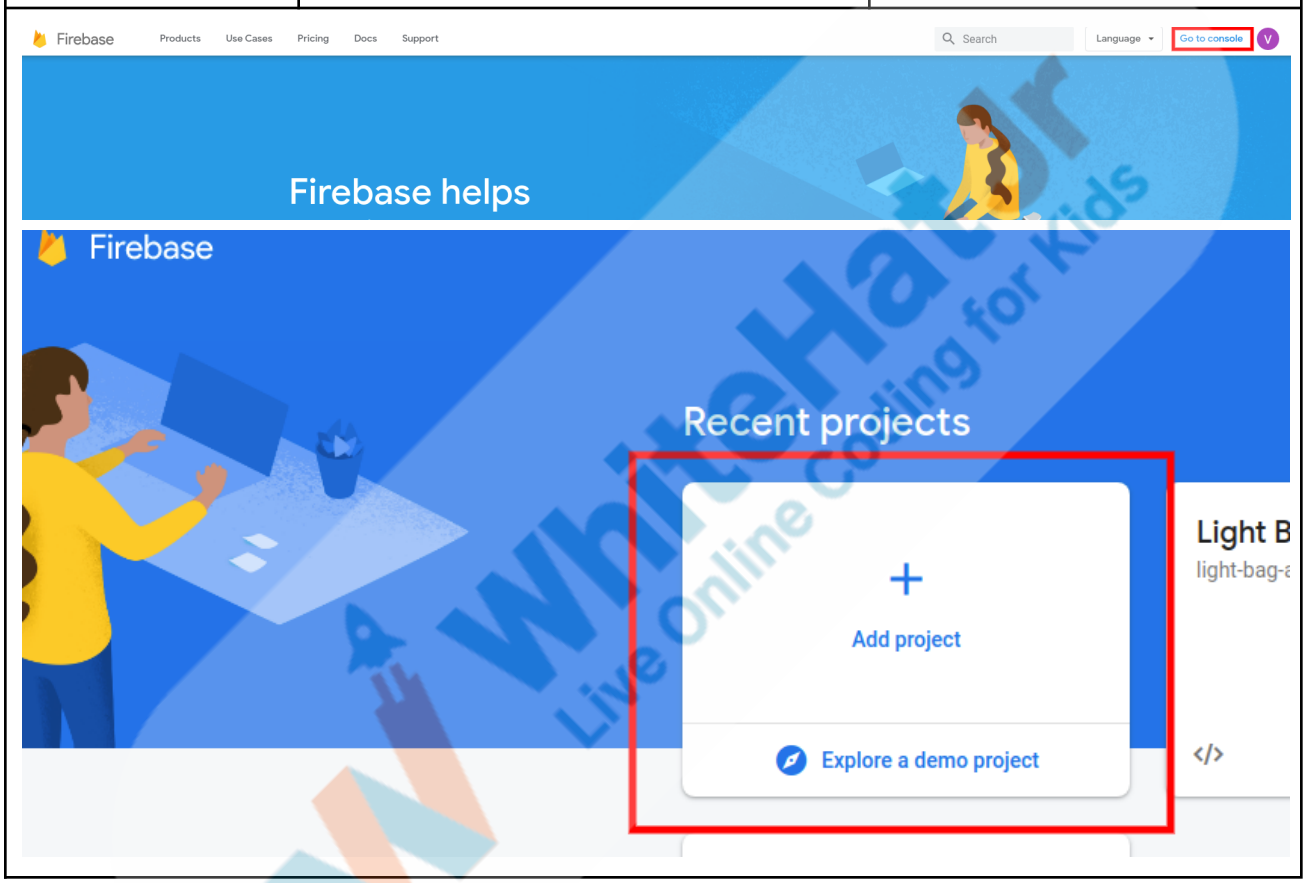
Topic	React Native Databases	
Class Description	**Students learn how to connect their react native application to Firebase Realtime Database. They learn about the concept of timestamp. They also write code to rank the order in which the team pressed the buttons.**	
Class	C58	
Class time	45 mins	
Goal	• Connect the React Native Application to the Realtime database. • Create a timestamp for the button presses.	
Resources Required	• Teacher Resources ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen ○ Android/iOS Smartphone with Expo App installed ○ Expo snack account • Student Resources ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen ○ Android/iOS Smartphone with Expo App installed ○ Expo snack account	
Class structure	**Warm Up** **Teacher-led Activity** **Student-led Activity** **Wrap up**	**5 mins** **15 min** **15 min** **5 min**

