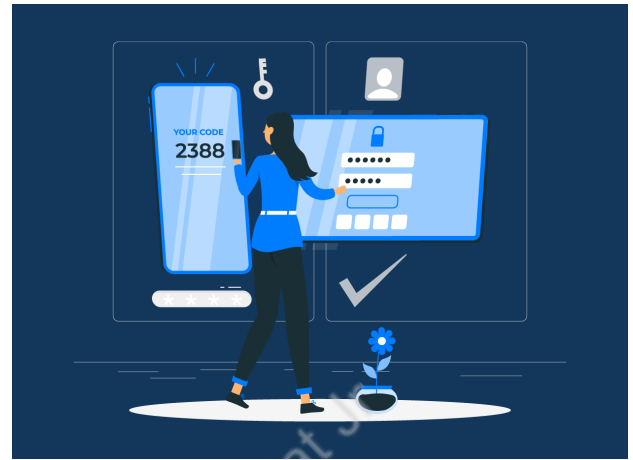


## AUTHENTICATION AND DB INTEGRATION



### What is our GOAL for this MODULE?

In this class, we continued to build the Storytelling application. We learned to implement the login feature and integrate the App with the Firebase database using Firebase Authentication.

### What did we ACHIEVE in the class TODAY?

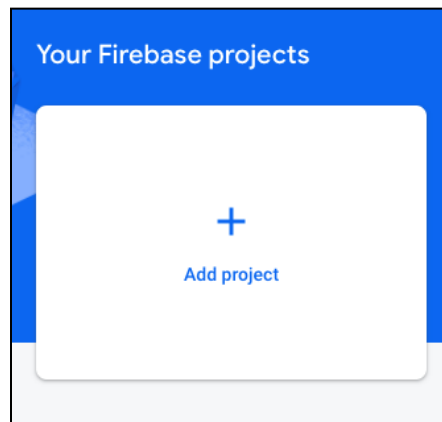
- Designed stack navigation to navigate from Login screen to the Feed screen.
- Implemented Login for the App by setting up the credentials in our Firebase App on the Firebase console.
- Integrated the App with Firebase using Firebase services for login authentication.

### Which CONCEPTS/ CODING BLOCKS did we cover today?

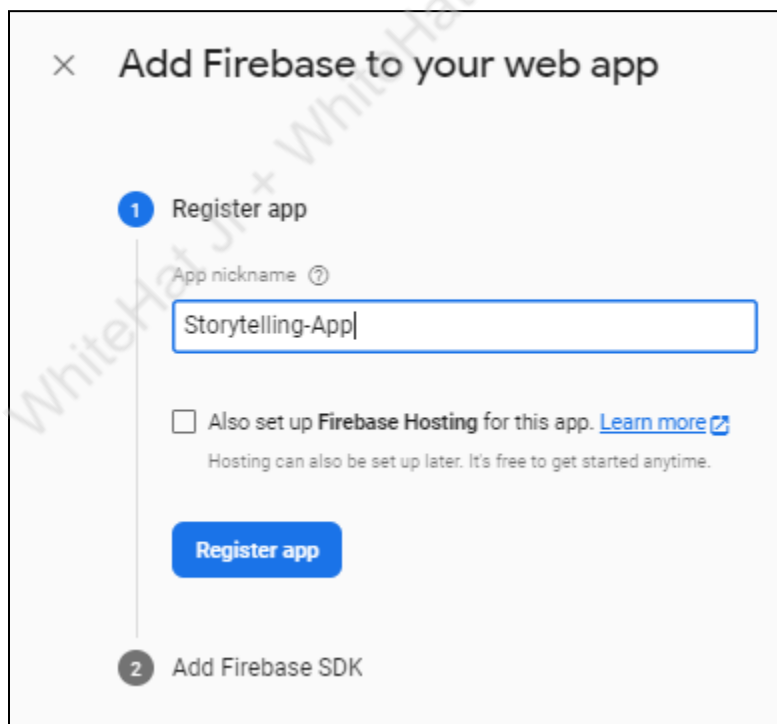
- Creating Login and Registration Screens
- Writing functions to implement authenticated login
- Writing functions to register a new user.

