

สารบัญ

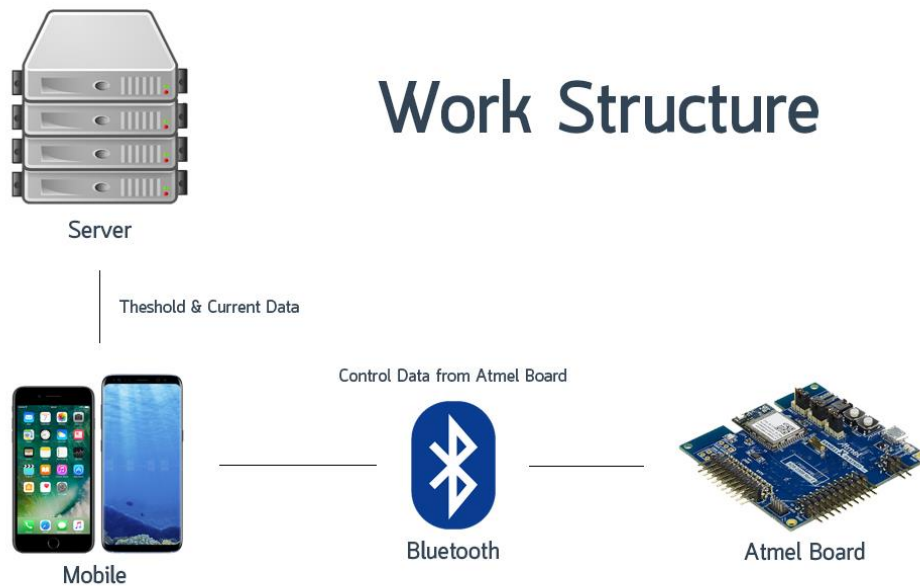
Contents

เป้าหมายของโครงการ	2
ภาพรวมของโครงการ.....	2
ภาพรวมของแอปพลิเคชัน.....	3
Flowchart กระบวนการถ่ายภาพ.....	4
Build Environment Setup and Dependencies	5
คำอธิบาย Source Code ที่สำคัญ	6
ผลการทดสอบ	13
สรุป.....	18

เป้าหมายของโครงการ

สร้างแอปพลิเคชันสำหรับมือถือ ทำการเชื่อมต่อกับอุปกรณ์ Breathe Max ผ่านสัญญาณ Bluetooth เพื่อรับ-ส่งข้อมูลในการทำกายภาพบำบัดของผู้ป่วย

ภาพรวมของโครงการ

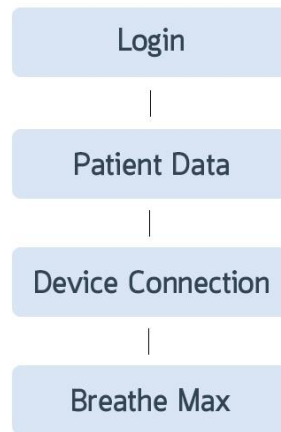


รูปภาพที่ 1 โครงสร้างของโครงการ

โครงการประกอบด้วย 3 ส่วนหลักคือ

1. **Server** ใช้ติดต่อเพื่อรับและส่งข้อมูลที่ใช้ในการทำกายภาพของผู้ป่วย
2. **Mobile** เป็นส่วนติดต่อกับผู้ป่วย แสดงผลการทำกายภาพ ข้อมูลสถานะของผู้ป่วยรวมถึงการเชื่อมต่อกับอุปกรณ์ทำกายภาพผ่านสัญญาณบลูทูธ
3. **Atmel Board** ใช้ในการเชื่อมต่อกับมือถือนวมถึงการรับข้อมูลจากเซ็นเซอร์แรงดันอากาศจากการทำกายภาพของผู้ป่วยส่งให้กับมือถือ