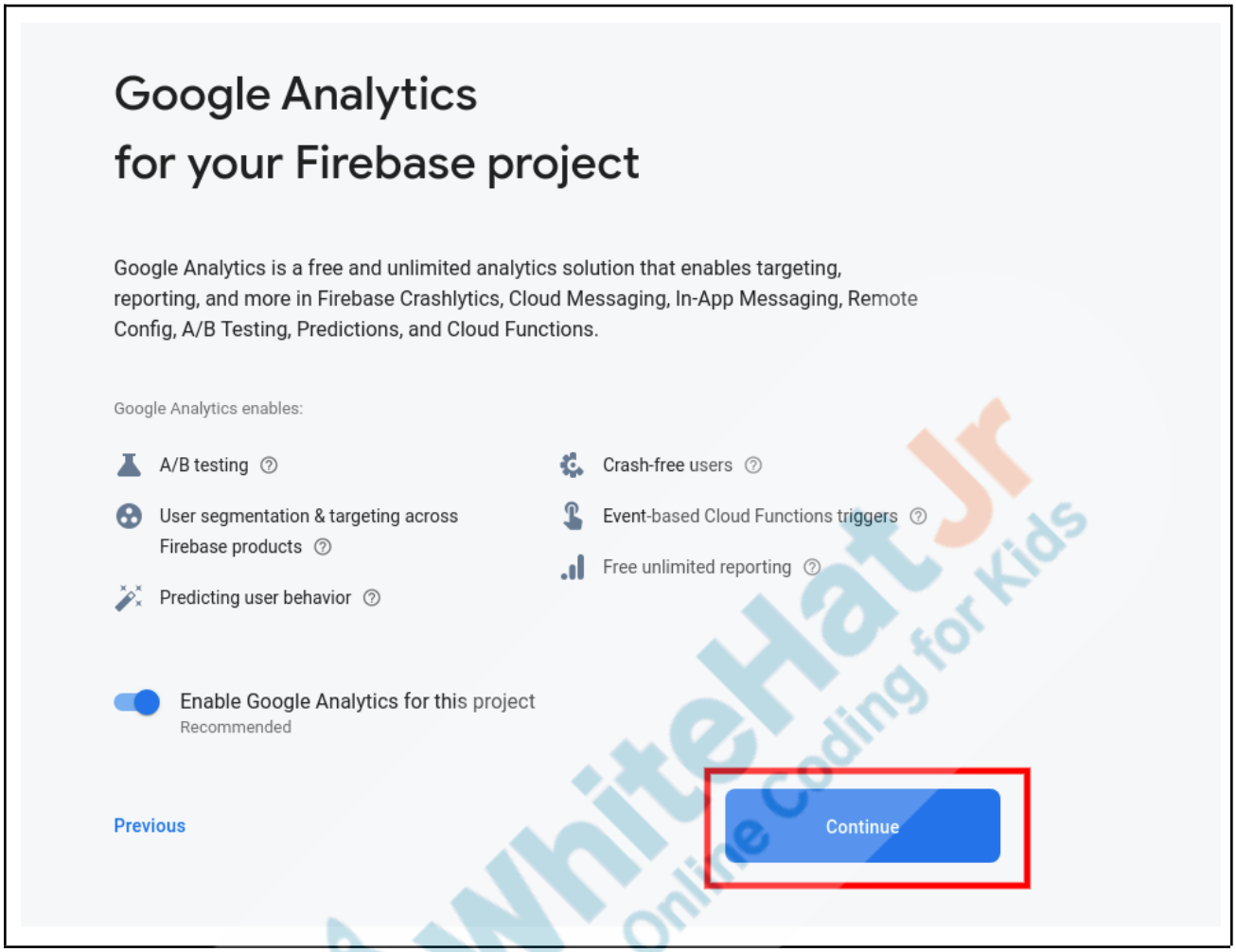
WARM-UP SESSION - 5 mins	
Teacher starts slideshow from slides 1 to 11 Refer to speaker notes and follow the instructions on each slide.	
Activity details	**Solution/Guidelines**

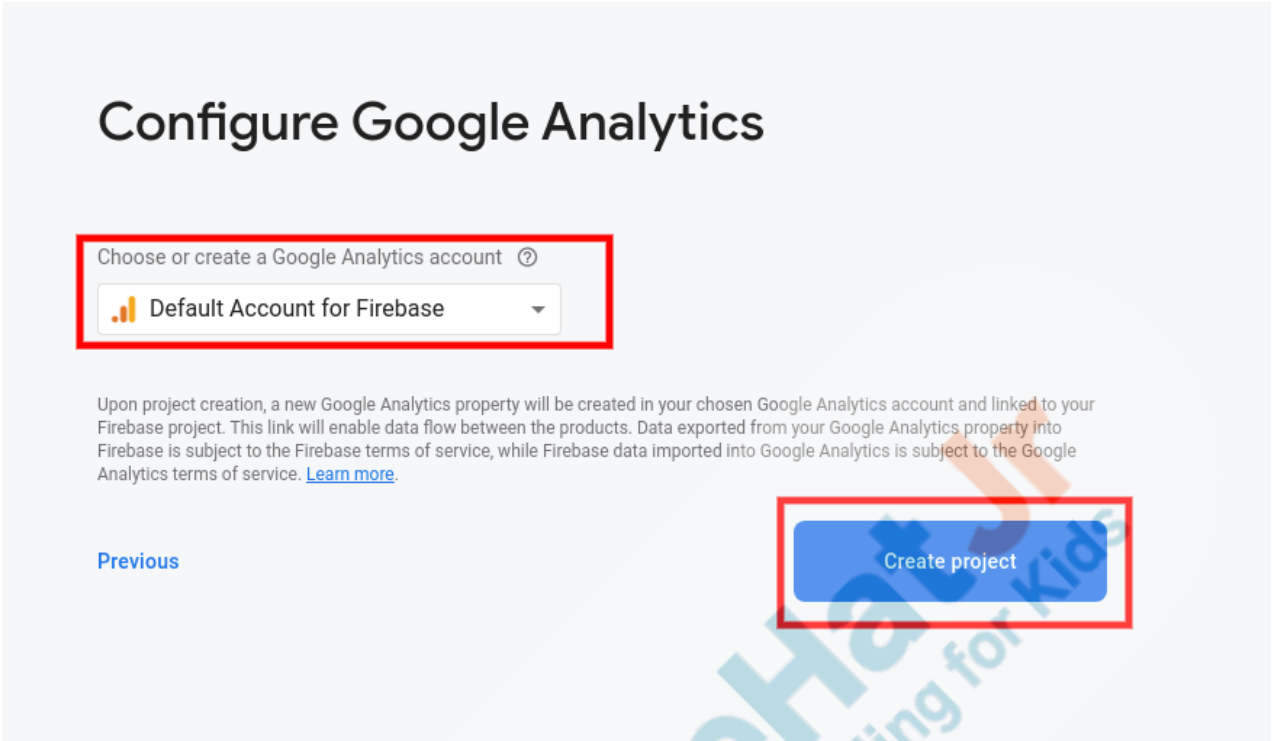
Hi, how have you been? Are you excited to learn something new? ***Run the presentation from slide 1 to slide 3.*** **The following are the warm-up session deliverables:** ● Reconnect with previous class topics. ● Warm-Up quiz session.	**ESR**: Varied Response. Click on the slide show tab and present the slides.
QnA Session	
Question	**Answer**
Which of these components holds the two screens of the app and the App Navigator together? A. createSwitchNavigator B. createAppContainer C. Neither A nor B D. Both A and B	**B**
What do “HomeScreen : HomeScreen” and “BuzzerScreen : BuzzerScreen” represent in the given code snippet? var AppNavigator = createSwitchNavigator({ HomeScreen : HomeScreen, BuzzerScreen : BuzzerScreen }) A. List of screens : key names B. Key names : list of screens C. Route of navigation D. Components : Props	**A**
Continue the warm-up session	
Activity details	**Solution/Guidelines**
Run the presentation from slide 4 to slide 11 to set the problem statement.	

