University of Engineering and Technology, Taxila

MAD

QUIZ 2

Submitted to:

Dr Kanwal Yousaf

Submitted by:

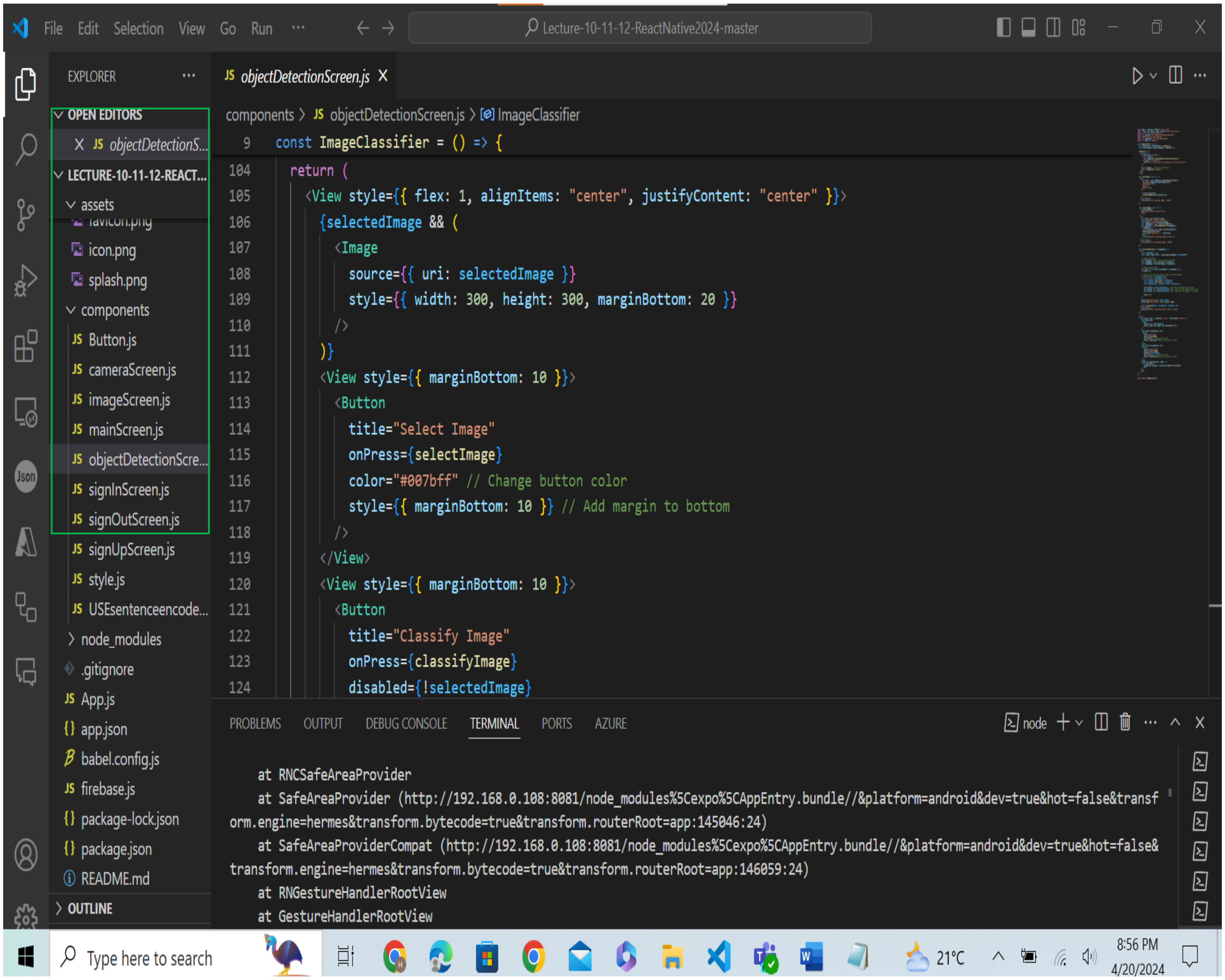
Hifza Eman

21-SE-15

Dated:

21/4/24

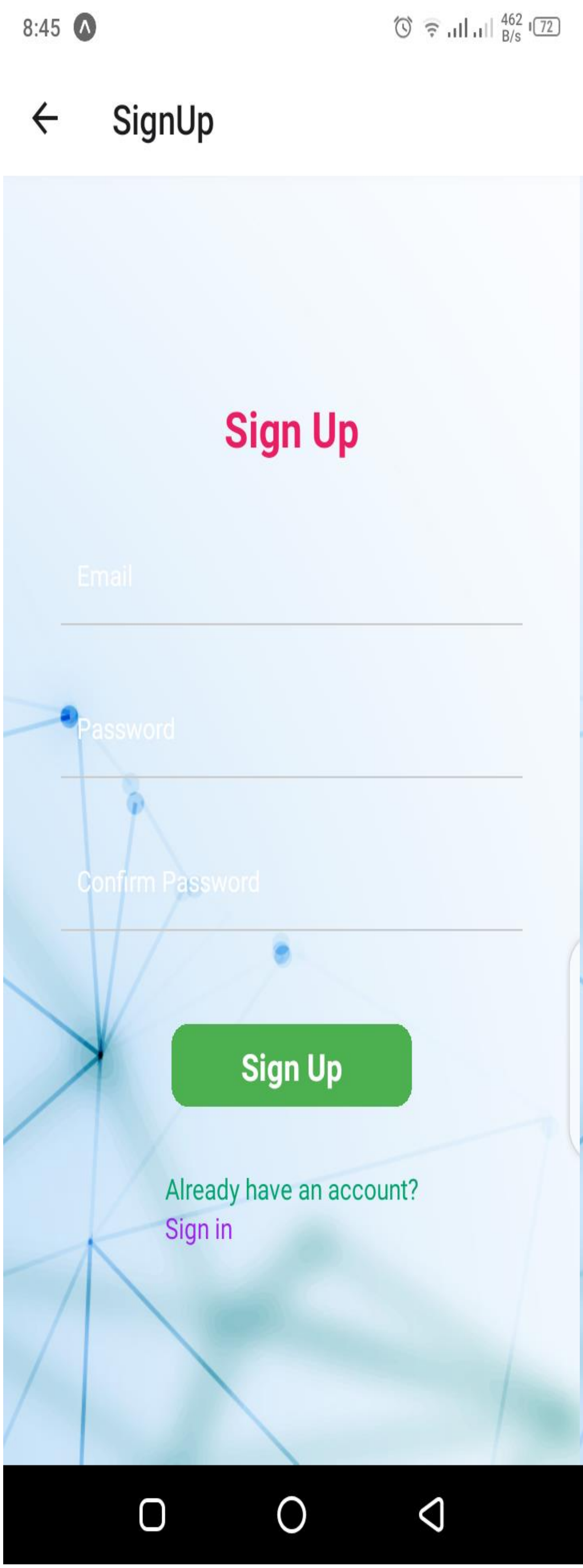
1. Code Reuse and Cleanup:

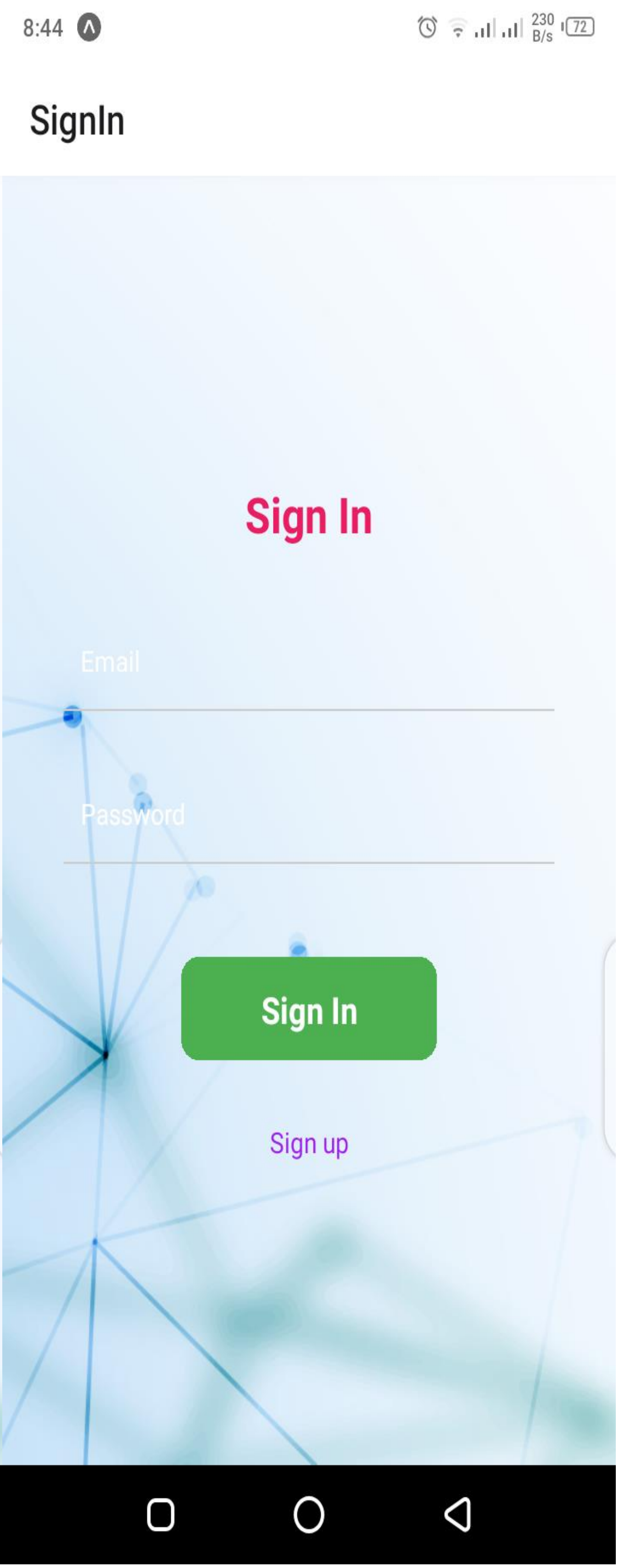


```
const ImageClassifier = () => {
  return (
    <View style={{ flex: 1, alignItems: "center", justifyContent: "center" }}>
      {selectedImage && (
        <Image
          source={{ uri: selectedImage }}
          style={{ width: 300, height: 300, marginBottom: 20 }}
        />
      )}
      <View style={{ marginBottom: 10 }}>
        <Button
          title="Select Image"
          onPress={selectImage}
          color="#007bff" // Change button color
          style={{ marginBottom: 10 }} // Add margin to bottom
        />
      </View>
      <View style={{ marginBottom: 10 }}>
        <Button
          title="Classify Image"
          onPress={classifyImage}
          disabled={!selectedImage}
```

```
    at RNCSafeAreaProvider
    at SafeAreaProvider (http://192.168.0.108:8081/node_modules%5Cexpo%5CAppEntry.bundle//&platform=android&dev=true&hot=false&transf
orm.engine=hermes&transform.bytecode=true&transform.routerRoot=app:145046:24)
    at SafeAreaProviderCompat (http://192.168.0.108:8081/node_modules%5Cexpo%5CAppEntry.bundle//&platform=android&dev=true&hot=false&
transform.engine=hermes&transform.bytecode=true&transform.routerRoot=app:146059:24)
    at RNGestureHandlerRootView
    at GestureHandlerRootView
```

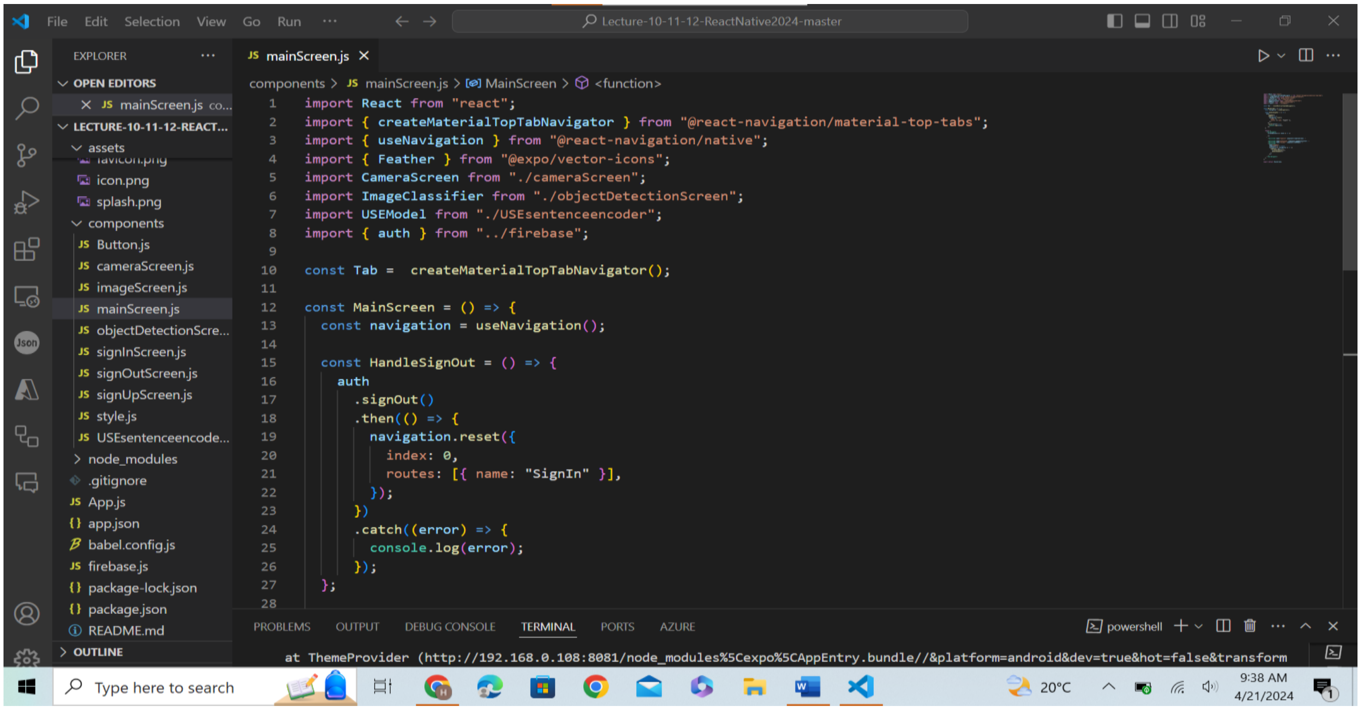
2. UI Enhancement:


8:45
SignUp
Sign Up
Email
Password
Confirm Password
Sign Up
Already have an account?
Sign in


