แอปพลิเคชันแจ้งเหตุฉุกเฉินสำหรับผู้ป่วยโรคเรื้อรัง

CHRONIC PATIENT EMERGENCY APPLICATION

โดย

นายเผ่าพันธุ์ ทองอินทร์

นายสหัสวรรษ จันทนวรานนท์

นางสาวอัฐภิญญา ทรัพย์พันธุ์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2564

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แอปพลิเคชันแจ้งเหตุฉุกเฉินสำหรับผู้ป่วยโรคเรื้อรัง

CHRONIC PATIENT EMERGENCY APPLICATION

โดย

นายเผ่าพันธุ์	ทองอินทร์	61010667
นายสหัสวรรษ	จันทนวรานนท์	61011096
นางสาวอัฐภิญญา	ทรัพย์พันธุ์	61011435

อาจารย์ที่ปรึกษา

ดร.พีระเมศร์ โชติกวีกิจญาดา

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2564

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์ปีการศึกษา 2564

ภาควิชาวิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง แอปพลิเคชันแจ้งเหตุฉุกเฉินสำหรับผู้ป่วยโรคเรื้อรัง

CHRONIC PATIENT EMERGENCY APPLICATION

ผู้จัดทำ

1. นายเผ่าพันธุ์ ทองอินทร์ 61010667
2. นายสหัสวรรษ จันทนวรานนท์ 61011096
3. นางสาวอัฐภิญญา ทรัพย์พันธุ์ 61011435

..... **อาจารย์ที่ปรึกษา**

(ดร.พีระเมศร์ โชติกวีกิจญาดา)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

การดำเนินปริญญานิพนธ์เรื่อง “แอปพลิเคชันแจ้งเหตุฉุกเฉินสำหรับผู้ป่วยโรคเรื้อรัง” จะไม่สามารถสำเร็จลุล่วงไปด้วยดีหากไม่ได้รับความช่วยเหลือ และความอนุเคราะห์อย่างดียิ่งจาก ดร.พีระเมศร์ โชติกวีกิจญาดา ที่กรุณาให้คำแนะนำ คำปรึกษา และแนวทางการแก้ไขปัญหาที่เป็นประโยชน์ต่อการศึกษา ค้นคว้าวิจัยให้ปริญญานิพนธ์นี้สำเร็จสมบูรณ์ยิ่งขึ้น รวมถึงสนับสนุนสถานที่ เครื่องมือ และอุปกรณ์ต่าง ๆ ที่ใช้ระหว่างการจัดทำปริญญานิพนธ์

ขอขอบคุณคณาจารย์และเจ้าหน้าที่ประจำภาควิชาวิศวกรรมโทรคมนาคม คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังทุกท่าน ที่ได้อบรมสั่งสอนประสิทธิ์วิชา ความรู้ กำลังใจ และประสบการณ์ให้แก่ผู้จัดทำ

ขอกราบขอบพระคุณ บิดา มารดา ครอบครัว พี่ ๆ และ เพื่อน ๆ ที่มอบความรัก ความห่วงใย และเป็นกำลังใจที่สำคัญเสมอมาและที่สำคัญคือการสนับสนุนให้โอกาสทางการศึกษาอันมีค่ายิ่งแก่ผู้จัดทำ

นายเผ่าพันธุ์ ทองอินทร์
นายสหัสวรรษ จันทนวรานนท์
นางสาวอัฐภิญญา ทรัพย์พันธุ์
ผู้จัดทำ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แอปพลิเคชันแจ้งเหตุฉุกเฉินสำหรับผู้ป่วยโรคเรื้อรัง
CHRONIC PATIENT EMERGENCY APPLICATION

โดย นายเผ่าพันธุ์ ทองอินทร์ 61010667
นายสหัสวรรษ จันทนวรานนท์ 61011096
นางสาวอัฐภิญญา ทรัพย์พันธุ์ 61011435

