งาน4: State และ Props ใน React Native

จัดทำโดย

65122250018 นายวรพล อุดม

เสนอ

ผศ.ดร. เสถียร จันทร์ปลา

ภาคเรียนที่ 2 ปีการศึกษา 2567

หลักสูตรวิทยาการคอมพิวเตอร์และนวัตกรรมข้อมูล

มหาวิทยาลัยราชภัฏสวนสุนันทา

แบบฝึกหัดที่ 4

State และ Props ใน React Native

คำสั่ง

1. ส่งงานให้ตรงเวลา
2. จัดเอกสารตามรูปแบบการทำรายงาน
3. ห้ามลอกกัน

โปรแกรมในเอกสาร

1. GreetingApp.js
    - โปรแกรม

```
import React from 'react';
import { View, Text, StyleSheet } from 'react-native';
// คอมโพเนนต์สำหรับแสดงข้อมูลผู้ใช้ โดยรับ Props user ที่เป็น Object
const UserCard = ({ user }) => {
    return (
        <View>
            {/* แสดงชื่อและอายุที่ส่งมาผ่าน Props user */}
            <Text style={styles.text}>Name: {user.name}</Text>
            <Text style={styles.text}>Age: {user.age}</Text>
        </View>
    );
};
// คอมโพเนนต์หลักที่ส่ง Object user ไปยัง UserCard
const UserCardApp = () => {
    const userInfo = { name: 'Woraphon', age: 21 }; // Object ที่เก็บข้อมูลผู้ใช้
    return (
        <View style={styles.container}>
            <Text style={styles.title}>แบบหลายค่า (Object Props)</Text>
            <Text style={styles.subtitle}>Programmer Name: Woraphon Udom</Text>
            {/* ส่ง Object userInfo ให้ UserCard */}
            <UserCard user={userInfo} />
        </View>
    );
};
const styles = StyleSheet.create({
    container: { flex: 1, justifyContent: 'center', alignItems: 'center', backgroundColor: "#98ff8a"},
    text: { fontSize: 18, },
    title: { fontSize: 20, fontWeight: "bold", position: "absolute", top: 50,color: "#ff0000" },
    subtitle: { fontSize: 16, position: "absolute", top: 80, color: "#ff6161", },
});
export default UserCardApp;
```

- ผลลัพธ์ของโปรแกรม

2. UserCardApp.js

- โปรแกรม

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

// คอมโพเนนต์สำหรับแสดงข้อมูลผู้ใช้ โดยรับ Props user ที่เป็น Object
const UserCard = ({ user }) => {
    return (
        <View>
            {/* แสดงชื่อและอายุที่ส่งมาผ่าน Props user */}
            <Text style={styles.text}>Name: {user.name}</Text>
            <Text style={styles.text}>Age: {user.age}</Text>
        </View>
    );
};

// คอมโพเนนต์หลักที่ส่ง Object user ไปยัง UserCard
const UserCardApp = () => {
    const userInfo = { name: 'Woraphon', age: 21 }; // Object ที่เก็บข้อมูลผู้ใช้

    return (
        <View style={styles.container}>
            <Text style={styles.title}>แบบหลายค่า (Object Props)</Text>
            <Text style={styles.subtitle}>Programmer Name: Woraphon Udom</Text>
            {/* ส่ง Object userInfo ให้ UserCard */}
            <UserCard user={userInfo} />
        </View>
    );
};

const styles = StyleSheet.create({
    container: { flex: 1, justifyContent: 'center', alignItems: 'center', backgroundColor: "#98ff8a"},
    text: { fontSize: 18, },
    title: { fontSize: 20, fontWeight: "bold", position: "absolute", top: 50,color: "#ff0000" },
    subtitle: { fontSize: 16, position: "absolute", top: 80, color: "#ff6161", },
});

export default UserCardApp;
```

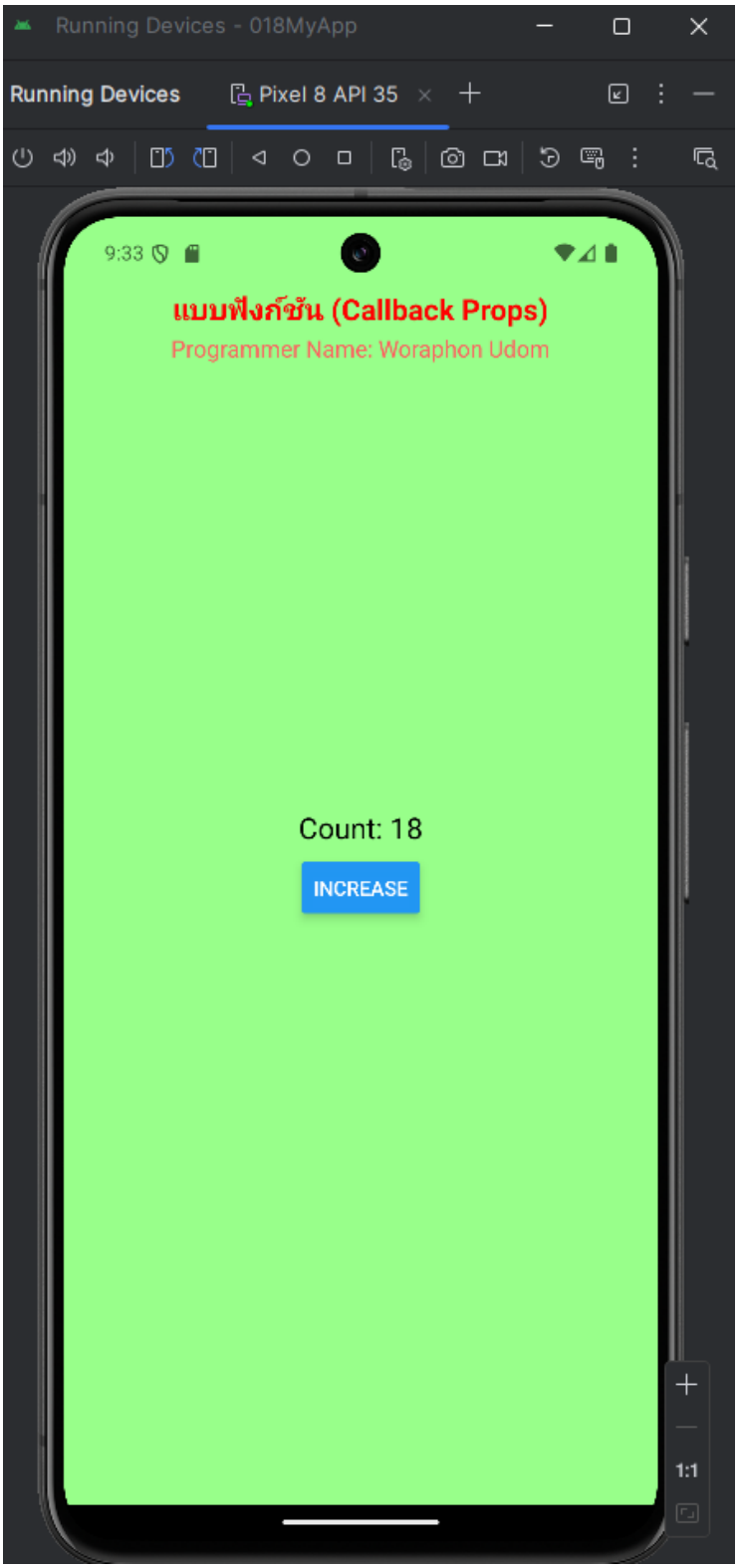
- ผลลัพธ์ของโปรแกรม

3. CounterApp.js

- โปรแกรม

