ชื่อ นายคุณากร โฆสิตสกุล
รหัสนักศึกษา 61070020

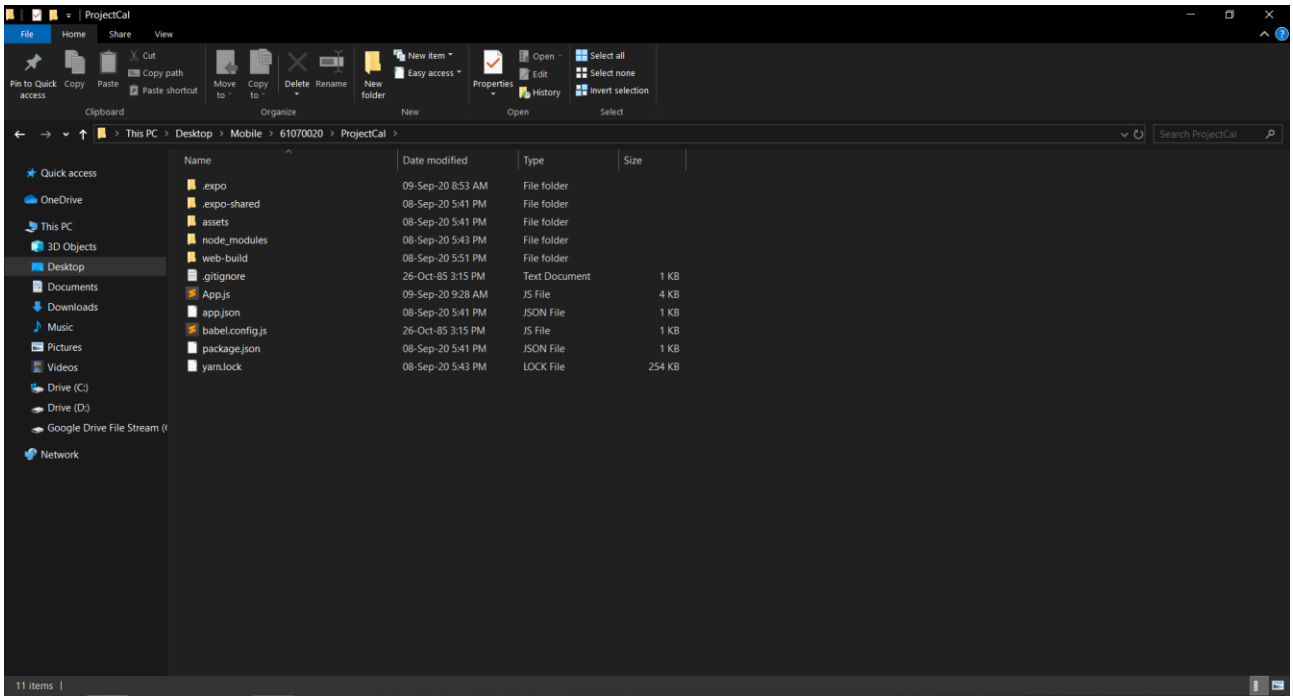
Lab 4 : JavaScript for Mobile Device Programming

1. ให้นักศึกษาทำการสร้าง New Project ใหม่ ชื่อ ProjectCal ใน Folder Mobile\<รหัสนักศึกษา>\ProjectCal แบบ Blank Project โดยใช้คำสั่ง

   Expo init <path\folder\StudentID\Project name>

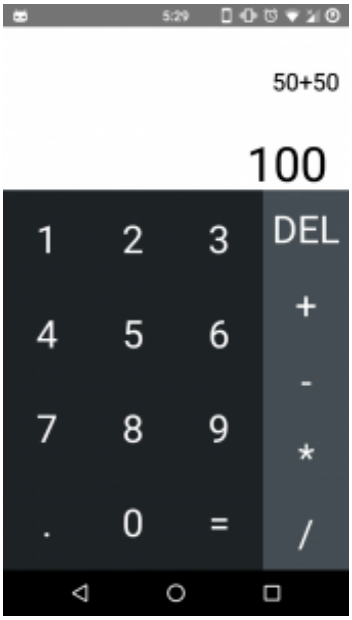
   บันทึกผลการทดลอง:

   ทำการสร้างโปรเจคใหม่ชื่อ ProjectCal ในโฟลเดอร์ 61070020



ชื่อ นายคุณากร โฆสิตสกุล
รหัสนักศึกษา 61070020

2. ให้นักศึกษาทำการสร้างโปรแกรมเครื่องคิดเลขบนอุปกรณ์เคลื่อนที่ ดังแสดงตามรูปข้างล่างนี้



บันทึกผลการทดลอง :

