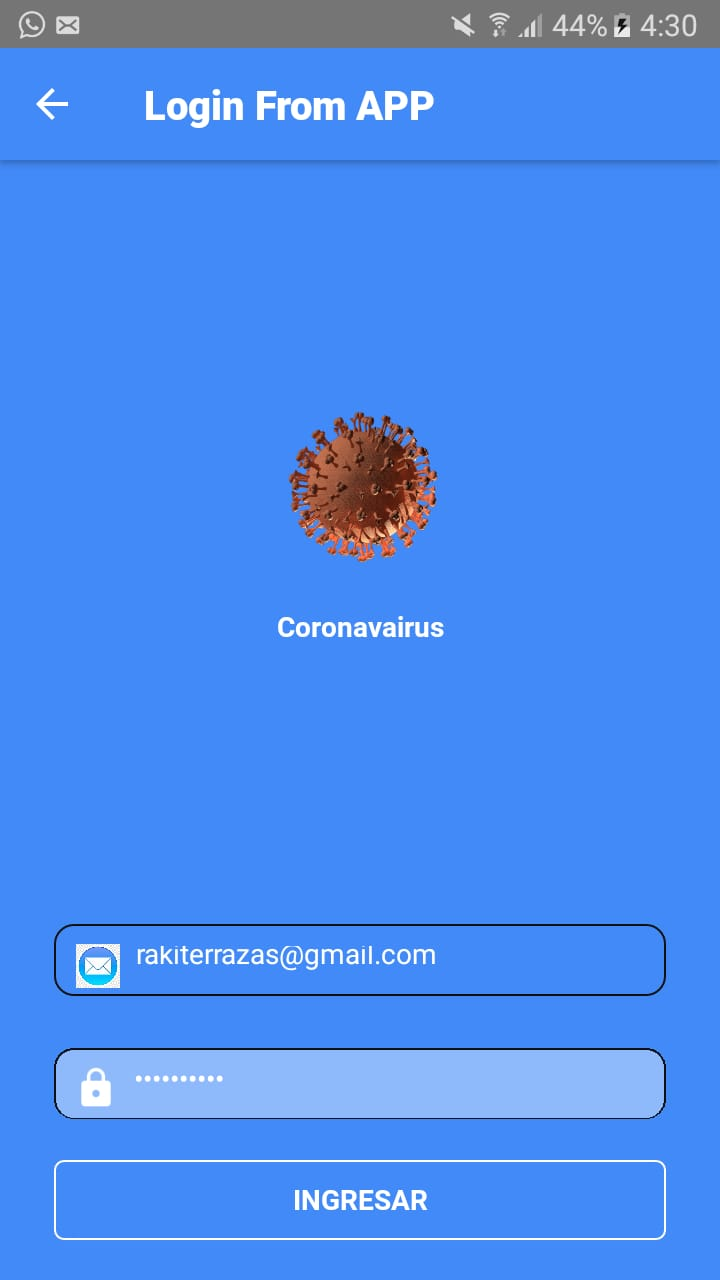
    textAlign:'center',

  }

});

export default Firebase;

* Creación del componente LOGIN (4ta vista)



import React, { useState } from 'react';

import PropTypes from 'prop-types';

import { StyleSheet, View, TouchableOpacity, Text, SafeAreaView, KeyboardAvoidingView, KeyboardAvoidingViewBase, Alert } from 'react-native';

import ButtonLogin from '../../Components/login/button';

import ButtonFooter from '../../Components/login/ButtonsFooter';

import TextInputLogin from '../../Components/login/TextInputForm';

import LogoLogin from '../../Components/login/logo';

import EmailTextField from '../../Components/login/EmailTextField';

import DimissKeyboard from '../../Components/login/DimissKeyboard';

import FirebasePlugin from '../../plugins/firebase/Firebase'

import Constants from '../../Config/Constants';

import Colors from '../../Config/Colors';

import Images from '../../Config/Images';

import imgUsername from '../../../assets/username.png';

import imgPassword from '../../../assets/pass.png';

import Utils from '../../utils/utils';

import { set } from 'react-native-reanimated';

const LOGIN = ({ navigation }) => {

    const [email, setEmail] = useState('');

    const [errorEmail, setErrorEmail] = useState('');

    const [password, setPassword] = useState('');

    const [errorPassword, setErrorPassword] = useState('');

    const [isLoading, setIsLoading] = useState(false);

    const \_validateEmailAddress = () => {

        let isvalidEmail = Utils.isValidEmail(email);

        isvalidEmail ? setErrorEmail('') : setErrorEmail(Constants.STRINGS.EMAIL\_ERROR);

        return isvalidEmail;