```
import React, { useState } from 'react';
import { View, Text, Button, StyleSheet } from 'react-native';

// คอมโพเนนต์สำหรับแสดงปุ่ม โดยรับ Props onIncrease ซึ่งเป็นฟังก์ชัน
const Counter = ({ onIncrease }) => {
    return <Button title="Increase" onPress={onIncrease} />;
};

// คอมโพเนนต์หลักที่ใช้ State เพื่อจัดการค่า count
const CounterApp = () => {
    const [count, setCount] = useState(0); // กำหนดค่าเริ่มต้นของ count เป็น 0

    // ฟังก์ชันเพิ่มค่า count
    const increaseCount = () => {
        setCount(count + 1);
    };

    return (
        <View style={styles.container}>
            <Text style={styles.title}>แบบฟังก์ชัน (Callback Props)</Text>
            <Text style={styles.subtitle}>Programmer Name: Woraphon Udom</Text>
            {/* แสดงค่าปัจจุบันของ count */}
            <Text style={styles.text}>Count: {count}</Text>
            {/* ส่งฟังก์ชัน increaseCount ไปยัง Counter ผ่าน Props onIncrease */}
            <Counter onIncrease={increaseCount} />
        </View>
    );
};

const styles = StyleSheet.create({
    container: { flex: 1, justifyContent: 'center', alignItems: 'center', backgroundColor: "#98ff8a"},
    text: { fontSize: 20, marginBottom: 10 },
    title: { fontSize: 20, fontWeight: "bold", position: "absolute", top: 50, color: "#ff0000" },
    subtitle: { fontSize: 16, position: "absolute", top: 80, color: "#ff6161",},
});

export default CounterApp;
```

- ผลลัพธ์ของโปรแกรม

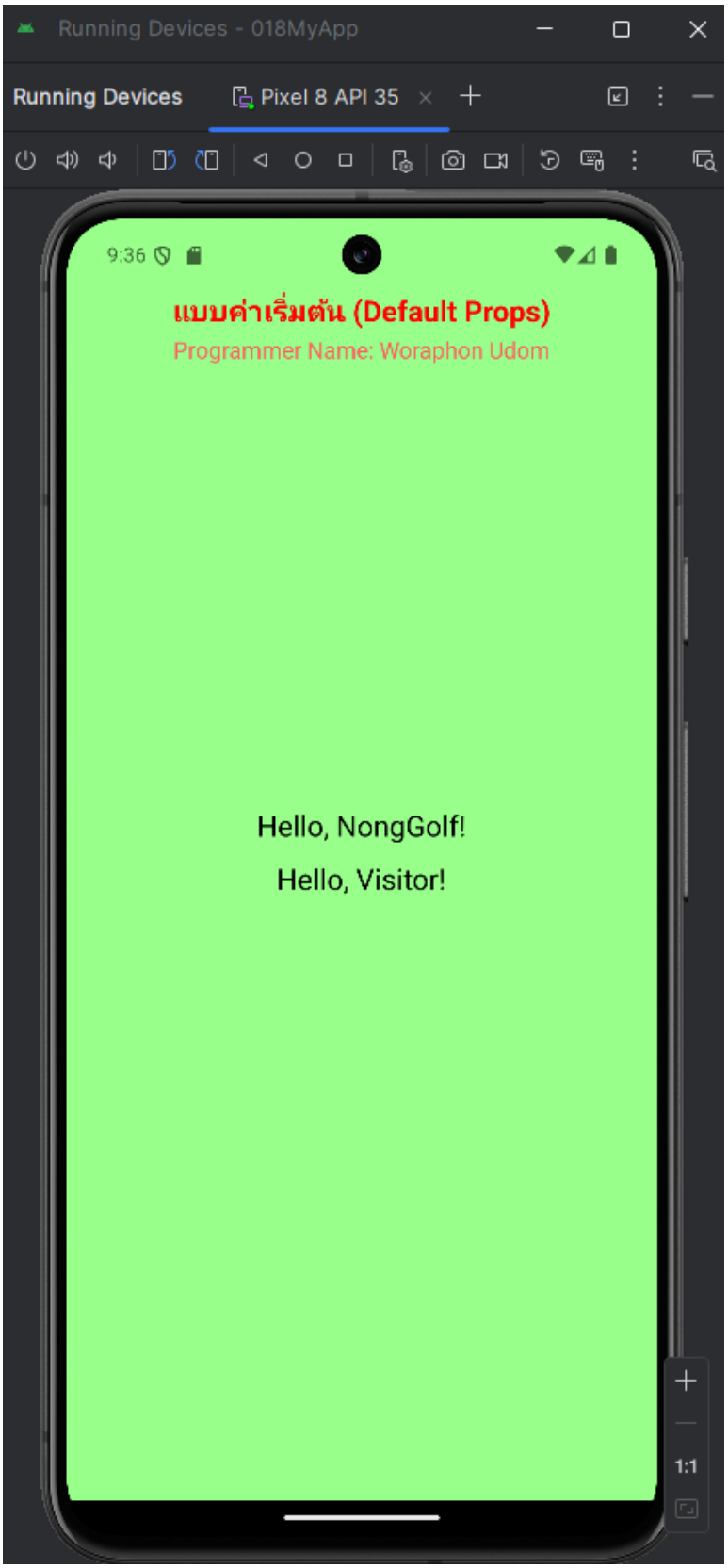


4. DefaultPropsApp.js

- โปรแกรม

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

// ใช้ Default Parameters ในฟังก์ชันแทน defaultProps
const Greeting = ({ name = 'Visitor' }) => {
    return <Text style={styles.text}>Hello, {name}!</Text>;
};

const DefaultPropsApp = () => {
    return (
        <View style={styles.container}>
            <Text style={styles.title}>แบบค่าเริ่มต้น (Default Props)</Text>
            <Text style={styles.subtitle}>Programmer Name: Woraphon Udom</Text>
            {/* ส่งค่า name เป็น 'Nonglak' */}
            <Greeting name="NongGolf" />
            {/* ไม่ส่งค่า name จะใช้ค่า default 'Visitor' */}
            <Greeting />
        </View>
    );
};

const styles = StyleSheet.create({
    container: { flex: 1, justifyContent: 'center', alignItems: 'center', backgroundColor: "#98ff8a"},
    text: { fontSize: 20, marginBottom: 10 },
    title: { fontSize: 20, fontWeight: "bold", position: "absolute", top: 50, color: "#ff0000" },
    subtitle: { fontSize: 16, position: "absolute", top: 80, color: "#ff6161",},
});
export default DefaultPropsApp;
```

