การพัฒนาโปรแกรมประยุกต์สำหรับอุปกรณ์เคลื่อนที่
เรื่อง State และ Props ใน React Native

จัดทำโดย

นายปฏิวัฒน์ กฤษฎิ์สุภารัตน์ 031

เสนอ

ผศ. เสถียร จันทร์ปลา

รหัสวิชา CSD3201

ภาคเรียนที่ 2

คณะวิทยาศาสตร์และเทคโนโลยี

มหาวิทยาลัยราชภัฏสวนสุนันทา

2567

แบบฝึกหัดที่ 4

State และ Props ใน React Native

คำสั่ง

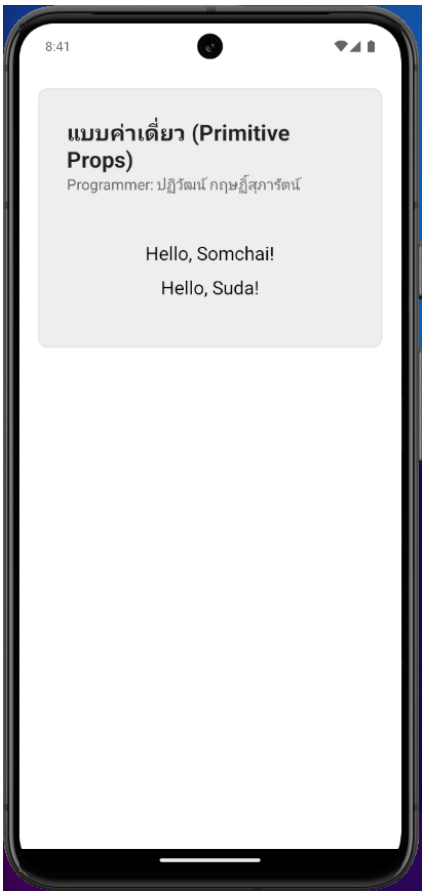
1. ส่งงานให้ตรงเวลา
2. จัดเอกสารตามรูปแบบการทำรายงาน
3. ห้ามลอกกัน

โปรแกรมในเอกสาร

1. GreetingApp.js
    - โปรแกรม

```
import React from "react";
import { View, Text } from "react-native";
import styles from "../../styles/styles";

const Greeting = ({ name }) => {
  return <Text style={styles.text}>Hello, {name}!</Text>;
};

const GreetingApp = () => {
  return (
    <View style={styles.programContainer}>
      <Greeting name="Somchai" />
      <Greeting name="Suda" />
    </View>
  );
};

export default GreetingApp;
```

- ผลลัพธ์ของโปรแกรม

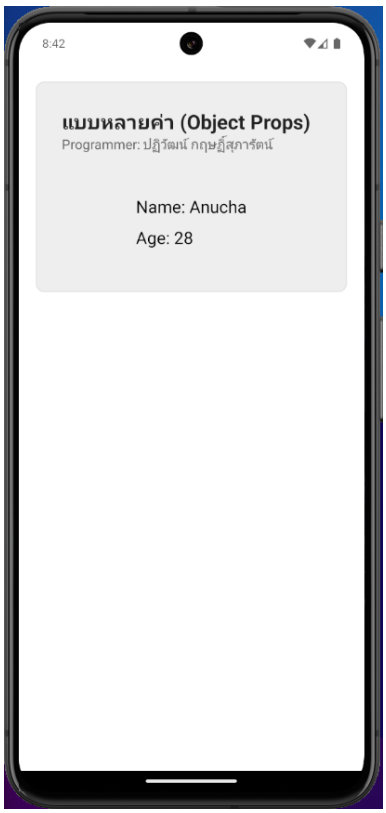


2. UserCardApp.js

- โปรแกรม

```
import React from "react";
import { View, Text } from "react-native";
import styles from "../../styles/styles";

const UserCard = ({ user }) => {
  return (
    <View>
      <Text style={styles.text}>Name: {user.name}</Text>
      <Text style={styles.text}>Age: {user.age}</Text>
    </View>
  );
};

const UserCardApp = () => {
  const userInfo = { name: "Anucha", age: 28 };

  return (
    <View style={styles.programContainer}>
      <UserCard user={userInfo} />
    </View>
  );
};

export default UserCardApp;
```