    }

    const \_validatePassword = () => {

        let isvalidPassword = Utils.isValidField(password);

        isvalidPassword ? setErrorPassword('') : setErrorPassword(Constants.STRINGS.PASSWORD\_ERROR);

        return isvalidPassword;

    }

    const \_onPress = () => {

        let emailData = \_validateEmailAddress();

        let passwordData = \_validatePassword();

        if (emailData && passwordData) {

            loginApp(email, password);

        } else {

            Alert.alert(Constants.STRINGS.EMPTY\_TITLE, Constants.STRINGS.EMPTY\_VALUES);

        }

    };

    const loginApp = (email, password) => {

        try {

            setIsLoading(true);

            FirebasePlugin.auth()

                .signInWithEmailAndPassword(email, password)

                .then(user => {

                    setIsLoading(false);

                    navigation.navigate('MIAPP');

                })

                .catch(error => {

                    FirebasePlugin.auth()

                        .createUserWithEmailAndPassword(email, password)

                        .then(user => {

                            setIsLoading(false);

                            navigation.navigate('MIAPP');

                        })

                        .catch(error => {

                            setIsLoading(false);

                            Alert.alert('Invalid Values', error.message);

                        });

                });

        } catch (error) {

            setIsLoading(true);

            Alert.alert('Firebase Error', error.message);

        }

    };

    return (

        <DimissKeyboard>

            <KeyboardAvoidingView style={stylesLoginScreen.container} behavior="height" enable>

                <View style={stylesLoginScreen.container}>

                    <LogoLogin style={stylesLoginScreen.logo} />

                    <View style={stylesLoginScreen.form}>

                        <EmailTextField

                            onChangeText={(email) => {

                                 setEmail(email) }}

                            onEndEditing={\_validateEmailAddress}

                            error={errorEmail}

                            source={Images.EMAIL}

                            placeholder={Constants.STRINGS.EMAIL}

                            securetextEntry={false}

                            autoCorrect={false}>

                        </EmailTextField>

                        <TextInputLogin

                            onChangeText={(password) => { setPassword(password) }}

                            source={imgPassword}

                            error={errorPassword}

                            onEndEditing={\_validatePassword}

                            placeholder={Constants.STRINGS.PASS}

                            secureTextEntry={true}

                            autoCorrect={false}>

                        </TextInputLogin>

                        <ButtonLogin

                            isLoading={isLoading}

                            onPress={\_onPress}

                            titleButton={Constants.STRINGS.TITLE\_BUTTON}>

                        </ButtonLogin>

                    </View>

                </View>

            </KeyboardAvoidingView>

        </DimissKeyboard>

    );

