Topic	COMPLETING THE FACE RECOGNITION APP	
Class Description	**Students will revisit the concepts of stack navigation in react native to complete the face recognition app.**	
Class	**C185**	
Class time	**45 mins**	
Goal	• Learn to add stack navigation in the app to create the first page in the App UI.	
Resources Required	• Teacher Resources: ○ Visual Studio Code Editor ○ laptop with internet connectivity ○ smartphone ○ earphones with mic ○ notebook and pen • Student Resources: ○ Visual Studio Code Editor ○ laptop with internet connectivity ○ smartphone ○ earphones with mic ○ notebook and pen	
Class structure	**Warm-Up** **Teacher-led Activity** **Student-led Activity** **Wrap-Up**	**5 mins** **15 mins** **20 mins** **5 mins**

CONTEXT

- **Design First Page of the App UI**

Class Steps	Teacher Action	Student Action

Step 1: Warm-Up (5 mins)	Hi, how are you? Great!	**ESR:** I am good!
	Can you tell me what we have learned in the previous class? Great! Today, we will be completing our Face Recognition App. For this, we will add Stack Navigation to our code and then create a home screen for our App! Do you remember what stack navigation is? Great! So, are you excited for today's class? Let's get started then.	**ESR:** ● We learned how to create categories and then we created categories for our filters! **ESR**: It is a form of navigation available in react native to navigate between different screens. It stacks the screens on top of one another. **ESR**: Yes.

	Click on the  *button on the bottom right corner of your screen to start the In-Class Quiz.* *A quiz will be visible to both you and the student.* *Encourage the student to answer the quiz question.* *The student may choose the wrong option, help the student to think correctly about the question and then answer again.* *After the student selects the correct option, the*  *button will start appearing on your screen.* *Click the End quiz to close the quiz pop-up and continue the class.*	
Teacher Initiates Screen Share		
CHALLENGE • **Design App UI** • **Adding Stack Navigation to the App**		

<table>
<tr><td>Step 2:
Teacher-led
Activity
(15 mins)</td><td>[Teacher Activity 1]

To implement stack navigation in our App, we have to install certain modules in our app! Let’s take the time to install them one by one so that we are ready to have our Stack Navigation integrated with our app -</td><td></td></tr>
<tr><td colspan="3">expo install react-native-gesture-handler

expo install react-native-reanimated

expo install react-native-screens

expo install react-native-safe-area-context

expo install @react-native-community/masked-view

The things we just installed are libraries that the navigation uses. All these modules are also mentioned in the documentation for stack navigation (refer to Teacher Activity 1 / Student Activity 1), and now we can install our navigation -

yarn add @react-navigation/native

yarn add @react-navigation/stack</td></tr>
<tr><td></td><td>Okay, now the first thing that we want to do is, we will create a StackNavigator and then create our navigation structure in our App.js file -</td><td></td></tr>
</table>

App.js

```
import 'react-native-gesture-handler';
import * as React from 'react';
import { NavigationContainer } from '@react-navigation/native';
import { createStackNavigator } from '@react-navigation/stack';

import Home from "./screens/Home";
import Main from "./screens/Main";

const Stack = createStackNavigator();

function App() {
 return (
   <NavigationContainer>
     <Stack.Navigator initialRouteName="Home" screenOptions={{
       headerShown: false
     }}>
       <Stack.Screen name="Home" component={Home} />
       <Stack.Screen name="Main" component={Main} />
     </Stack.Navigator>
   </NavigationContainer>
 );
}

export default App;
```

Here, we are first importing ***NavigationContainer & createStackNavigator***.

createStackNavigator helps us create a navigator where we can define our Screens and these screens should always be inside a Navigation Container.

When we were in the React Native module, we used navigation like Drawer Navigation and Tab Navigation. All these navigations require their screens to be inside a **<NavigationContainer>** component.
We had our **<Main>** component already completed in the previous class, which is our screen with the camera where the user can try out different frames, but we have also imported a new screen here called **<Home>**. This is what we will be building up today!

	For now, let's create a file called ***Home.js*** inside our ***screens*** folder and give the following boilerplate code to it -	

./screens/Home.js

```
import React, { Component } from 'react';
import { Text, View } from 'react-native';

export default class Home extends Component {
   render() {
       return (
           <View
               style={{
                   flex: 1,
                   justifyContent: "center",
                   alignItems: "center"
               }}>
               <Text>Home Screen!</Text>
           </View>
       )
   }
}
```

	Awesome! If we run the app now, we can see just **"Home Screen!"** written on the screen. That's because we specified in our Stack Navigator that we want our default screen (or the initialRouteName) to be Home.	

	As you might have guessed already, now it's going to be your turn to complete this app by building the home screen and adding the final navigation from Home screen to the Main Screen!	
Teacher Stops Screen Share		
	Now it's your turn. Please share your screen with me.	
• **Ask the student to press the ESC key to come back to the panel.** • **Guide the student to start screen share.** • **Teacher gets into fullscreen.**		
ACTIVITY • **Design the app UI.** • **Create the final screen and complete the App!**		
Step 3: Student-led Activity (20 mins)	***[Student Activity 1]*** ***Note**: The student will repeat teacher activity for adding stack navigation.* *Guide the student to achieve the same and share the code snippets.*	
	Discuss with the student how the UI should look like. *The final result would look like -*	