8:44
SignIn
Sign In
Email
Password
Sign In
Sign up

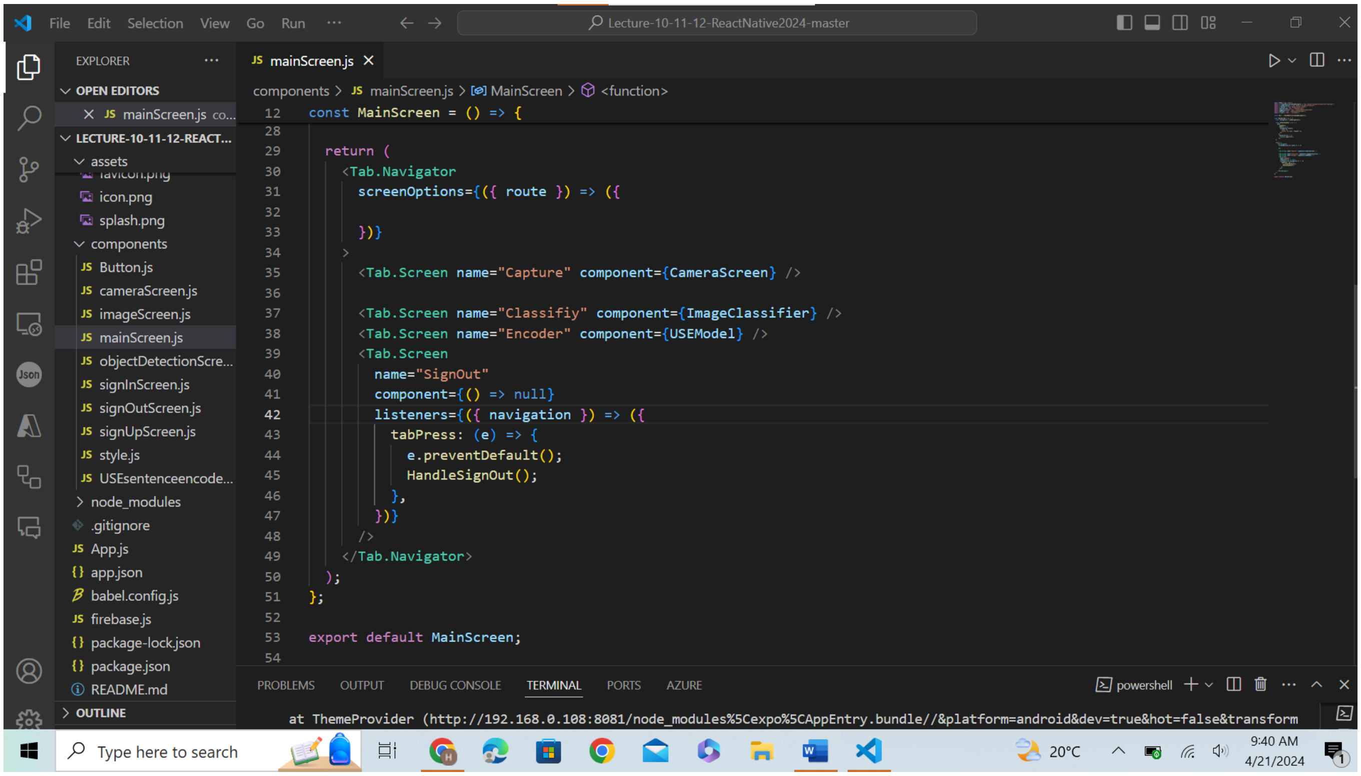
3. Menu Integration:

mainScreen.js



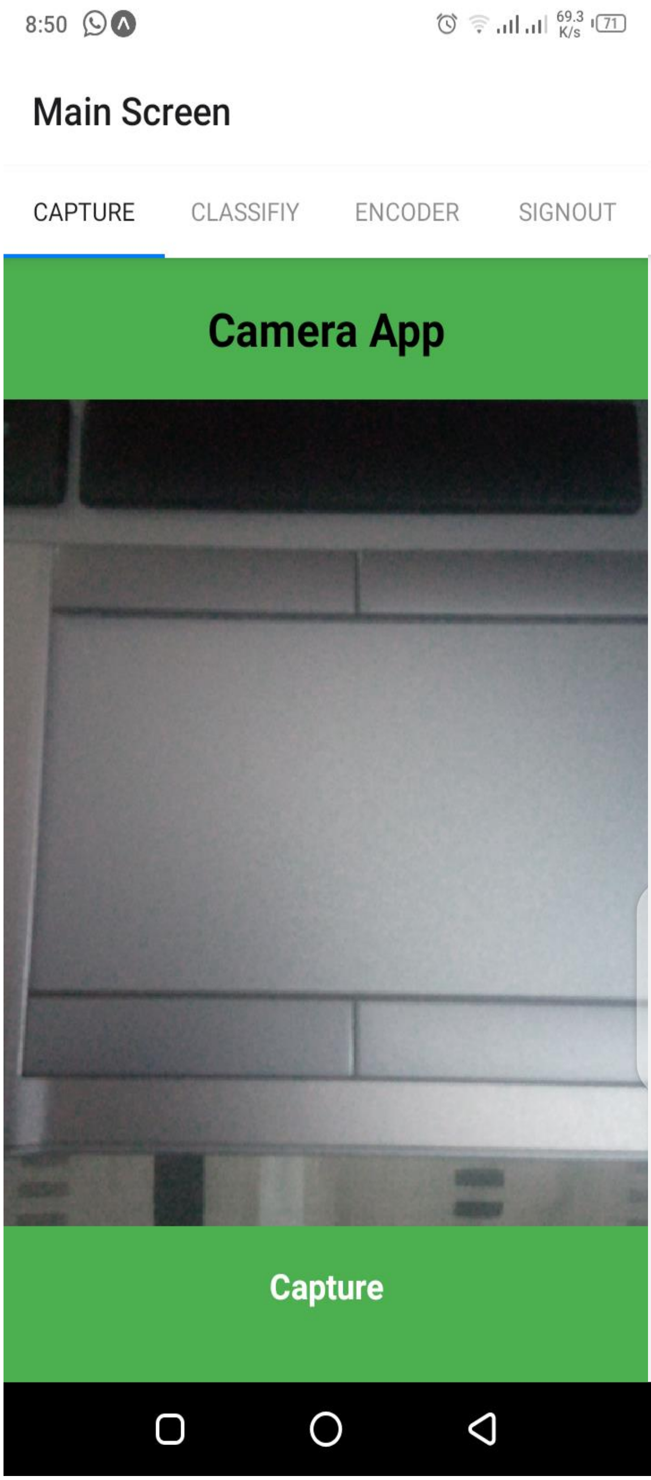
```
import React from "react";
import { createMaterialTopTabNavigator } from "@react-navigation/material-top-tabs";
import { useNavigation } from "@react-navigation/native";
import { Feather } from "@expo/vector-icons";
import CameraScreen from "./cameraScreen";
import ImageClassifier from "./objectDetectionScreen";
import USEModel from "./USEsentenceencoder";
import { auth } from "../firebase";

const Tab =  createMaterialTopTabNavigator();

const MainScreen = () => {
  const navigation = useNavigation();

  const HandleSignOut = () => {
    auth
      .signOut()
      .then(() => {
        navigation.reset({
          index: 0,
          routes: [{ name: "SignIn" }],
        });
      })
      .catch((error) => {
        console.log(error);
      });
  };