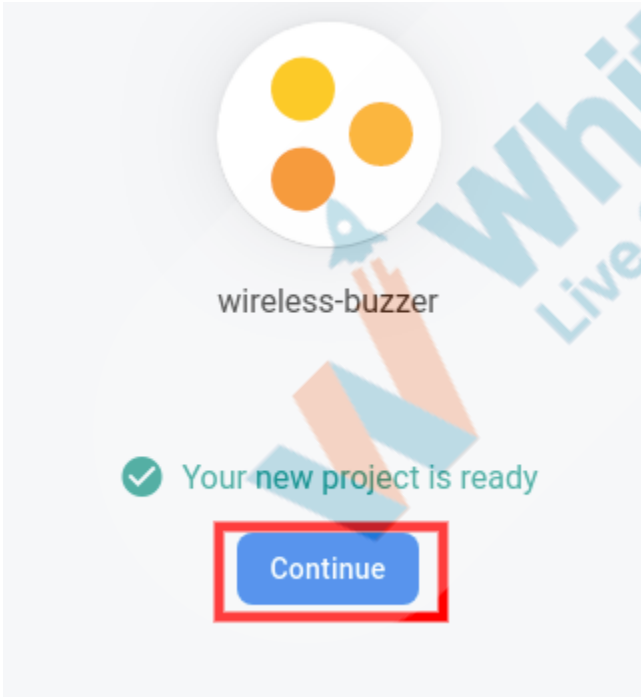
The following are the warm-up session deliverables: • **Review code from the previous class.** • **Introduce the problem of finding out who pressed the Buzzer first.**		
Teacher ends slideshow		
TEACHER-LED ACTIVITY - 15 mins		
Teacher Initiates Screen Share		
<u>CHALLENGE</u> • **Connect the React Native App to the Database.** • **Detect Buzzer Button presses.**		
Step 2: Teacher-led Activity (15 min)	Teacher Opens **<u>Teacher Activity 1</u>** Before we start, like in the previous classes, can you go over the code and explain what we have done so far here? Note: Allow the student to explain the different code blocks and help them wherever their understanding is inconsistent with what is taught in the class.	The student goes through the code and explains what different blocks are doing.
	Alright. So now let's get started with connecting our react native application to firebase. First, we need to create a firebase realtime database for our application. Let's login to console.firebase.com and create a firebase database.	The student observes how to create a new Firebase Database.

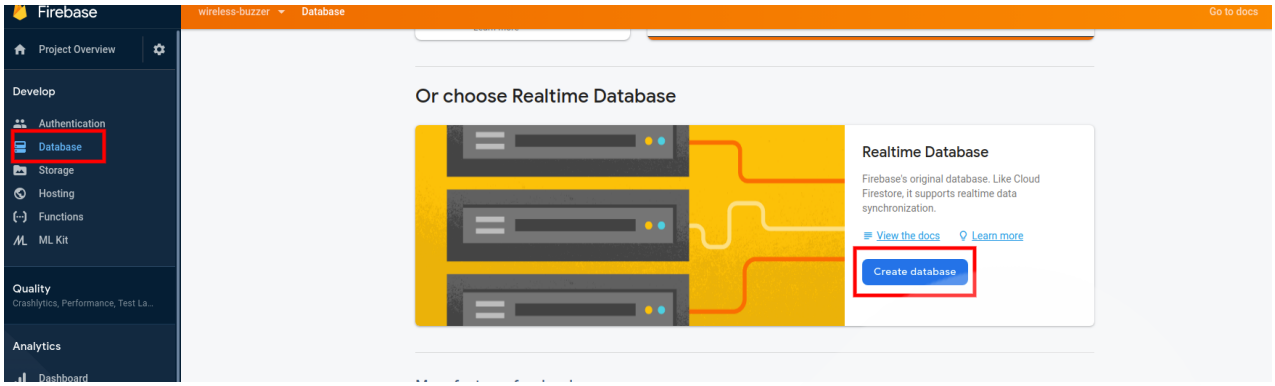
	Teacher logs in to the console.firebase.com. She creates a new Realtime database called "Wireless Buzzer". Note: Create the database in test mode. This will keep the read, write permissions for all users to be true.	

