

Topic	FACE DETECTION	
Class Description	Students will learn about face detection to build a face filters app using React Native.	
Class	C181	
Class time	45 mins	
Goal	<ul style="list-style-type: none"> Learn about face detection. 	
Resources Required	<ul style="list-style-type: none"> Teacher Resources: <ul style="list-style-type: none"> Visual Studio Code Editor laptop with internet connectivity smartphone earphones with mic notebook and pen Student Resources: <ul style="list-style-type: none"> 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
WARM-UP SESSION - 5 mins		
<u>CONTEXT</u> <ul style="list-style-type: none"> Understand the face detection. 		



Teacher Starts Slideshow

Slide 1 to 3

Refer to speaker notes and follow the instructions on each slide.

Hey <student's name>. How are you? It's great to see you!
Are you excited to learn something new today?

Following are the WARM-UP session deliverables:

- Greet the student.
- Revision of previous class activities.

ESR: Hi, thanks!
Yes I am excited about it!

Click on the slide show tab
and present the slides

WARM-UP QUIZ

Click on In-Class Quiz



Continue WARM-UP Session

Slide 4 to 11

Following are the session deliverables:

- Appreciate the student.
- Narrate the story by using hand gestures and voice modulation methods to bring in more interest in students.

Class Steps	Teacher Action	Student Action
Step 1: Warm-Up (5 mins)	Hi, how are you? Great!	ESR: I am good!
	Any ideas on what we are going to learn today? Today we will begin to learn how to make an augmented reality (AR) face filter mobile app.	ESR: Varied!

	<p>Can you tell me what face filters are?</p> <p>Superb!</p> <p>Note: Encourage the student to discuss what they remember and help them to be more involved.</p> <p>Filters are also used for changing the background or frames in the pictures.</p> <p>Using filters in the pictures clicked is very popular these days.</p> <p>Filters are computer generated effects which are placed on the real world image.</p> <p>Since these filters/effects are added on the real life image after clicking by front/back camera, these are called Augmented reality filters or AR filters.</p> <p>You must have heard about various apps using face filters these days, right?</p> <p>These apps are designed in such a way where use can apply filters on your face and click selfies.</p>	<p>ESR: These are effects which are applied over the face while clicking a selfie.</p> <p>Note: The student discusses his/her views with the teacher.</p> <p>ESR: Yes!</p>
--	--	---

	<p>Can you tell me a few apps with these filters?</p> <p>Great!</p> <p>It's so fun to use those filters while taking a selfie, isn't it?</p> <p>Have you ever wondered what would be working behind the scenes to make these apps?</p> <p>What's the first thing that comes to your mind to be able to start building these mobile apps?</p> <p>Let's try to formulate the requirements:</p> <ul style="list-style-type: none"> • We would definitely need a way to open a smartphone camera. • Then we should know how to recognize the face of the person, right? • To be more specific, we should be able to detect facial features-eyes, nose and mouth head etc, to place filters over them. <p>How do you think facial features are detected?</p>	<p>ESR: Snapchat, Instagram.</p> <p>ESR: Yes!</p> <p>ESR: Varied.</p> <p>ESR: Varied.</p> <p>ESR: Yes.</p> <p>ESR: Varied.</p>
--	--	--

