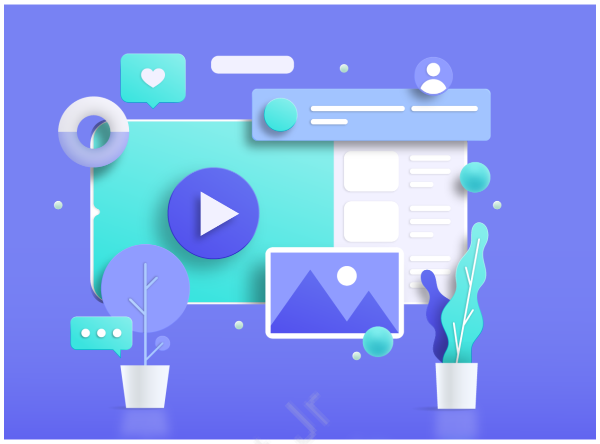
WhiteHat Jr
Live Online Coding for Kids

CUSTOM STYLED DRAWER NAVIGATION



What is our GOAL for this MODULE?

In this class, we learned to style our Drawer Navigation, and we also integrated the like functionality on our stories.

What did we ACHIEVE in the class TODAY?

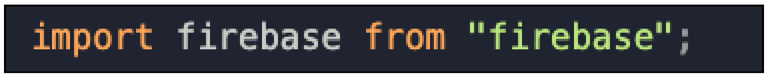
- Custom styled our Drawer Navigation Tab.
- Added the like functionality for stories in our app.

Which CONCEPTS/ CODING BLOCKS did we cover today?

- Styling of Drawer Navigation.
- “Like” functionality.

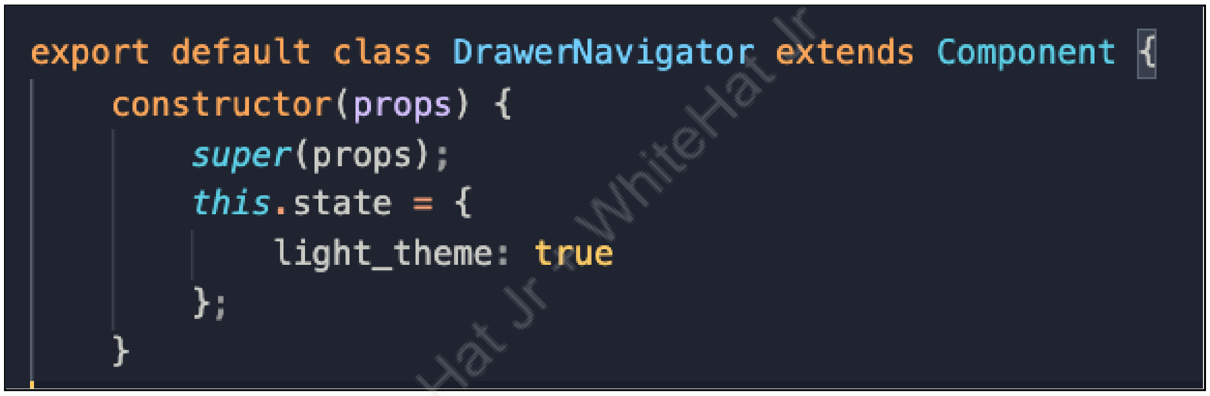
How did we DO the activities?

1. Convert a **Drawer Navigator** into a class to incorporate **construtor()** and **ComponentDidMount()** as follows:
    - Import **firebase.**