### How did we DO the activities?

1. Go to the [firebase console](#) and set up a new firebase database.
  - Click on **Add Project** and add a name for your project.



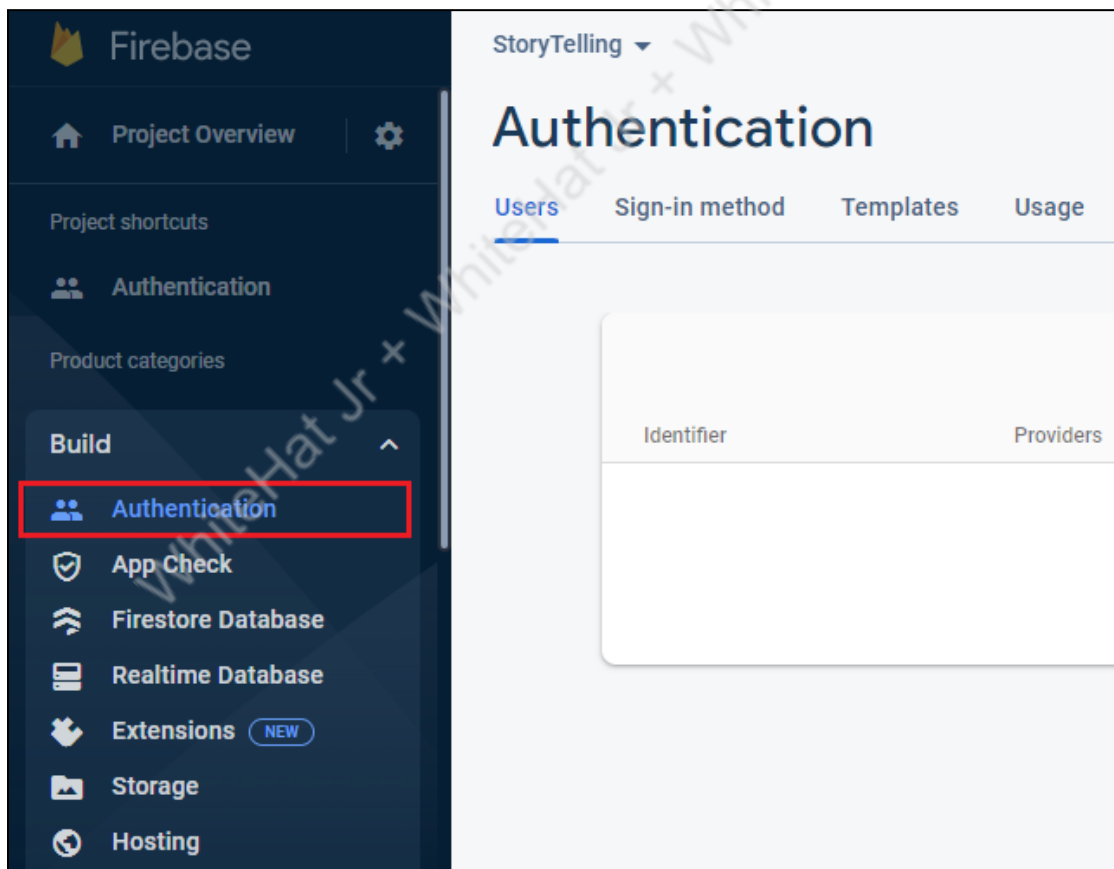
- Once the project is created, click on the web button, register our app by giving it a name and copy the config keys from there -