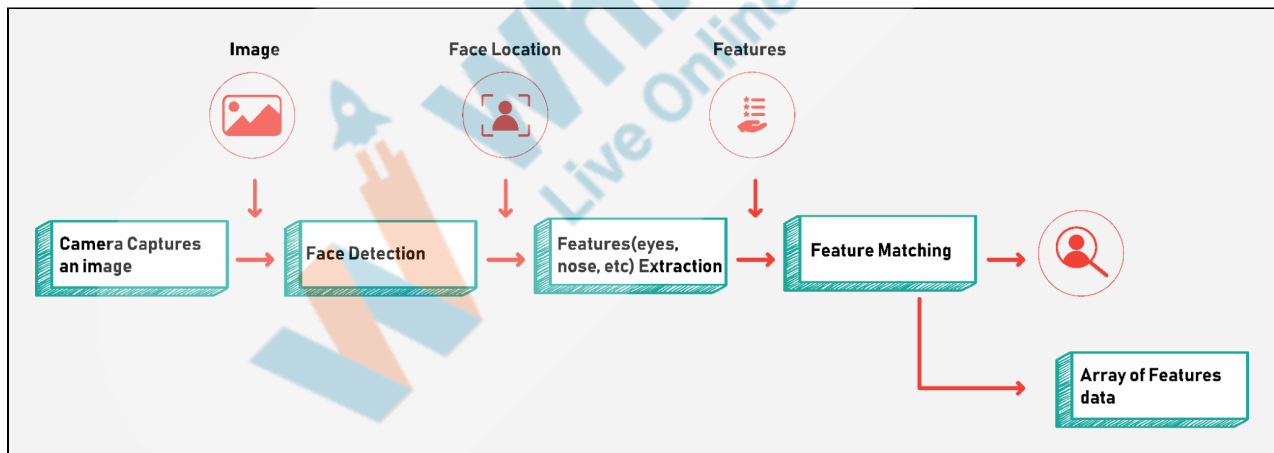
Note: Encourage the student to discuss what they remember and help them to be more involved.

Well, detecting a face requires a lot of algorithms to work together.

But let's understand it by dividing it into a few broad steps:


- Capturing face images.
- Face detection algorithm.
- Detecting facial features.
- Matching the face.
- Generate Faces ID and Array of face objects with other data related to faces.

Face Detection Image Link



We won't be deep diving into working on each step, but we can certainly use some predefined libraries/APIs to get this final data.

We will use expo React Native API to get this data.

<p>Are you excited?</p> <p>Let's get started then.</p>		ESR: Yes.
<p>Teacher Ends Slideshow </p>		
TEACHER-LED ACTIVITY - 15 mins		
Teacher Initiates Screen Share		
<p><u>CHALLENGE</u></p> <ul style="list-style-type: none"> Get Face Detection data using react native. 		
<p>Step 2: Teacher-led Activity (15 mins)</p>	<p>To begin with we will need to create the react app.</p> <p>We would call this the “Face Recognition App (FRAPP)”.</p> <p><i>Note: Encourage the student to help you with the steps to create the app as they have done the react native module.</i></p> <p>Set up the project using the following command:</p> <p>expo init frapp</p>	

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19042.928]
(c) Microsoft Corporation. All rights reserved.

D:\WhiteHatJr\Classes\Class 180>expo init frapp
```

```
C:\> npm
Microsoft Windows [Version 10.0.19042.928]
(c) Microsoft Corporation. All rights reserved.

