Muhammad Mudassir

2112193

Bscs8c

HYBRID MOBILE APP DEVELOPMENT EXAM

Q1) LIFE CYCLE METHODS IN REACT NATIVE FUNCTIONAL COMPONENETS:

- In React Native functional components, lifecycle events are handled using React Hooks (mainly `useEffect`).
- Mounting: `useEffect(() => { ... }, [])` runs once when the component mounts (like `componentDidMount`).
- Updating: `useEffect(() => { ... }, [dependency])` runs when the specified dependency changes (like `componentDidUpdate`).
- Unmounting: `useEffect(() => { return () => { ... }; }, [])` cleanup function runs when the component unmounts (like `componentWillUnmount`).
- Hooks provide a way to perform side effects, fetch data, set up subscriptions, and clean up resources in functional components.

Q2) Stratgies used to optimize performance of React Native App:

- Use FlatList or SectionList for rendering large lists efficiently instead of ScrollView.
- Minimize unnecessary re-renders by using React.memo, useCallback, and useMemo.
- Optimize images by resizing and compressing them before use.
- Use lazy loading and code splitting to load components only when needed.
- Avoid inline functions and object/array literals in render methods.
- Reduce the number of components mounted at once.
- Use native modules and libraries for heavy computations or animations.
- Remove unused dependencies and assets to reduce app size.
- Enable Hermes engine for better JavaScript performance on Android.
- Profile and monitor performance using tools like Flipper and React DevTools.

Q3) Differnces between navigations:

Stack Navigation

- Organizes screens in a stack (like a deck of cards).
- Allows users to push and pop screens (navigate forward and back).
- Common for workflows where users move deeper into content (e.g., details pages).
- Back button returns to the previous screen.

Tab Navigation

- Displays tabs (usually at the bottom or top) for switching between main sections.
- Each tab has its own navigation stack.
- Good for apps with a few top-level screens (e.g., Home, Search, Profile).
- Users can quickly switch between sections.