- ผลลัพธ์ของโปรแกรม

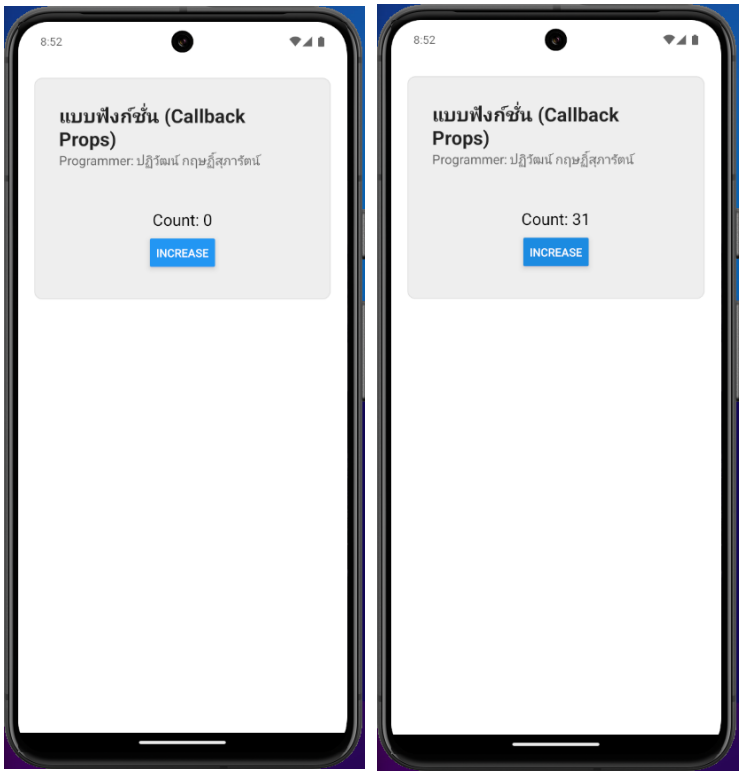


3. CounterApp.js

- โปรแกรม

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

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

const CounterApp = () => {
  const [count, setCount] = useState(0);

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

  return (
    <View style={styles.programContainer}>
      <Text style={styles.text}>Count: {count}</Text>
      <Counter onIncrease={increaseCount} />
    </View>
  );
};

export default CounterApp;
```

- ผลลัพธ์ของโปรแกรม

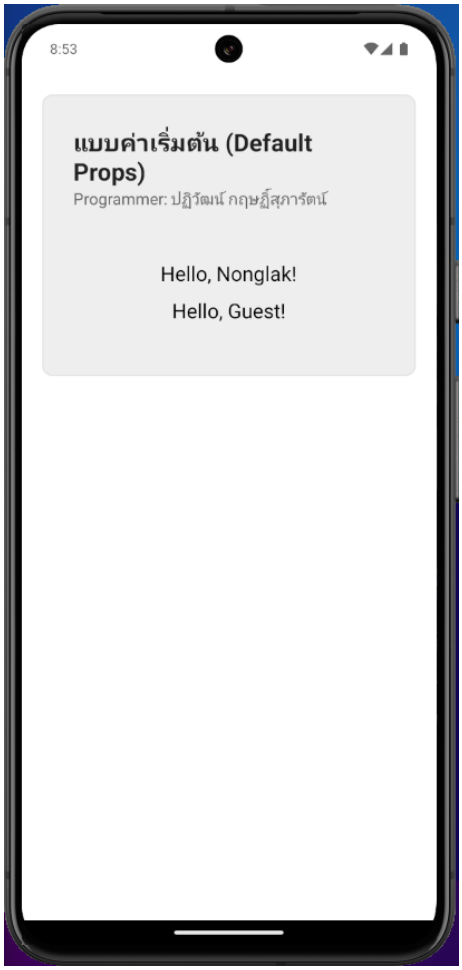


4. DefaultPropsApp.js

- โปรแกรม

```
import React from "react";
import { View, Text } from "react-native";
import styles from "../../styles/styles";

const Greeting = ({ name = "Guest" }) => {
  return <Text style={styles.text}>Hello, {name}!</Text>;
};

const DefaultPropsApp = () => {
  return (
    <View style={styles.programContainer}>
      <Greeting name="Nonglak" />
      <Greeting />
    </View>
  );
};