- ผลลัพธ์ของโปรแกรม



5. SpreadPropsApp.js

- โปรแกรม

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

// คอมโพเนนต์แสดงข้อมูลผู้ใช้ โดยรับ Props name และ age
const UserCard = ({ name, age }) => {
    return (
        <View>
            <Text style={styles.text}>Name: {name}</Text>
            <Text style={styles.text}>Age: {age}</Text>
        </View>
    );
};

// คอมโพเนนต์หลักที่ใช้ Spread Operator เพื่อส่ง Props หลายค่า
const SpreadPropsApp = () => {
    const userInfo = { name: 'Woraphon', age: 21 }; // Object ที่เก็บข้อมูลผู้ใช้

    return (
        <View style={styles.container}>
            <Text style={styles.title}>แบบกระจายค่า (Spread Props)</Text>
            <Text style={styles.subtitle}>Programmer Name: Woraphon Udom</Text>
            {/* ใช้ Spread Operator ส่ง Props */}
            <UserCard {...userInfo} />
        </View>
    );
};

const styles = StyleSheet.create({
    container: { flex: 1, justifyContent: 'center', alignItems: 'center', backgroundColor: "#98ff8a"},
    text: { fontSize: 20, marginBottom: 10 },
    title: { fontSize: 20, fontWeight: "bold", position: "absolute", top: 50, color: "#ff0000" },
    subtitle: { fontSize: 16, position: "absolute", top: 80, color: "#ff6161",},
});

export default SpreadPropsApp;
```

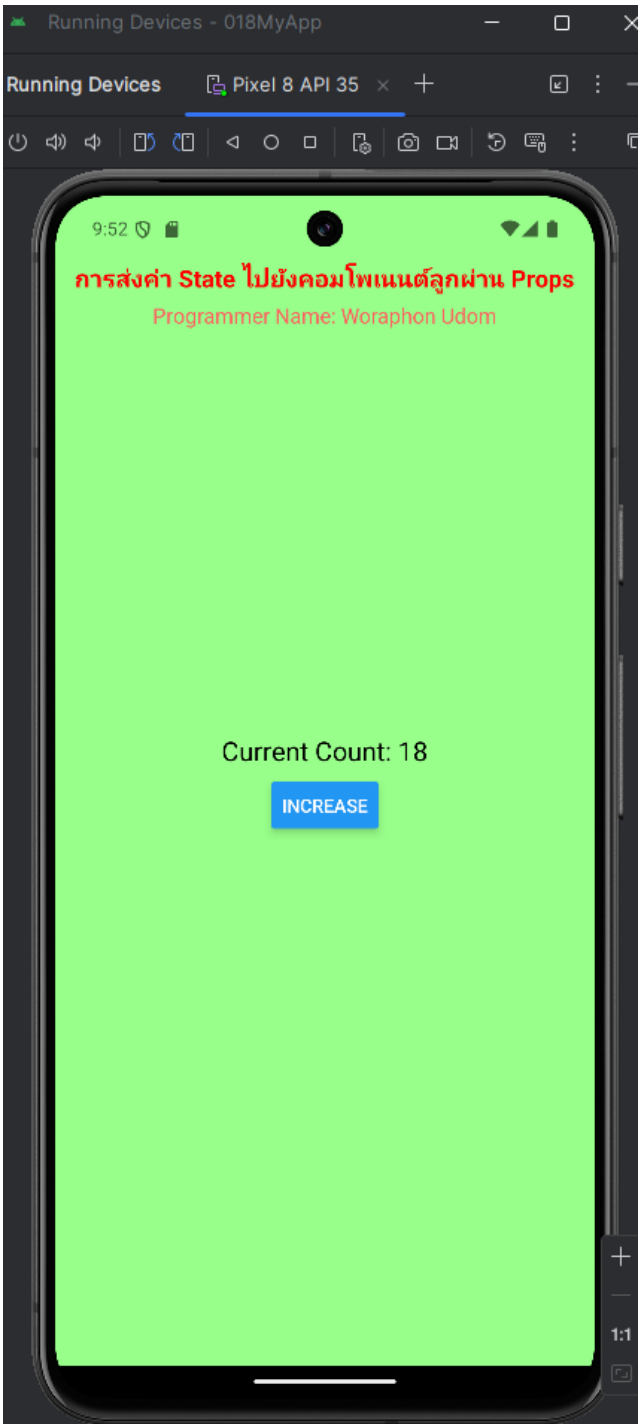
- ผลลัพธ์ของโปรแกรม

6. StateToPropsParent.js

- โปรแกรม

```
import React, { useState } from 'react';
import { View, Text, Button, StyleSheet } from 'react-native';

// คอมโพเนนต์ลูก รับค่า count ผ่าน Props
const CounterDisplay = ({ count }) => {
  return <Text style={styles.text}>Current Count: {count}</Text>;
};

// คอมโพเนนต์แม่
const StateToPropsParent = () => {
  const [count, setCount] = useState(0); // State ของคอมโพเนนต์แม่

  return (
    <View style={styles.container}>
      <Text style={styles.title}>การส่งค่า State ไปยังคอมโพเนนต์ลูกผ่าน Props</Text>
      <Text style={styles.subtitle}>Programmer Name: Woraphon Udom</Text>
      {/* ส่ง State count ไปยัง CounterDisplay ผ่าน Props */}
      <CounterDisplay count={count} />
      <Button title="Increase" onPress={() => setCount(count + 1)} />
    </View>
  );
};

const styles = StyleSheet.create({
  container: { flex: 1, justifyContent: 'center', alignItems: 'center', backgroundColor: "#98ff8a"},
  text: { fontSize: 20, marginBottom: 10 },
  title: { fontSize: 18, fontWeight: "bold", position: "absolute", top: 50, color: "#ff0000" },
  subtitle: { fontSize: 16, position: "absolute", top: 80, color: "#ff6161",},
});

export default StateToPropsParent;
```

- ผลลัพธ์ของโปรแกรม

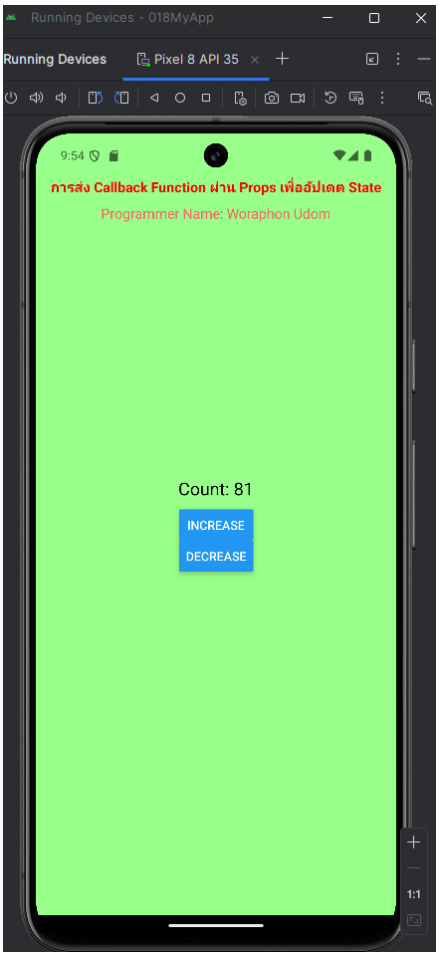


7. CallbackPropsParent.js

- โปรแกรม