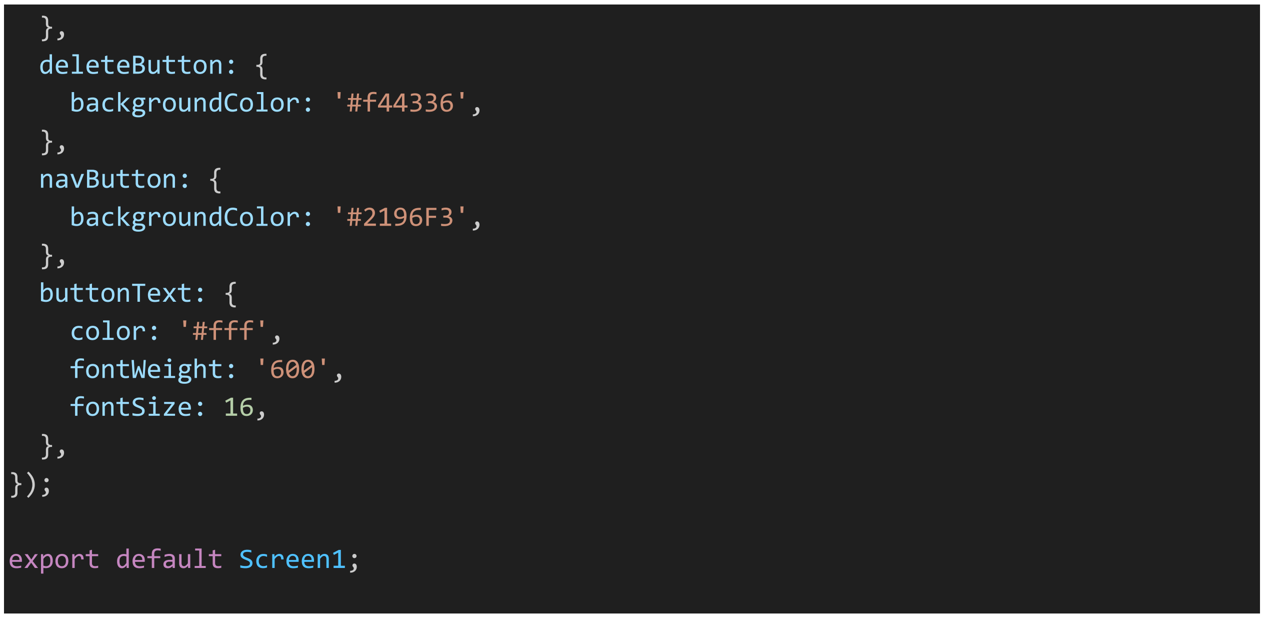
Drawer Navigation

- Provides a side menu (drawer) that slides in from the left or right.
- Menu contains links to different screens or sections.
- Useful for apps with many sections or settings.
- Frees up screen space compared to tabs.

SECTION B PRACTICAL

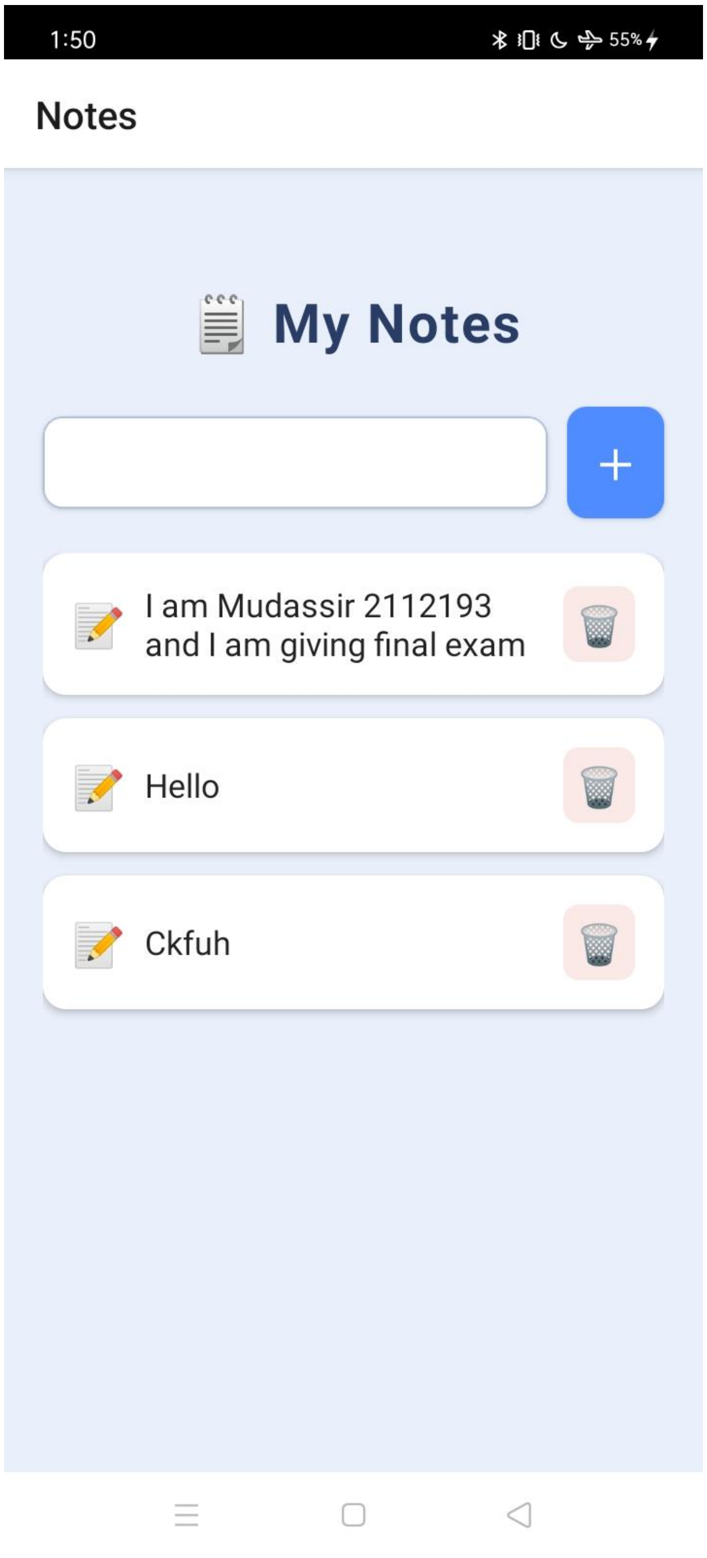
Q1)

```
import React, { useState, useEffect } from 'react';

import { View, Text, TextInput, TouchableOpacity, StyleSheet } from 'react-native';
import AsyncStorage from '@react-native-async-storage/async-storage';
import { NativeStackScreenProps } from '@react-navigation/native-stack';
import { RootStackParamList } from '../App';

type Props = NativeStackScreenProps<RootStackParamList, 'Screen1'>;

const Screen1: React.FC<Props> = ({ navigation }) => {
  const [input, setInput] = useState('');
  const [storedValue, setStoredValue] = useState('');

  useEffect(() => {
    getData();
  }, []);

  const saveData = async () => {
    await AsyncStorage.setItem('userInput', input);
    setStoredValue(input);
    setInput('');
  };

  const getData = async () => {
    const value = await AsyncStorage.getItem('userInput');
    if (value) setStoredValue(value);
  };

  const deleteData = async () => {
    await AsyncStorage.removeItem('userInput');
    setStoredValue('');
  };

  return (
    <View style={styles.container}>
      <Text style={styles.title}>Welcome to Screen 1</Text>
      <Text style={styles.label}>
        Stored Value: <Text style={styles.value}>{storedValue || 'None'}</Text>
      </Text>

      <TextInput
        style={styles.input}