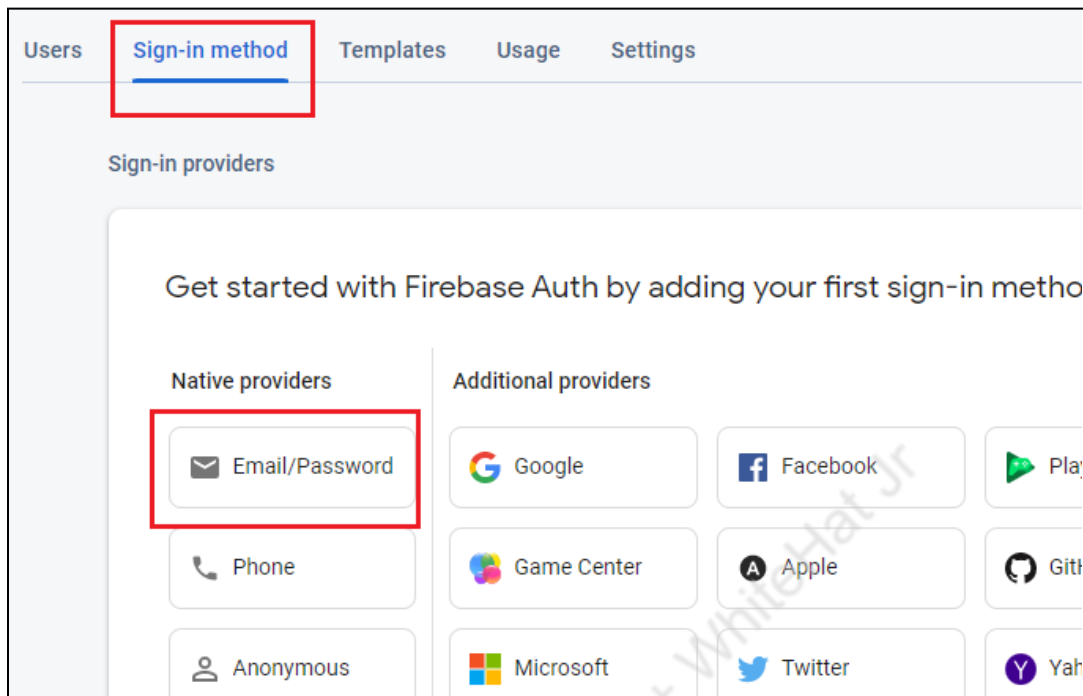
- Copy the SDK code after you have clicked the **Register app** button.
- To save these config keys, let's create a new file **config.js** in our root folder of the project.

```
export const firebaseConfig = {
  apiKey: [REDACTED],
  authDomain: "storytelling-app-cab54.firebaseio.com",
  projectId: "storytelling-app-cab54",
  storageBucket: "storytelling-app-cab54.appspot.com",
  messagingSenderId: "843153669971",
  appId: "1:843153669971:web:05101931886d9498266ba6"
};
```

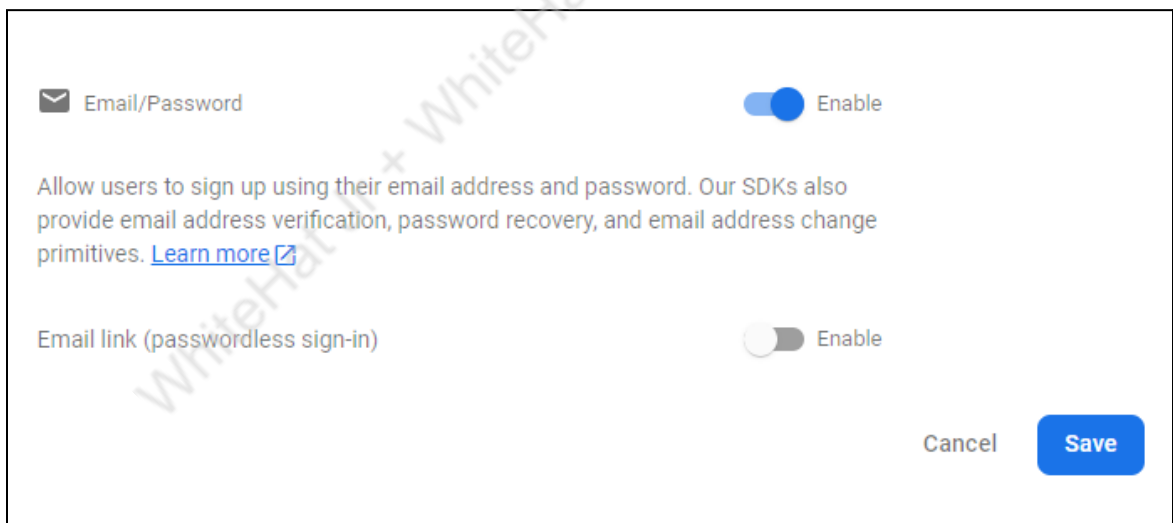
2. Enable Email / Password authentication in firebase-
  - Go to the **StoryTelling** project that was just created in the [firebase console](#) → go to the left hand side of the screen and expand the **build** option → Click on **Authentication**.