ภาพรวมของแอปพลิเคชัน

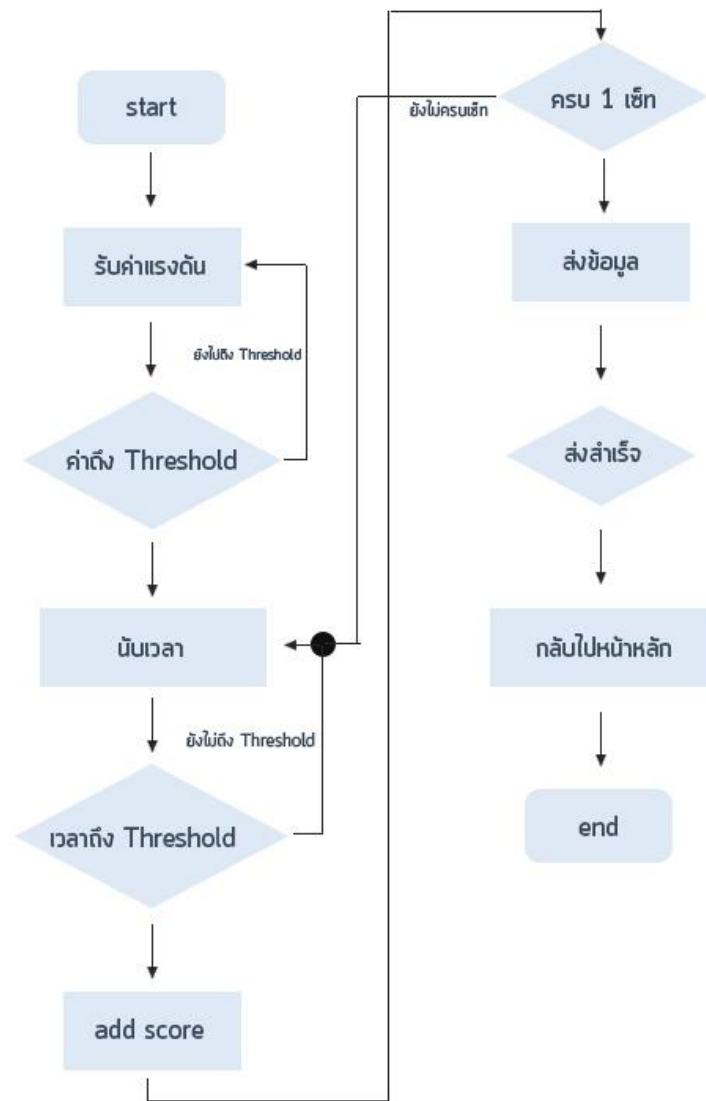


รูปภาพที่ 2 ภาพรวมของแอปพลิเคชัน

จากภาพด้านบนโครงสร้างภาพรวมจะประกอบ

1. **Login** เข้าใช้ระบบด้วยการกรอกข้อมูลรหัสผู้ป่วย
2. **Patient Data** เป็นหน้าหลักแสดงข้อมูลการถ่ายภาพในปัจจุบัน
3. **Device Connection** ทำการเชื่อมต่อกับอุปกรณ์ถ่ายภาพเพื่อเริ่มการปฏิบัติ
4. **Breathe Max** เมื่อเชื่อมต่ออุปกรณ์จะสามารถเริ่มถ่ายภาพได้จากหน้านี้
เมื่อเสร็จจะทำการส่งผลการถ่ายภาพสู่เซิร์ฟเวอร์และกลับสู่หน้า Patient Data

Flowchart ระยะเวลาการถ่ายภาพ



รูปภาพที่ 3 โฟลว์ชาร์ทของการถ่ายภาพ

Build Environment Setup and Dependencies

Dependencies

1. Java SDK
2. Android Studio + Android SDK
3. Nodejs + Node Package Control(NPM)
4. Ionic 2 Framework

Setup Environments

1. ติดตั้ง Nodejs เพื่อใช้ Node Package Control ในการติดตั้ง Ionic 2 Framework
ดาวน์โหลดและติดตั้งผ่าน <https://nodejs.org/en/download/>
2. ติดตั้ง Ionic 2 Framework ผ่าน npm
command -> run npm install -g ionic cordova
3. ติดตั้ง Java SDK
ดาวน์โหลดและติดตั้งผ่าน
<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>
เซตพาร์ท
https://www.mkyong.com/java/how-to-set-java_home-on-windows-10/
4. ติดตั้ง Android Studio พร้อมกับ Android SDK
ดาวน์โหลดและติดตั้งผ่าน
<https://developer.android.com/studio/index.html>
วิธีการติดตั้ง
<https://developer.android.com/studio/install.html>