        placeholder="Enter something..."
        value={input}
        onChangeText={setInput}
        placeholderTextColor="#999"
      />

      <TouchableOpacity style={styles.button} onPress={saveData}>
        <Text style={styles.buttonText}>💾 Save</Text>
      </TouchableOpacity>

      <TouchableOpacity style={[styles.button, styles.deleteButton]}
onPress={deleteData}>
        <Text style={styles.buttonText}>🗑 Delete</Text>
      </TouchableOpacity>
      <TouchableOpacity style={[styles.button, styles.navButton]} onPress={() =>
        navigation.navigate('Screen2', { passedValue: storedValue })}>
        <Text style={styles.buttonText}>➞ Go to Screen 2</Text>
      </TouchableOpacity>
    </View>
  );
};

const styles = StyleSheet.create({
  container: {
    padding: 20,
    paddingTop: 60,
    backgroundColor: '#f4f4f4',
    flex: 1,
  },
  title: {
    fontSize: 22,
    fontWeight: '700',
    marginBottom: 15,
    color: '#333',
  },
  label: {
    fontSize: 16,
    marginBottom: 10,
  },
  value: {
    fontWeight: 'bold',
    color: '#555',
  },
  input: {
    borderWidth: 1,
    borderColor: '#ccc',
    padding: 12,
    borderRadius: 10,
    backgroundColor: '#fff',
    marginBottom: 15,
    fontSize: 16,
  },
  button: {
    backgroundColor: '#4CAF50',
    padding: 14,
    borderRadius: 10,
    marginBottom: 10,
    alignItems: 'center',
```



```
  },
  deleteButton: {
    backgroundColor: '#f44336',
  },
  navButton: {
    backgroundColor: '#2196F3',
  },
  buttonText: {
    color: '#fff',
    fontWeight: '600',
    fontSize: 16,
  },
});

export default Screen1;
```