App.js

61070020 > ProjectCal > App.js > App > render

```
import { StatusBar } from 'expo-status-bar';
import React, { useState } from 'react';
import { StyleSheet, Text, View, TextInput, Button, TouchableOpacity } from 'react-native';

export default class App extends React.Component {
  constructor() {
    super()
    this.state = {
      resultText: '',
      calculateText: ''
    }
  }

  calculateResult = () => {
    const text = this.state.calculateText
    this.setState({ resultText: eval(text) })
  }

  checkLastChar = (char) => {
    const lastChar = this.state.calculateText.slice(-1)
    if (lastChar == char) {
      // pass (กด เครื่องหมาย ซ้ำไม่ได้)
    }
    else if (lastChar == '+' || lastChar == '-' || lastChar == '*' || lastChar == '/') {
      if (char == '.') {
        // pass (กด เครื่องหมายอื่น แล้วเปลี่ยนเป็น . ไม่ได้)
      } else {
        this.setState({ calculateText: this.state.calculateText.slice(0, this.state.calculateText.length - 1) + char })
      }
    }
    else {
      this.setState({ calculateText: this.state.calculateText + char })
    }
```

ชื่อ นายคุณากร โฆสิตสกุล
รหัสนักศึกษา 61070020

App.js

61070020 > ProjectCal > App.js > App > render

```
    }
  }

  buttonPressed = (text) => {
    if (text == 'DEL') {
      this.setState({
        // ลบตัวสุดท้าย
        calculateText: this.state.calculateText.slice(0, this.state.calculateText.length - 1)
        // ลบหมด
        // calculateText: this.state.calculateText.slice(-1,0)
      })
    } else if (text == '=') {
      this.calculateResult()
    } else if (text == '+' || text == '-' || text == '*' || text == '/' || text == '.') {
      this.checkLastChar(text)
    } else {
      this.setState({ calculateText: this.state.calculateText + text })
    }
  }

```

App.js

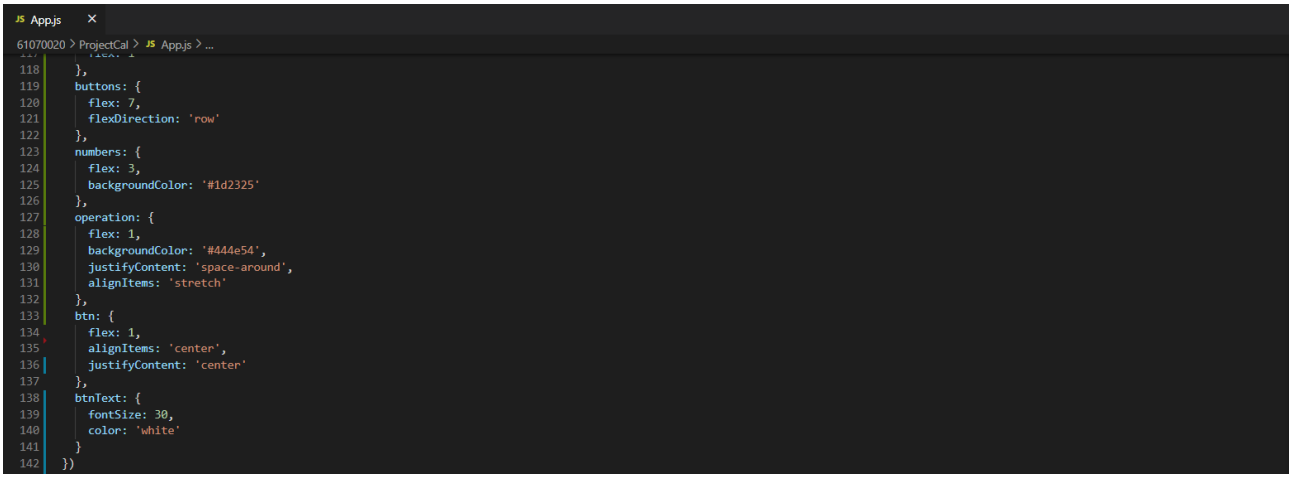
61070020 > ProjectCal > App.js > styles > result > alignItems

```
  }
}

render() {
  let rows = []
  let operate = []
  let nums = [[1, 2, 3], [4, 5, 6], [7, 8, 9], ['.', 0, '=']]
  let operations = ['DEL', '+', '-', '*', '/']
  for (let i = 0; i < 4; i++) {
    let row = []
    for (let j = 0; j < 3; j++) {
      row.push(<TouchableOpacity onPress={() => this.buttonPressed(nums[i][j])} style={styles.btn}><Text style={styles.btnText}>{nums[i][j]}</Text></TouchableOpacity>)
    }
    rows.push(<View style={styles.row}>{row}</View>)
  }
  for (let i = 0; i < 5; i++) {
    operate.push(<View style={styles.row}><TouchableOpacity onPress={() => this.buttonPressed(operations[i])} style={styles.btn}><Text style={styles.btnText}>{operations[i]}</Text></TouchableOpacity></View>)
  }

  return (
    <View style={styles.container}>
      <View style={styles.textCal}>
        <Text style={styles.calculateText}>{this.state.calculateText}</Text>
      </View>
      <View style={styles.result}>
        <Text style={styles.resultText}>{this.state.resultText}</Text>
      </View>
      <View style={styles.buttons}>
        <View style={styles.numbers}>
          {rows}
        </View>
        <View style={styles.operation}>
          {operate}
        </View>
      </View>
    </View>
  )
  }
}