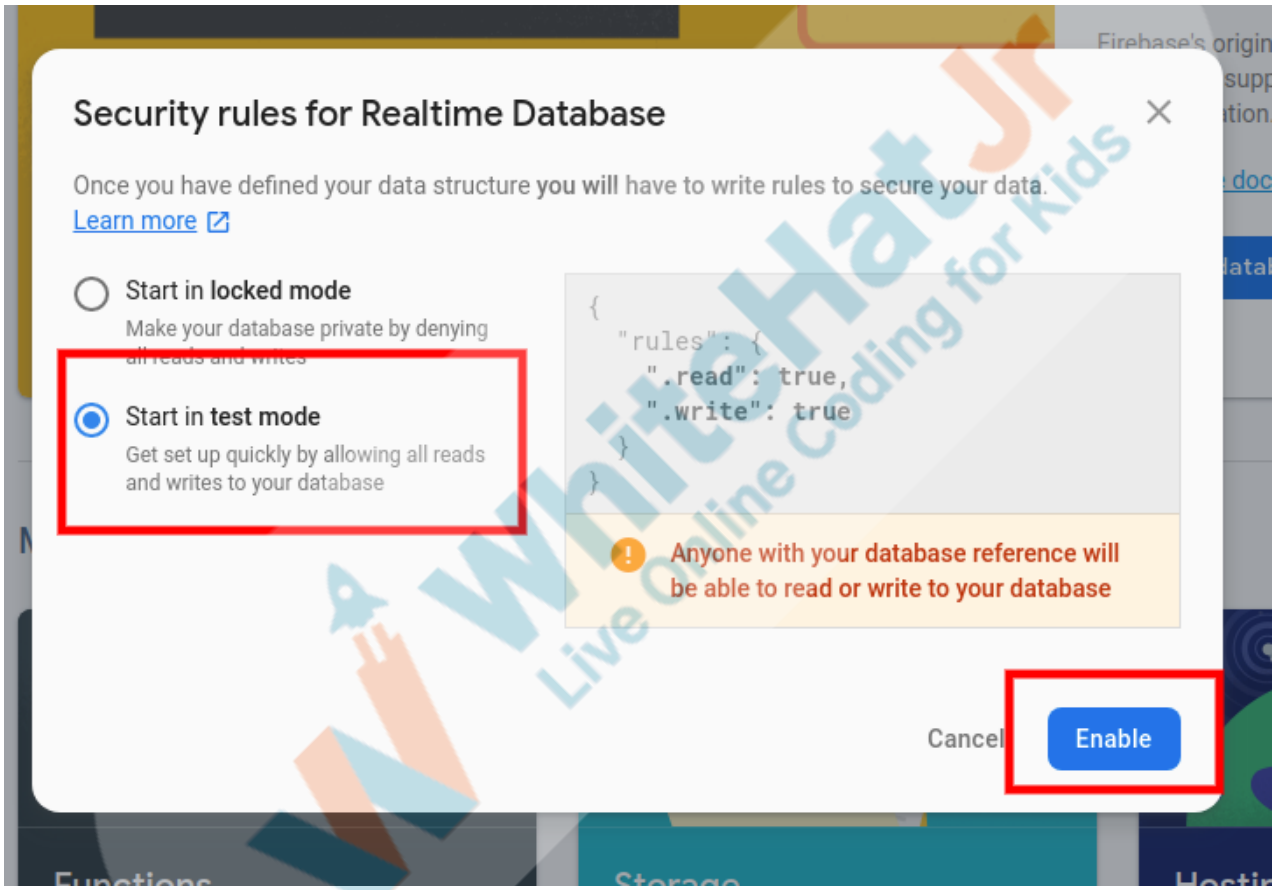






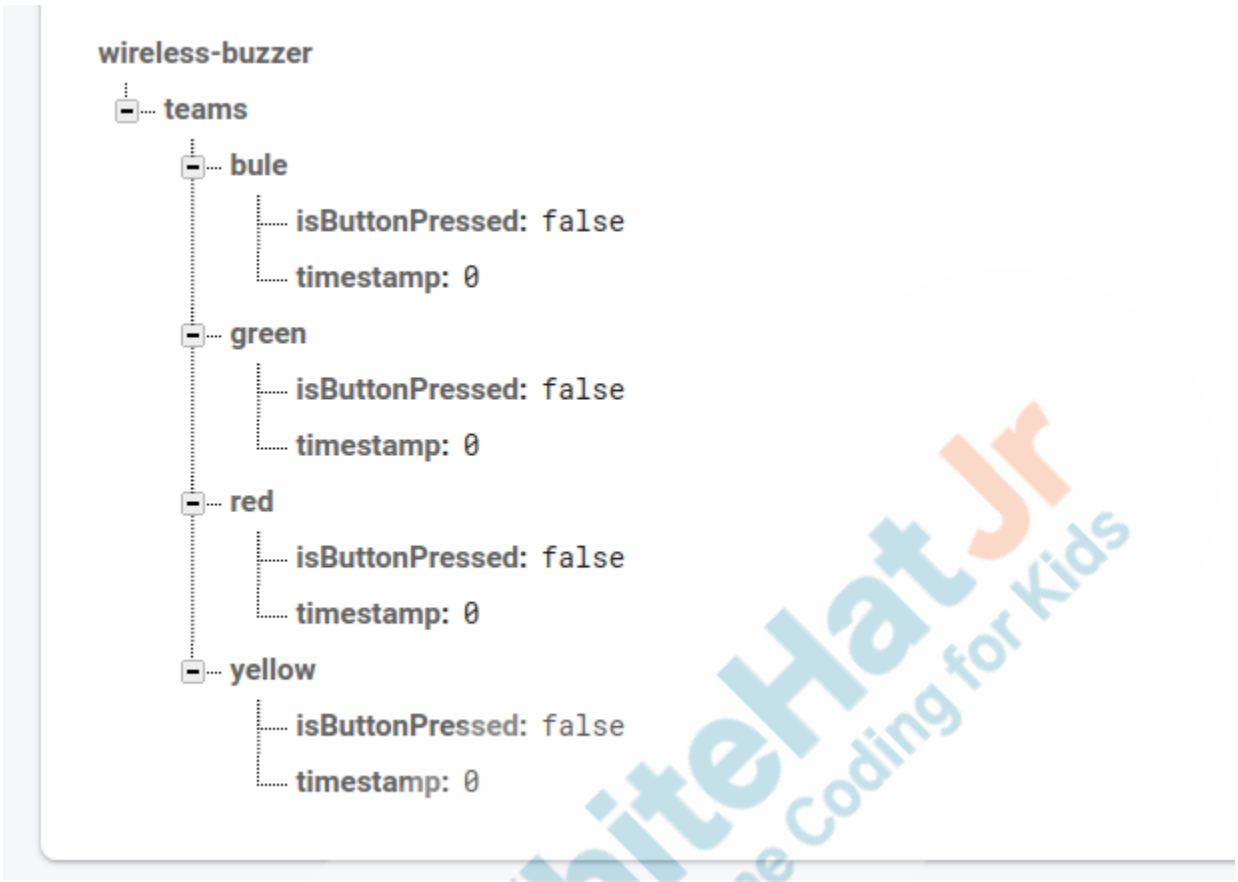




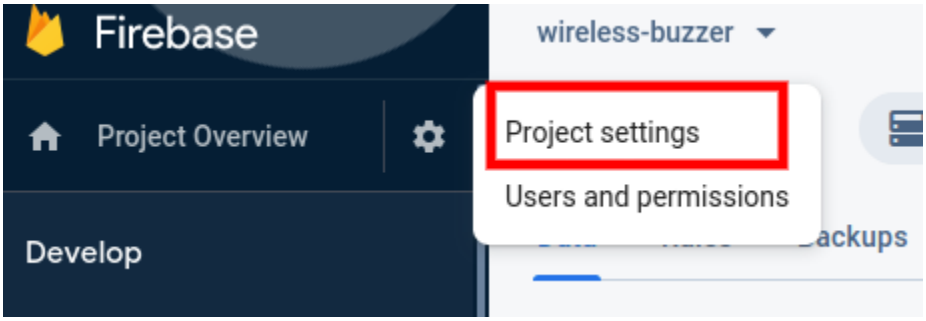


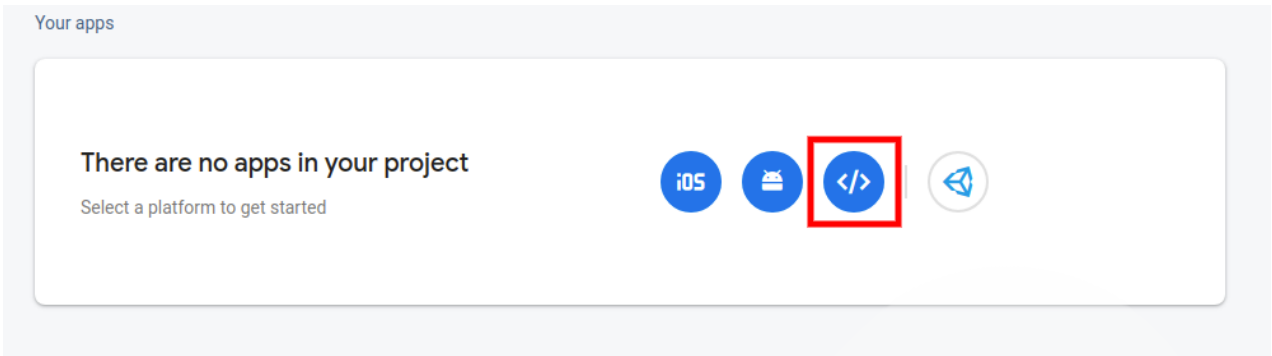
	Let's create data fields in our database. What do you think we should write in our database fields? Remember, the firebase database is similar to a JSON object.	ESR: varied

	Every field has a key name and a value. Each key can also contain more fields nested inside them.	
	We will have a data field called teams. Inside teams, we will have the teams - red, green, blue and yellow. For each team, we are going to have two fields - ‘isButtonPressed’ and ‘timestamp’. Initially ‘isButtonPressed’ is going to have the value of "false". Whenever the team button is pressed, this value will turn to “true”. ‘timestamp’ will capture the time at which the button is pressed. It will contain a default value of 0. Teacher creates these data fields in the application.	The student observes and learns how to create a firebase database.



	We will now need to register our react native app to use the database. Let's do that first. Teacher navigates to Project Settings and creates/registers for a new web application to connect to the database.	The student observes how to register the application to use the database.





	Do you remember how we connected the firebase database to our Car Racing Application last time?	The student tries to recall the database project config keys located under the project settings.