export default DefaultPropsApp;
```

- ผลลัพธ์ของโปรแกรม

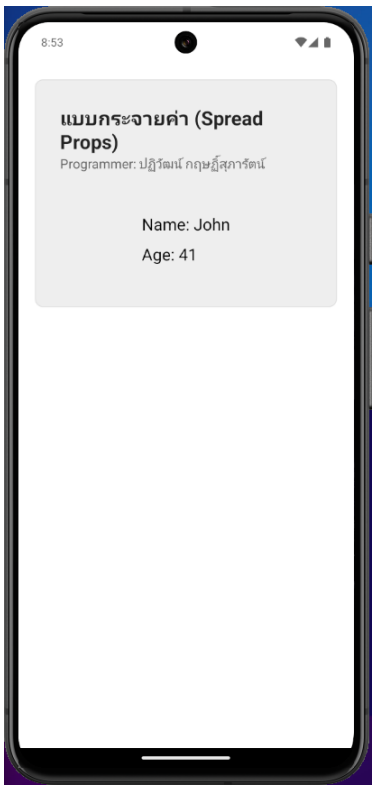


5. SpreadPropsApp.js

- โปรแกรม

```
import React from "react";
import { View, Text } from "react-native";
import styles from "../../styles/styles";

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

const SpreadPropsApp = () => {
  const userInfo = { name: "John", age: 41 };

  return (
    <View style={styles.programContainer}>
      <UserCard {...userInfo} />
    </View>
  );
};

export default SpreadPropsApp;
```

- ผลลัพธ์ของโปรแกรม

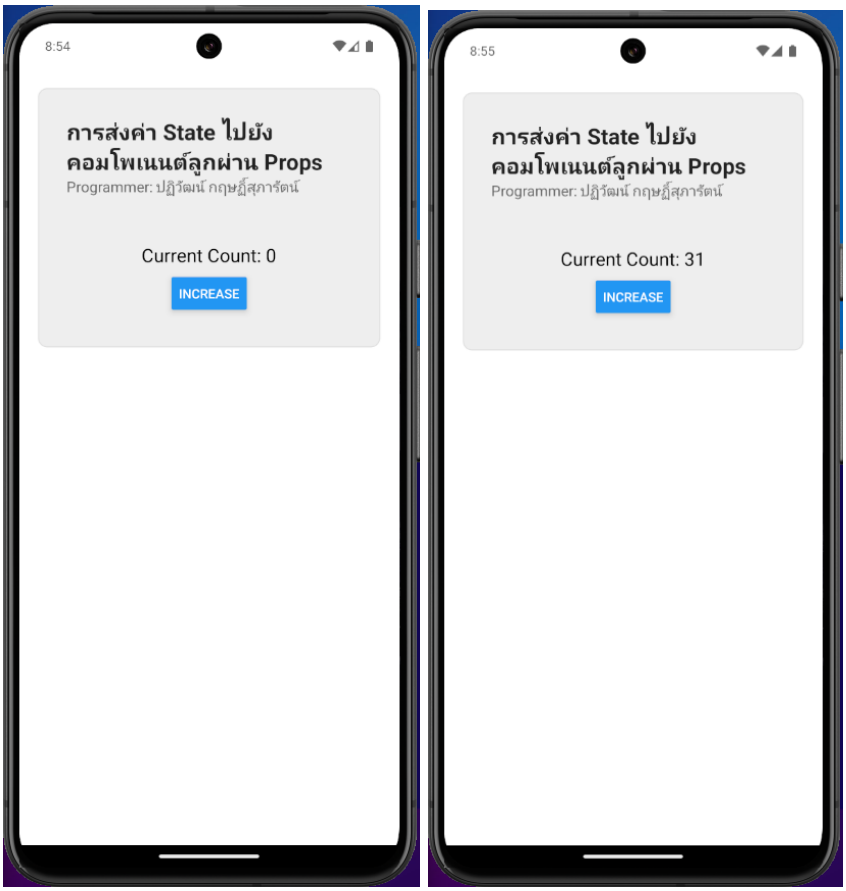


6. StateToPropsParent.js

- โปรแกรม

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

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

const StateToPropsParent = () => {
  const [count, setCount] = useState(0);

  return (
    <View style={styles.programContainer}>
      <CounterDisplay count={count} />
      <Button title="Increase" onPress={() => setCount(count + 1)} />
    </View>
  );
};

export default StateToPropsParent;
```