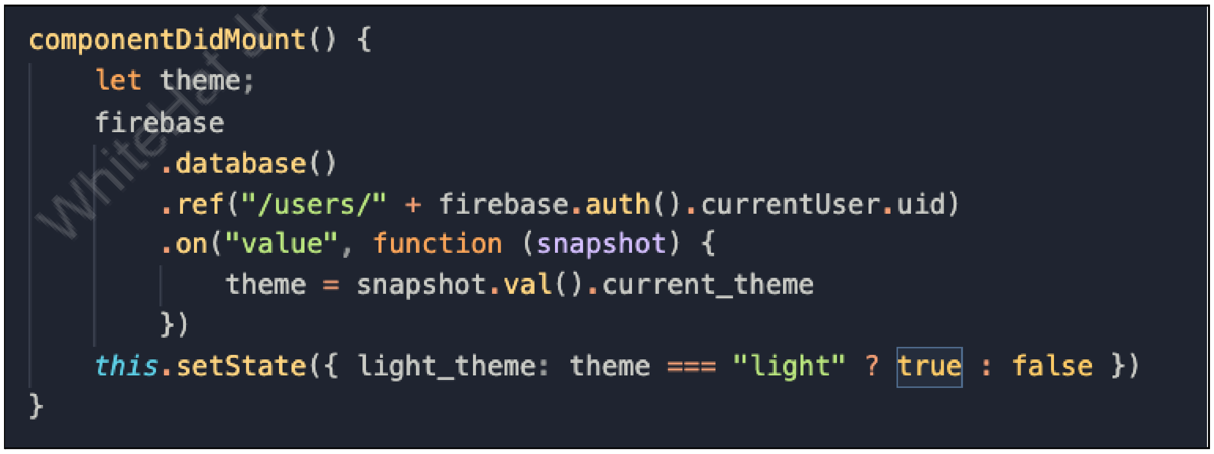
```
import firebase from "firebase";
```

- Create a **constructor()** and declare a property.



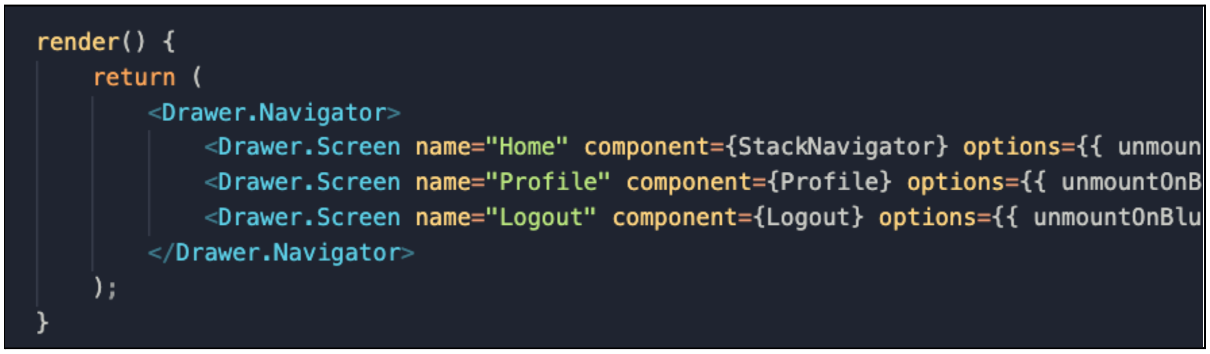
```
export default class DrawerNavigator extends Component {
    constructor(props) {
        super(props);
        this.state = {
            light_theme: true
        };
    }
```

- Query database to fetch the theme selected by the user previously, and set it to the property.