```
import React, { useState } from "react";
import { View, Text, Button, StyleSheet } from "react-native";

// คอมโพเนนต์ลูก รับฟังก์ชัน onIncrease และ onDecrease ผ่าน Props
const CounterControls = ({ onIncrease, onDecrease }) => {
  return (
    <View>
      <Button title="Increase" onPress={onIncrease} />
      <Button title="Decrease" onPress={onDecrease} />
    </View>
  );
};

// คอมโพเนนต์แม่
const CallbackPropsParent = () => {
  const [count, setCount] = useState(0); // State ของคอมโพเนนต์แม่

  return (
    <View style={styles.container}>
      <Text style={styles.title}>การส่ง Callback Function ผ่าน Props เพื่ออัปเดต State</Text>
      <Text style={styles.subtitle}>Programmer Name: Woraphon Udom</Text>
      <Text style={styles.text}>Count: {count}</Text>
      {/* ส่งฟังก์ชัน setCount ผ่าน Props ไปยัง CounterControls */}
      <CounterControls
        onIncrease={() => setCount(count + 1)}
        onDecrease={() => setCount(count - 1)}
      />
    </View>
  );
};

const styles = StyleSheet.create({
  container: { flex: 1, justifyContent: 'center', alignItems: 'center', backgroundColor: "#98ff8a"},
  text: { fontSize: 20, marginBottom: 10 },
  title: { fontSize: 16, fontWeight: "bold", position: "absolute", top: 50, color: "#ff0000" },
  subtitle: { fontSize: 16, position: "absolute", top: 80, color: "#ff6161",},
});

export default CallbackPropsParent;
```

- ผลลัพธ์ของโปรแกรม

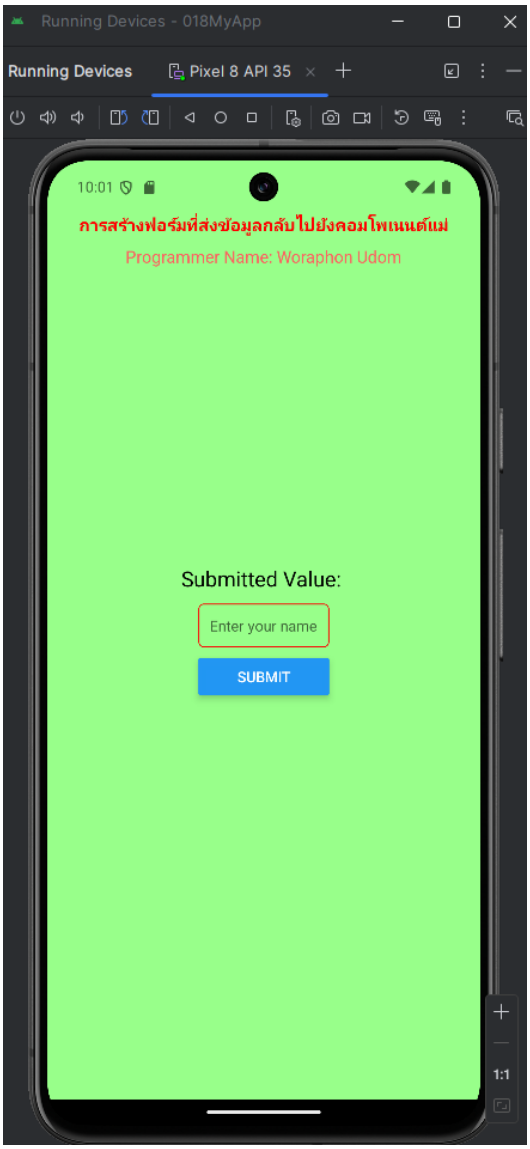


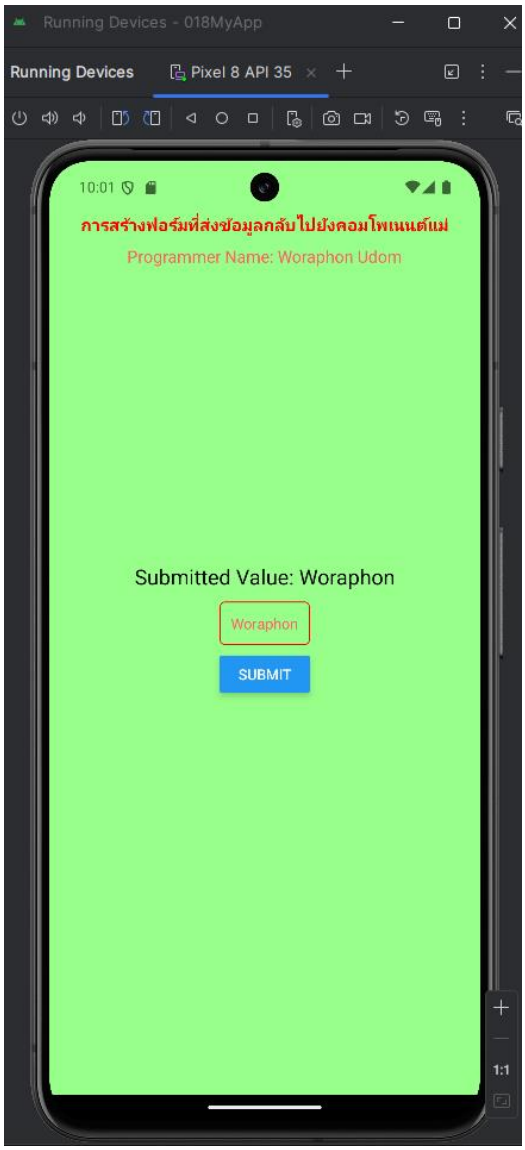
8. FormParent.js

- โปรแกรม

```
import React, { useState } from 'react';
import { View, Text, TextInput, Button, StyleSheet } from 'react-native';

// คอมโพเนนต์ลูกสำหรับฟอร์ม
const InputForm = ({ onSubmit }) => {
  const [inputValue, setInputValue] = useState(''); // State ภายในฟอร์ม
  return (
    <View>
      <TextInput
        style={styles.input}
        value={inputValue}
        onChangeText={setInputValue}
        placeholder="Enter your name"
      />
      {/* เรียกฟังก์ชัน onSubmit เมื่อกดปุ่ม */}
      <Button title="Submit" onPress={() => onSubmit(inputValue)} />
    </View>
  );
};
// คอมโพเนนต์แม่
const FormParent = () => {
  const [submittedValue, setSubmittedValue] = useState(''); // State ของคอมโพเนนต์แม่

  return (
    <View style={styles.container}>
      <Text style={styles.title}>การสร้างฟอร์มที่ส่งข้อมูลกลับไปยังคอมโพเนนต์แม่</Text>
      <Text style={styles.subtitle}>Programmer Name: Woraphon Udom</Text>
      <Text style={styles.text}>Submitted Value: {submittedValue}</Text>
      {/* ส่งฟังก์ชัน setSubmittedValue ไปยัง InputForm */}
      <InputForm onSubmit={(value) => setSubmittedValue(value)} />
    </View>
  );
};