	Guide the student to create the header of the screen with App Name and App's Catch Phrase	

```
            <View style={styles.container}>
               <SafeAreaView style={styles.droidSafeArea} />
               <View style={styles.headingContainer}>
                   <View style={{ flexDirection: 'row', flexWrap: 'wrap' }}>
                       <Text style={styles.titleText1}>FR</Text><Text
style={styles.titleText2}>APP</Text>
                   </View>
                   <View style={{ flexDirection: 'row', flexWrap: 'wrap' }}>
                       <Text style={styles.subheading1}>Try Out</Text><Text
style={styles.subheading2}> Cool Frames</Text>
                   </View>
               </View>
            </View>
```

```
container: {
       flex: 1,
       backgroundColor: "#6278e4"
   },
   droidSafeArea: {
       marginTop: Platform.OS === "android" ? StatusBar.currentHeight : 0
   },
   headingContainer: {
       flex: 0.2,
       alignItems: 'center',
       justifyContent: 'center'
   },
   titleText1: {
       fontSize: RFValue(30),
       fontWeight: "bold",
       color: "#efb141",
       fontStyle: 'italic',
       textShadowColor: 'rgba(0, 0, 0, 0.75)',
       textShadowOffset: { width: -3, height: 3 },
       textShadowRadius: 1
   },
   titleText2: {
       fontSize: RFValue(30),
       fontWeight: "bold",
       color: "white",
```

```
        fontStyle: 'italic',
        textShadowColor: 'rgba(0, 0, 0, 0.75)',
        textShadowOffset: { width: -3, height: 3 },
        textShadowRadius: 1
    },
    subheading1: {
        fontSize: RFValue(20),
        color: "#efb141",
        fontStyle: 'italic',
        textShadowColor: 'rgba(0, 0, 0, 0.75)',
        textShadowOffset: { width: -3, height: 3 },
        textShadowRadius: 1
    },
    subheading2: {
        fontSize: RFValue(20),
        color: "white",
        fontStyle: 'italic',
        textShadowColor: 'rgba(0, 0, 0, 0.75)',
        textShadowOffset: { width: -3, height: 3 },
        textShadowRadius: 1
    },
```

Guide the student to create the second section, for text and images -

```
<View style={styles.contentContainer}>
                    <View style={{ flex: 0.5 }}>
                        <Text style={styles.contentText}>Experience the virtual
experience of trying out different frames from our wide collection on your mobile phone
before making any purchase, just how you would in an offline store!</Text>
                    </View>
                    <View style={{ flexDirection: "row", flex: 0.25 }}>
                        <View style={{ flex: 0.5 }}>
                            <Image source={require('../assets/Frapp-03.png')} style={{
height: 64, width: 160 }} />
                        </View>
                        <View style={{ flex: 0.5 }}>
```

```
                        <Image source={require('../assets/Frapp-09.png')} style={{
height: 64, width: 160 }} />
                    </View>
                </View>
                <View style={{ flexDirection: "row", flex: 0.25 }}>
                    <View style={{ flex: 0.5 }}>
                        <Image source={require('../assets/Frapp-02.png')} style={{
height: 64, width: 160 }} />
                    </View>
                    <View style={{ flex: 0.5 }}>
                        <Image source={require('../assets/Frapp-08.png')} style={{
height: 64, width: 160 }} />
                    </View>
                </View>
            </View>
```

```
contentContainer: {
       flex: 0.6,
       margin: RFValue(5),
       borderRadius: RFValue(15),
       backgroundColor: "white",
       height: "100%",
       padding: RFValue(20)
   },
   contentText: {
       fontSize: RFValue(17),
       fontStyle: 'italic',
       fontWeight: "bold"
   },
```

	Guide the student to create the final section with the button -	

```
<View style={styles.buttonContainer}>
                    <TouchableOpacity onPress={() =>
this.props.navigation.navigate("Main")}>
                        <View style={styles.button}>
                            <Text style={styles.buttonText}>Try Now!</Text>
                        </View>
                    </TouchableOpacity>
                </View>
```

```
buttonContainer: {
        flex: 0.2,
        justifyContent: "center",
        alignItems: "center"
    },
    button: {
        backgroundColor: "#efb141",
        paddingLeft: RFValue(50),
        paddingRight: RFValue(50),
        paddingTop: RFValue(20),
        paddingBottom: RFValue(20),
        borderRadius: RFValue(20)
    },
    buttonText: {
        fontSize: RFValue(25),
        fontStyle: 'italic',
        color: "white",
        textShadowColor: 'rgba(0, 0, 0, 0.75)',
        textShadowOffset: { width: -1, height: 1 },
        textShadowRadius: 1
    }
```

Don’t forget to add the navigation on the **onPress()** event of our **<TouchableOpacity/>**

	Guide the student to test the output.	

Teacher Guides Student to Stop Screen Share

FEEDBACK • **Compliment the student for her/his effort in the class.** • **Encourage the student to think and come up with their own solutions.**		
Step 4: **Wrap-Up** **(5 mins)**	Let's quickly wrap-up today's class. What did we learn?	**ESR**: • We learned to apply face filters using the data received from FaceDetector API expo module.
	Amazing work today! In the next class, we will be revising some of the concepts we had learned earlier for Virtual Reality, so we can create something completely out of our own imagination!	*The student listens.*
	You get a “hats-off”. Alright. See you in next class.	*Make sure you have given at least 2 Hats Off during the class for:* 

Project Overview	**NAME** **Goal of the Project:** **Story:** <Yet To Updated>	
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. ○ The 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 their reflections in a reflection journal.*

Activity	Activity Name	Links
Teacher Activity 1	Previous Class Code	https://github.com/whitehatjr/PRO-C184-Code-Ref
Teacher Activity 2	Stack Navigation	https://reactnavigation.org/docs/getting-started

Teacher Activity 3	Final Reference Code	https://github.com/whitehatjr/PRO-C185-Code-Ref
Student Activity 1	Previous Class Code	https://github.com/whitehatjr/PRO-C184-Code-Ref
Student Activity 2	Stack Navigation	https://reactnavigation.org/docs/getting-started