Copy and paste these scripts into the bottom of your <body> tag, but before you use any Firebase services:

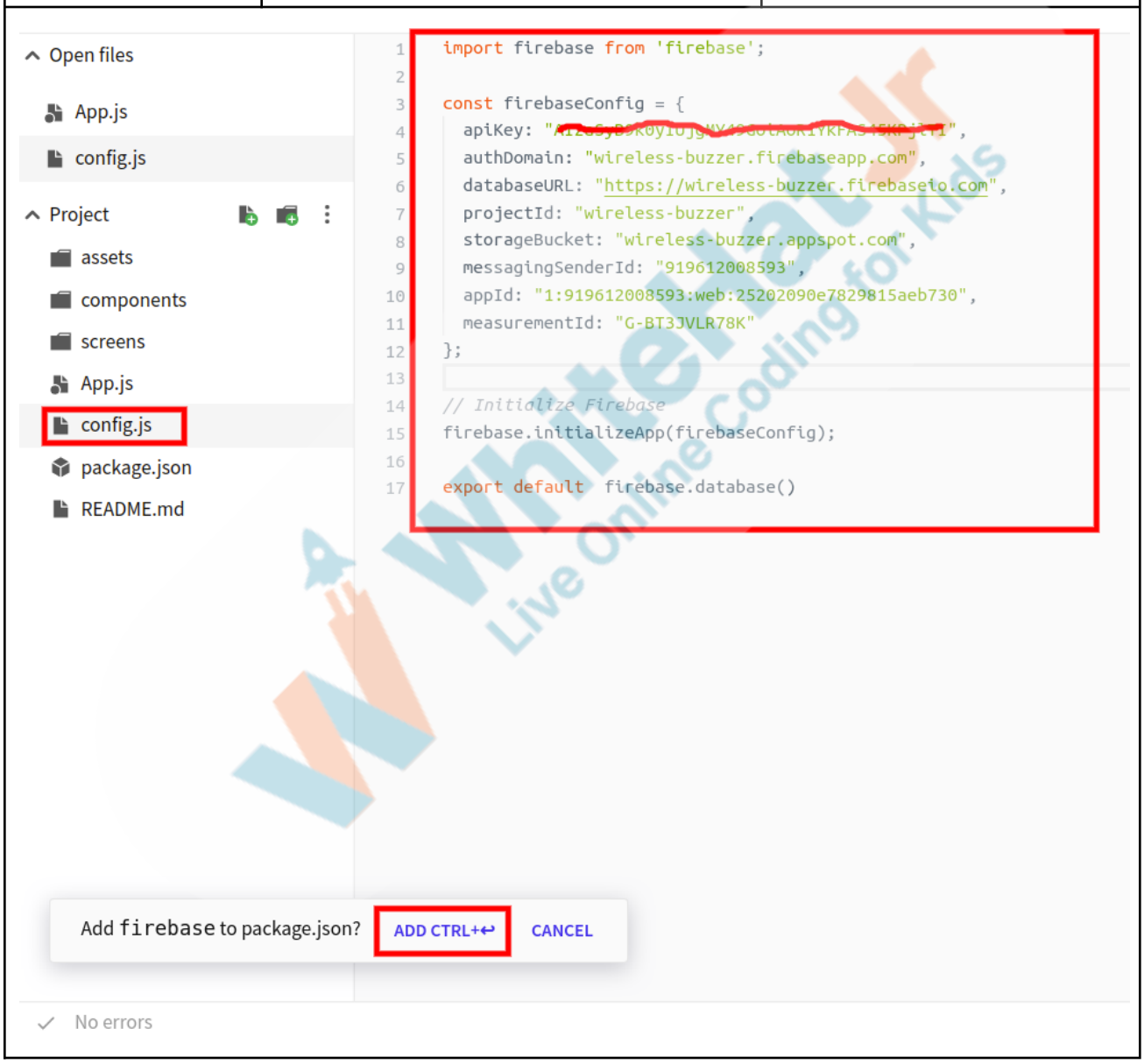
```
<!-- The core Firebase JS SDK is always required and must be listed first -->
<script src="https://www.gstatic.com/firebasejs/7.6.1/firebase-app.js"></script

<!-- TODO: Add SDKs for Firebase products that you want to use
     https://firebase.google.com/docs/web/setup#available-libraries -->
<script src="https://www.gstatic.com/firebasejs/7.6.1/firebase-analytics.js"></

<script>
  // Your web app's Firebase configuration
  var firebaseConfig = {
    apiKey: "[illegible]",
    authDomain: "wireless-buzzer.firebaseapp.com",
    databaseURL: "https://wireless-buzzer.firebaseio.com",
    projectId: "wireless-buzzer",
    storageBucket: "wireless-buzzer.appspot.com",
    messagingSenderId: "919612008593",
    appId: "1:919612008593:web:25202090e7829815aeb730",
    measurementId: "G-BT3JVLR78K"
  };
  // Initialize Firebase
  firebase.initializeApp(firebaseConfig);
  firebase.analytics();
</script>
```

Learn more about Firebase for web: Get Started, Web SDK API Reference, Samples

Continue to console

	These config keys contain the address and access permissions to allow us to use the database from our application. We will be using this in our application.	
	Let's create a new file called "config.js" in our application folder. This file will contain the config keys for our database. We will use it to initialize firebase in our application.	The student learns how to initialize firebase using the firebase config keys.