- Now, click on the **Sign-in method** tab → select **Email/Password**.

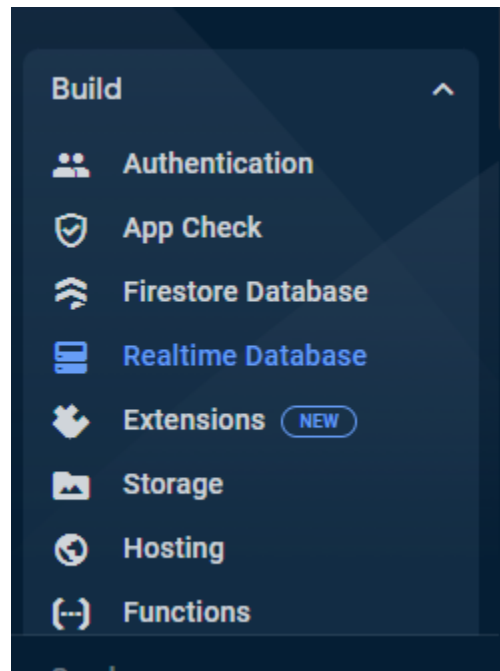


- Enable the Email/Password option and click on **Save**.

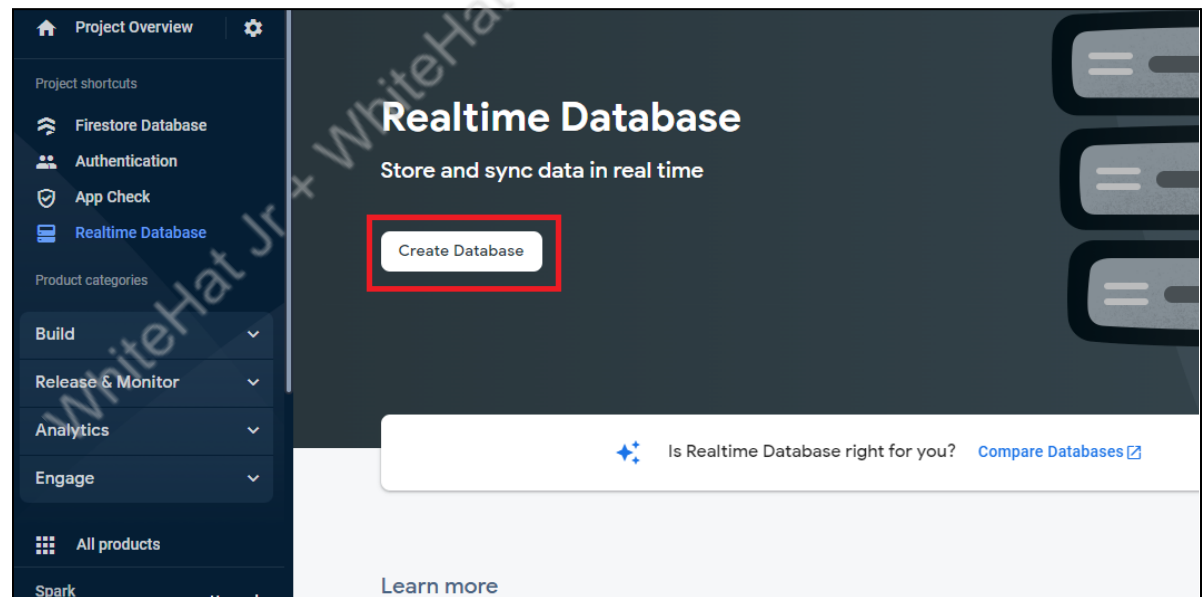


### 3. Create a **Realtime Database**.

- go to the left hand side of the screen and expand the **build** option → Click on the **Realtime Database** option.



- Click on the **Create Database** button.



4. Write code to create a Stack Navigator using **createStackNavigator()** function which would contain:
  - **Login Screen**
  - **Registration Screen**
  - **DrawerNavigator** (which was created earlier)

Also, import **firebase** and **firebaseConfig** into **App.js** and write an **if** statement to initialize firebase.

This is how App.js should look like-

```
import * as React from "react";
import { NavigationContainer } from '@react-navigation/native';

import { createStackNavigator } from "@react-navigation/stack";
import LoginScreen from "./screens/LoginScreen";
import RegisterScreen from "./screens/Register";

import DrawerNavigator from "./navigation/DrawerNavigator";

import * as firebase from "firebase";
import { firebaseConfig } from "./config";

if (!firebase.apps.length) {
  firebase.initializeApp(firebaseConfig);
} else {
  firebase.app();
}

const Stack = createStackNavigator();

const StackNav = () => {
  return(
    <Stack.Navigator initialRouteName="Login" screenOptions={{
      headerShown: false,
      gestureEnabled: false
    }}>
      <Stack.Screen name="Login" component={LoginScreen} />
      <Stack.Screen name="RegisterScreen" component={RegisterScreen} />
      <Stack.Screen name="Dashboard" component={DrawerNavigator} />
    </Stack.Navigator>
  )
}
```

```
    </Stack.Navigator>
  }

  export default function App() {
    return (
      <NavigationContainer>
        <StackNav/>
      </NavigationContainer>
    )
  }
```

5. Write code to add the login feature in the **LoginScreen.js**. Use the **firebase.auth().signInWithEmailAndPassword()** method for this.

```
import React, { Component } from "react";
import {
  View,
  StyleSheet,
  SafeAreaView,
  Platform,
  StatusBar,
  Image,
  TextInput,
  Alert,
  TouchableOpacity,
  Text
} from "react-native";

import firebase from "firebase";
import { RFValue } from "react-native-responsive-fontsize";
import * as Font from "expo-font";

import * as SplashScreen from 'expo-splash-screen';
SplashScreen.preventAutoHideAsync();
```

```
let customFonts = {
  "Bubblegum-Sans": require("../assets/fonts/BubblegumSans-Regular.ttf"),