const styles = StyleSheet.create({
  container: { flex: 1, justifyContent: 'center', alignItems: 'center', backgroundColor: "#98ff8a"},
  text: { fontSize: 20, marginBottom: 10 },
  input: { borderWidth: 1, borderColor: '#ff0000', padding: 10, width: '80%',
    marginBottom: 10, borderRadius: 5, color: "#ff6161"},
  title: { fontSize: 16, fontWeight: "bold", position: "absolute", top: 50, color: "#ff0000" },
  subtitle: { fontSize: 16, position: "absolute", top: 80, color: "#ff6161",},
});

export default FormParent;
```

- ผลลัพธ์ของโปรแกรม





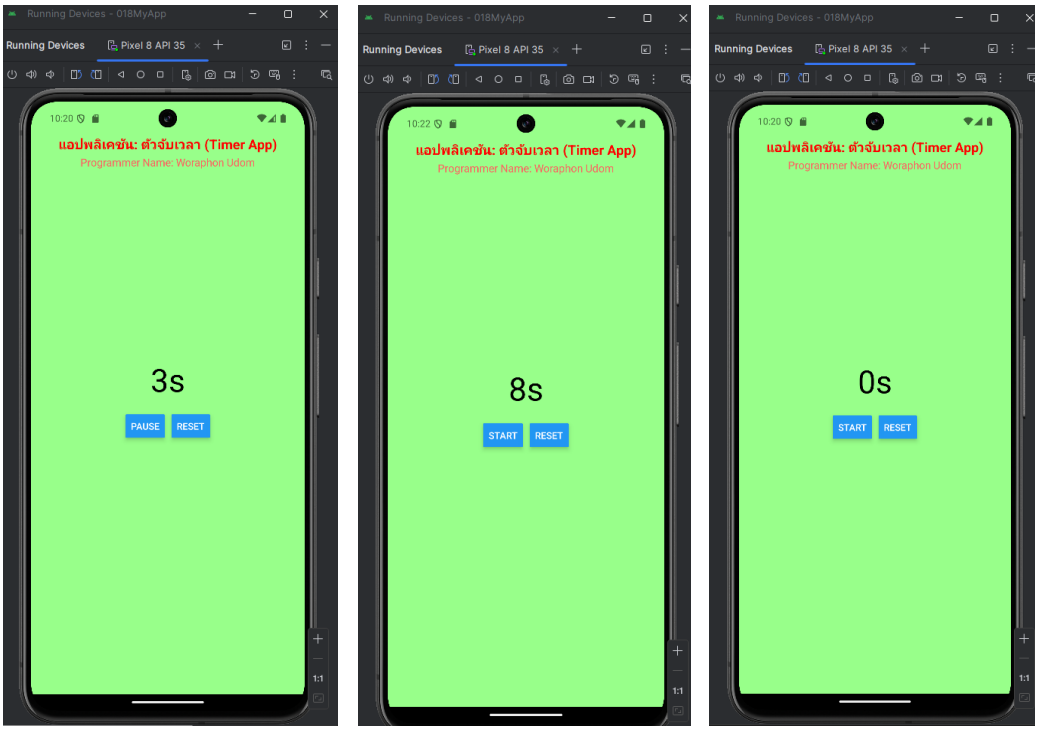
9. TimerApp.js

- โปรแกรม

```
import React, { useState, useEffect } from 'react';
import { View, Text, Button, StyleSheet } from 'react-native';
// คอมโพเนนต์หลักสำหรับแอปพลิเคชันจับเวลา
const TimerApp = () => {
  // State สำหรับเก็บเวลา (นับเป็นวินาที)
  const [time, setTime] = useState(0);
  // State สำหรับเก็บสถานะการจับเวลา (เริ่ม/หยุด)
  const [isRunning, setIsRunning] = useState(false);
  // ใช้ useEffect สำหรับตั้งค่าและล้าง Timer
  useEffect(() => {
    let timer; // ตัวแปรสำหรับเก็บ Timer
    if (isRunning) {
      // ถ้า isRunning เป็น true ให้เริ่มจับเวลา
      timer = setInterval(() => {
        setTime((prevTime) => prevTime + 1); // เพิ่มเวลา 1 วินาที
      }, 1000); // ตั้งค่า Timer ให้ทำงานทุก ๆ 1 วินาที
    } else {
      // ถ้า isRunning เป็น false ให้หยุดจับเวลา
      clearInterval(timer); // ล้าง Timer
    }
    // Cleanup function สำหรับล้าง Timer เมื่อคอมโพเนนต์ถูกถอดออก หรือก่อนเริ่ม Timer ใหม่
    return () => clearInterval(timer);
  }, [isRunning]); // รัน useEffect เมื่อค่า isRunning เปลี่ยนแปลง
  // ฟังก์ชันรีเซ็ตเวลา
  const resetTimer = () => {
    setTime(0); // ตั้งค่าเวลาเป็น 0
    setIsRunning(false); // หยุดการจับเวลา
  };
  return (
    <View style={styles.container}>
      <Text style={styles.title}>แอปพลิเคชัน: ตัวจับเวลา (Timer App)</Text>
      <Text style={styles.subtitle}>Programmer Name: Woraphon Udom</Text>
      {/* แสดงเวลาในรูปแบบวินาที */}
      <Text style={styles.timer}>{time}s</Text>
      <View style={styles.buttonContainer}>
        {/* ปุ่มสำหรับเริ่มหรือหยุดจับเวลา */}
        <Button
          title={isRunning ? 'Pause' : 'Start'}
          onPress={() => setIsRunning(!isRunning)} // เปลี่ยนสถานะ isRunning
        />
        {/* ปุ่มสำหรับรีเซ็ตเวลา */}
        <Button title="Reset" onPress={resetTimer} />
      </View>
    </View>
  );
};
```

```
// การตั้งค่า Styles สำหรับ UI
const styles = StyleSheet.create({
  container: { flex: 1, justifyContent: 'center', alignItems: 'center', backgroundColor: "#98ff8a" },
  timer: { fontSize: 48, marginBottom: 20 }, // ขนาดตัวอักษรของตัวจับเวลา
  buttonContainer: { flexDirection: 'row', gap: 10 }, // ปุ่มเรียงในแนวนอนพร้อมระยะห่าง
  title: { fontSize: 20, fontWeight: "bold", position: "absolute", top: 50, color: "#ff0000"},
  subtitle: { fontSize: 16, position: "absolute", top: 80, color: "#ff6161", },
});

export default TimerApp; // ส่งออกคอมโพเนนต์ TimerApp
```

- ผลลัพธ์ของโปรแกรม



10. TriangleAreaApp.js

- โปรแกรม

```
import React, { useState } from 'react';
import { View, Text, TextInput, Button, StyleSheet } from 'react-native';

const TriangleAreaApp = () => {
    // State สำหรับเก็บค่าฐาน, ความสูง และผลลัพธ์
    const [base, setBase] = useState('');
    const [height, setHeight] = useState('');
    const [area, setArea] = useState(null);