คำอธิบาย Source Code ที่สำคัญ

Core Technology: Ionic 2 Framework

Ionic 2 Framework ทำงานอยู่บนภาษา Typescript ซึ่งถือเป็น Super Set ของ JavaScript ที่มีความสามารถกว้างกว่า ซึ่งมีโครงสร้างเป็นแบบ Component-Base คือมองส่วนต่างๆของหนึ่งหน้าในแอปพลิเคชันเป็นส่วนประกอบที่เกิดจากการนำมาเรียงต่อกันจนเกิดเป็นหน้าหนึ่งของแอปพลิเคชัน โดยแต่ละส่วนประกอบหรือ component จะประกอบด้วยไฟล์ 3 ชนิดคือ

Typescript ส่วนของการควบคุมการทำงานของ component

Html ส่วนของการแสดงผล(View)

Sass ส่วนของการตกแต่งส่วนการแสดงผล คล้ายกับ css แต่มีความสะดวกต่อการอ้างอิง element เป็นแบบ Hierarchy ได้

การเชื่อมต่อกับเซิร์ฟเวอร์

เนื่องจากการติดต่อรับส่งข้อมูลกับเซิร์ฟเวอร์เกิดขึ้นแบบ Asynchronous โค้ดของการเชื่อมต่อจึงไม่ควรอยู่ภายในแต่ละ component จึงต้องแยกออกมาเขียนเป็น Service และชนิดของข้อมูลที่ได้รับจากเซิร์ฟเวอร์เป็นแบบ Xml Object ซึ่งยังไม่สามารถนำมาใช้งานได้ จึงต้องทำการแปลงเป็น Json สำหรับการนำไปใช้ภายในแอปพลิเคชัน ต้องใช้ไลบรารี X2JS เข้ามาช่วยได้โค้ดดังนี้

```
parseXtoJ(xml){  
  let l = JSON.parse(JSON.stringify(xml));  
  let parser : any = new X2JS();  
  let json = parser.xml2js(l._body);  
  return json;  
}
```

รูปภาพที่ 4 โค้ดแปลงชนิดของข้อมูล

ตัวอย่างการรับค่า Threshold จากเซิร์ฟเวอร์

```
getTheshold(username){
  let url = "http://www.nbtcrehab.eng.psu.ac.th:8080/"+
    "ConfigurationServer/webresources/getthreshold?Patient_ID=" + username +
    "&Device_ID=7";
  let header = new Headers({'Content-Type': 'text/plain'});
  let options = new RequestOptions({headers: header});
  let response = this.http.get(url, options)
    .map(res => this.parseXtoJ(res));
  // return XML json
  return response;
}
```

รูปภาพที่ 5 การเรียกข้อมูลภายใน service

เนื่องจากการเชื่อมต่อเป็นแบบ GET Method การใส่ข้อมูลจากจึงอยู่ภายใน url ได้เลย
จากนั้นใช้ http เพื่อเรียก GET และใส่ url ลงไปโดย response จะส่งค่ากลับมาในรูปของ xml
object จึงต้องเปลี่ยนให้เป็น json ด้วยฟังก์ชัน parseXtoJ()

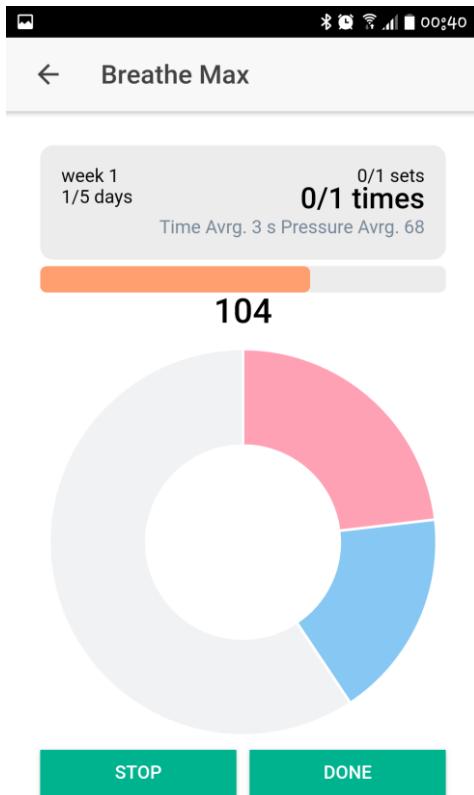
ทำการเรียกใช้ Service ภายในไฟล์ Typescript

```
getCurrentData(){
  this.patientService.getCurrent().subscribe(
    data=>{
      this.treatInfo = data.current;
    },
    err=>{
      console.log(err);
    })
}
```