- ผลลัพธ์ของโปรแกรม

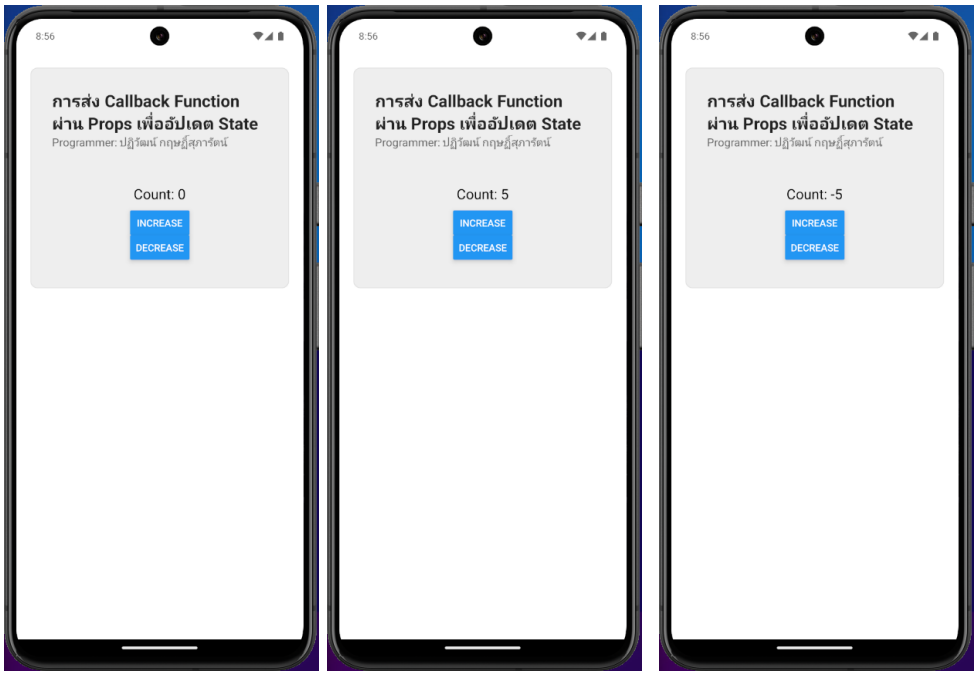


7. CallbackPropsParent.js

- โปรแกรม

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

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

const CallbackPropsParent = () => {
  const [count, setCount] = useState(0);

  return (
    <View style={styles.programContainer}>
      <Text style={styles.text}>Count: {count}</Text>
      <CounterControls
        onIncrease={() => setCount(count + 1)}
        onDecrease={() => setCount(count - 1)}
      />
    </View>
  );
};

export default CallbackPropsParent;
```

- ผลลัพธ์ของโปรแกรม

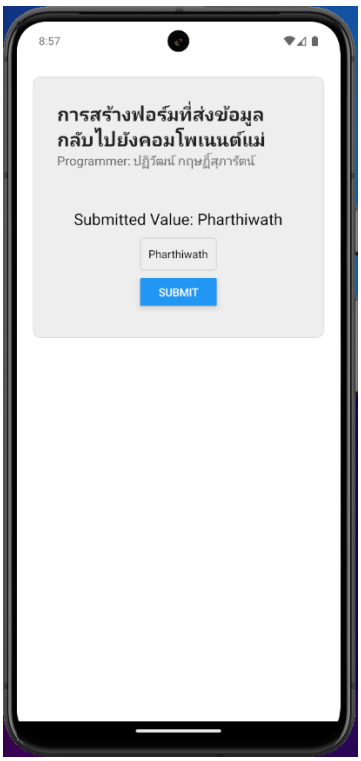


8. FormParent.js

- โปรแกรม

```
import React, { useState } from "react";
import { View, Text, TextInput, Button } from "react-native";
import styles from "../../styles/styles";

const InputForm = ({ onSubmit }) => {
  const [inputValue, setInputValue] = useState("");

  return (
    <View>
      <TextInput
        style={styles.input}
        value={inputValue}
        onChangeText={setInputValue}
        placeholder="Enter your name"
      />
      <Button title="Submit" onPress={() => onSubmit(inputValue)} />
    </View>
  );
};

const FormParent = () => {
  const [submittedValue, setSubmittedValue] = useState("");

  return (
    <View style={styles.programContainer}>
      <Text style={styles.text}>Submitted Value: {submittedValue}</Text>
      <InputForm onSubmit={(value) => setSubmittedValue(value)} />
    </View>
  );
};

export default FormParent;
```

- ผลลัพธ์ของโปรแกรม

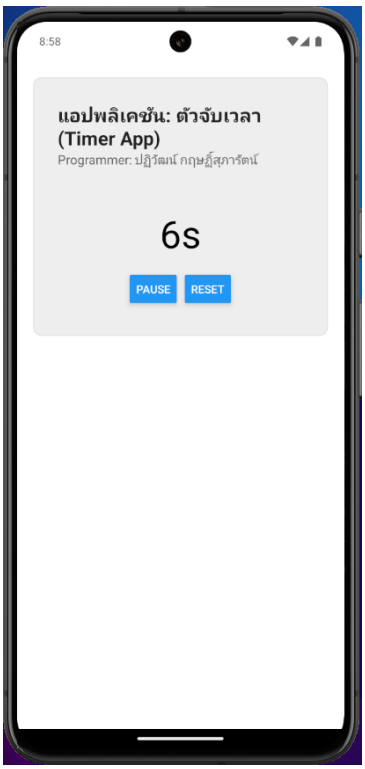


9. TimerApp.js

- โปรแกรม