    // ฟังก์ชันคำนวณพื้นที่สามเหลี่ยม
    const calculateArea = () => {
        const baseValue = parseFloat(base); // แปลงฐานเป็นตัวเลข
        const heightValue = parseFloat(height); // แปลงความสูงเป็นตัวเลข

        if (!isNaN(baseValue) && !isNaN(heightValue)) {
            const calculatedArea = 0.5 * baseValue * heightValue; // คำนวณพื้นที่
            setArea(calculatedArea.toFixed(2)); // เก็บผลลัพธ์ใน State และจำกัดทศนิยม 2 ตำแหน่ง
        } else {
            setArea('Invalid Input'); // กรณีค่าที่กรอกไม่ถูกต้อง
        }
    }; //end calculateArea Function

    return (
        <View style={styles.container}>
            <Text style={styles.title}>Triangle Area Calculator</Text>
            <Text style={styles.subtitle}>Programmer Name: Woraphon Udom</Text>

            {/* ช่องกรอกค่าฐาน */}
            <TextInput
                style={styles.input}
                placeholder="Enter base"
                keyboardType="numeric"
                value={base}
                onChangeText={setBase}
            />

            {/* ช่องกรอกค่าความสูง */}
            <TextInput
                style={styles.input}
                placeholder="Enter height"
                keyboardType="numeric"
                value={height}
                onChangeText={setHeight}
            />
```

```
            {/* ปุ่มคำนวณ */}
            <Button title="Calculate" onPress={calculateArea} />
            {/* แสดงผลลัพธ์ */}
            {area !== null && ( // ตรวจสอบว่า State area ไม่ใช่ null
              <Text style={styles.result}>
                {/* ตรวจสอบว่า area เป็นตัวเลขหรือไม่ */}
                {isNaN(area)
                  ? area // ถ้า area ไม่ใช่ตัวเลข (NaN) แสดงข้อความ "Invalid Input"
                  : `Area: ${area} square units` // ถ้า area เป็นตัวเลข แสดงผลลัพธ์พื้นที่พร้อมหน่วย
                }
              </Text>
            )}
        </View>
    );
};
const styles = StyleSheet.create({
    container: { flex: 1, justifyContent: 'center', alignItems: 'center', backgroundColor: "#98ff8a"},
    title: { fontSize: 20, fontWeight: "bold", position: "absolute", top: 50, color: "#ff0000" },
    subtitle: { fontSize: 16, position: "absolute", top: 80, color: "#ff6161",},
    input: {borderWidth: 1,borderColor: '#ccc',padding: 10, marginBottom: 10, borderRadius: 5, width: '80%',},
    result: { fontSize: 18, marginTop: 20, color: '#333' },
});
export default TriangleAreaApp;
```

- ผลลัพธ์ของโปรแกรม

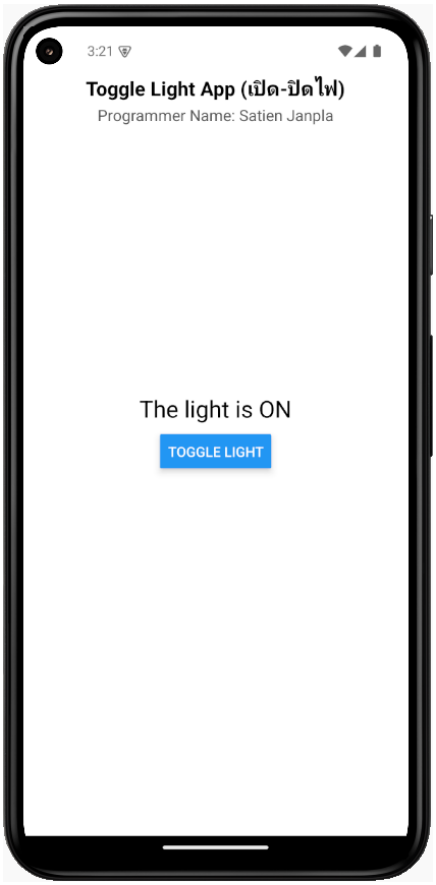
แบบฝึกหัด

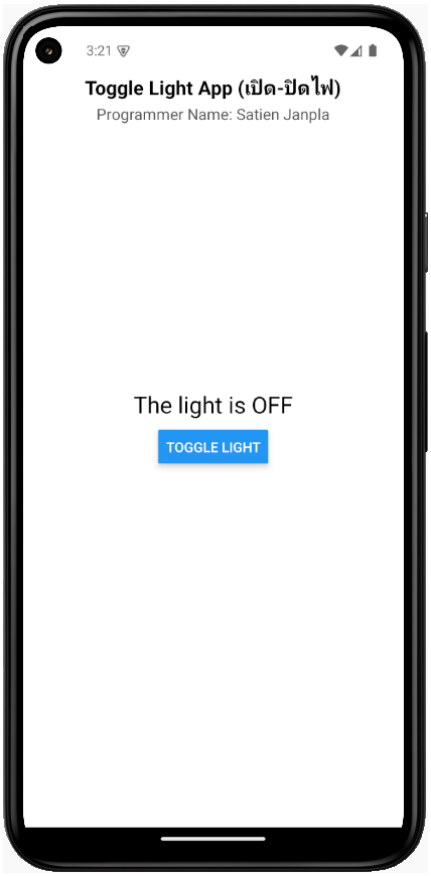
โจทย์ที่ 1: ชื่อแอป: Toggle Light App (เปิด-ปิดไฟ)

รายละเอียดการทำงานของแอป

- แอปนี้แสดงสถานะของหลอดไฟ (เปิด/ปิด)
- ผู้ใช้สามารถกดปุ่ม Toggle Light เพื่อเปลี่ยนสถานะไฟจาก "ON" เป็น "OFF" หรือจาก "OFF" เป็น "ON"
- ใช้ State ในการเก็บสถานะของหลอดไฟ

ตัวอย่างผลลัพธ์





- โปรแกรม

```
import React, { useState } from 'react';
import { View, Text, Button, StyleSheet } from 'react-native';

const ToggleLightApp = () => {
    const [isLightOn, setIsLightOn] = useState(false);

    const toggleLight = () => {
        setIsLightOn(!isLightOn);
    };

    return (
        <View style={styles.container}>
            <Text style={styles.title}>Toggle Light App</Text>
            <Text style={styles.subtitle}>Programmer Name: Woraphon Udom</Text>
            <Text style={styles.status}>The Light is : {isLightOn ? 'ON' : 'OFF'}</Text>
            <Button title="Toggle Light" onPress={toggleLight} />
        </View>
    );
};

const styles = StyleSheet.create({
    container: { flex: 1, justifyContent: 'center', alignItems: 'center', backgroundColor: "#98ff8a"},
    text: { fontSize: 20, marginBottom: 10 },
    title: { fontSize: 20, fontWeight: "bold", position: "absolute", top: 50, color: "#ff0000" },
    subtitle: { fontSize: 16, position: "absolute", top: 80, color: "#ff6161",},
    status:{ fontSize: 30, padding: 10 }
});