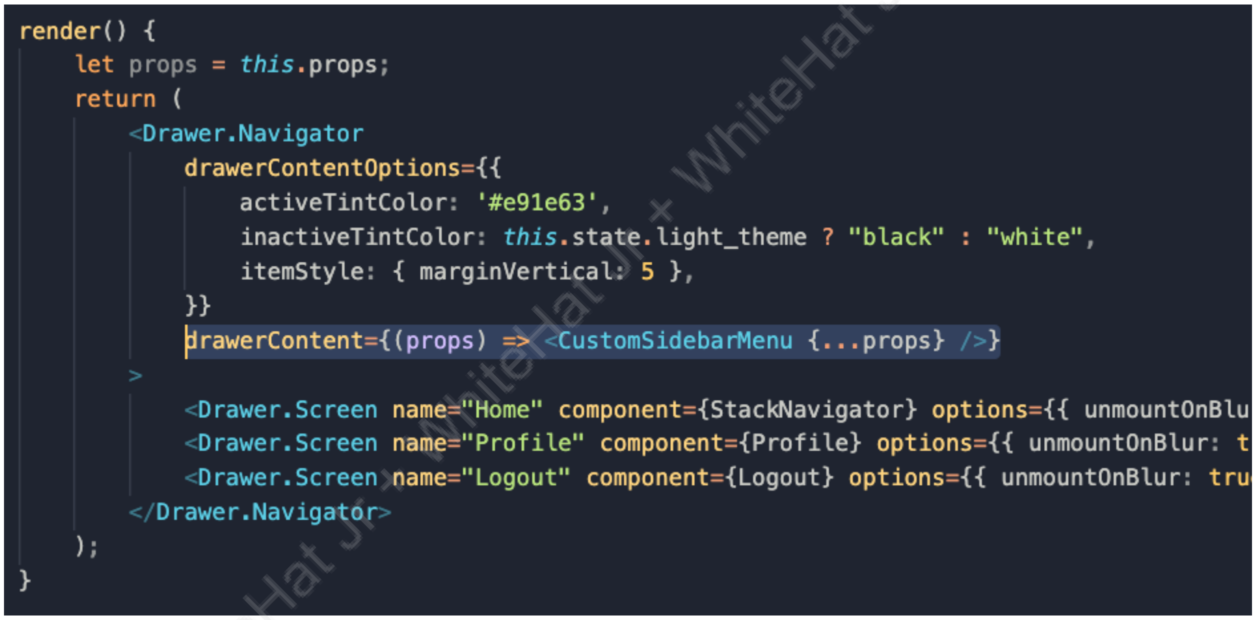
```
componentDidMount() {
    let theme;
    firebase
        .database()
        .ref("/users/" + firebase.auth().currentUser.uid)
        .on("value", function (snapshot) {
            theme = snapshot.val().current_theme
        })
    this.setState({ light_theme: theme === "light" ? true : false })
}
```

- Create the **render()** function, as **DrawerNavigator** is now a class and not a function.



```
render() {
    return (
        <Drawer.Navigator>
            <Drawer.Screen name="Home" component={StackNavigator} options={{ unmoun
            <Drawer.Screen name="Profile" component={Profile} options={{ unmountOnB
            <Drawer.Screen name="Logout" component={Logout} options={{ unmountOnBlu
        </Drawer.Navigator>
    );
}
```

2. Add an attribute **drawerContentOptions** and **drawerContent** to our **<Drawer.Navigator>** component.



```
render() {
    let props = this.props;
    return (
        <Drawer.Navigator
            drawerContentOptions={{
                activeTintColor: '#e91e63',
                inactiveTintColor: this.state.light_theme ? "black" : "white",
                itemStyle: { marginVertical: 5 },
            }}
            drawerContent={(props) => <CustomSidebarMenu {...props} />}
        >
            <Drawer.Screen name="Home" component={StackNavigator} options={{ unmountOnBlu
            <Drawer.Screen name="Profile" component={Profile} options={{ unmountOnBlur: t
            <Drawer.Screen name="Logout" component={Logout} options={{ unmountOnBlur: tru
        </Drawer.Navigator>
    );
}
```

3. Create the **CustomSidebarMenu.js** screen in the **screen** folder as follows:

- Import the necessary components.

```
import React, { Component } from 'react';
import {
  SafeAreaView,
  StyleSheet,
  Image,
} from 'react-native';
import firebase from "firebase";
import {
  DrawerContentScrollView,
  DrawerItemList,
} from '@react-navigation/drawer';
```

- Create the class **CustomSideBarMenu.**

```
export default class CustomSidebarMenu extends Component {

}
```

- Inside this class, create a **constructor()** and **componentDidMount()** to fetch and save the user-selected theme from the database.

```
constructor(props) {
    super(props);
    this.state = {
        light_theme: true
    };
}

componentDidMount() {
    let theme;
    firebase
        .database()
        .ref("/users/" + firebase.auth().currentUser.uid)
        .on("value", function (snapshot) {
            theme = snapshot.val().current_theme
        })
    this.setState({ light_theme: theme === "light" ? true : false })
}
```