D:\WhiteHatJr\Classes\Class 180>expo init frapp
✓ Choose a template: » blank          a minimal app as clean as an empty canvas
✓ Downloaded and extracted project files.
⌘ Using npm to install packages.
\ Installing JavaScript dependencies.
```

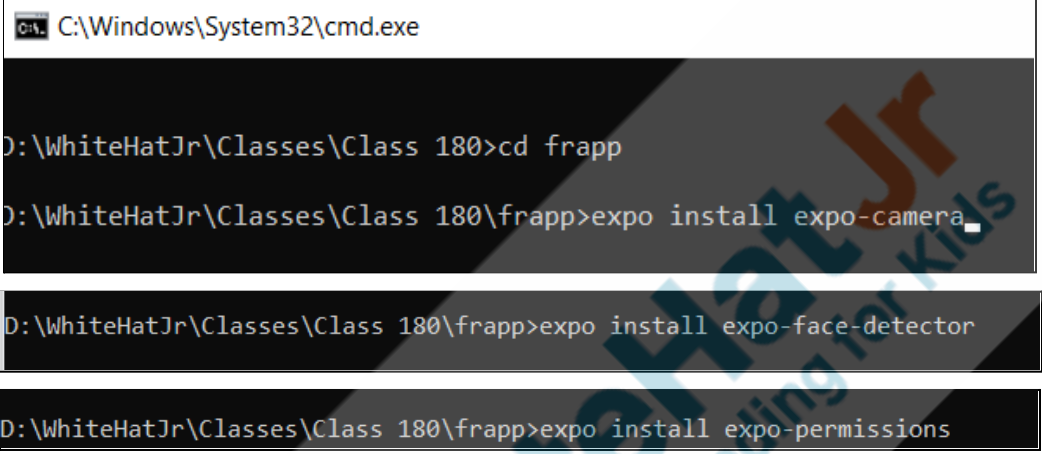
Next, we would want to use a camera and then recognize various elements of a user's face from it, like the position of the eyes, nose, etc.

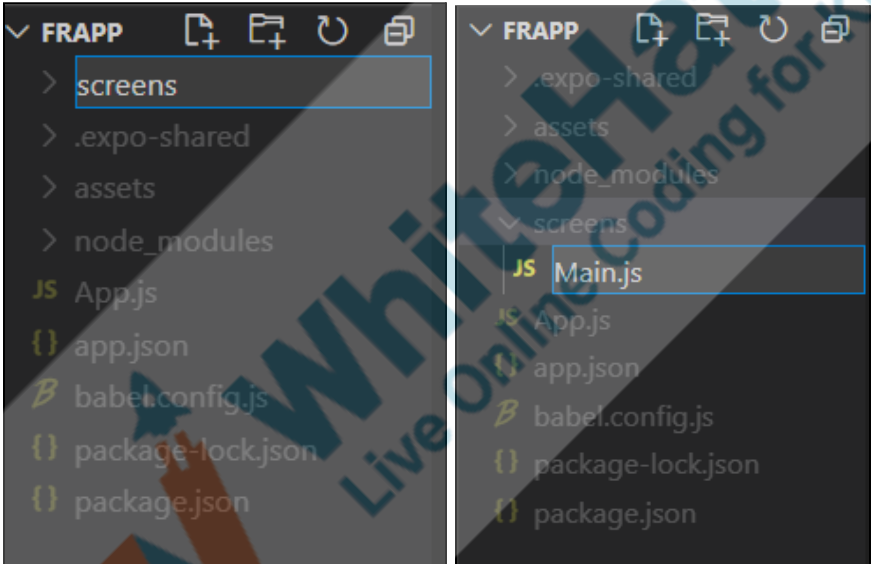
Through this, we will be able to position our filters better on their face.

For that, let's install the following dependencies:

```
expo install expo-camera
expo install expo-face-detector
expo install expo-permissions
```

We have installed a new dependency, which is our *expo-face-detector*.

	<p>FaceDetector is a module developed by Expo to help detect faces from the camera.</p> <p>Once it detects the faces, it gives us a lot of information about them.</p>	
		
	<p>Now let's structure the app. What do you think we should have for the app functionality?</p> <p>Yes, that's correct.</p> <p>We will have two screens in the app:</p> <ul style="list-style-type: none"> • One screen will be the camera screen where we will ask the user to use their camera so we can apply filters to their faces. This screen will also allow the users to click a selfie with the filter applied. • The second screen would be the preview of this image where we will provide the user with a 	<p>ESR: We need a screen to open the camera with which we detect the face.</p>

	<p>share button, so they can share this image at different social media platforms.</p> <p>Let's start by building the basic version of the first screen.</p> <p>We will create a new folder called screens in our root directory and then create a file Main.js inside this folder:</p>	
		
	<p>Once we install the dependencies, what's the next step?</p> <p>Yes. Can you help me to import the dependencies?</p> <p>Note: The importing dependencies have been covered in the react module. Help the student to recollect the dependencies they have used.</p>	<p>ESR: We need to import these dependencies.</p> <p>Note: The student helps the teacher to import the dependencies required.</p>

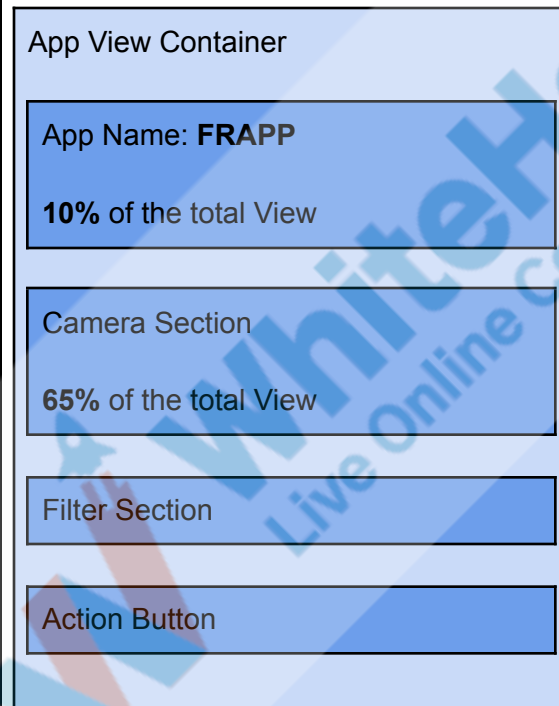
	<p>SafeAreaView is used to render content within the safe area boundaries of a device.</p>	
<pre> JS Main.js X screens > JS Main.js 1 import React from 'react'; 2 import { StyleSheet, Text, View, SafeAreaView, StatusBar, Platform } from 'react-native'; 3 import { Camera } from 'expo-camera'; 4 import * as Permissions from "expo-permissions"; 5 import * as FaceDetector from 'expo-face-detector'; </pre>		
	<p>Now we would write the “Main” class component and import it in the App.js.</p> <p>Note: Update App.js to import <Main/>.</p>	
<pre> export default class Main extends React.Component { } import React from 'react'; import Main from "../screens/Main"; export default function App() { return (<Main />) } </pre>		
	<p>Now we would have to decide how the container of our first screen will look.</p>	

	<p>The main container should cover the whole screen.</p> <p>For this we will set the flex property in the styles to 1.</p>	
<pre>const styles = StyleSheet.create({ container: { flex: 1 } }); export default class Main extends React.Component { render() { return (<View style={styles.container}> </View>) } }</pre>		
	<p>The first screen is where we want to use a camera so that we can apply filters to their faces. And this screen will also allow the users to click a selfie with the filter applied.</p> <p>So considering this, what all components can we use on the first screen?</p> <p>Great!</p>	<p>ESR: We can have:</p> <ul style="list-style-type: none"> • A <View> for the camera component. • And a view for the capture button.

And we will also need a section to filter options to choose filters.

We should also have one view component to add an app name at the top.

We can summarise the container view as:



```
const styles = StyleSheet.create({
  container: {
    flex: 1
  },
  droidSafeArea: {
    marginTop: Platform.OS === "android" ? StatusBar.currentHeight : 0
  },
  headingContainer: {
    flex: 0.1,
    alignItems: 'center',
    justifyContent: 'center'
  },
  titleText: {
    fontSize: 30
  },
  cameraStyle: {
    flex: 0.65
  },
  filterContainer: {},
  actionContainer: {}
});
```

```

return (
  <View style={styles.container}>
    <SafeAreaView style={styles.droidSafeArea} />
    App Name <View style={styles.headingContainer}>
      <Text style={styles.titleText}>FRAPP</Text>
    </View>
    Camera Section <View style={styles.cameraStyle}>
      <Camera />
    </View>
    Filters Section <View style={styles.filterContainer}>
    </View>
    Action or Capture <View style={styles.actionContainer}>
    </View>
  </View>
)

```

We are going to use the **<Camera/>** component to set the **FaceDetector** properties which will return “**faces**” array.

We can set the type of camera that we want to use.

type: type of the camera—front/back

To set the **FaceDetector** object properties:

- **mode**
(FaceDetector.Constants.Mode)

	<p>: Whether to detect faces in fast or accurate mode.</p> <ul style="list-style-type: none"> • detectLandmarks(FaceDetector.Constants.Landmarks): Whether to detect and return landmark positions on the face (ears, eyes, mouth, cheeks, nose). Valid values: all, none. • runClassifications(FaceDetector.Constants.Classifications): Whether to run additional classifications on detected faces (smiling probability, open eye probabilities). Valid values: all, none. 	
<pre><View style={styles.cameraStyle}> <Camera style={{ flex: 1 }} type={Camera.Constants.Type.front} faceDetectorSettings={{ mode: FaceDetector.Constants.Mode.fast, detectLandmarks: FaceDetector.Constants.Landmarks.all, runClassifications: FaceDetector.Constants.Classifications.all }} /> </View></pre>		
	<p>Once we have set the FaceDetector properties, we will have camera permissions for the app.</p> <p>Also we will need to get the “faces” array from the FaceDetector object.</p>	

	<p>For this we'll set the state:</p> <pre>{ hasCameraPermission: null faces: [] }</pre> <p>hasCameraPermission: This state property will be used to update the camera permission of the app. It will be set to 'granted' once the permissions are given.</p> <p>faces: This state property will be updated once the face/faces are detected in the camera view.</p> <p>We can use the setState() method to update the state value.</p> <p>We will write these functions:</p> <p>componentDidMount(): To ask(alert) for the camera permission after the component is mounted.</p> <p>onCameraPermission(): Set the hasCameraPermission value to 'granted'.</p> <p>onFacesDetected(): Set the faces array.</p> <p>onFaceDetectionError(): Display error message using console.log() method.</p>	
--	---	--