	Teacher creates a config.js file where she: - imports firebase library. - stores firebase config keys. - initializes firebase app using the config keys. - exports firebase.database() to be used in other files.	



	When do we want to connect to the database? - in which component?	ESR: In ‘SoundButton’ when a button is pressed.
	Great. Let's import the firebase.database() as db from config.js file inside ‘SoundButton.js’. Note: ‘config.js’ file by default exports firebase.database(). The name "db" could be anything.	Student observes and asks questions.

Student Activity 1: Switch Navigator Reference

All changes saved less than 20 seconds ago. See previous saves.

Open files

- App.js
- SoundButton.js
- BuzzerScreen.js
- config.js

Project

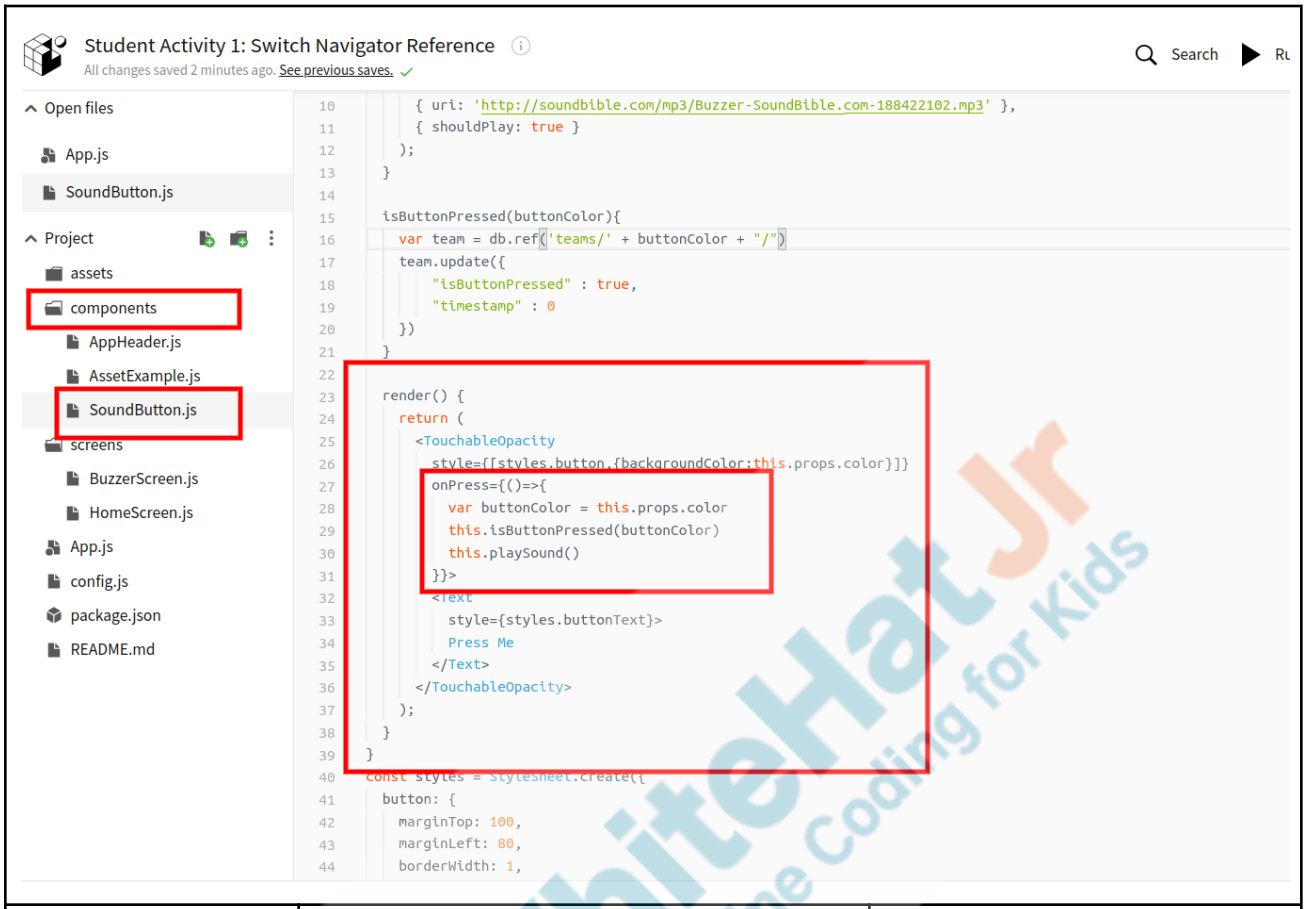
- assets
- components
  - AppHeader.js
  - AssetExample.js
  - SoundButton.js
- screens
  - BuzzerScreen.js
  - HomeScreen.js
- App.js
- config.js
- package.json
- README.md

```
import * as React from 'react';
import { Text, View, TouchableOpacity, StyleSheet } from 'react-native';
import {Audio} from 'expo-av';

import db from '../config';

class SoundButton extends React.Component {
    playSound = async () => {
      await Audio.Sound.createAsync(
        { uri: 'http://soundbible.com/mp3/Buzzer-SoundBible.com-188422102.mp3' },
        { shouldPlay: true }
      );
    }

    render() {
      return (
        <TouchableOpacity
          style={[styles.button,{backgroundColor:this.props.color}]}
          onPress={this.playSound}>
          <Text
            style={styles.buttonText}>
            Press Me
          </Text>
        </TouchableOpacity>
      );
    }
  }
const styles = StyleSheet.create({
  button: {
    marginTop: 100,
    marginLeft: 80,
    borderWidth: 1,
    borderColor: 'rgba(0,0,0,0.2)',
    alignItems: 'center',
    justifyContent: 'center',
```