```

```
at ThemeProvider (http://192.168.0.108:8081/node_modules%5Cexpo%5CAppEntry.bundle//&platform=android&dev=true&hot=false&transform
```

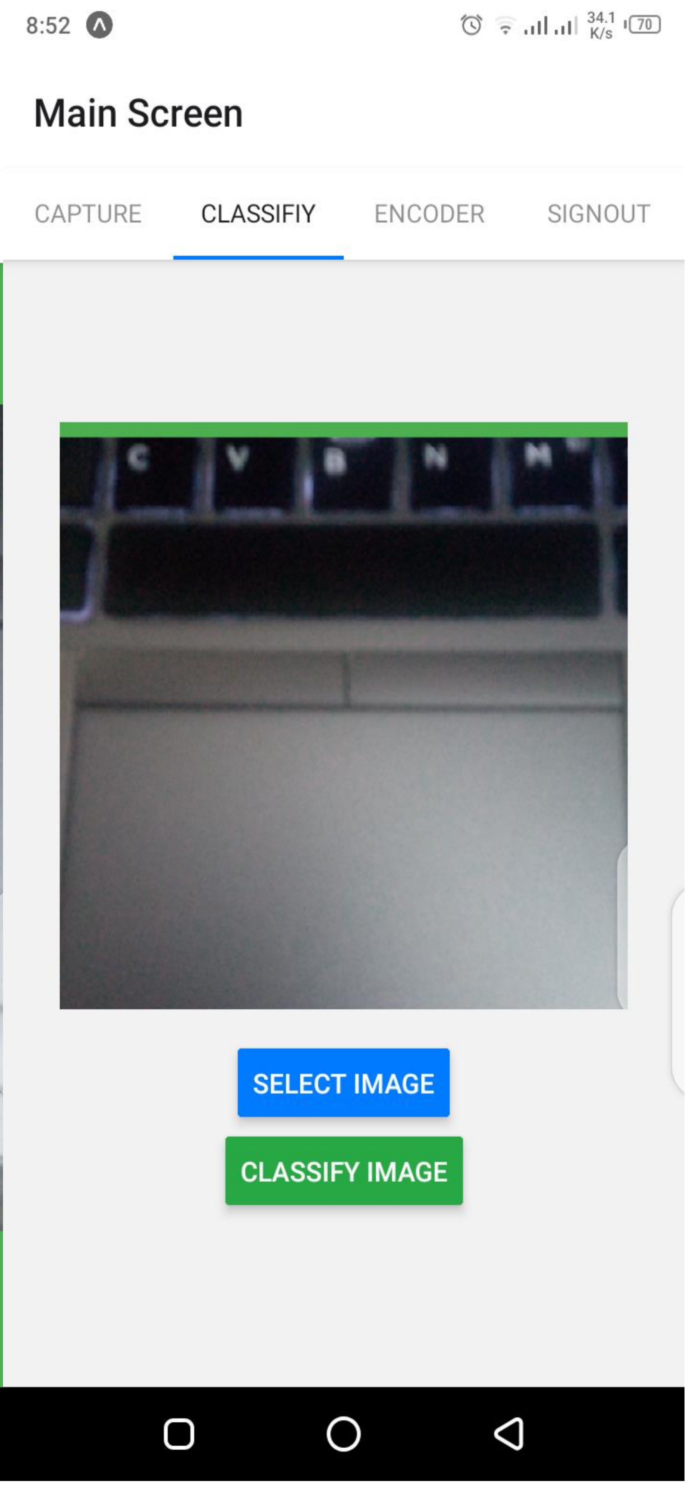


```
const MainScreen = () => {

  return (
    <Tab.Navigator
      screenOptions={({ route }) => ({

      })}
    >
      <Tab.Screen name="Capture" component={CameraScreen} />

      <Tab.Screen name="Classifiy" component={ImageClassifier} />
      <Tab.Screen name="Encoder" component={USEModel} />
      <Tab.Screen
        name="SignOut"
        component={() => null}
        listeners={({ navigation }) => ({
          tabPress: (e) => {
            e.preventDefault();
            HandleSignOut();
          },
        })}
      />
    </Tab.Navigator>
  );
};

export default MainScreen;

```

```
at ThemeProvider (http://192.168.0.108:8081/node_modules%5Cexpo%5CAppEntry.bundle//&platform=android&dev=true&hot=false&transform
```



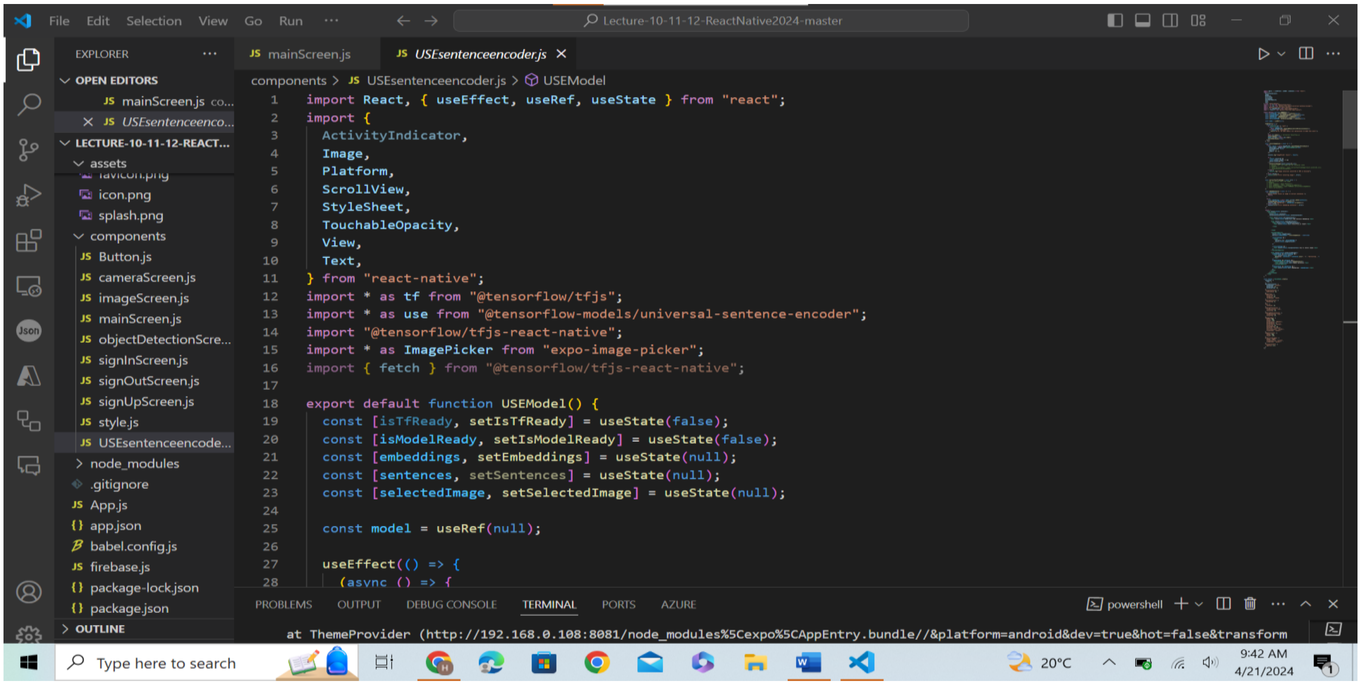
8:50
69.3 K/s
71
Main Screen
CAPTURE
CLASSIFIY
ENCODER
SIGNOUT
Camera App
Capture