}

const stylesLoginScreen = StyleSheet.create({

    container: {

      flex: 1,

      backgroundColor: Colors.blue,

      justifyContent: 'center',

      alignItems: 'center',

    },

    logo: {

      width: '100%',

      resizeMode: 'contain',

      alignSelf: 'center',

    },

    form: {

      justifyContent: 'center',

      width: '80%',

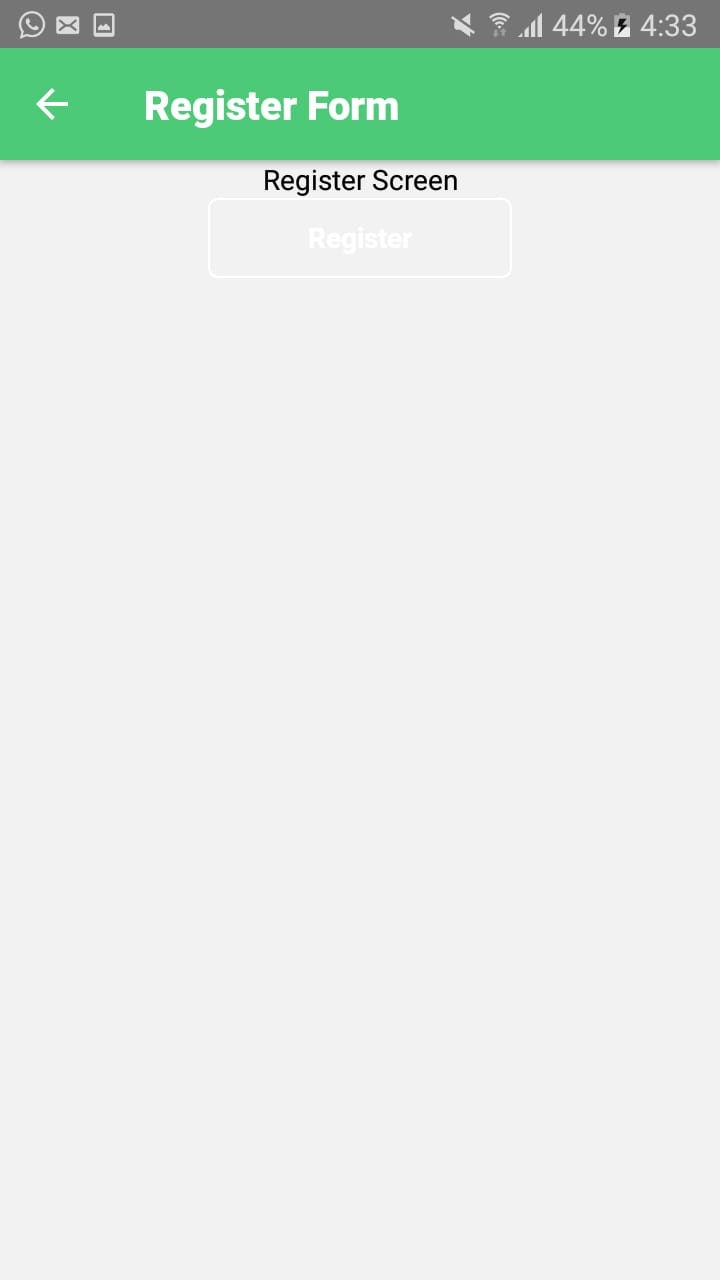
      marginBottom: 20,

    },

  });

export default LOGIN;

* Creacion del componente MIAPP (Vista que se mostrara despues de hacer LOGIN).



import React from 'react';

import {StyleSheet, View, TextInput,Text, Image} from 'react-native';

import PropTypes from 'prop-types';

import ButtonLogin from'../../Components/login/button';

import Colors from '../../Config/Colors';

import Constants from'../../Config/Constants';

import Images from'../../Config/Images';

const MIAPP = ({navigation}) => {

    const onPress = () => {

      console.log('register');

    };

    return (

      <View style={styles.container}>

        <Text>Register Screen</Text>

        <ButtonLogin onPress={onPress} titleButton={Constants.STRINGS.REGISTER} />

      </View>

    );

  };

const styles = StyleSheet.create({

    container: {

      flex: 1,

      backgroundColor: 'transparent',

      alignItems: 'center',

    },

    text: {

      color: Colors.white,

      textAlign: 'center',

      fontWeight: 'bold',

      height: 20,

    },

  });

export default MIAPP;

Sin embargo es muy probable que necesite crear componentes adicionales:

* Button,

import React from 'react';

import {

  TouchableOpacity,

  StyleSheet,

  Text,

  View,

  ActivityIndicator,

} from 'react-native';

import Colors from '../../Config/Colors';

const Button = ({titleButton, onPress, isLoading}) => {

  const loader = () => {

    return <ActivityIndicator animating={isLoading} />;

  };

  const button = () => {

    return (

      <TouchableOpacity onPress={onPress}>

        <Text style={styles.text}>{titleButton}</Text>

      </TouchableOpacity>

    );

  };

  return <View style={styles.button}>{isLoading ? loader() : button()}</View>;