รูปภาพที่ 6 ใช้ฟังก์ชันของ service ภายใน typescript

การเรียกใช้ข้อมูลจาก service จะรีเทิร์นเป็น Observable ซึ่งประกาศผ่านการ .subscribe ภายใน
ประกอบด้วย การ ข้อมูลและความข้อความหากเกิดความผิดพลาดคือ data และ err ตามลำดับ

โค้ดส่วนที่เกี่ยวข้องกับการทำกายภาพ



รูปภาพที่ 7 หน้าทำกายภาพ

หน้าการทำกายภาพ

ประกอบด้วย 2 ส่วนประกอบหลักคือ

1. เกทเวลาเมื่อสามารถค้างแรงดันได้
2. แผนภาพแรงดันแสดงค่าแรงดันที่ส่งมาจากอุปกรณ์

การทำงานทั้งหมดประกอบด้วย

1. เริ่มรับค่าแรงดันจากอุปกรณ์

```
this.ble.startNotification(  
  this.device.peripheralId,  
  this.device.service,  
  this.device.measurement)  
.subscribe(  
  buffer =>{  
    let dd = new Uint8Array(buffer);  
    this.d = "" + dd[1];  
    this.calPressAvg(dd[1]);  
    this.updateChart(dd[1]);  
    this.checkForCountDown(dd[1]);  
  },  
  err =>{  
    console.log("ERROR FROM STARTNOTIFICATION " + err);  
  }  
);
```

รูปภาพที่ 8 เริ่มรับค่าแรงดัน

ใช้โมดูล ble เพื่อเรียกค่าฟังก์ชัน startNotification เพื่อรับค่าจากอุปกรณ์ จากนั้นเมื่อได้รับค่าในแต่ละครั้งให้ทำการ คำนวณค่าเฉลี่ยแรงดัน อัปเดตแผนภาพแรงดันและเช็คการนับเวลา

2. นำค่าที่ได้ไปอัปเดตแผนภาพแรงดัน

```
updateChart(value){
  if(value <= 59){
    this.doughnutChartData[0] = value;
    this.doughnutChartData[1] = 0;
    this.doughnutChartData[2] = 0;
  }else if(value < 120){
    this.doughnutChartData[0] = 59;
    this.doughnutChartData[1] = value-59;
    this.doughnutChartData[2] = 0;
  }else if(value <= 255){
    this.doughnutChartData[0] = 59;
    this.doughnutChartData[1] = 120 - 59;
    this.doughnutChartData[2] = value - 120;
  }
  this.doughnutChartData[3] = 255 - value;
  this.chart.chart.update();
}
```

รูปภาพที่ 9 ปรับแผนภาพตามค่าที่รับมา

ค่าที่ได้ต้องมาอัปเดตในแผนภาพโดยที่จะต้องจัดแต่งแต่ละสีภายในแผนภาพใหม่ทุกครั้ง

3. ตรวจสอบแรงดันว่าผ่านค่าแรงดัน Threshold ที่กำหนดหรือไม่ หากผ่านจะเริ่มนับเวลาตาม Threshold และเพิ่มแถบเวลาสีส้มดังรูป

```
checkForCountDown(value){
  if(value >= 80){
    if(!this.isCountDown){
      this.countDown(3);
      this.isCountDown = true;
      console.log("isCountDown!");
    }
  }else{
    this.wi = 0 + "%";
    this.isCountDown = false;
    console.log("is not CountDown");
    clearInterval(this.interval);
  }
}
```

รูปภาพที่ 10 เช็คดาวน์เวลา

หากค่าที่ได้จากอุปกรณ์มากกว่าหรือเท่ากับค่าที่กำหนด ให้ทำการนับถอยหลังเวลาได้ แต่หากไม่ถึงให้ทำการรีเซ็ตเททเวลาและการนับใหม่