8:52
34.1 K/s
70
Main Screen
CAPTURE
CLASSIFIY
ENCODER
SIGNOUT
SELECT IMAGE
CLASSIFY IMAGE

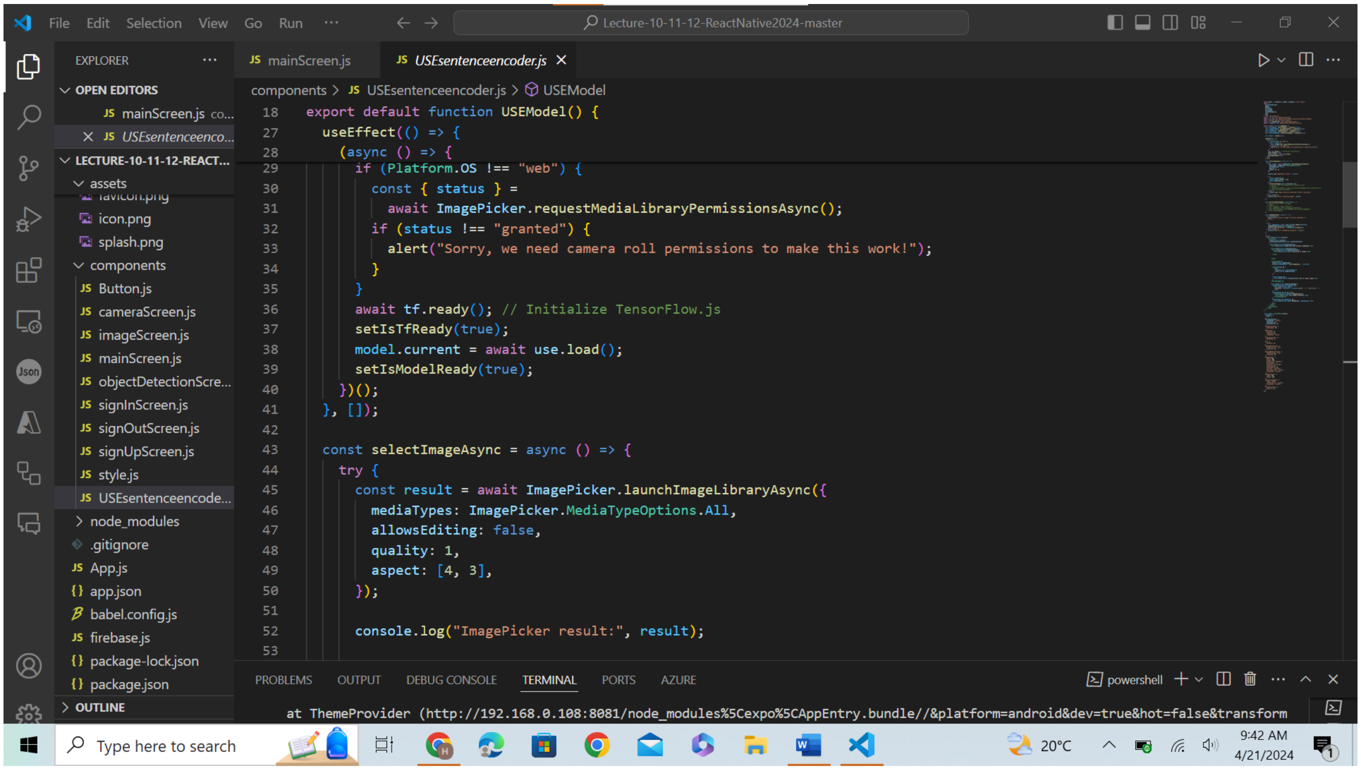
4. AI Model Integration:

USEsentenceencoder.js



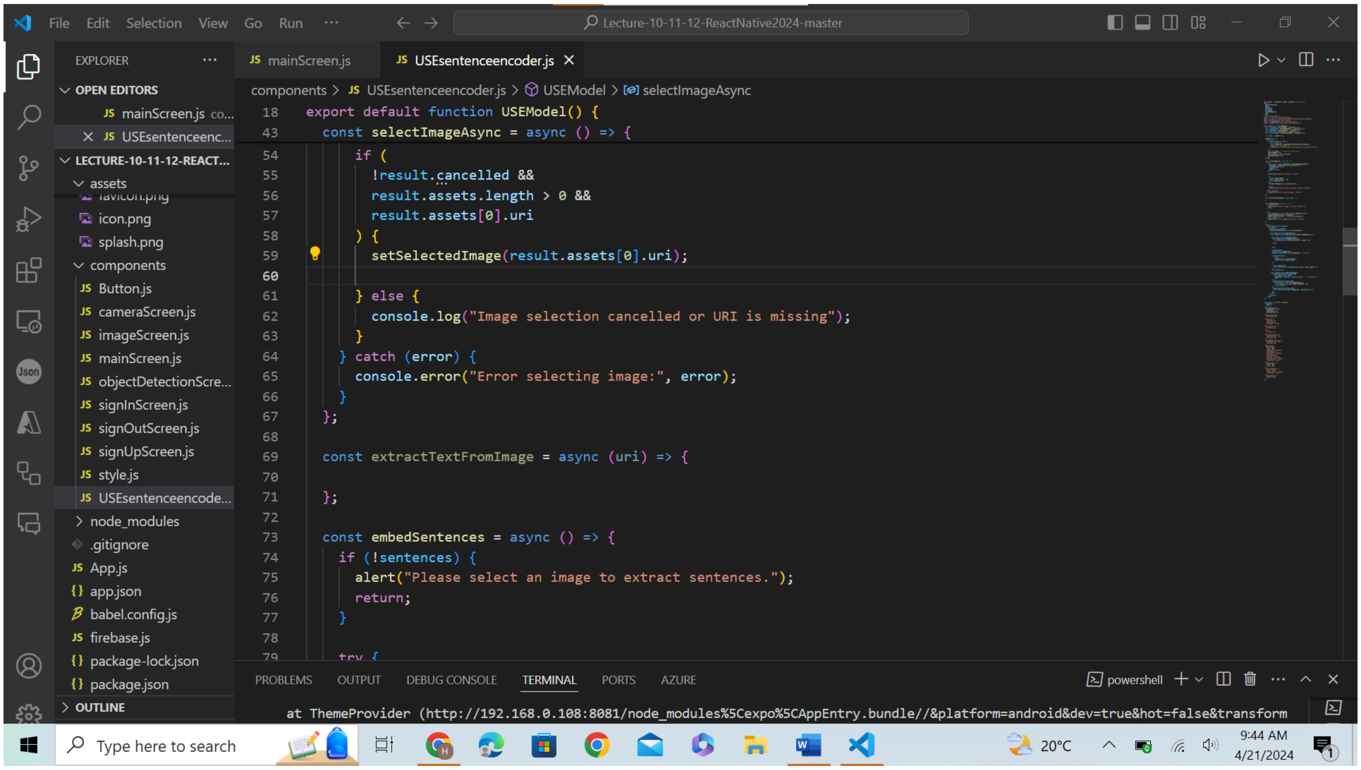
```javascript
import React, { useEffect, useRef, useState } from "react";
import {
  ActivityIndicator,
  Image,
  Platform,
  ScrollView,
  StyleSheet,
  TouchableOpacity,
  View,
  Text,
} from "react-native";
import * as tf from "@tensorflow/tfjs";
import * as use from "@tensorflow-models/universal-sentence-encoder";
import "@tensorflow/tfjs-react-native";
import * as ImagePicker from "expo-image-picker";
import { fetch } from "@tensorflow/tfjs-react-native";

export default function USEModel() {
  const [isTfReady, setIsTfReady] = useState(false);
  const [isModelReady, setIsModelReady] = useState(false);
  const [embeddings, setEmbeddings] = useState(null);
  const [sentences, setSentences] = useState(null);
  const [selectedImage, setSelectedImage] = useState(null);

  const model = useRef(null);

  useEffect(() => {
    (async () => {
```

```
at ThemeProvider (http://192.168.0.108:8081/node_modules%5Cexpo%5CAppEntry.bundle//&platform=android&dev=true&hot=false&transform
```