```

constructor(props) {
  super(props)
  this.state = {
    hasCameraPermission: null,
    faces: []
  }
}

componentDidMount() {
  Permissions
    .askAsync(Permissions.CAMERA)
    .then(this.onCameraPermission)
}

onCameraPermission = (status) => {
  this.setState({ hasCameraPermission: status.status === 'granted' })
}

onFacesDetected = (faces) => {
  this.setState({ faces: faces })
}

onFaceDetectionError = (error) => {
  console.log(error)
}

```



Now inside the `render()` method, we will check if the **hasCameraPermission** is null, then return an empty view.

If the permission is denied, set the `<Text></Text>` view component, "No access to camera".

Call the **onFacesDetected()** and **onFaceDetectionError()** method in the `<Camera/>`.

	And display the values of the faces array to get the data of the face/faces detected.	
<pre> render() { const { hasCameraPermission } = this.state; if (hasCameraPermission === null) { return <View /> } if (hasCameraPermission === false) { return (<View style={styles.container}> <Text>No access to camera</Text> </View>) } console.log(this.state.faces) </pre> <pre> <Camera style={{ flex: 1 }} type={Camera.Constants.Type.front} faceDetectorSettings={{ mode: FaceDetector.Constants.Mode.fast, detectLandmarks: FaceDetector.Constants.Landmarks.all, runClassifications: FaceDetector.Constants.Classifications.all }} onFacesDetected={this.onFacesDetected} onFacesDetectionError={this.onFacesDetectionError} /> </pre>		
	Now let's test the output using expo.	