4. เมื่อการนับเวลาครบตามที่กำหนด ทำการเพิ่มคะแนนให้กับ จำนวนครั้ง เซ็ต วัน และสัปดาห์ตามลำดับ

```
countDown(sec){
  let milli = 0;
  let percent = 0;
  this.interval = setInterval(()=>{
    milli += 1000;
    this.wi = percent + "%";
    if(milli == (sec * 1000)){
      this.numOfPassTime++;
      percent = 100;
      console.log("success");
      this.calPassTimeAvg(sec);
      this.countScore();
    }else if(milli < (sec * 1000)){
      percent = (milli / (sec * 1000)) * 100;
    }else{
      percent = 100;
      this.calPassTimeAvg(1);
    }
  }, 1000);
}
```

รูปภาพที่ 11 การนับเวลา

ทุก 1 วินาทีที่จะเพิ่มขนาดของเกทเวลา และเมื่อครบตามเวลาที่กำหนด จะคำนวณค่าเฉลี่ยของเวลาที่ค้างแรงดันได้ จับนั่นเพิ่มคะแนน

```
countScore(){
  this.treatInfo.Time_NO++;
  if (this.treatInfo.Time_NO == this.treatInfo.NoTimeinSet){
    this.treatInfo.Time_NO = 0;
    this.treatInfo.Set_NO++;
    this.isDone = true;
    this.endDate = new Date();
  }
  if (this.treatInfo.Set_NO == this.treatInfo.NoSetinDay){
    this.treatInfo.Set_NO = 0;
    this.treatInfo.Day_NO++;
  }
  if (this.treatInfo.Day_NO == this.treatInfo.NoDayinWeek){
    this.treatInfo.Day_NO = 0;
    this.treatInfo.WeekNO++;
  }
}
```

รูปภาพที่ 12 การเพิ่มคะแนน

การนับคะแนนจะเริ่มจากการเพิ่มค่าจำนวนครั้งได้จนถึงการเพิ่มสัปดาห์ โดยเมื่อครบทุกๆ หนึ่งเซ็ท จะจับเวลาที่ทำเสร็จด้วย

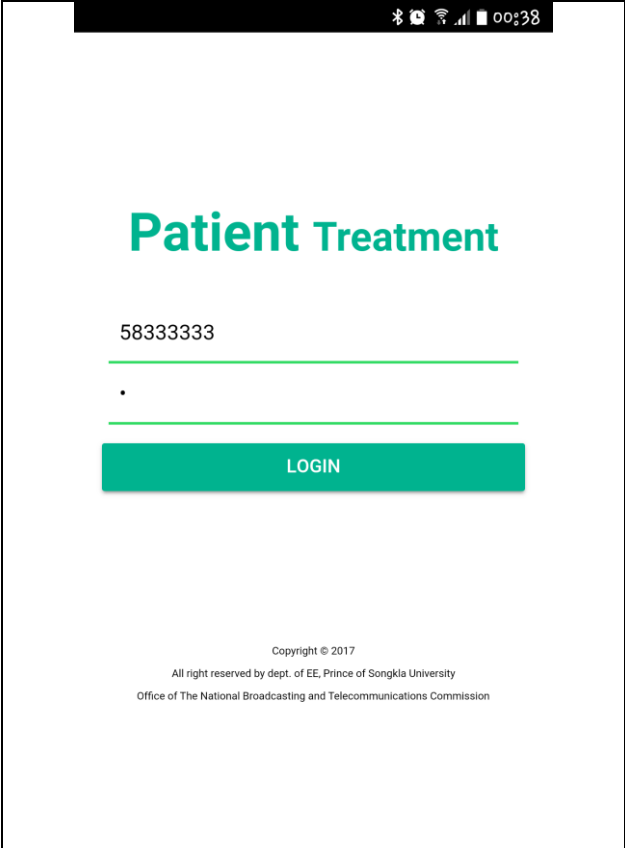
5. เมื่อครบ 1 เซ็ททำการส่งข้อมูลไปยังเซิร์ฟเวอร์

```
done(){
  this.storage.get('PatientID').then(userId=>{
    this.treatInfo["pressAvrg"] = this.pressAvrg;
    this.treatInfo["passTimeAvrg"] = this.passTimeAvrg;
    this.treatInfo["sumOfPassTime"] = this.sumOfPassTime;
    this.patientService.sendResult(
      userId,
      this.treatInfo,
      this.startDate.getTime(),
      this.endDate.getTime()
    ).subscribe(
      data=>{
        console.log(data);
        this.toast("Good Job!, You Did it", 3);
        this.navCtrl.popToRoot().then(()=>console.log("go to root"));
      },
      err=>{
        this.toast("Can't connect to server", 3);
      }
    )
  })
}
```