export default ToggleLightApp;
```

- คำอธิบายโปรแกรม

การตั้งค่า State:

◉ใช้ useState เพื่อเก็บสถานะของหลอดไฟ:

-true: ไฟเปิด (ON)

-false: ไฟปิด (OFF)

ฟังก์ชัน toggleLight:

◉เมื่อผู้ใช้กดปุ่ม "Toggle Light", ฟังก์ชันนี้จะสลับสถานะของ isLightOn:

-ถ้า isLightOn เป็น true เปลี่ยนเป็น false

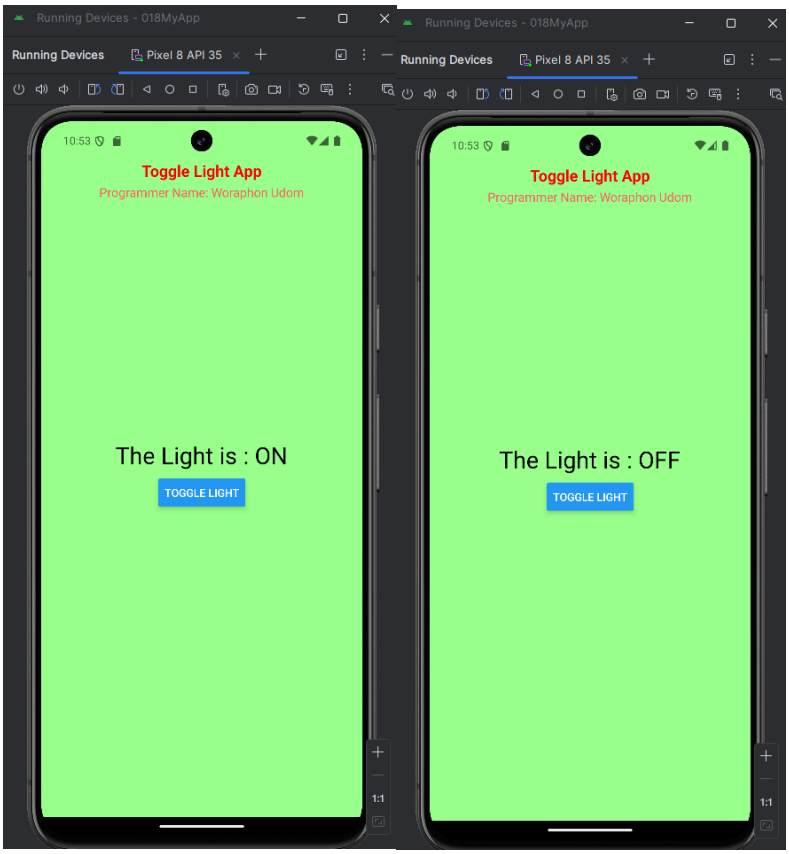
-ถ้า isLightOn เป็น false เปลี่ยนเป็น true

การแสดงผล:

-ข้อความ "The Light is: ON" หรือ "The Light is: OFF" จะแสดงตามสถานะ isLightOn

-ปุ่ม "Toggle Light" ใช้สำหรับเปลี่ยนสถานะ

- ผลลัพธ์ของโปรแกรม

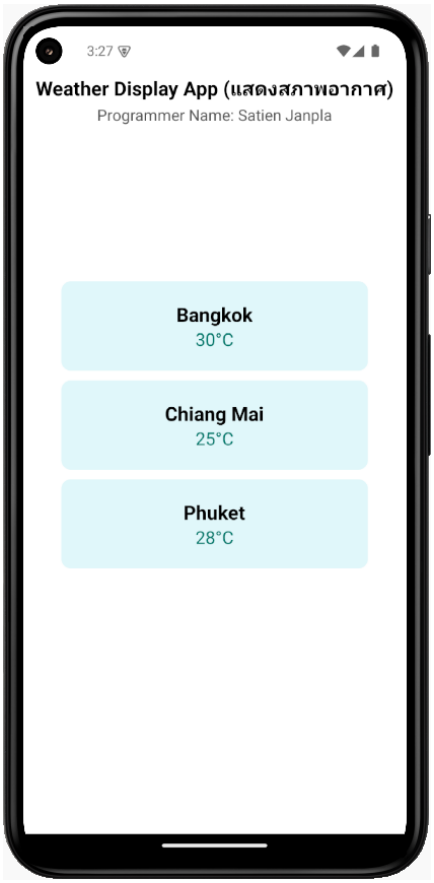


โจทย์ที่ 2: ชื่อแอป: Weather Display App (แสดงสภาพอากาศ)

รายละเอียดการทำงานของแอป

- แอปนี้แสดงข้อมูลสภาพอากาศ เช่น ชื่อเมืองและอุณหภูมิ
- คอมโพเนนต์แม่ส่งข้อมูลเมืองและอุณหภูมิผ่าน Props ไปยังคอมโพเนนต์ลูก
- แอปนี้มีข้อมูลตัวอย่างเมือง 3 แห่ง: Bangkok, Chiang Mai, และ Phuket

ตัวอย่างผลลัพธ์



- โปรแกรม

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

const WeatherDisplayApp = () => {
    const cities = [
        { name: 'Bangkok', temperature: 30 },
        { name: 'Chiang Mai', temperature: 25 },
        { name: 'Phuket', temperature: 28 },
    ];

    return (
        <View style={styles.container}>
            <Text style={styles.title}>Weather Display App (แสดงสภาพอากาศ)</Text>
            <Text style={styles.subtitle}>Programmer Name: Woraphon Udom</Text>
            {cities.map((city, index) => (
                <View key={index} style={styles.cityBox}>
                    <Text style={styles.cityName}>{city.name}</Text>
                    <Text style={styles.temperature}>{city.temperature}°C</Text>
                </View>
            ))}
        </View>
    );
};

const styles = StyleSheet.create({
    container: { flex: 1, justifyContent: 'center', alignItems: 'center', backgroundColor: '#98ff8a' },
    title: { fontSize: 20, fontWeight: "bold", position: "absolute", top: 50,color: "#ff0000" },
    subtitle: { fontSize: 16, position: "absolute", top: 80, color: "#ff6161", },
    cityBox: { width: '80%', padding: 15, marginVertical: 10, borderRadius: 10, backgroundColor: '#ff1f1f', alignItems: 'center', },
    cityName: { fontSize: 20, fontWeight: 'bold', marginBottom: 5, color: "#ffffff" },
    temperature: { fontSize: 18, color: '#00796b', color: "#ffffff" },
});

export default WeatherDisplayApp;
```

- คำอธิบายโปรแกรม

ข้อมูล:

-เก็บชื่อเมืองและอุณหภูมิใน array ชื่อ cities เช่น Bangkok: 30°C, Chiang Mai: 25°C.

แสดงข้อมูลใน Box:

-ใช้ map() เพื่อสร้าง View สำหรับแต่ละเมือง:

-ชื่อเมือง (city.name) แสดงเป็นข้อความขนาดใหญ่.

-อุณหภูมิ (city.temperature) แสดงพร้อมหน่วย °C.