};

const applcon = require("../assets/logo.png");

export default class LoginScreen extends Component {
  constructor(props) {
    super(props);
    this.state = {
      email: "",
      password: "",
      fontsLoaded: false,
      userSignedIn: false
    };
  }
  async _loadFontsAsync() {
    await Font.loadAsync(customFonts);
    this.setState({ fontsLoaded: true });
  }
  componentDidMount() {
    this._loadFontsAsync();
  }

  signIn = async (email, password) => {
    firebase
      .auth()
      .signInWithEmailAndPassword(email, password)
      .then(() => {
        this.props.navigation.replace("Dashboard");
      })
      .catch(error => {
```



```
Alert.alert(error.message);
});
};

render() {
  if (this.state.fontsLoaded) {
    SplashScreen.hideAsync();
    const { email, password } = this.state;

    return (
      <View style={styles.container}>
        <SafeAreaView style={styles.droidSafeArea} />

        <Text style={styles.appTitleText}>Story Telling</Text>
        <Image source={applcon} style={styles.applcon} />

        <TextInput
          style={styles.textinput}
          onChangeText={text => this.setState({ email: text })}
          placeholder={"Enter Email"}
          placeholderTextColor={"#FFFFFF"}
          autoFocus
        />
        <TextInput
          style={[styles.textinput, { marginTop: 20 }]}
          onChangeText={text => this.setState({ password: text })}
          placeholder={"Enter Password"}
          placeholderTextColor={"#FFFFFF"}
          secureTextEntry
        />
        <TouchableOpacity
          style={[styles.button, { marginTop: 20 }]}
          onPress={() => this.signIn(email, password)}
        />
      </View>
    );
  }
}
```

```

    >
    <Text style={styles.buttonText}>Login</Text>
  </TouchableOpacity>
  <TouchableOpacity
    onPress={() => this.props.navigation.navigate("RegisterScreen")}
  >
    <Text style={styles.buttonTextNewUser}>New User ?</Text>
  </TouchableOpacity>
</View>
);
marginTop: Platform.OS === "android" ? StatusBar.currentHeight : RFValue(35)
},
apIcon: {
  width: RFValue(200),
  height: RFValue(200),
  resizeMode: "contain",
  marginBottom: RFValue(20)
},
appTitleText: {
  color: "white",
  textAlign: "center",
  fontSize: RFValue(40),
  fontFamily: "Bubblegum-Sans",
  marginBottom: RFValue(20)
},
textInput: {
  width: RFValue(250),
  height: RFValue(50),
  padding: RFValue(10),
  borderColor: "#FFFFFF",
  borderWidth: RFValue(4),
  borderRadius: RFValue(10),
  fontSize: RFValue(20),
  color: "#FFFFFF",