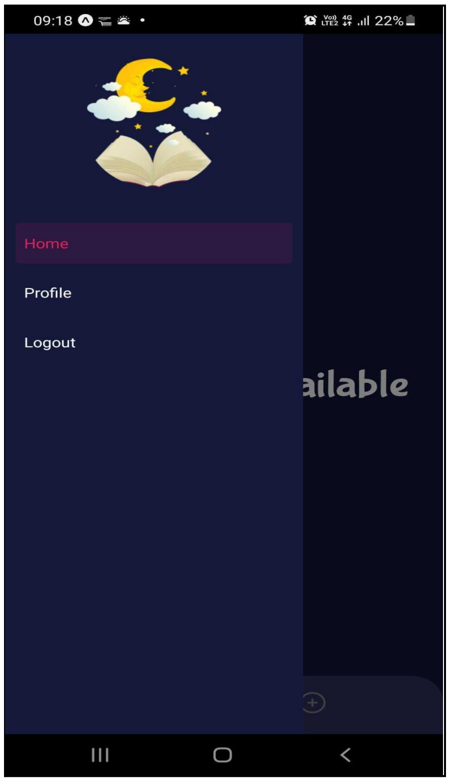
- Add a **render()** function to create the **DrawerContent** component.

```
render() {
    let props = this.props;
    return (
        <SafeAreaView style={{ flex: 1, backgroundColor:
this.state.light_theme ? "white" : "#15193c" }}>
           <Image source={require("../assets/logo.png")}
style={styles.sideMenuProfileIcon}></Image>
           <DrawerContentScrollView {...props}>
              <DrawerItemList {...props} />
           </DrawerContentScrollView>
        </SafeAreaView>
     );
  }
```

- Add styles to the image displayed in **customSideBarMenu()**.

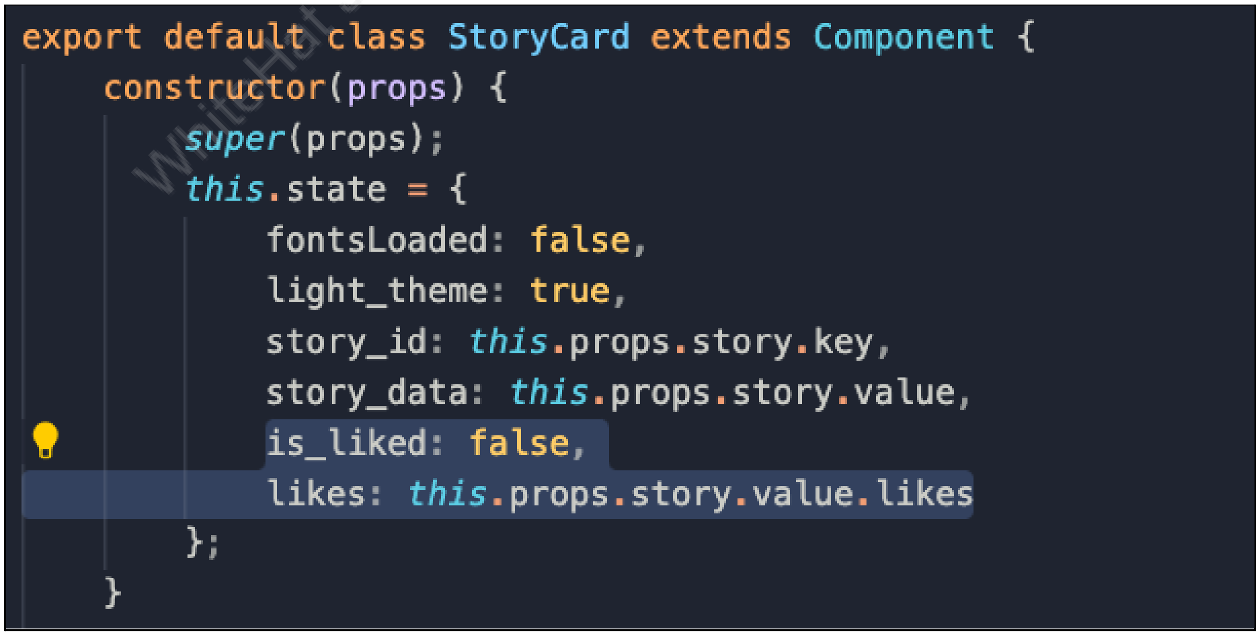
```
const styles = StyleSheet.create({
 sideMenuProfileIcon: {
   width: RFValue(140),
   height: RFValue(140),
   borderRadius: RFValue(70),
   alignSelf: "center",
   marginTop: RFValue(60),
   resizeMode: "contain"
 }
});
```

4. Run the code to check the output.



Now, to add the like button functionality in our app.