AS IN THE Q1 YOU CAN SEE ADD NOTE, LIST OF NOTE,ASYNC STORAGE AND DELETE BUTTON WHICH IS FUNCTIONAL.

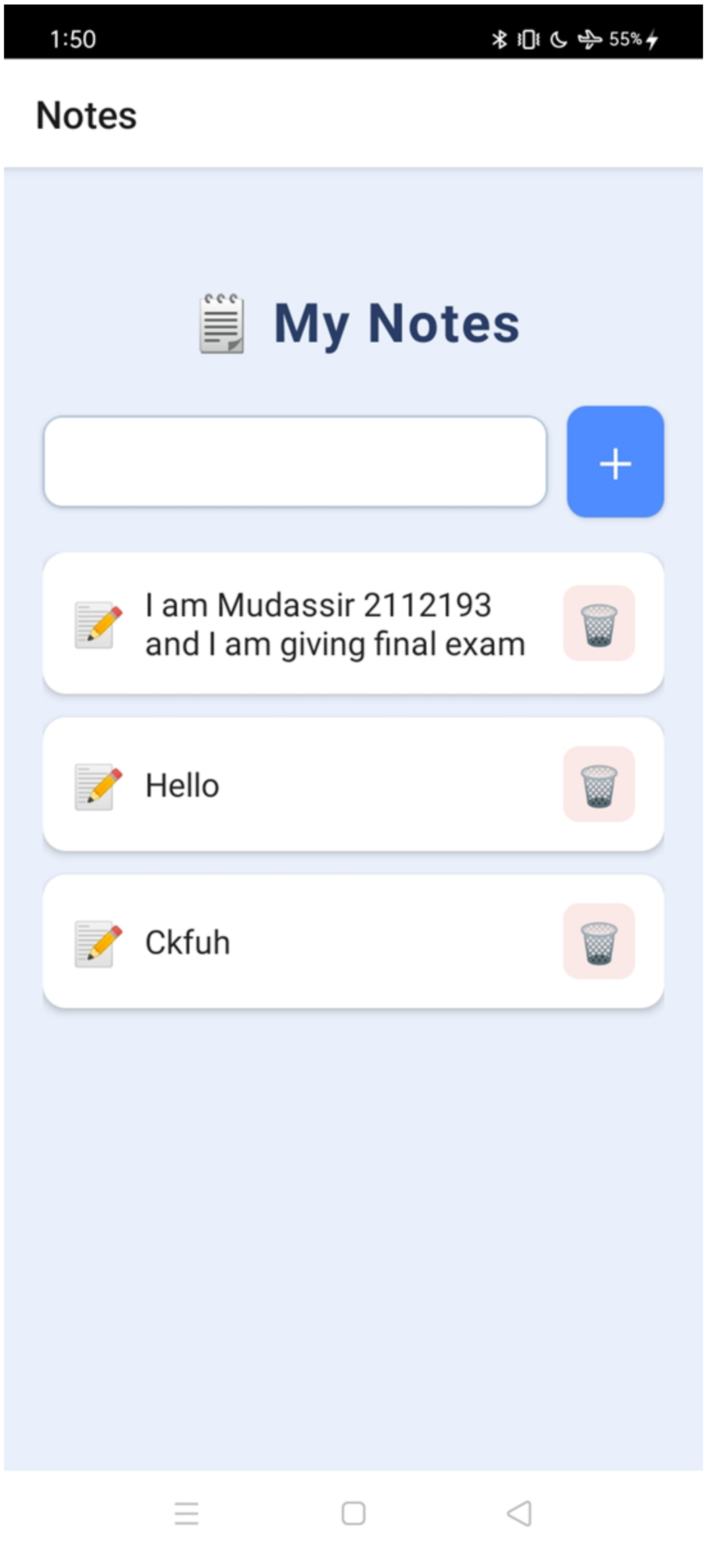


My Notes