```
import React, { useState, useEffect } from "react";
import { View, Text, Button } from "react-native";
import styles from "../../styles/styles";

const TimerApp = () => {
  const [time, setTime] = useState(0);

  const [isRunning, setIsRunning] = useState(false);

  useEffect(() => {
    let timer;
    if (isRunning) {
      timer = setInterval(() => {
        setTime((prevTime) => prevTime + 1);
      }, 1000);
    } else {
      clearInterval(timer);
    }

    return () => clearInterval(timer);
  }, [isRunning]);

  const resetTimer = () => {
    setTime(0);
    setIsRunning(false);
  };

  return (
    <View style={styles.programContainer}>
      <Text style={styles.timer}>{time}s</Text>
      <View style={styles.buttonContainer}>
        <Button
          title={isRunning ? "Pause" : "Start"}
          onPress={() => setIsRunning(!isRunning)}
        />
        <Button title="Reset" onPress={resetTimer} />
      </View>
    </View>
  );
};

export default TimerApp;
```

- ผลลัพธ์ของโปรแกรม

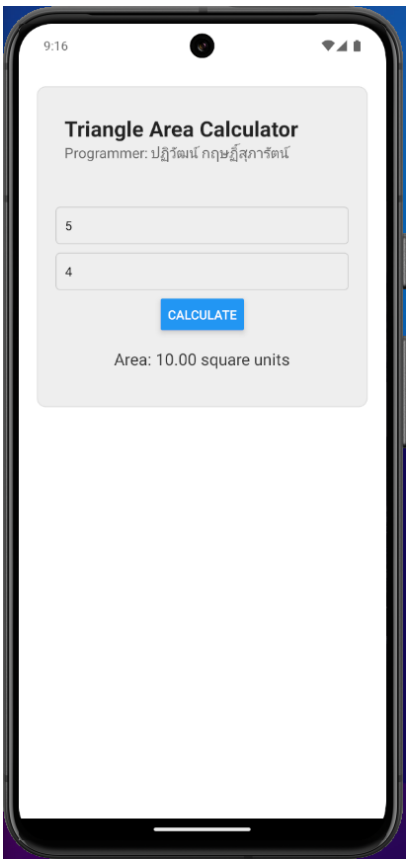


10. TriangleAreaApp.js

- โปรแกรม

```
import React, { useState } from "react";
import { View, Text, TextInput, Button } from "react-native";
import styles from "../../styles/styles";

const TriangleAreaApp = () => {
  const [base, setBase] = useState("");
  const [height, setHeight] = useState("");
  const [area, setArea] = useState(null);

  const calculateArea = () => {
    const baseValue = parseFloat(base);
    const heightValue = parseFloat(height);

    if (!isNaN(baseValue) && !isNaN(heightValue)) {
      const calculatedArea = 0.5 * baseValue * heightValue;
      setArea(calculatedArea.toFixed(2));
    } else {
      setArea("Invalid Input");
    }
  };

  return (
    <View style={styles.programContainer}>
      <TextInput
        style={styles.input}
        placeholder="Enter base"
        keyboardType="numeric"
        value={base}
        onChangeText={setBase}
      />

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

      <Button title="Calculate" onPress={calculateArea} />
      {area !== null && (
        <Text style={styles.result}>
          {isNaN(area) ? area : `Area: ${area} square units`}
        </Text>
      )}
    </View>
  );
};