การตกแต่ง Styles:

-container: จัดหน้าจอให้อยู่ตรงกลางพร้อมพื้นหลังสีเขียวอ่อน

-title และ subtitle:

ชื่อแอปและชื่อโปรแกรมเมอร์อยู่ด้านบนสุด

-cityBox:

กล่องของแต่ละเมืองมีพื้นหลังสีแดง, มุมโค้ง, และขอบเขตที่ชัดเจน

-cityName และ temperature:

ข้อความชื่อเมืองและอุณหภูมิใช้ฟอนต์สีขาว

- ผลลัพธ์ของโปรแกรม

โจทย์ที่ 3: ชื่อแอป: BMI Calculator (คำนวณดัชนีมวลกาย)

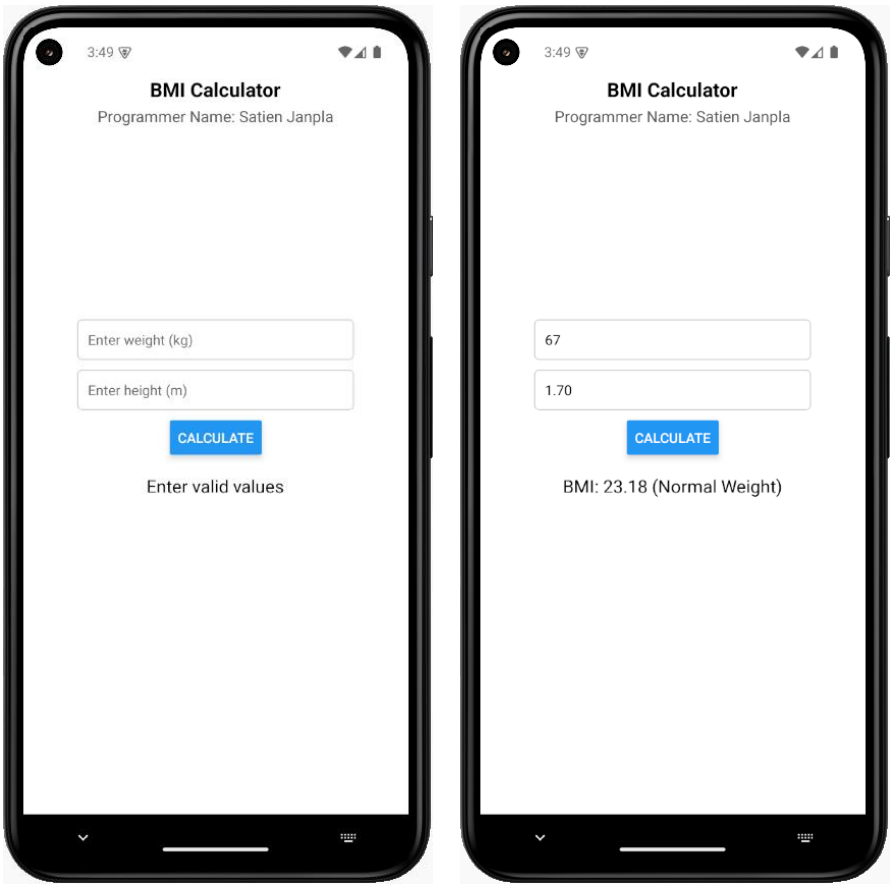
รายละเอียดการทำงานของแอป

- ผู้ใช้กรอกน้ำหนัก (กิโลกรัม) และส่วนสูง (เมตร)
- เมื่อกดปุ่ม "Calculate" แอปจะแสดงค่า BMI (ดัชนีมวลกาย) พร้อมข้อความแนะนำ เช่น "Normal Weight" หรือ "Overweight"

Category ของ BMI

ช่วงค่า BMI	Category
BMI < 18.5	Underweight น้ำหนักต่ำกว่าเกณฑ์
18.5 ≤ BMI < 24.9	Normal Weight น้ำหนักปกติ
25 ≤ BMI < 29.9	Overweight น้ำหนักเกิน
BMI ≥ 30	Obesity โรคอ้วน

ตัวอย่างผลลัพธ์



- โปรแกรม

```
import React, { useState } from 'react';
import { View, Text, TextInput, Button, StyleSheet } from 'react-native';

const BMICalculator = () => {
    const [weight, setWeight] = useState('');
    const [height, setHeight] = useState('');
    const [bmiResult, setBmiResult] = useState('');

    const calculateBMI = () => {
        const weightNum = parseFloat(weight);
        const heightNum = parseFloat(height);

        if (!weightNum || !heightNum) {
            setBmiResult('Enter valid values');
            return;
        }

        const bmi = weightNum / (heightNum * heightNum);
        let category;

        if (bmi < 18.5) {
            category = 'Underweight';
        } else if (bmi < 24.9) {
            category = 'Normal Weight';
        } else if (bmi < 29.9) {
            category = 'Overweight';
        } else {
            category = 'Obesity';
        }

        setBmiResult(`BMI: ${bmi.toFixed(2)} (${category})`);
    };

    return (
        <View style={styles.container}>
            <Text style={styles.title}>BMI Calculator</Text>
            <Text style={styles.subtitle}>Programmer Name: Woraphon Udom</Text>
            <TextInput
                style={styles.input}
                placeholder="Enter weight (kg)"
                keyboardType="numeric"
                value={weight}
                onChangeText={setWeight}
            />
```

```
            <TextInput
                style={styles.input}
                placeholder="Enter hight (m)"
                keyboardType="numeric"
                value={height}
                onChangeText={setHeight}
            />
            <Button title="Calculate" onPress={calculateBMI} />
            <Text style={styles.result}>{bmiResult}</Text>
        </View>
    );
};

const styles = StyleSheet.create({
    container: { flex: 1, justifyContent: 'center', alignItems: 'center', backgroundColor: '#98ff8a' },
    title: { fontSize: 20, fontWeight: "bold", position: "absolute", top: 50,color: "#ff0000" },
    subtitle: { fontSize: 16, position: "absolute", top: 80, color: "#ff6161", },
    input: { borderWidth: 1, borderColor: '#ccc', padding: 10, marginBottom: 10, width: '80%', borderRadius: 5 },
    result: { fontSize: 18, marginTop: 20 },
});

export default BMICalculator;
```

- คำอธิบายโปรแกรม

การตั้งค่า State

weight: เก็บค่าน้ำหนักที่ผู้ใช้กรอก.

height: เก็บค่าส่วนสูงที่ผู้ใช้กรอก.

BmiResult: เก็บผลลัพธ์ของค่า BMI พร้อมคำแนะนำ

ฟังก์ชัน calculateBMI

คำนวณ BMI

การทำงาน:

แปลงน้ำหนักและส่วนสูงจาก string เป็น float ด้วย parseFloat

ตรวจสอบค่าว่างหรือค่าที่ไม่ใช่ตัวเลข:

ถ้าผิดพลาด แสดงข้อความ "Enter valid values"

คำนวณค่า BMI:

ใช้สูตรคำนวณและจัดหมวดหมู่ BMI:

< 18.5: Underweight

18.5 - 24.9: Normal Weight

25 - 29.9: Overweight

≥ 30: Obesity

แสดงผลในรูปแบบ BMI: 22.86 (Normal Weight)

การตกแต่ง Styles

container:

จัดองค์ประกอบให้อยู่ตรงกลางหน้าจอ พร้อมพื้นหลังสีเขียวอ่อน

title และ subtitle:

ชื่อโปรแกรมและชื่อโปรแกรมเมอร์อยู่ด้านบนสุด

input:

ช่องกรอกข้อมูลมีขอบ, ระยะห่าง และมุมโค้ง

result:

แสดงผลลัพธ์ด้วยฟอนต์ขนาดใหญ่และระยะห่างที่เหมาะสม

- ผลลัพธ์ของโปรแกรม