```
Finished building JavaScript bundle in 25ms.
Running application on Redmi Note 5 Pro.
Array []
Object {
  "faces": Array [
    Object {
      "bottomMouthPosition": Object {
        "x": 179.14767733487213,
        "y": 393.75616267811165,
      },
      "bounds": Object {
        "origin": Object {
          "x": 49.636363636363626,
          "y": 170.0248106060606,
        },
        "size": Object {
          "height": 272.81363636363636,
          "width": 301.09090909090907,
        },
      },
      "faceID": -1,
      "leftCheekPosition": Object {
        "x": 261.3562150435014,
        "y": 353.27145982222123,
      },
      "leftEarPosition": Object {
        "x": 328.16231606223363,
        "y": 343.1396335255016,
      },
      "leftEyeOpenProbability": 0.9871754050254822,
      "leftEyePosition": Object {
        "x": 254.82401899857953,
        "y": 285.08886475996536,
      },
      "leftMouthPosition": Object {
        "x": 231.05218505859372,
        "y": 385.97691208810517,
      },
      "noseBasePosition": Object {
        "x": 204.5574729225852,
        "y": 315.6004593589089,
      },
      "rightCheekPosition": Object {
        "x": 119.24881813742894,
        "y": 326.1953313654119,
      },
      "rightEarPosition": Object {
        "x": 82.58466131036931,
```

	<p>That's really amazing!</p> <p>We were able to detect faces and get the data. Now you will have to create an app to get the facial data.</p> <p>Are you excited?</p>	<p>ESR: Yes!</p>
Teacher Stops Screen Share		
	<p>Now it's your turn. Please share your screen with me.</p>	
<p>Teacher Starts Slideshow </p> <p>Slide 12 to 13</p> <p>Refer to speaker notes and follow the instructions on each slide.</p>		
	<p>We have one more class challenge for you. Can you solve it?</p> <p>Let's try. I will guide you through it.</p>	
<p>Teacher Ends Slideshow </p>		
STUDENT-LED ACTIVITY - 20 mins		
<ul style="list-style-type: none"> • 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. 		
<p><u>ACTIVITY</u></p> <ul style="list-style-type: none"> • Get Face Detection Data using expo react. 		

Step 3: Student-led Activity (20 mins)	<p><i>The teacher guides the student to create a new react native app project.</i></p> <p>Note: <i>The student will repeat teacher activity.</i></p>	
	<p><i>Guide the student to create and set up the react project.</i></p> <ul style="list-style-type: none"> • <i>Create Main.js file in screens folder.</i> • <i>Import the <Main/> component in App.js.</i> 	
<pre>import React from 'react'; import { StyleSheet, Text, View, SafeAreaView, StatusBar, Platform } from 'react-native'; import { Camera } from 'expo-camera'; import * as Permissions from "expo-permissions"; import * as FaceDetector from 'expo-face-detector';</pre>		
	<p><i>Guide the student to set state values.</i></p>	

```
constructor(props) {  
  super(props)  
  this.state = {  
    hasCameraPermission: null,  
    faces: []  
  }  
}  
  
componentDidMount() {  
  Permissions  
    .askAsync(Permissions.CAMERA)  
    .then(this.onCameraPermission)  
}  
  
onCameraPermission = (status) => {  
  this.setState({ hasCameraPermission: status.status === 'granted' })  
}  
  
onFacesDetected = (faces) => {  
  this.setState({ faces: faces })  
}  
  
onFaceDetectionError = (error) => {  
  console.log(error)  
}
```

*Guide the student to write the **Main** class component.*

```
render() {
  const { hasCameraPermission } = this.state;
  if (hasCameraPermission === null) {
    return <View />
  }
  if (hasCameraPermission === false) {
    return (
      <View style={styles.container}>
        <Text>No access to camera</Text>
      </View>
    )
  }
  console.log(this.state.faces)
  return (
    <View style={styles.container}>
      <SafeAreaView style={styles.droidSafeArea} />
      <View style={styles.headingContainer}>
        <Text style={styles.titleText}>FRAPP</Text>
      </View>
      <View style={styles.cameraStyle}>
        <Camera
          style={{ flex: 1 }}
          type={Camera.Constants.Type.front}
          faceDetectorSettings={{
            mode: FaceDetector.Constants.Mode.fast,
            detectLandmarks: FaceDetector.Constants.Landmarks.all,
            runClassifications: FaceDetector.Constants.Classifications.all
          }}
          onFacesDetected={this.onFacesDetected}
          onFacesDetectionError={this.onFacesDetectionError}
        />
      </View>
      <View style={styles.filterContainer}>
      </View>
      <View style={styles.actionContainer}>
      </View>
    </View>
  )
}
```

```
const styles = StyleSheet.create({
  container: {
    flex: 1
  },
  droidSafeArea: {
    marginTop: Platform.OS === "android" ? StatusBar.currentHeight : 0
  },
  headingContainer: {
    flex: 0.1,
    alignItems: 'center',
    justifyContent: 'center'
  },
  titleText: {
    fontSize: 30
  },
  cameraStyle: {
    flex: 0.65
  },
  filterContainer: {},
  actionContainer: {}
});
```

Guide the student to test the output.