export default TriangleAreaApp;
```

- ผลลัพธ์ของโปรแกรม



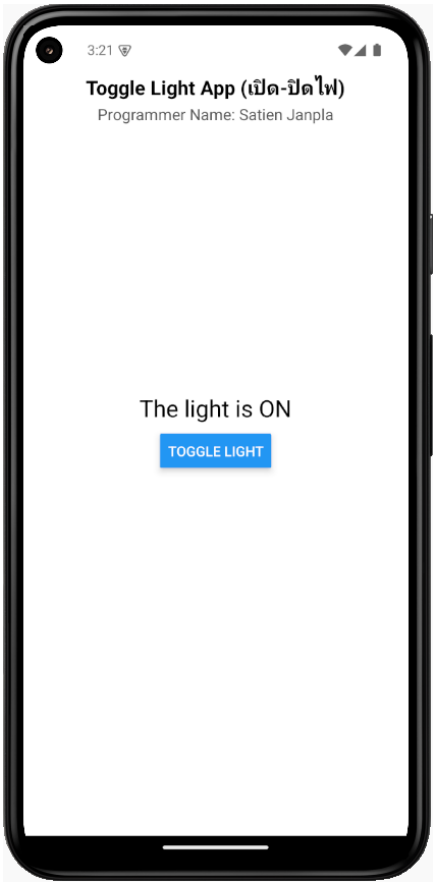
แบบฝึกหัด

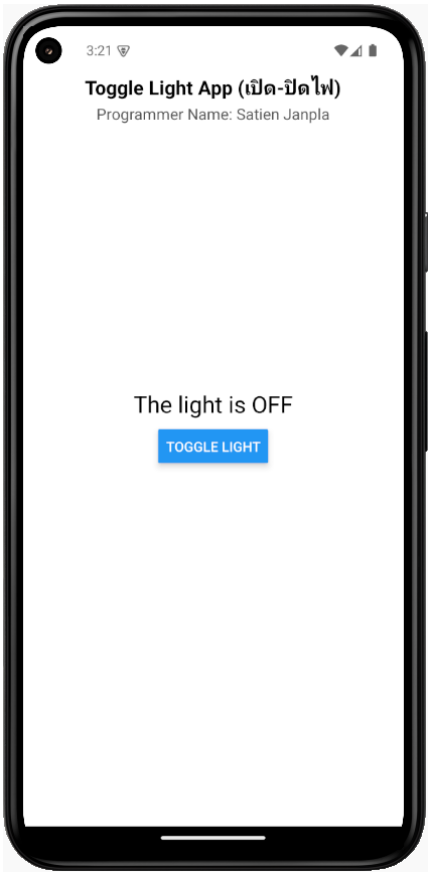
โจทย์ที่ 1: ชื่อแอป: Toggle Light App (เปิด-ปิดไฟ)

รายละเอียดการทำงานของแอป

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

ตัวอย่างผลลัพธ์





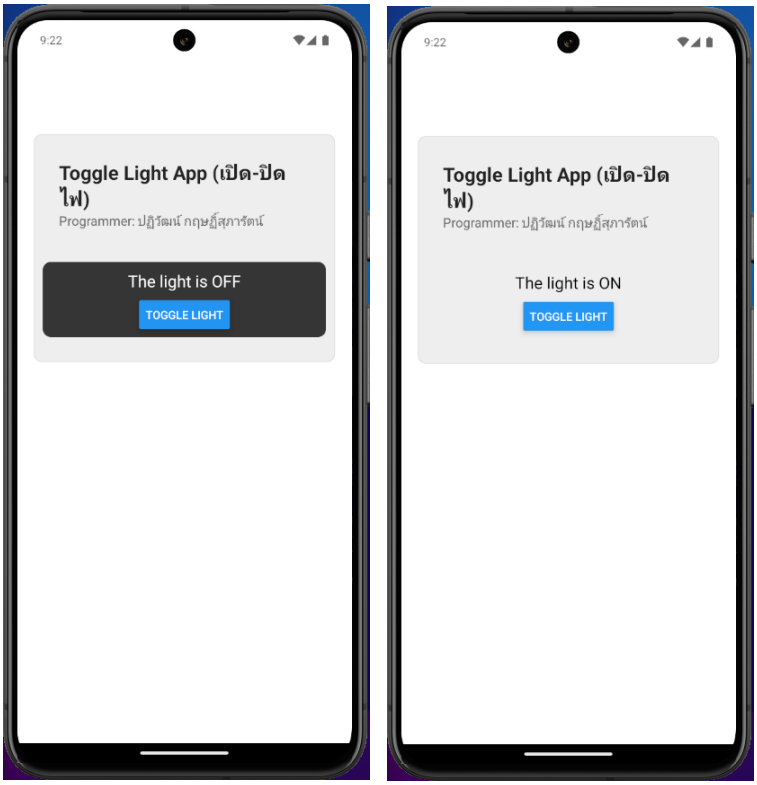
- โปรแกรม

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

const ToggleLight = () => {
  const [isOn, setIsOn] = useState(false);

  const toggleLight = () => {
    setIsOn(!isOn);
  };

  return (
    <View style={isOn ? styles.programContainer : styles.programContainerDark}>
      <Text style={isOn ? styles.text : styles.textDark}>
        The light is {isOn ? "ON" : "OFF"}
      </Text>
      <Button title="Toggle Light" onPress={toggleLight} />
    </View>
  );
};

export default ToggleLight;
```