I am Mudassir 2112193 and I am giving final exam

Hello

Ckfuh



Notes

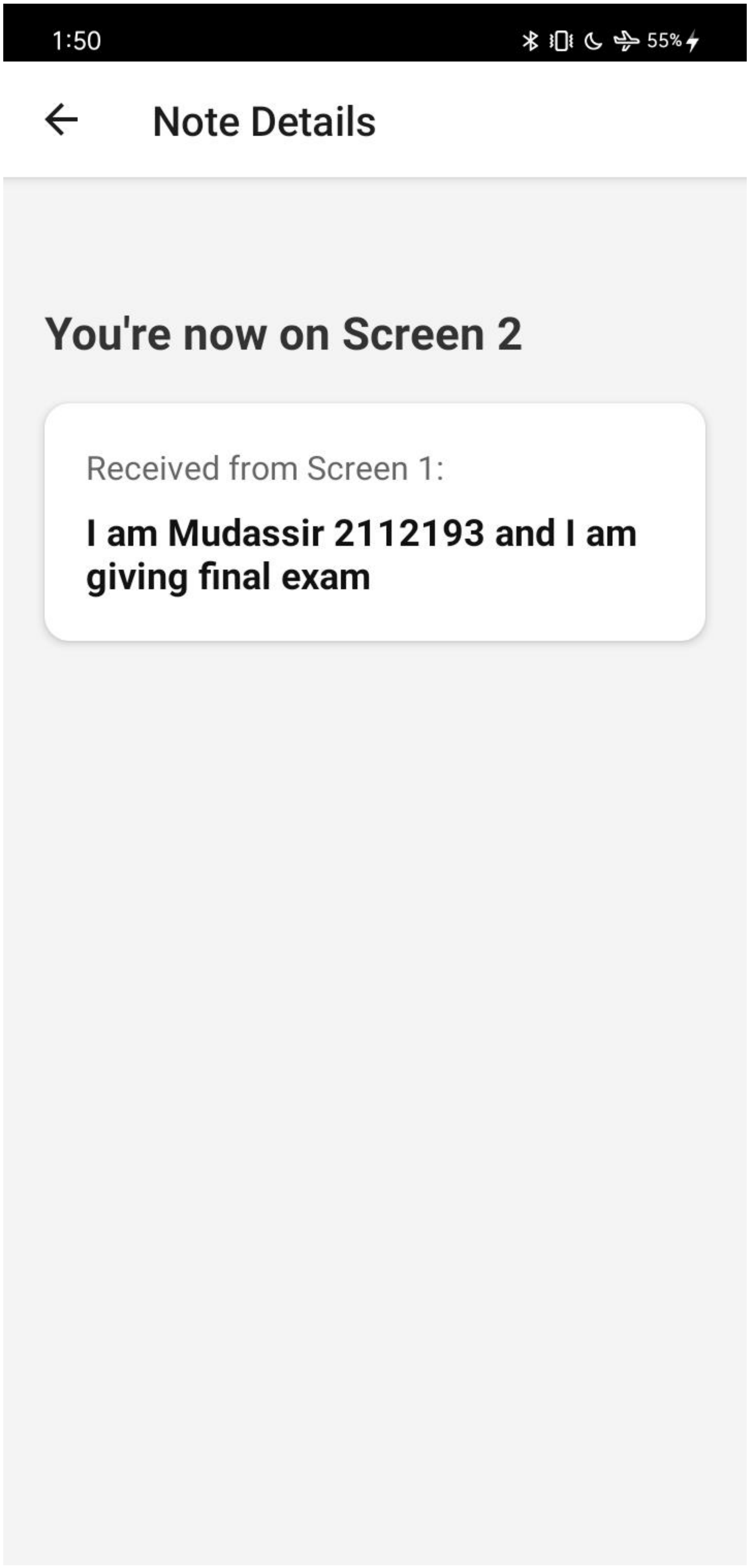
🗒️ My Notes

+