```
Finished building JavaScript bundle in 25ms.
Running application on Redmi Note 5 Pro.
Array []
Object {
  "faces": Array [
    Object {
      "bottomMouthPosition": Object {
        "x": 179.14767733487213,
        "y": 393.75616267811165,
      },
      "bounds": Object {
        "origin": Object {
          "x": 49.636363636363626,
          "y": 170.0248106060606,
        },
        "size": Object {
          "height": 272.81363636363636,
          "width": 301.09090909090907,
        },
      },
      "faceID": -1,
      "leftCheekPosition": Object {
        "x": 261.3562150435014,
        "y": 353.27145982222123,
      },
      "leftEarPosition": Object {
        "x": 328.16231606223363,
        "y": 343.1396335255016,
      },
      "leftEyeOpenProbability": 0.9871754050254822,
      "leftEyePosition": Object {
        "x": 254.82401899857953,
        "y": 285.08886475996536,
      },
      "leftMouthPosition": Object {
        "x": 231.05218505859372,
        "y": 385.97691208810517,
      },
      "noseBasePosition": Object {
        "x": 204.5574729225852,
        "y": 315.6004593589089,
      },
      "rightCheekPosition": Object {
        "x": 119.24881813742894,
        "y": 326.1953313654119,
      },
      "rightEarPosition": Object {
        "x": 82.58466131036931,
```

Teacher Guides Student to Stop Screen Share

WRAP UP SESSION - 5 mins

© 2020 - WhiteHat Education Technology Private Limited.

Note: This document is the original copyright of WhiteHat Education Technology Private Limited.

Please don't share, download or copy this file without permission.



Teacher Starts Slideshow
Slide 14 to 18

Activity details

Following are the WRAP-UP session deliverables:

- Appreciate the student.
- Revise the current class activities.
- Discuss the quizzes.

WRAP-UP QUIZ
Click on In-Class Quiz



Continue WRAP-UP Session
Slide 19 to 24

Activity Details




Following are the session deliverables:

- Explain the facts and trivia
- Next class challenge
- Project for the day
- Additional Activity (Optional)

FEEDBACK

- **Appreciate and compliment the student for trying to learn a difficult concept.**
- **Get to know how they are feeling after the session.**
- **Review and check their understanding.**

Teacher Action	Student Action
You get Hats off for your excellent work!	<i>Make sure you have given at least 2 Hats Off during the class for:</i>

		<div>Creatively Solved Activities  +10</div> <div>Great Question  +10</div> <div>Strong Concentration  +10</div>
<p align="center">PROJECT OVERVIEW DISCUSSION</p> <p align="center">Refer the document below in Activity Links Sections</p>		
<p align="center">Teacher Clicks</p>		<div>✕ End Class</div>
<p>Additional Activities</p>	<p><i>Encourage the student to write reflection notes in their reflection journal using markdown.</i></p> <p>Use these as guiding questions:</p> <ul style="list-style-type: none"> • What happened today? <ul style="list-style-type: none"> ○ 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? 	<p><i>The student uses the markdown editor to write their reflections in a reflection journal.</i></p>

Activity	Activity Name	Links
Teacher Activity 1	Face Detection Image Link	https://s3-whjr-v2-prod-bucket.whjr.online/03a994cd-0f62-45df-b8e3-1d9b59767f1f.png
Teacher Activity 2	Expo FaceDetector	https://docs.expo.io/versions/latest/sdk/facedetector/
Teacher Activity 3	Final Reference Code	https://github.com/whitehatjr/PRO-C181-Code-Ref
Student Activity 1	Expo FaceDetector	https://docs.expo.io/versions/latest/sdk/facedetector/
Teacher Reference 1	Project Document	https://s3-whjr-curriculum-uploads.whjr.online/dfd3bc32-7472-4df6-afe7-c1bdb6c3d060.pdf
Teacher Reference 2	Project Solution	https://github.com/whitehatjr/AR-PRO-C181
Teacher Reference 3	Visual-Aid	https://s3-whjr-curriculum-uploads.whjr.online/dac54939-a4c9-4329-94e0-479c0f946bcd.html
Teacher Reference 4	In-Class Quiz	https://s3-whjr-curriculum-uploads.whjr.online/3c420b42-dd7b-4b81-b76b-58224a7b60fc.pdf