- คำอธิบายโปรแกรม

- ใช้ useState เพื่อสร้างสถานะ isOn ที่เก็บสถานะของไฟ (เปิดหรือปิด)
- ฟังก์ชัน toggleLight จะสลับสถานะของ isOn เมื่อกดปุ่ม "Toggle Light"
- แสดงข้อความ "The light is ON" หรือ "The light is OFF" ตามสถานะของ isOn
- เปลี่ยนแปลงสไตล์ของ View และ Text ตามสถานะของไฟ (เปิดหรือปิด)

- ผลลัพธ์ของโปรแกรม

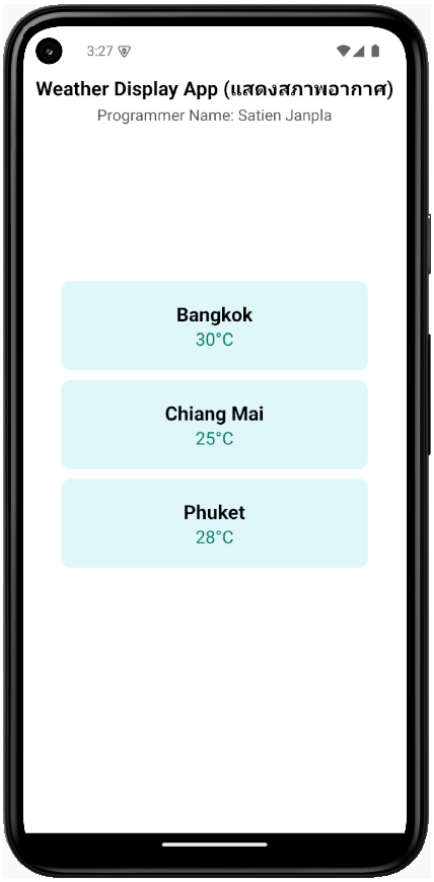


โจทย์ที่ 2: ชื่อแอป: Weather Display App (แสดงสภาพอากาศ)

รายละเอียดการทำงานของแอป

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

ตัวอย่างผลลัพธ์



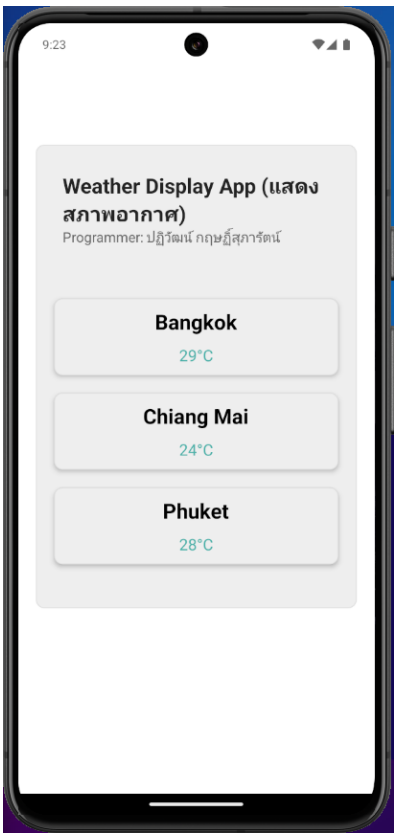
- โปรแกรม

```
import React from "react";
import { View, Text } from "react-native";
import styles from "../../styles/styles";

const weatherData = [
  { city: "Bangkok", temp: 29 },
  { city: "Chiang Mai", temp: 24 },
  { city: "Phuket", temp: 28 },
];

const WeatherCard = ({ city, temp }) => {
  return (
    <View style={styles.weatherCard}>
      <Text style={styles.textCity}>{city}</Text>
      <Text style={styles.textTemp}>{temp}°C</Text>
    </View>
  );
};

const WeatherDisplay = () => {
  return (
    <View style={styles.programContainer}>
      {weatherData.map((data, index) => (
        <WeatherCard key={index} {...data} />
      ))}
    </View>
  );
};

export default WeatherDisplay;
```