รูปภาพที่ 13 ทำการส่งข้อมูลเมื่อเสร็จ

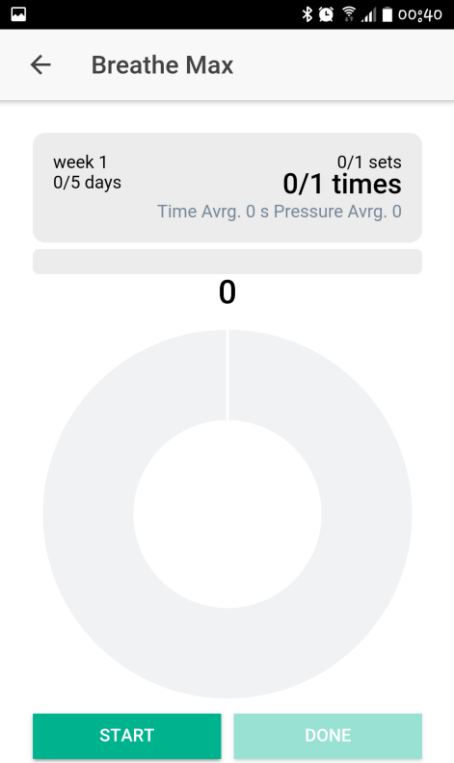
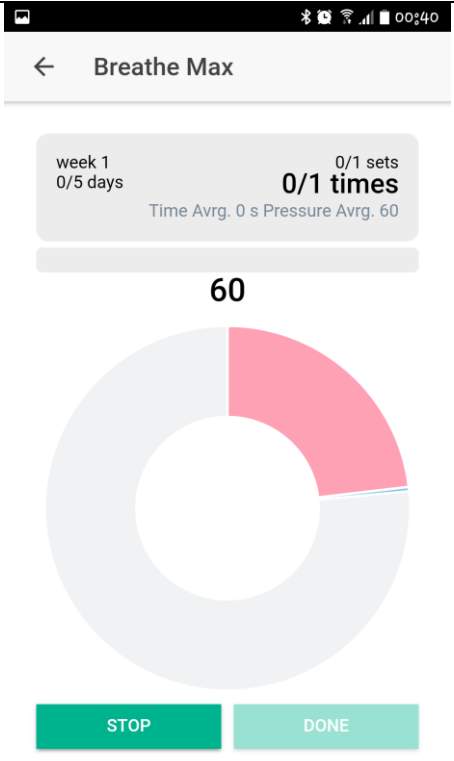
เมื่อครบ 1 เซ็ทจะส่งค่าทำการรวมข้อมูลที่ server ต้องการจากนั้นส่งผ่าน sendResult และ
กลับไปยังหน้า patientData ซึ่งเป็นหน้าหลักของแอปพลิเคชัน