📝 I am Mudassir 2112193 and I am giving final exam 🗑️

📝 Hello 🗑️

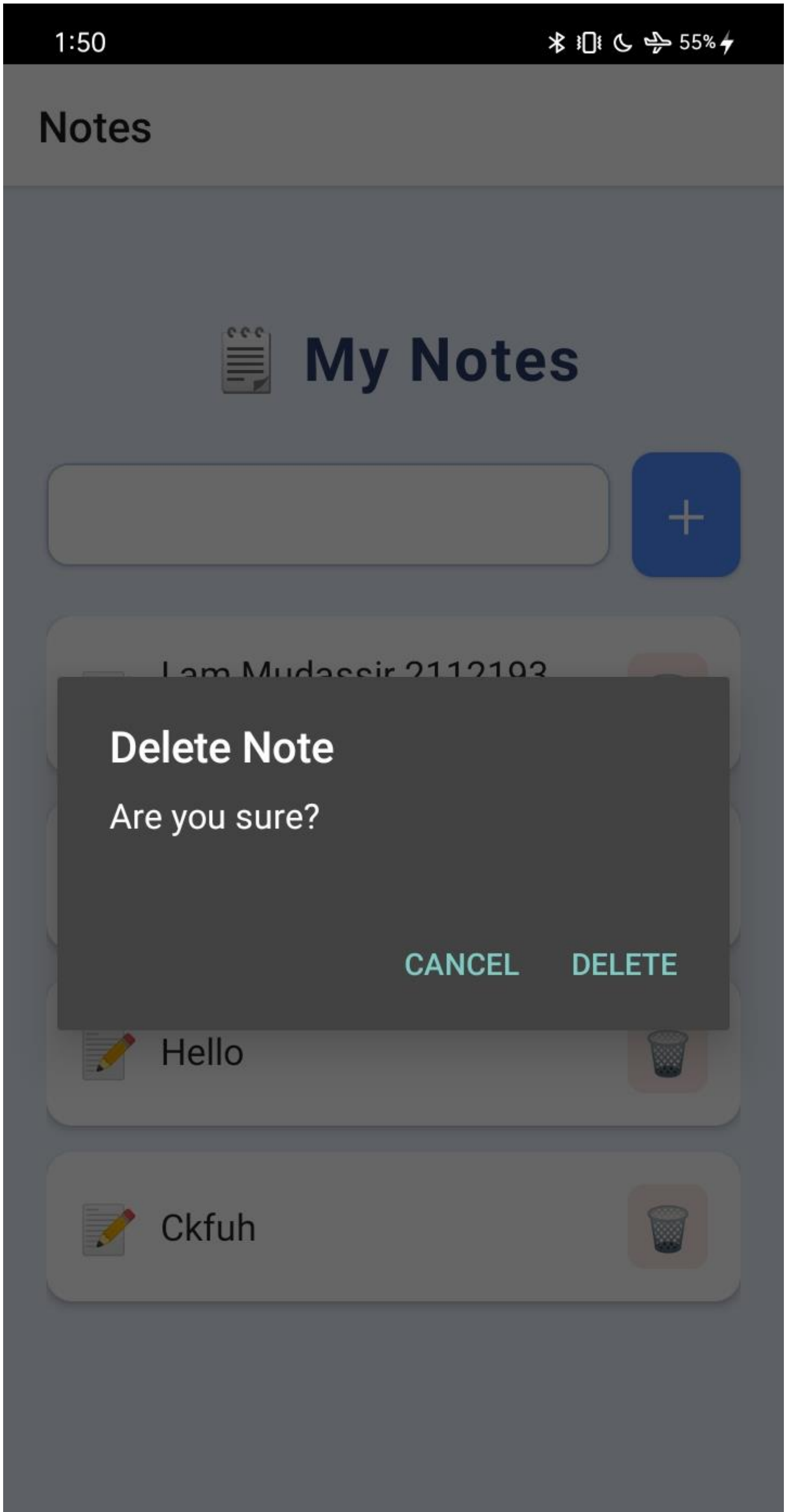
📝 Ckfuh 🗑️



You're now on Screen 2

Received from Screen 1:

I am Mudassir 2112193 and I am giving final exam



1:50 55%

Notes

My Notes

+

I am Mudassir 2112193

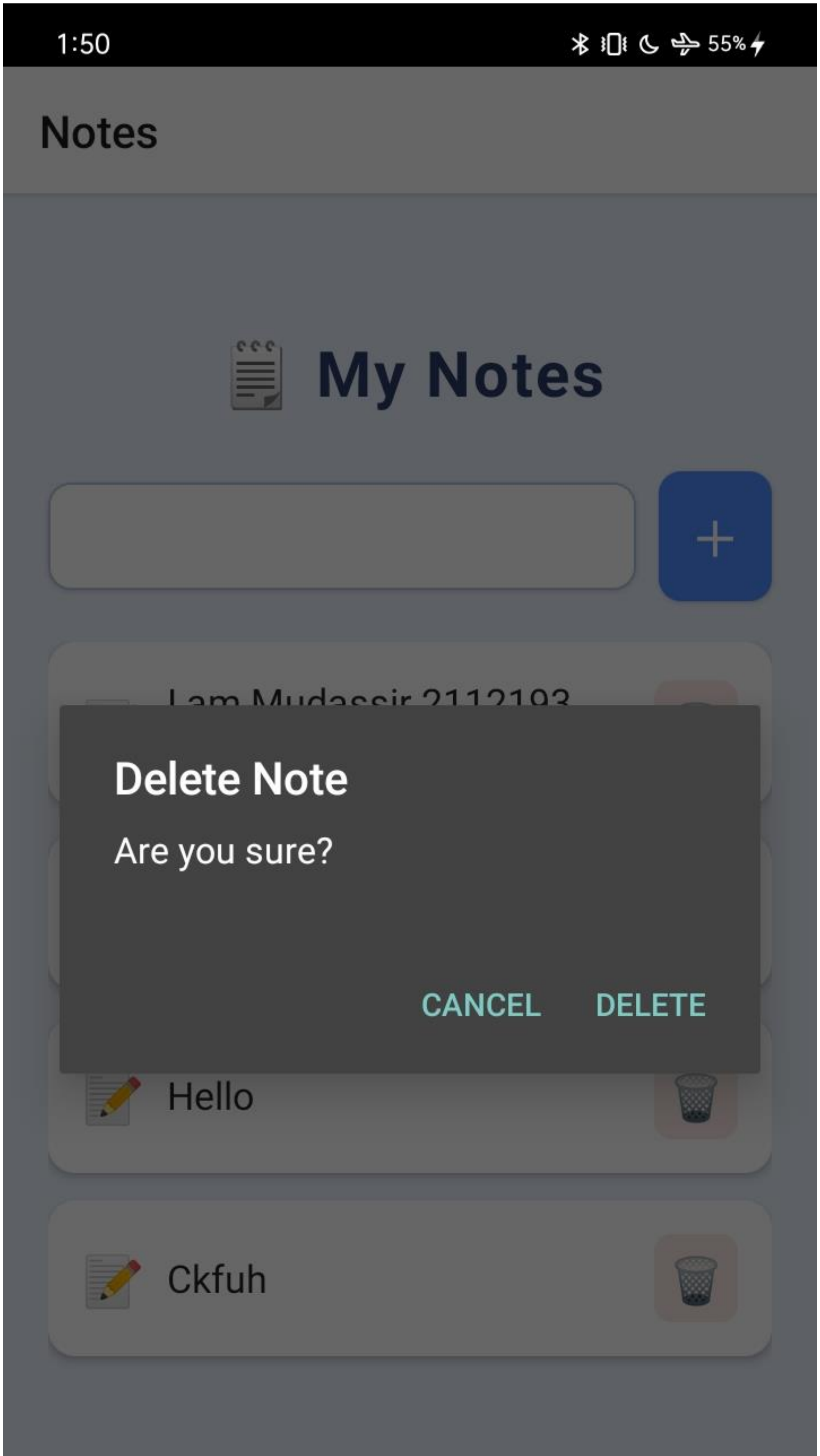
Delete Note

Are you sure?