```

```
    backgroundColor: "#15193c",
    fontFamily: "Bubblegum-Sans"
  },
  button: {
    width: RFValue(250),
    height: RFValue(50),
    flexDirection: "row",
    justifyContent: "space-evenly",
    alignItems: "center",
    borderRadius: RFValue(30),
    backgroundColor: "white",
    marginBottom: RFValue(20)
  },
  buttonText: {
    fontSize: RFValue(24),
    color: "#15193c",
    fontFamily: "Bubblegum-Sans"
  },
  buttonTextNewUser: {
    fontSize: RFValue(12),
    color: "#FFFFFF",
    fontFamily: "Bubblegum-Sans",
    textDecorationLine: 'underline'
  }
}); }
}
}

const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: "#15193c",
    alignItems: "center",
    justifyContent: "center"
```

```
},  
droidSafeArea: {
```

6. Write code to add the login feature in the **RegisterScreen.js**. Use the **firebase.auth().createUserWithEmailAndPassword()** method for this.

```
import React, { Component } from "react";
```

```
import {
```

```
  View,
```

```
  StyleSheet,
```

```
  SafeAreaView,
```

```
  Platform,
```

```
  StatusBar,
```

```
  Image,
```

```
  TextInput,
```

```
  Alert,
```

```
  TouchableOpacity,
```

```
  Text
```

```
} from "react-native";
```

```
import firebase from "firebase";
```

```
import { RFValue } from "react-native-responsive-fontsize";

import * as Font from "expo-font";

import * as SplashScreen from 'expo-splash-screen';

SplashScreen.preventAutoHideAsync();

let customFonts = {

  "Bubblegum-Sans": require("../assets/fonts/BubblegumSans-Regular.ttf"),

};

const applcon = require("../assets/logo.png");

export default class RegisterScreen extends Component {

  constructor(props) {

    super(props);

    this.state = {

      first_name: "",

      last_name: "",

      email: "",

      password: "",

      confirmPassword: "",

      fontsLoaded: false
```

```
};  
  
}  
  
async _loadFontsAsync() {  
  
    await Font.loadAsync(customFonts);  
  
    this.setState({ fontsLoaded: true });  
  
}  
  
componentDidMount() {  
  
    this._loadFontsAsync();  
  
}  
  
registerUser = (email, password, confirmPassword, first_name, last_name) => {  
  
    if(password==confirmPassword){  
  
        firebase  
  
        .auth()  
  
        .createUserWithEmailAndPassword(email, password)  
  
        .then((userCredential) => {  
  
            Alert.alert("User registered!!!");  
  
            console.log(userCredential.user.uid)  
  