};

const styles = StyleSheet.create({

  button: {

    display: 'flex',

    height: 40,

    borderRadius: 5,

    justifyContent: 'center',

    alignItems: 'center',

    backgroundColor: 'transparent',

    shadowColor: Colors.blue,

    shadowOpacity: 0.4,

    shadowOffset: {height: 10, width: 10},

    shadowRadius: 20,

    borderColor: Colors.white,

    borderWidth: 1,

  },

  text: {

    width: 150,

    color: Colors.white,

    textAlign: 'center',

    fontWeight: 'bold',

    height: 20,

  },

});

export default Button;

import React from 'react';

import {

  TouchableOpacity,

  StyleSheet,

  Text,

  View,

  ActivityIndicator,

} from 'react-native';

import Colors from '../../Config/Colors';

const Button = ({titleButton, onPress, isLoading}) => {

  const loader = () => {

    return <ActivityIndicator animating={isLoading} />;

  };

  const button = () => {

    return (

      <TouchableOpacity onPress={onPress}>

        <Text style={styles.text}>{titleButton}</Text>

      </TouchableOpacity>

    );

  };

  return <View style={styles.button}>{isLoading ? loader() : button()}</View>;

};

const styles = StyleSheet.create({

  button: {

    display: 'flex',

    height: 30,

    borderRadius: 20,

    justifyContent: 'center',

    alignItems: 'center',

    backgroundColor: 'transparent',

    shadowColor: Colors.blue,

    shadowOpacity: 0.4,

    shadowOffset: {height: 10, width: 10},

    shadowRadius: 20,

    borderColor: Colors.white,

    borderWidth: 1,

  },

  text: {

    width: 100,

    color: Colors.white,

    textAlign: 'center',

    fontWeight: 'bold',

    height: 20,

  },

});

export default Button;

import React, { Component } from 'react';

import { StyleSheet, View, TextInput, Text, Image } from 'react-native';

import PropTypes from 'prop-types';

import Colors from '../../Config/Colors';

import Button from '../../Components/login/ButtonPyN'

import Constants from '../../Config/Constants';

const ButtonsFooter = ({  onPressP,onPressN, isLoading, navigation }) => {

    return (

        <View style={styleButtonsFooter.container}>

            <Button

                isLoading={isLoading}

                onPress={onPressP}

                titleButton={Constants.STRINGS.PREV}>

            </Button>

            <Button

                isLoading={isLoading}

                onPress={onPressN}

                titleButton={Constants.STRINGS.NEXT}>

            </Button>

        </View>

    );

}

* TextField,

import React, {Component} from 'react';

import {StyleSheet, View, Text, TextInput, Image} from 'react-native';

import PropTypes from 'prop-types';

import Colors from '../../Config/Colors';

import Constants from '../../Config/Constants';

export default class TextInputForm extends Component {

  constructor(props) {

    super(props);

  }

  render() {

    return (

      <View>

        <Text style={stylesTextInput.errorText}>{this.props.error}</Text>

        <View>

          <Image style={stylesTextInput.inlineImg} source={this.props.source} />

          <TextInput

            style={stylesTextInput.textInput}

            onChangeText={this.props.onChangeText}

            onEndEditing={this.props.onEndEditing}

            placeholder={this.props.placeholder}

            secureTextEntry={this.props.secureTextEntry}

            autoCorrect={this.props.autoCorrect}

            selectionColor={Colors.blue}

            placeholderTextColor={Colors.white}

            underlineColorAndroid="transparent"

          />

        </View>

      </View>

    );

  }

}

TextInputForm.propTypes = {

  source: PropTypes.number.isRequired,

  placeholder: PropTypes.string.isRequired,

  autoCorrect: PropTypes.bool,

  secureTextEntry: PropTypes.bool,

};

const stylesTextInput = StyleSheet.create({

  textInput: {

    height: Constants.CONFIG.HEADER\_HEIGHT \* 0.06,

    width: Constants.CONFIG.SCREEN\_WIDTH \* 0.85,

    backgroundColor: 'rgba(255, 255, 255, 0.4)',

    alignItems: 'center',

    color: Colors.white,

    paddingLeft: 40,

    borderRadius: 10,

    borderWidth: 1,

    borderColor: Colors.black2,

    borderBottomWidth: StyleSheet.hairlineWidth,

    marginBottom: 20,

  },

  inlineImg: {

    position: 'absolute',

    zIndex: 99,

    width: 22,

    height: 22,

    left: 10,

    top: 9,

  },

  errorText: {

    fontSize: 12,

    color: Colors.red,

    marginHorizontal: 20,

  },

});