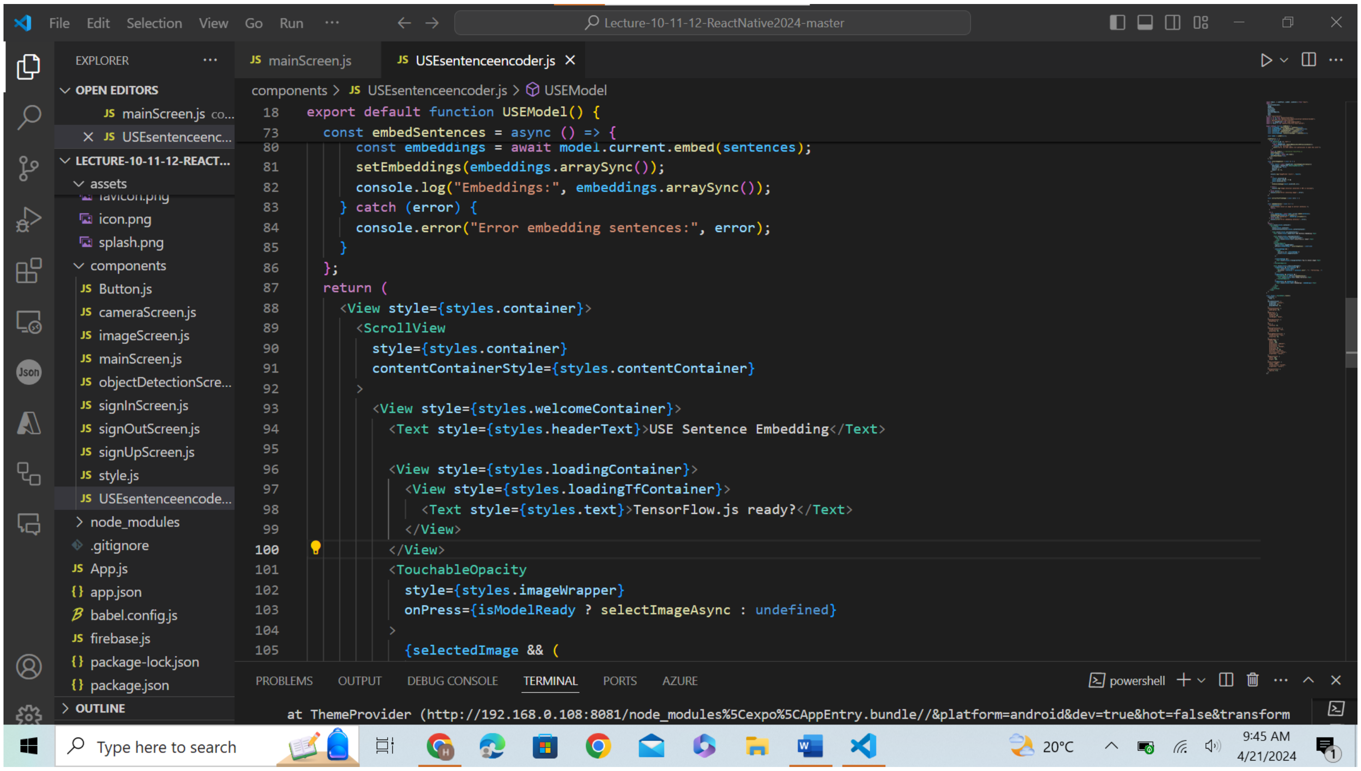
```javascript
export default function USEModel() {
  useEffect(() => {
    (async () => {
      if (Platform.OS !== "web") {
        const { status } =
          await ImagePicker.requestMediaLibraryPermissionsAsync();
        if (status !== "granted") {
          alert("Sorry, we need camera roll permissions to make this work!");
        }
      }
      await tf.ready(); // Initialize TensorFlow.js
      setIsTfReady(true);
      model.current = await use.load();
      setIsModelReady(true);
    })();
  }, []);

  const selectImageAsync = async () => {
    try {
      const result = await ImagePicker.launchImageLibraryAsync({
        mediaTypes: ImagePicker.MediaTypeOptions.All,
        allowsEditing: false,
        quality: 1,
        aspect: [4, 3],
      });

      console.log("ImagePicker result:", result);

```

```
at ThemeProvider (http://192.168.0.108:8081/node_modules%5Cexpo%5CAppEntry.bundle//&platform=android&dev=true&hot=false&transform
```



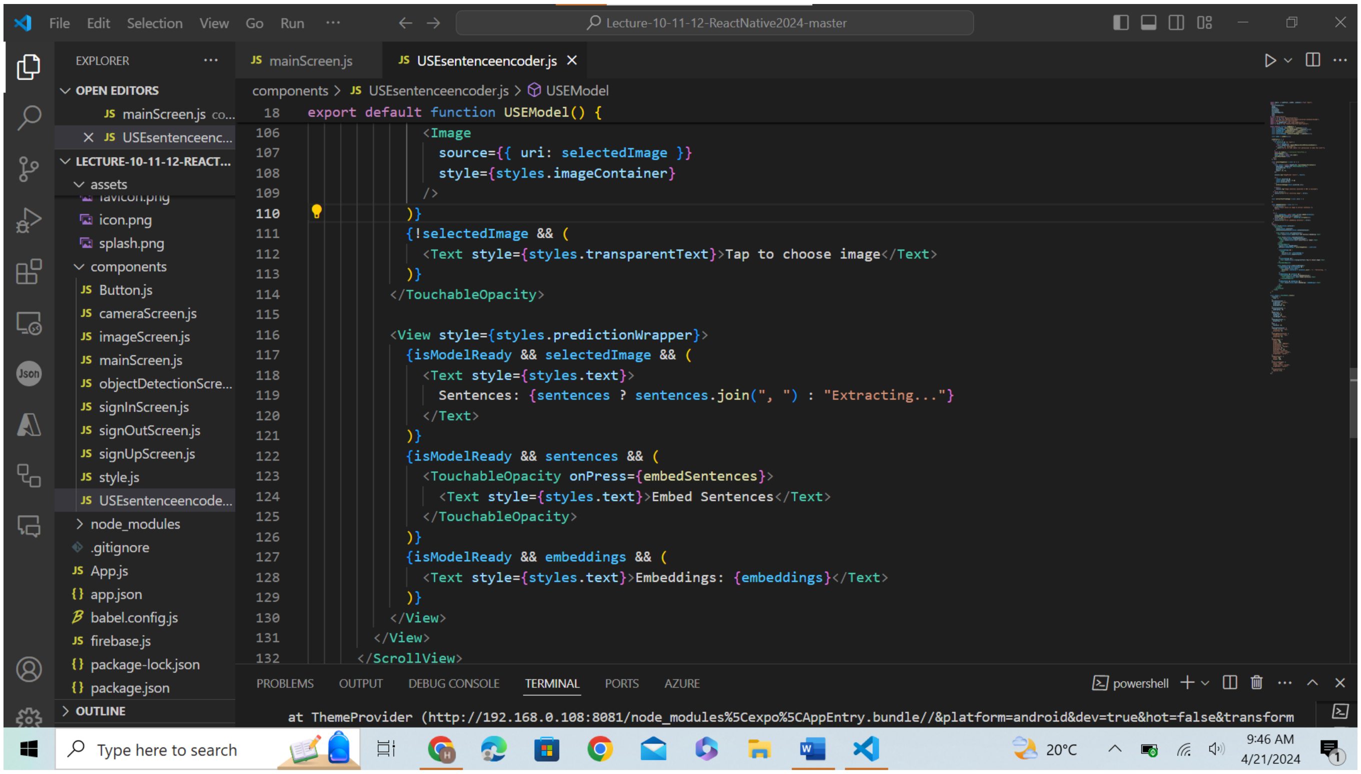
```
export default function USEModel() {
  const selectImageAsync = async () => {
      if (
        !result.cancelled &&
        result.assets.length > 0 &&
        result.assets[0].uri
      ) {
        setSelectedImage(result.assets[0].uri);

      } else {
        console.log("Image selection cancelled or URI is missing");
      }
    } catch (error) {
      console.error("Error selecting image:", error);
    }
  };

  const extractTextFromImage = async (uri) => {

  };

  const embedSentences = async () => {
    if (!sentences) {
      alert("Please select an image to extract sentences.");
      return;
    }

    try {
```

```
at ThemeProvider (http://192.168.0.108:8081/node_modules%5Cexpo%5CAppEntry.bundle//&platform=android&dev=true&hot=false&transform
```