CANCEL DELETE

Hello

Ckfuh

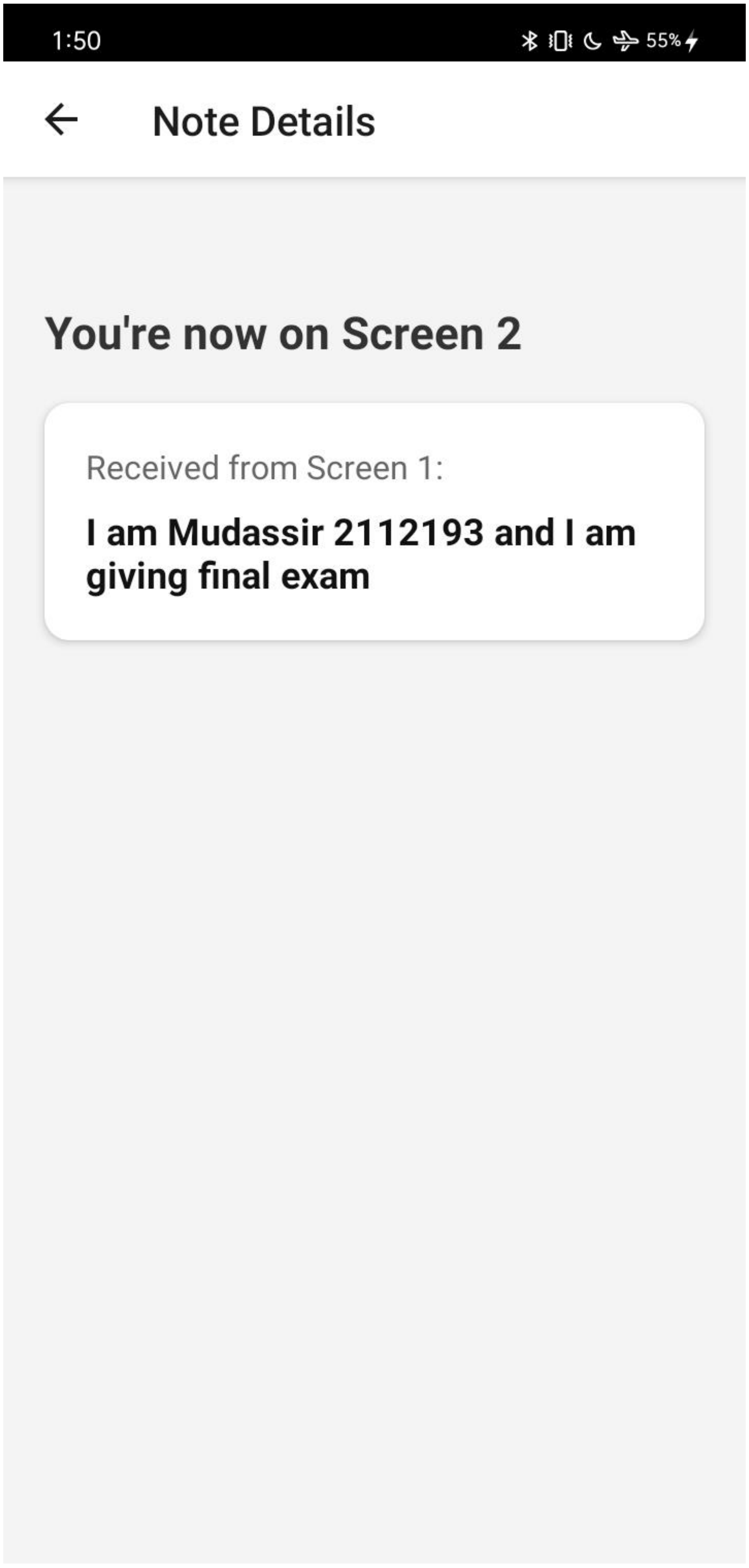


Delete Note

Are you sure?

Q2)
```
import React from 'react';

import { View, Text, StyleSheet } from 'react-native';
import { NativeStackScreenProps } from '@react-navigation/native-stack';
import { RootStackParamList } from '../App';

type Props = NativeStackScreenProps<RootStackParamList, 'Screen2'>;

const Screen2: React.FC<Props> = ({ route }) => {
  const { passedValue } = route.params;

  return (
    <View style={styles.container}>
      <Text style={styles.title}>You're now on Screen 2</Text>

      <View style={styles.card}>
        <Text style={styles.label}>Received from Screen 1:</Text>
        <Text style={styles.value}>
          {passedValue ? passedValue : 'No data received'}
        </Text>
      </View>
    </View>
  );
};

const styles = StyleSheet.create({
  container: {
    padding: 20,
    paddingTop: 60,
    backgroundColor: '#f4f4f4',
    flex: 1,
  },
  title: {
    fontSize: 22,
    fontWeight: '700',
    marginBottom: 20,
    color: '#333',
  },
  card: {
    backgroundColor: '#fff',
    borderRadius: 12,
    padding: 20,
    elevation: 3,
    shadowColor: '#000',
    shadowOpacity: 0.1,
    shadowRadius: 5,
    shadowOffset: { width: 0, height: 2 },
  },
  label: {
    fontSize: 16,
    color: '#666',
    marginBottom: 8,
  },
  value: {
    fontSize: 18,
    fontWeight: 'bold',
    color: '#111',
  },
});

export default Screen2;
```

YOU CAN SEE SAME DATA PASSED DOWN IN SCREEN 2



You're now on Screen 2

Received from Screen 1:

I am Mudassir 2112193 and I am giving final exam