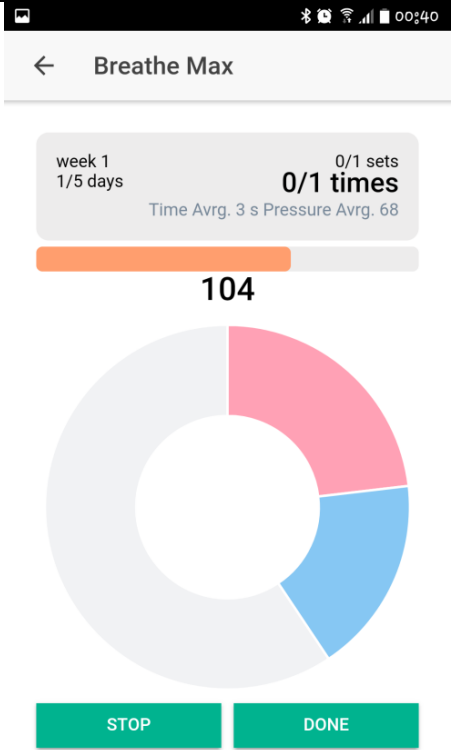
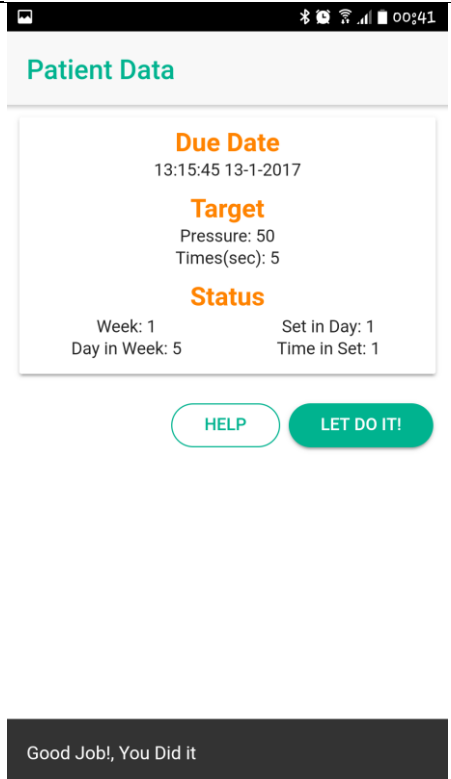
ผลการทดสอบ

	<p>การลงชื่อเข้าใช้ทดลอง 3 เหตุการณ์ ไม่ได้เชื่อมต่ออินเทอร์เน็ตกรอกข้อมูลผิด เชื่อมต่อได้สำเร็จ ซึ่ง 3 เหตุการณ์สามารถรับรู้ และรองรับไว้ได้แล้ว</p>
--	---

	<p>หน้าการช่วยเหลือจะขึ้นมาอัตโนมัติเมื่อเข้าใช้ครั้งแรกเท่านั้นซึ่งเป็นไปได้ตามกำหนด</p>
	<p>สามารถเรียกข้อมูลจากเซิร์ฟเวอร์และนำมาแสดงผลได้ การเรียกหน้าช่วยเหลือสามารถทำได้ และสามารถเข้าสู่ขั้นตอนการเชื่อมต่ออุปกรณ์ได้แล้ว</p>

 <p>← Device Connection</p> <p>connect your device...</p> <p>CONNECT</p>	<p>เข้าสู่หน้าการเชื่อมต่อกับอุปกรณ์</p>
 <p>← Device Connection</p> <p>Connected</p> <p>NEXT</p>	<p>เมื่อพบอุปกรณ์และกรอกรหัสผ่านเรียบร้อยแล้ว ปุ่ม next จะปรากฏเพื่อไปหน้าของ การทำกายภาพ</p>

 <p>week 1 0/5 days</p> <p>0/1 sets 0/1 times Time Avrg. 0 s Pressure Avrg. 0</p> <p>0</p> <p>START DONE</p>	<p>หน้าทำกายภาพสามารถรับข้อมูลมาแสดงได้</p>
 <p>week 1 0/5 days</p> <p>0/1 sets 0/1 times Time Avrg. 0 s Pressure Avrg. 60</p> <p>60</p> <p>STOP DONE</p>	<p>เมื่อกด start จะเริ่มรับค่าแรงดันจากอุปกรณ์ และแสดงผลผ่านกราฟได้</p>

	<p>เมื่อค่าแรงดันเกณฑ์เวลาจะเพิ่มขึ้นได้จนครบเวลาที่กำหนด</p>
	<p>เมื่อกดปุ่ม done ข้อมูลจะถูกส่งขึ้นสู่เซิร์ฟเวอร์ แสดงข้อความหากส่งสำเร็จและกลับสู่หน้า Patient Data</p>

สรุป

การพัฒนาแอปพลิเคชันสามารถนำมาใช้งานได้ตามวัตถุประสงค์และมีความสามารถที่ใช้งานได้ตามปกติรวมถึงการรับ-ส่งข้อมูล รองรับความผิดพลาดที่อาจเกิดขึ้นจากผู้ใช้งานในระดับหนึ่ง