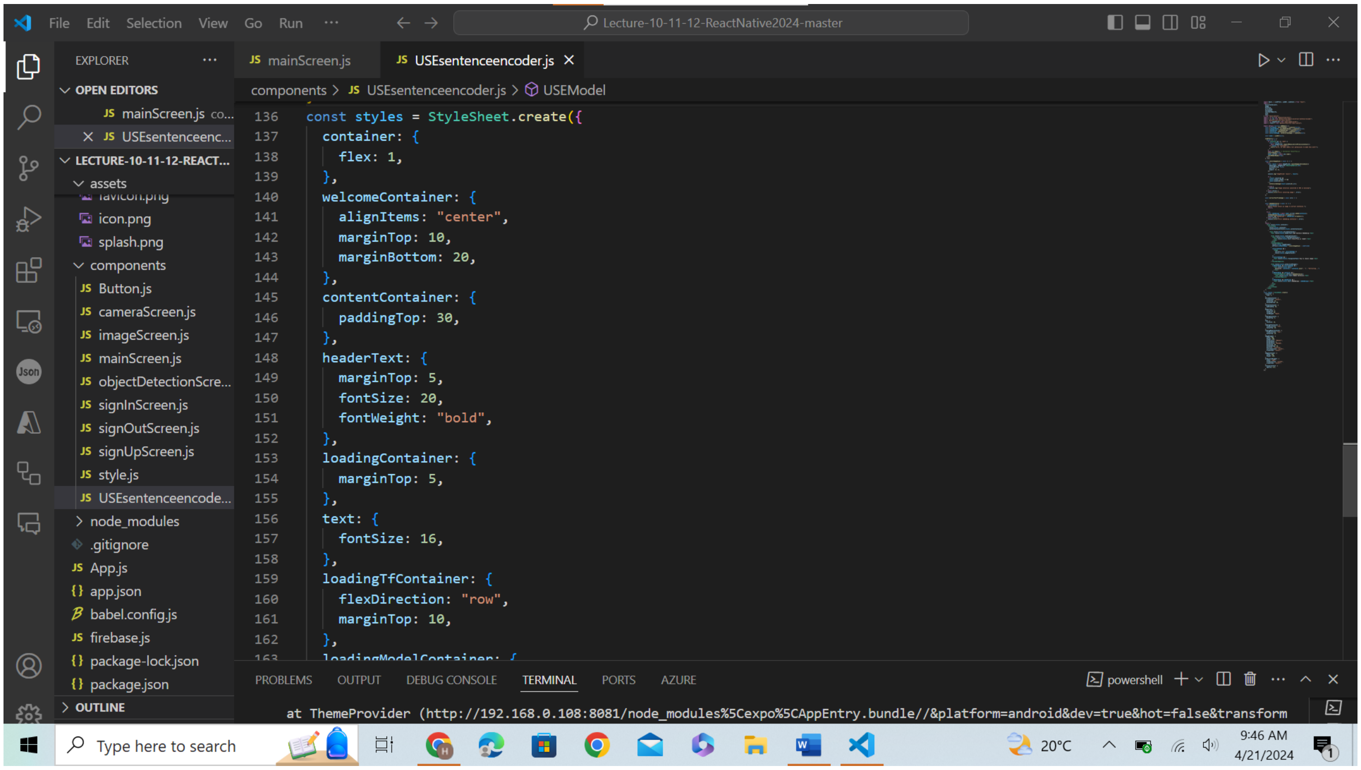
```
export default function USEModel() {
  const embedSentences = async () => {
      const embeddings = await model.current.embed(sentences);
      setEmbeddings(embeddings.arraySync());
      console.log("Embeddings:", embeddings.arraySync());
    } catch (error) {
      console.error("Error embedding sentences:", error);
    }
  };
  return (
    <View style={styles.container}>
      <ScrollView
        style={styles.container}
        contentContainerStyle={styles.contentContainer}
      >
        <View style={styles.welcomeContainer}>
          <Text style={styles.headerText}>USE Sentence Embedding</Text>

          <View style={styles.loadingContainer}>
            <View style={styles.loadingTfContainer}>
              <Text style={styles.text}>TensorFlow.js ready?</Text>
            </View>
          </View>
          <TouchableOpacity
            style={styles.imageWrapper}
            onPress={isModelReady ? selectImageAsync : undefined}
          >
            {selectedImage && (
```

```
at ThemeProvider (http://192.168.0.108:8081/node_modules%5Cexpo%5CAppEntry.bundle//&platform=android&dev=true&hot=false&transform
```