            this.props.navigation.replace("Login");  
  
            firebase.database().ref("/users/" + userCredential.user.uid)  
  
                .set({
```

```
        email: userCredential.user.email,

        first_name: first_name,

        last_name: last_name,

        current_theme: "dark"

    })

  })

  .catch(error => {

    Alert.alert(error.message);

  });

} else {

  Alert.alert("Passwords don't match!");

}

};

render() {

  if (this.state.fontsLoaded) {

    SplashScreen.hideAsync();

    const { email, password, confirmPassword, first_name, last_name } = this.state;

    return (

      <View style={styles.container}>
```

```
<SafeAreaView style={styles.droidSafeArea} />

<Text style={styles.appTitleText}>Register</Text>

<TextInput

  style={styles.textinput}

  onChangeText={text => this.setState({ first_name: text })}

  placeholder={"First name"}

  placeholderTextColor={"#FFFFFF"}

/>

<TextInput

  style={styles.textinput}

  onChangeText={text => this.setState({ last_name: text })}

  placeholder={"Last name"}

  placeholderTextColor={"#FFFFFF"}

/>

<TextInput

  style={styles.textinput}

  onChangeText={text => this.setState({ email: text })}

  placeholder={"Enter Email"}
```



```
placeholderTextColor={"#FFFFFF"}

/>

<TextInput

  style={styles.textinput}

  onChangeText={text => this.setState({ password: text })}

  placeholder={"Enter Password"}

  placeholderTextColor={"#FFFFFF"}

  secureTextEntry

/>

<TextInput

  style={styles.textinput}

  onChangeText={text => this.setState({ confirmPassword: text })}

  placeholder={"Re-enter Password"}

  placeholderTextColor={"#FFFFFF"}

  secureTextEntry

/>

<TouchableOpacity

  style={[styles.button, { marginTop: 20 }]}

  onPress={() => this.registerUser(email, password,
confirmPassword,first_name,last_name)}

  >
```

```

    <Text style={styles.buttonText}>Register</Text>

    </TouchableOpacity>

    <TouchableOpacity

      onPress={() => this.props.navigation.replace("Login")}

    >

    <Text style={styles.buttonTextNewUser}>Login ?</Text>

    </TouchableOpacity>

  </View>

);

}

}

}

const styles = StyleSheet.create({

  container: {

    flex: 1,

    backgroundColor: "#15193c",

    alignItems: "center",

    justifyContent: "center"

  },

  droidSafeArea: {

    marginTop: Platform.OS === "android" ? StatusBar.currentHeight : RFValue(35)
  }
});