* EmailField,

import React from 'react';

import {StyleSheet, View, TouchableOpacity,Text, TextInput,Image} from 'react-native';

import Colors from '../../Config/Colors';

import Constants from'../../Config/Constants';

import Images from'../../Config/Images';

const EmailTextField = ({

  onChangeText,

  onEndEditing,

  error,

  source,

  placeholder,

  secureTextEntry,

  autoCorrect,

}) => {

  return (

    <View>

      <Text style={styles.errorText}>{error}</Text>

      <View style={styles.textFieldView}>

        <Image style={styles.inlineImg} source={source} />

        <TextInput

          style={styles.textField}

          onChangeText={onChangeText}

          onEndEditing={onEndEditing}

          placeholder={placeholder}

          secureTextEntry={secureTextEntry}

          autoCorrect={autoCorrect}

          selectionColor={Colors.blue}

          placeholderTextColor={Colors.white}

          underlineColorAndroid="transparent"

        />

      </View>

    </View>

  );

};

const styles = StyleSheet.create({

  textField: {

    fontSize: 14,

    flex: 1,

    paddingLeft: 20,

    marginHorizontal: 20,

    color: Colors.white,

  },

  textFieldView: {

    height: Constants.CONFIG.HEADER\_HEIGHT \* 0.06,

    width: Constants.CONFIG.SCREEN\_WIDTH \* 0.85,

    borderRadius: 10,

    marginTop: 5,

    marginBottom: 10,

    borderColor: Colors.black2,

    borderWidth: 1,

    justifyContent: 'center',

    backgroundColor: Colors.fieldBackgroud,

  },

  inlineImg: {

    position: 'absolute',

    zIndex: 99,

    width: 22,

    height: 22,

    left: 10,

    top: 9,

  },

  errorText: {

    fontSize: 12,

    color: Colors.red,

    marginBottom: -5,

    marginHorizontal: 20,

  },

});

export default EmailTextField;

* Logo,

import React, {Component} from 'react'

import {

  StyleSheet,

  View,

  Text,

  Image

} from 'react-native';

import Colors from '../../Config/Colors';

import Constants from'../../Config/Constants';

import Images from'../../Config/Images';

export default class logo extends Component{

    constructor(props){

        super(props);

    }

    render(){

        return(

            <>

                <View style={styles.container}>

                    <Image source={Images.LOGO} style={styles.image}/>

                    <Text style={styles.text}>

                        Coronavairus

                    </Text>

                </View>

            </>

        );

    }

}

const styles = StyleSheet.create({

    container:{

        flex:1,

        alignItems:'center',

        justifyContent:'center',

    },

    image:{

        width:80,

        height:80,

    },

    text:{

        color:'white',

        fontWeight:'bold',

        backgroundColor:'transparent',

        marginTop:20,

    }

});

* Image,

export default{

    LOGO: require('../../assets/coronavirus.png'),

    PASSWORD: require('../../assets/pass.png'),

    USERNAME: require('../../assets/username.png'),

    EMAIL: require('../../assets/email.png'),

    IMG1: require('../../assets/img1.png'),

    IMG2: require('../../assets/img2.png'),

    IMG3: require('../../assets/img3.png'),

}

* Descripción

import React, { Component } from 'react';

import { StyleSheet, View, TextInput, Text, Image, TextBase } from 'react-native';

import PropTypes from 'prop-types';

import Colors from '../../Config/Colors';

import Button from '../../Components/login/ButtonPyN'

import Constants from '../../Config/Constants';

import Images from '../../Config/Images';

const Descripcion = ({ source, textoDesc, textoTit,navigation }) => {

    return (

        <View style={styleDescripcion.container}>

            <Image  style={styleDescripcion.image} source={source} />

            <View >

                <Text style={styleDescripcion.text1}>

                    {textoTit}

                </Text>

                <Text style={styleDescripcion.text}

                multiline={true}>

                    {textoDesc}

                </Text>

            </View>

        </View>

    );

}

No se tiene restricciones para poder resolver el problema puede crear los componentes sugeridos u otros en su caso.