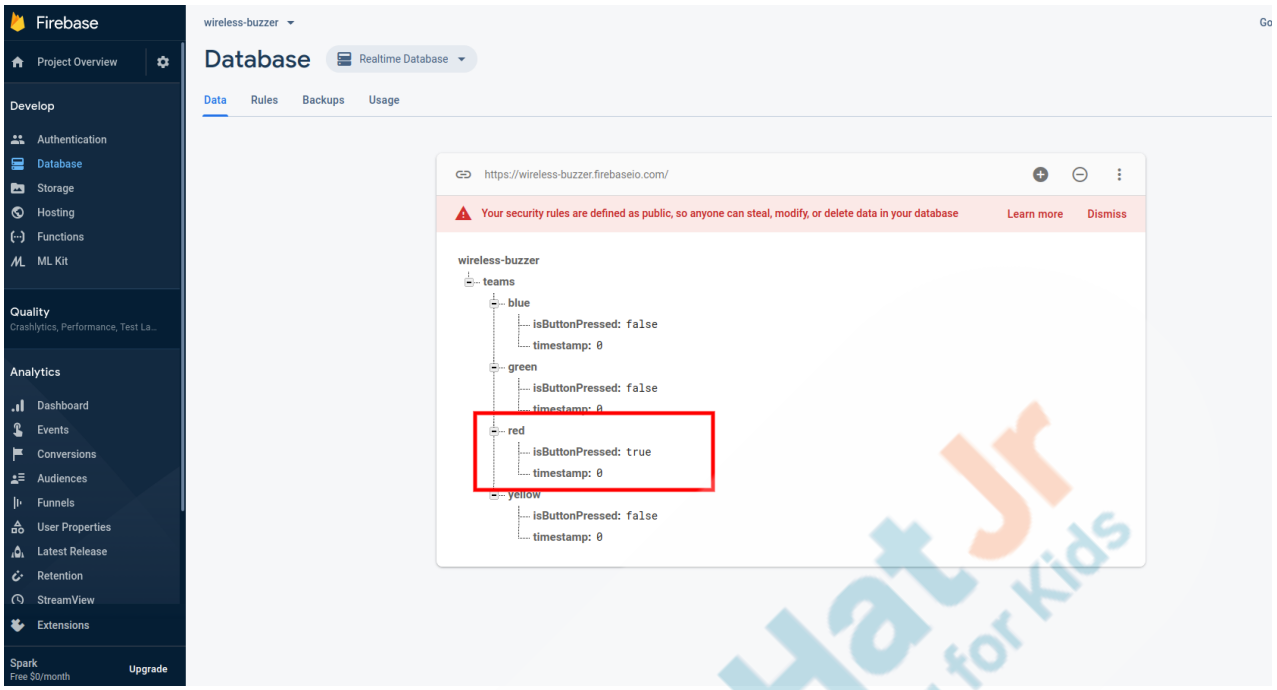
	Inside the SoundButton class, let's write a function called 'isButtonPressed()' which takes teamColor as an input(argument). This function should connect to the database and update the 'isButtonPressed' field in our database from "false" to "true". Can you help me on how to do that? Some guided questions: • What do we need to write to the field in the database? • Which database function will help us write to the database?	The student helps the teacher in writing the questions. ESR: We need a reference to the field in the data. ESR: databaseRef.update() function.

	When do we call this function isButtonPressed()?	ESR: When the buzzer Button is pressed. Inside 'onPress' prop for the 'TouchableOpacity'.

	But we are already calling another function called ‘playSound()’ when the button is pressed. How do we call one more function now? Allow time for the student to think.	The student comes up with varied responses.
	A good way to solve this would be to create a third function which first calls ‘isButtonPressed()’ and then calls ‘playSound()’. We can call this function inside the ‘isButtonPressed()’ function. Also, we can create this function inside the ‘onPress’ prop itself inside { } Teacher uses arrow keys to create a function which calls both ‘isButtonPressed()’ and ‘playSound()’ functions. This function does not have a name and is called an anonymous function.	The student learns how to write an anonymous function.



	We have done great so far. Let's run the app and see what happens. Teacher shows the change in the 'isButtonPressed' field for the team when the button of that color is pressed.	The student runs and tests the app on their phone. He/She chooses a team color and presses the buzzer.



	Awesome. So now I hope you know how to connect to the firebase database from your react native app and update any field. Can you now connect firebase database to your own app and update the timeStamp field for the team when the button is pressed? The timeStamp should contain the time at which the button is pressed. Note: Remind the student about the Date object that they have already learned earlier.	ESR: I can try.
	Great. And obviously I am there to support you.	-
Teacher Stops Screen Share		

	Now it's your turn. Please share your screen with me.	
STUDENT-LED ACTIVITY - 15 mins		
• **Ask Student to press ESC key to come back to panel** • **Guide Student to start Screen Share** • **Teacher gets into Fullscreen**		
<u>ACTIVITY</u> • **Add Timestamp for each button press.**		
Teacher starts slideshow from slides 12 to 13 Refer to speaker notes and follow the instructions on each slide.		
Step 3: Student-Led Activity (15 min)	Guide the student to create a firebase database and create a new database - wireless-buzzer	The student visits console.firebase.com and creates a new firebase database.
	Guide the student to create new fields in the database.	The student creates the fields in the database.
	Guide the student to register the app and get the config keys for their app database.	The student generates config keys for the app by registering their app.
	Guide the student to create config.js file in their project. Guide them to initialize the firebase app using config keys and export firebase.database.	The student creates config.js file, initializes the firebase function and exports firebase.database().
	Guide the student to import db from the config file.	The student imports db from config.js file inside the SoundButton Component.

	Guide the student to create the isButtonPressed function.	The student creates a new function 'isButtonPressed'.
	Inside isButtonPressed, let's create a new Date object. This will store the current date/time. We can convert the date into milliseconds using getTime. In computing, time is always measured since Jan 1, 1970. time.getTime() will give us the number of milliseconds passed since that time!!	The student creates a new date object which stores the current time. He/She converts the time into milliseconds.
	Now, let's get a reference to our team in the database and update both isButtonPressed and timeStamp when the button is pressed.	The student gets the database reference for the team and updates both isButtonPressed and timeStamp.