5. In **StoryCard.js,** start with creating **constructor()** and states. The first state **is_liked** is, for if the story is liked, which is **false** by default. We also have stored the number of likes separately for now.



```
export default class StoryCard extends Component {
    constructor(props) {
        super(props);
        this.state = {
            fontsLoaded: false,
            light_theme: true,
            story_id: this.props.story.key,
            story_data: this.props.story.value,
            is_liked: false,
            likes: this.props.story.value.likes
        };
    }
```

6. Add a **<TouchableOpacity>** component around our like button so that we can make it clickable.

```
<TouchableOpacity onPress={() => this.likeAction()}>
                        <View style={this.state.is_liked ? styles.likeButtonLiked :
styles.likeButtonDisliked}>
                           <View style={styles.likeIcon}>
                              <Ionicons name={"heart"} size={30}
color={this.state.light_theme ? "black" : "white"} style={{ width: 30, marginLeft: 20,
marginTop: 5 }} />
                           </View>
                           <View>
                              <Text style={this.state.light_theme ? styles.likeTextLight :
styles.likeText}>{this.state.likes}</Text>
                           </View>
                        </View>
                     </TouchableOpacity>
```

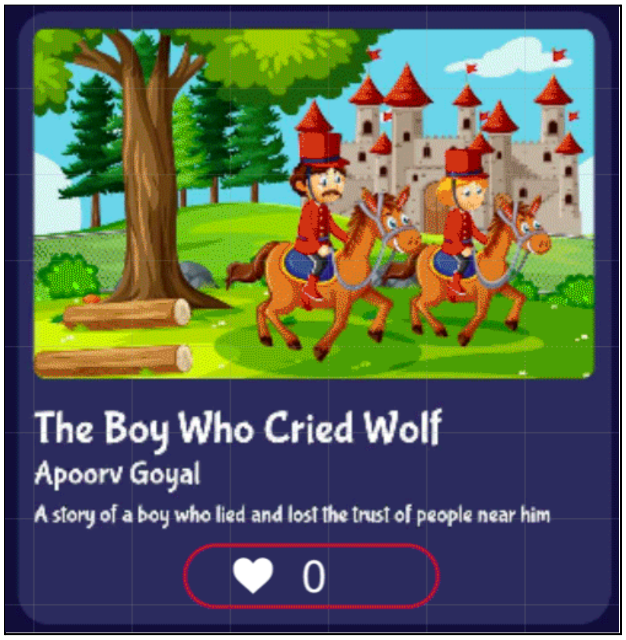
7. Add styles to **TouchableOpacity** to update its style when the button is clicked or not selected.

```
likeButtonLiked: {
     backgroundColor: "#eb3948",
     borderRadius: 30,
     width: 160,
     height: 40,
     flexDirection: "row"
  },
  likeButtonDisliked: {
     borderColor: "#eb3948",
     borderWidth: 2,
     borderRadius: 30,
     width: 160,
     height: 40,
    flexDirection: "row"
},
```

8. Now, finally code the **likeAction().**

```
likeAction = () => {
    if (this.state.is_liked) {
        firebase.database()
            .ref('posts')
            .child(this.state.story_id)
            .child('likes')
            .set(firebase.database.ServerValue.increment(-1))
        this.setState({ likes: this.state.likes -= 1, is_liked: false })
    } else {
        firebase.database()
            .ref('posts')
            .child(this.state.story_id)
            .child('likes')
            .set(firebase.database.ServerValue.increment(1))
        this.setState({ likes: this.state.likes += 1, is_liked: true })
    }
}
```

Output:


The Boy Who Cried Wolf
Apoorv Goyal
A story of a boy who lied and lost the trust of people near him
0

What's NEXT?

In the next class, we will brainstorm some ideas to create your app.

EXTEND YOUR KNOWLEDGE

Bookmark the following link to know more about DrawerNavigator:
https://reactnavigation.org/docs/drawer-based-navigation/