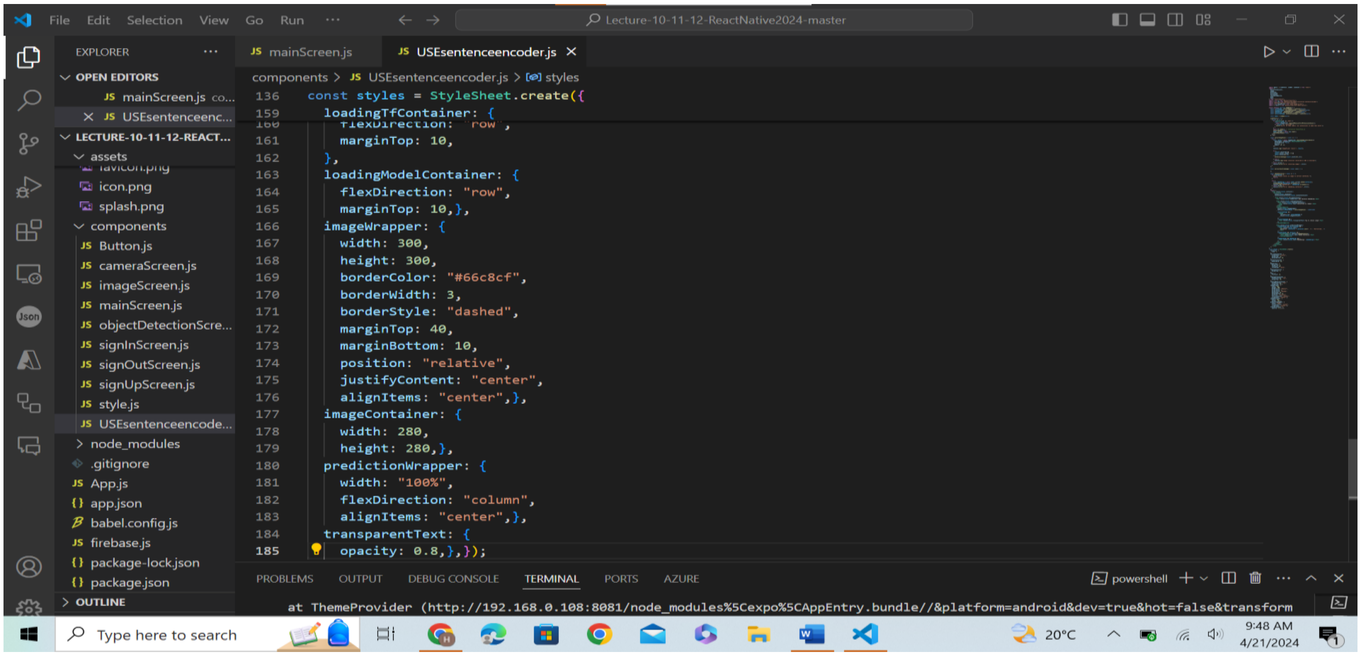
```
export default function USEModel() {
              <Image
                source={{ uri: selectedImage }}
                style={styles.imageContainer}
              />
            )}
            {!selectedImage && (
              <Text style={styles.transparentText}>Tap to choose image</Text>
            )}
          </TouchableOpacity>

          <View style={styles.predictionWrapper}>
            {isModelReady && selectedImage && (
              <Text style={styles.text}>
                Sentences: {sentences ? sentences.join(", ") : "Extracting..."}
              </Text>
            )}
            {isModelReady && sentences && (
              <TouchableOpacity onPress={embedSentences}>
                <Text style={styles.text}>Embed Sentences</Text>
              </TouchableOpacity>
            )}
            {isModelReady && embeddings && (
              <Text style={styles.text}>Embeddings: {embeddings}</Text>
            )}
          </View>
        </View>
      </ScrollView>
```

```
at ThemeProvider (http://192.168.0.108:8081/node_modules%5Cexpo%5CAppEntry.bundle//&platform=android&dev=true&hot=false&transform
```



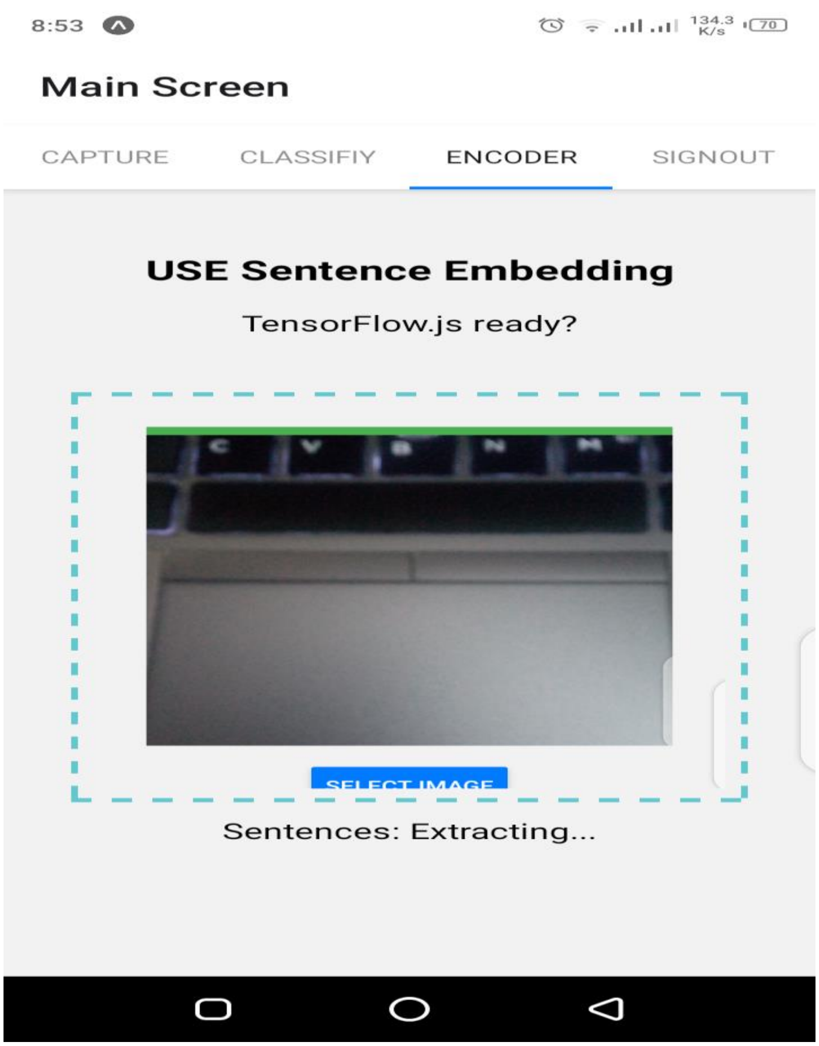
```
const styles = StyleSheet.create({
  container: {
    flex: 1,
  },
  welcomeContainer: {
    alignItems: "center",
    marginTop: 10,
    marginBottom: 20,
  },
  contentContainer: {
    paddingTop: 30,
  },
  headerText: {
    marginTop: 5,
    fontSize: 20,
    fontWeight: "bold",
  },
  loadingContainer: {
    marginTop: 5,
  },
  text: {
    fontSize: 16,
  },
  loadingTfContainer: {
    flexDirection: "row",
    marginTop: 10,
  },
  loadingModelContainer: {
```

```
at ThemeProvider (http://192.168.0.108:8081/node_modules%5Cexpo%5CAppEntry.bundle//&platform=android&dev=true&hot=false&transform
```



```
const styles = StyleSheet.create({
  loadingTfContainer: {
    flexDirection: "row",
    marginTop: 10,
  },
  loadingModelContainer: {
    flexDirection: "row",
    marginTop: 10,},
  imageWrapper: {
    width: 300,
    height: 300,
    borderColor: "#66c8cf",
    borderWidth: 3,
    borderStyle: "dashed",
    marginTop: 40,
    marginBottom: 10,
    position: "relative",
    justifyContent: "center",
    alignItems: "center",},
  imageContainer: {
    width: 280,
    height: 280,},
  predictionWrapper: {
    width: "100%",
    flexDirection: "column",
    alignItems: "center",},
  transparentText: {
    opacity: 0.8,},});
```

```
 at ThemeProvider (http://192.168.0.108:8081/node_modules%5Cexpo%5CAppEntry.bundle//&platform=android&dev=true&hot=false&transform
```



Main Screen

USE Sentence Embedding

TensorFlow.js ready?

Sentences: Extracting...