Student Activity 1: Switch Navigator Reference

All changes saved less than 10 seconds ago. See previous saves.

Search Run

Open files
- App.js
- SoundButton.js

Project
- assets
- components
  - AppHeader.js
  - AssetExample.js
  - SoundButton.js
- screens
  - BuzzerScreen.js
  - HomeScreen.js
- App.js
- config.js
- package.json
- README.md

```
import * as React from 'react';
import { Text, View, TouchableOpacity, StyleSheet } from 'react-native';
import {Audio} from 'expo-av';

import db from '../config';

class SoundButton extends React.Component {
    playSound = async () => {
     await Audio.Sound.createAsync(
        { uri: 'http://soundbible.com/mp3/Buzzer-SoundBible.com-188422102.mp3' },
        { shouldPlay: true }
     );
    }

  isButtonPressed(buttonColor){
    var time =  new Date().getTime();
    var team = db.ref('teams/' + buttonColor + "/")
    team.update({
        "isButtonPressed" : true,
        "timestamp" : time
    })
  }

  render() {
    return (
```

	Finally, we need to call both the functions isButtonPressed() and playSound() inside onPress prop. Let's do that.	The student creates an anonymous function inside onPress prop which calls both the functions.

Student Activity 1: Switch Navigator Reference

All changes saved 2 minutes ago. See previous saves.

Search

Open files

App.js

SoundButton.js

Project

assets

components

AppHeader.js

AssetExample.js

SoundButton.js

screens

BuzzerScreen.js

HomeScreen.js

App.js

config.js

package.json

README.md

```
10        { uri: 'http://soundbible.com/mp3/Buzzer-SoundBible.com-188422102.mp3' },
11        { shouldPlay: true }
12      );
13    }
14
15    isButtonPressed(buttonColor){
16      var team = db.ref('teams/' + buttonColor + "/")
17      team.update({
18          "isButtonPressed" : true,
19          "timestamp" : 0
20      })
21    }
22
23    render() {
24      return (
25        <TouchableOpacity
26          style={[styles.button,{backgroundColor:this.props.color}]}
27          onPress={()=>{
28            var buttonColor = this.props.color
29            this.isButtonPressed(buttonColor)
30            this.playSound()
31          }}>
32          <Text
33            style={styles.buttonText}>
34            Press Me
35          </Text>
36        </TouchableOpacity>
37      );
38    }
39  }
40  const styles = StyleSheet.create({
41    button: {
42      marginTop: 100,
43      marginLeft: 80,
44      borderWidth: 1,
```

	Ok! Now let's quickly run and test our app	The student tests the app on his.her phone. He/She checks for the change in the database when the sound button is pressed.
	Awesome! You have done it!	

Teacher Guides Student to Stop Screen Share

WRAP-UP SESSION - 5 Mins

Teacher starts slideshow from slide 14 to slide 24	
Activity details	**Solution/Guidelines**
Run the presentation from slide 14 to slide 24 **Following are the wrap-up session deliverables:** • **Explain the facts and trivias** • **Next class challenge** • **Project for the day** • **Additional Activity**	Guide the student to develop the project and share with us.
Quiz time - Click on in-class quiz	
Question	**Answer**
In computing, time is measured since January 1, 1970. Which function will give us the number of milliseconds passed since that time? A. time.convertTime() B. time.fixTime() C. time.getTime() D. time.stopTime()	**C**
In our program, what does timestamp do? A. Shows us the current system time of the user. B. Stops the app from functioning after a particular time. C. Displays the time taken for the app to load at the user's end. D. Captures the time at which the buzzer is pressed.	**D**
A function which does not have a name is called an _____. A. Nameless function B. Anonymous function C. Unnamed function D. Invisible function	**B**
End the quiz panel	

FEEDBACK • **Encourage the student to think about the bugs in their code.** • **Encourage the student to make reflection notes in the markdown format.** • **Complement the student for her/his effort in the class.**		
	So now teams can press the buzzer sound and you can look at the database to check who pressed the buzzer first. But you will have to compare the time in milliseconds everytime. You also have to look at your database every time. That is not a good solution....Is it? What are the other problems in our app?	ESR: No Once a button is pressed, we have to manually reset both the timestamp and isButtonPressed fields in the database.
	Yes. That's a problem too. We will solve this problem by creating a Quiz master app in coming classes. Our Quiz Master app can reset the database and read from the database to rank who pressed the button first.	
	You get a “hats off”. Great! See you in the next class then where we will be creating the Quiz Master App.	Make sure you have given at least 2 Hats Off during the class for:  

		Strong Concentration +10
Project Pointers and Cues (5 min)	## NEWSLETTER APP - 2 **Goal of the Project:** Today you have learnt about “React with Databases”. You have coded for a wireless buzzer where the first team who clicks on the buzzer will be registered along with the time stamp. In this project, you will apply your learning to add more functionality to the Newsletter App, which you started creating in the previous project. *This is a continuation of Project 57. So make sure you complete that project before you attempt this one.* **Story:** In a poll that you ran, ninety percent of your friends said that they would really benefit from a Newsletter type of app! You have already started building this awesome app for your friends. You have created different buttons for the user to quickly navigate to different screens. Now create a Firebase database and connect it to the app. I am very excited to see your project solution and I know you both will do really well.	

	Bye Bye!	
Teacher Clicks ✖ End Class		
Additional Activities	Encourage the student to write reflection notes in their reflection journal using markdown. Use these as guiding questions: ● What happened today? - Describe what happened - Code I wrote ● How did I feel after the class? ● What have I learned about programming and developing games? ● What aspects of the class helped me? What did I find difficult?	The student uses the markdown editor to write her/his reflection in a reflection journal.

Activity	Activity Name	Links
Teacher Activity 1	Previous Class Reference	https://snack.expo.io/@whitehatjr/pro-c57
Student Activity 1	Previous Class Reference	https://snack.expo.io/@whitehatjr/pro-c57
Teacher Activity 2	Teacher Reference	https://snack.expo.io/@whitehatjr/pro-c58
Teacher Reference visual aid link	Visual aid link	https://curriculum.whitehatjr.com/Visual+Project+Asset/PRO_VD/PRO_C58_withcues.html

Teacher Reference In-class quiz	In-class quiz	https://s3-whjr-curriculum-uploads.whjr.online/af0418ba-d973-459d-9e36-3236aa66dab3.pdf