```

App.js

61070020 > ProjectCal > App.js > ...

```
}

const styles = StyleSheet.create({
  container: {
    paddingTop: 30,
    flex: 1
  },
  result: {
    flex: 1,
    backgroundColor: 'white',
    justifyContent: 'center',
    alignItems: 'flex-end'
  },
  resultText: {
    fontSize: 30,
    paddingRight: 10
  },
  textCal: {
    flex: 1,
    backgroundColor: 'white',
    justifyContent: 'center',
    alignItems: 'flex-end'
  },
  calculateText: {
    fontSize: 30,
    paddingRight: 10
  },
  row: {
    flexDirection: 'row',
    flex: 1
  },
```



Source Code:

```
import { StatusBar } from 'expo-status-bar';
import React, { useState } from 'react';
import { StyleSheet, Text, View, TextInput, Button, TouchableOpacity } from 'react-
native';

export default class App extends React.Component {
  constructor() {
    super()
    this.state = {
      resultText: '',
      calculateText: ''
    }
  }

  calculateResult = () => {
    const text = this.state.calculateText
    this.setState({ resultText: eval(text) })
  }

  checkLastChar = (char) => {
    const lastChar = this.state.calculateText.slice(-1)
    if (lastChar == char) {
      // pass (กด เครื่องหมาย ซ้ำไม่ได้)
    }
    else if (lastChar == '+' || lastChar == '-
' || lastChar == '*' || lastChar == '/') {
      if (char == '.') {
        // pass (กด เครื่องหมายอื่น แล้วเปลี่ยนเป็น . ไม่ได้)
      } else {
```

```
        this.setState({ calculateText: this.state.calculateText.slice(0, this.state
.calculateText.length - 1) + char })
      }
    }
    else {
      this.setState({ calculateText: this.state.calculateText + char })
    }
  }

  buttonPressed = (text) => {
    if (text == 'DEL') {
      this.setState({
        // ลบตัวสุดท้าย
        calculateText: this.state.calculateText.slice(0, this.state.calculateText.l
ength - 1)
        // ลบหมด
        // calculateText: this.state.calculateText.slice(-1,0)
      })
    } else if (text == '=') {
      this.calculateResult()
    } else if (text == '+' || text == '-
' || text == '*' || text == '/' || text == '.') {
      this.checkLastChar(text)
    } else {
      this.setState({ calculateText: this.state.calculateText + text })
    }
  }

  render() {
    let rows = []
    let operate = []
    let nums = [[1, 2, 3], [4, 5, 6], [7, 8, 9], ['.', 0, '=']]
    let operations = ['DEL', '+', '-', '*', '/']
    for (let i = 0; i < 4; i++) {
      let row = []
      for (let j = 0; j < 3; j++) {
        row.push(<TouchableOpacity onPress={() => this.buttonPressed(nums[i][j])} s
tyle={styles.btn}><Text style={styles.btnText}>{nums[i][j]}</Text></TouchableOpacit
y>)
      }
      rows.push(<View style={styles.row}>{row}</View>)
    }
    for (let i = 0; i < 5; i++) {
```

```
      operate.push(<View style={styles.row}><TouchableOpacity onPress={() => this.b
uttonPressed(operations[i])} style={styles.btn}><Text style={styles.btnText}>{opera
tions[i]}</Text></TouchableOpacity></View>)
    }

    return (
      <View style={styles.container}>
        <View style={styles.textCal}>
          <Text style={styles.calculateText}>{this.state.calculateText}</Text>
        </View>
        <View style={styles.result}>
          <Text style={styles.resultText}>{this.state.resultText}</Text>
        </View>
        <View style={styles.buttons}>
          <View style={styles.numbers}>
            {rows}
          </View>
          <View style={styles.operation}>
            {operate}
          </View>
        </View>
      </View>
    )
  }
}

const styles = StyleSheet.create({
  container: {
    paddingTop: 30,
    flex: 1
  },
  result: {
    flex: 1,
    backgroundColor: 'white',
    justifyContent: 'center',
    alignItems: 'flex-end'
  },
  resultText: {
    fontSize: 30,
    paddingRight: 10
  },
  textCal: {
    flex: 1,
    backgroundColor: 'white',
    justifyContent: 'center',
```

```
    alignItems: 'flex-end'
  },
  calculateText: {
    fontSize: 30,
    paddingRight: 10
  },
  row: {
    flexDirection: 'row',
    flex: 1
  },
  buttons: {
    flex: 7,
    flexDirection: 'row'
  },
  numbers: {
    flex: 3,
    backgroundColor: '#1d2325'
  },
  operation: {
    flex: 1,
    backgroundColor: '#444e54',
    justifyContent: 'space-around',
    alignItems: 'stretch'
  },
  btn: {
    flex: 1,
    alignItems: 'center',
    justifyContent: 'center'
  },
  btnText: {
    fontSize: 30,
    color: 'white'
  }
})
```