- คำอธิบายโปรแกรม

- ข้อมูลสภาพอากาศ (weatherData) ถูกเก็บในรูปแบบของอาร์เรย์ที่มีวัตถุแต่ละตัวเก็บชื่อเมือง (city) และอุณหภูมิ (temp)
- คอมโพเนนต์ WeatherCard รับพารามิเตอร์ city และ temp เพื่อแสดงชื่อเมืองและอุณหภูมิ
- คอมโพเนนต์ WeatherDisplay ทำการวนลูปผ่าน weatherData และสร้าง WeatherCard สำหรับแต่ละเมือง
- ข้อมูลสภาพอากาศจะแสดงในรูปแบบของการ์ดที่มีชื่อเมืองและอุณหภูมิ

- ผลลัพธ์ของโปรแกรม



โจทย์ที่ 3: ชื่อแอป: BMI Calculator (คำนวณดัชนีมวลกาย)

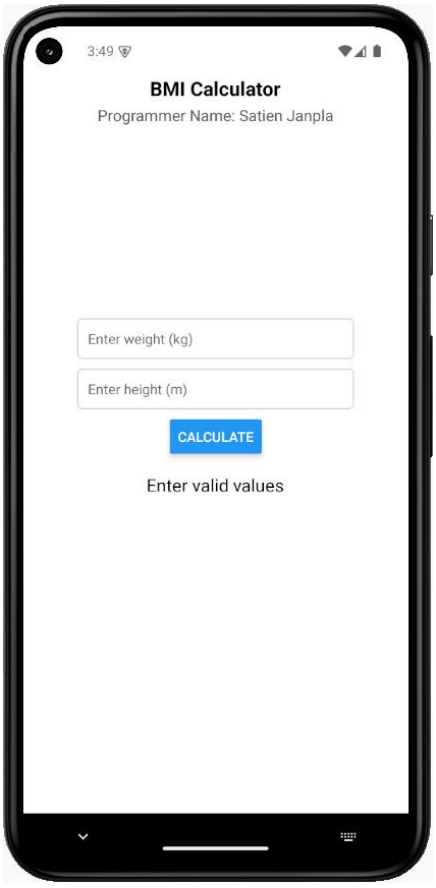
รายละเอียดการทำงานของแอป

- 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 } from "react-native";
import styles from "../../styles/styles";

const BMICalculator = () => {
  const [weight, setWeight] = useState("");
  const [height, setHeight] = useState("");
  const [bmi, setBMI] = useState(null);
  const [category, setCategory] = useState(null);

  const calculate = () => {
    const weightValue = parseFloat(weight);
    const heightValue = parseFloat(height);

    if (!isNaN(weightValue) && !isNaN(heightValue) && heightValue > 0) {
      const bmiValue = weightValue / (heightValue * heightValue);
      setBMI(bmiValue);

      let categoryValue = "";
      if (bmiValue < 18.5) {
        categoryValue = "Underweight";
      } else if (bmiValue < 24.9) {
        categoryValue = "Normal weight";
      } else if (bmiValue < 29.9) {
        categoryValue = "Overweight";
      } else {
        categoryValue = "Obesity";
      }
      setCategory(categoryValue);
    } else {
      alert("Enter valid values");
    }
  };

  return (
    <View style={styles.programContainer}>
      <TextInput
        style={styles.input}
        placeholder="Enter weight (kg)"
        keyboardType="numeric"
        value={weight}
        onChangeText={setWeight}
      />

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

```
      <Button title="Calculate" onPress={calculate} />

      {bmi !== null ? (
        <Text style={styles.result}>
          {`BMI: ${bmi.toFixed(2)} (${category})`}
        </Text>
      ) : (
        <Text style={styles.result}>{"Enter valid values"}</Text>
      )}
    </View>
  );
};

export default BMICalculator;
```

- คำอธิบายโปรแกรม

- ผู้ใช้กรอกน้ำหนัก (kg) และส่วนสูง (m) ลงใน TextInput
- เมื่อกดปุ่ม "Calculate" ฟังก์ชัน calculate จะถูกเรียกใช้
- ฟังก์ชัน calculate จะคำนวณค่า BMI จากน้ำหนักและส่วนสูงที่กรอก
- โปรแกรมจะแสดงค่า BMI และหมวดหมู่ (Underweight, Normal weight, Overweight, Obesity) ตามค่าที่คำนวณได้
- หากกรอกข้อมูลไม่ถูกต้อง จะแสดงข้อความเตือน "Enter valid values"

- ผลลัพธ์ของโปรแกรม