```

```
},  
  
applcon: {  
  
    width: RFValue(200),  
  
    height: RFValue(200),  
  
    resizeMode: "contain",  
  
    marginBottom:RFValue(20)  
  
},  
  
appTitleText: {  
  
    color: "white",  
  
    textAlign: "center",  
  
    fontSize: RFValue(40),  
  
    fontFamily: "Bubblegum-Sans",  
  
    marginBottom:RFValue(20)  
  
},  
  
textinput: {  
  
    width: RFValue(250),  
  
    height: RFValue(40),  
  
    padding: RFValue(10),  
  
    marginTop:RFValue(10),  
  
    borderColor: "#FFFFFF",  
  
    borderWidth: RFValue(4),  
  
    borderRadius: RFValue(10),
```

```
fontSize: RFValue(15),

color: "#FFFFFF",

backgroundColor: "#15193c",

fontFamily: "Bubblegum-Sans"

},

button: {

width: RFValue(250),

height: RFValue(50),

flexDirection: "row",

justifyContent: "space-evenly",

alignItems: "center",

borderRadius: RFValue(30),

backgroundColor: "white",

marginBottom: RFValue(20)

},

buttonText: {

fontSize: RFValue(24),

color: "#15193c",

fontFamily: "Bubblegum-Sans"

},

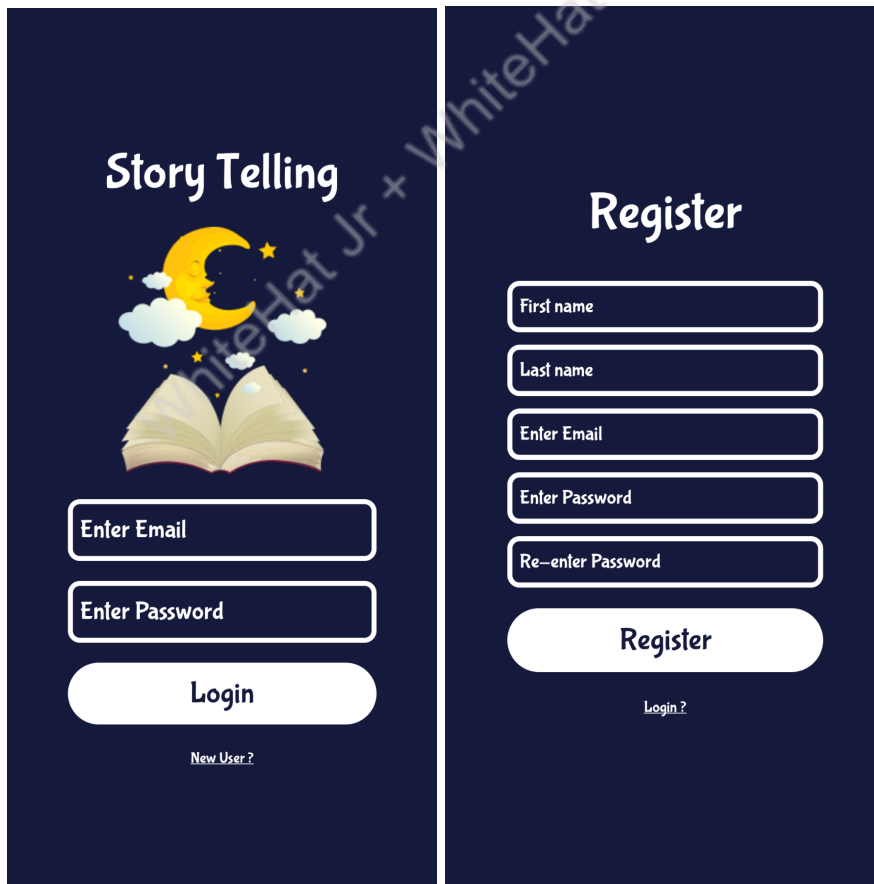
buttonTextNewUser: {

fontSize: RFValue(12),
```

```
color: "#FFFFFF",  
  
fontFamily: "Bubblegum-Sans",  
  
textDecorationLine: 'underline'  
  
}  
  
});
```

7. Test the App by running the app on Mobile and registering a new user. It should register the Email ID in the database.

OUTPUT:



The image displays two side-by-side mobile application screens. The left screen, titled 'Story Telling', features a dark blue background with a yellow crescent moon and stars above an open book. It contains input fields for 'Enter Email' and 'Enter Password', a 'Login' button, and a 'New User?' link. The right screen, titled 'Register', also has a dark blue background and includes input fields for 'First name', 'Last name', 'Enter Email', 'Enter Password', and 'Re-enter Password'. It features a 'Register' button, a 'Login?' link, and a 'New User?' link.

### What's next?

In the next class, we will add the logout feature and we will work on the profile screen.

### Expand your knowledge:

1. To explore more options for creating Authentication in Firebase:  
<https://firebase.google.com/docs/auth?authuser=0>

WhiteHat Jr + WhiteHat Jr + WhiteHat Jr