อาจารย์ที่ปรึกษา ดร.พีระเมศร์ โชติกวีกิจญาดา

บทคัดย่อ

โรคเรื้อรัง (CHRONIC DISEASE) คือโรคที่เป็นยาวนานและมีการเปลี่ยนแปลงไปอย่างช้าๆ ซึ่งโรคเรื้อรังนับเป็นสาเหตุของการเสียชีวิตลำดับต้น ๆ และเมื่อมีอาการของโรคเกิดขึ้นแล้ว ก็มักจะเกิดการเรื้อรังของโรคด้วย ประเทศไทยมีจำนวนของผู้สูงอายุจำนวนมากที่ป่วยเป็นโรคเรื้อรัง ความก้าวหน้าทางการแพทย์ในปัจจุบันทำให้อัตราการเสียชีวิตของผู้ป่วยเรื้อรังจากภาวะวิกฤตลดลง ผู้ป่วยเรื้อรังที่รอดจากภาวะวิกฤตและไม่สามารถดูแลตัวเองได้มีจำนวนมาก และต้องการการฟื้นฟูสภาพ ซึ่งถ้าอยู่รักษาในโรงพยาบาลจนดีขึ้นจะต้องใช้เวลานานมากและมีค่าใช้จ่ายที่สูง ผู้จัดทำจึงเล็งเห็นปัญหาการติดตามอาการของผู้ป่วยโรคเรื้อรังเมื่อรับการรักษาต่อที่บ้าน อาจเกิดเหตุฉุกเฉินหรือภาวะอาการกำเริบต่างๆที่อาจส่งผลต่อชีวิต ปริญญานิพนธ์นี้จึงได้ประยุกต์ใช้เทคโนโลยีการสื่อสารทางไกล WI-FI ซึ่งเป็นเทคโนโลยีที่ที่ช่วยให้อุปกรณ์อิเล็กทรอนิกส์ในการแลกเปลี่ยนข้อมูล หรือการเชื่อมต่ออินเทอร์เน็ตแบบไร้สาย สามารถเชื่อมโยงอุปกรณ์อิเล็กทรอนิกส์ที่สามารถตรวจวัดอาการต่างๆของผู้ป่วยโรคเรื้อรังเช่น อุปกรณ์ตรวจวัดชีพจร อุปกรณ์ตรวจจับการหกล้ม ร่วมกับระบบ GPS เพื่อระบุตำแหน่งของผู้ป่วยโรคเรื้อรัง เพื่อส่งแจ้งเตือนอาการที่วิกฤต หรือผิดปกติให้กับผู้ดูแลผู้ป่วย หรือ โรงพยาบาลได้ เพื่อส่งเสริมการตรวจสอบผู้ป่วยและการส่งมอบการดูแลรักษาได้อย่างทันท่วงที นอกจากนี้ยังสามารถพัฒนาเทคโนโลยีเหล่านี้เข้ากับแอปพลิเคชัน ที่สามารถระบุข้อมูลของผู้ป่วยได้อย่างเรียลไทม์ และยังสามารถเก็บรวบรวมข้อมูลเพื่อนำไปวิเคราะห์เพื่อการรักษาต่อไปอีกด้วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ABSTRACT

CHRONIC DISEASE is a disease that has been present for a long time and is slowly progressing. which chronic disease is the leading cause of death, and when the disease has already occurred It is frequently a long-term illness. Thailand has a large population of elderly people who are afflicted with chronic diseases. The mortality rate of chronic patients suffering from critical conditions has decreased as a result of recent medical advancements. Many chronic patients survive the crisis but lack the ability to care for themselves. and in desperate need of treatment If you are treated in a hospital until you recover, it will take a long time and cost a lot of money. As a result, the organizers anticipate a problem with chronic disease patients receiving treatment at home monitoring their symptoms. may experience an emergency or a variety of exacerbations that have a negative impact on life. WI-FI telecommunication technology was used in this project, which is a technology that allows electronic devices to exchange information. or a wireless internet connection can connect electronic equipment that assess a variety of symptoms in chronic disease patients, such as a pulse measuring device Patients with chronic illnesses can be tracked using a fall detection device and a GPS system. To alert caregivers or hospitals about critical or atypical symptoms in order to improve patient monitoring and care delivery. These technologies can be turned into programs that can recognize patient information in real time and collect data for further research and therapy

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

		หน้า
กิตติกรรมประกาศ		I
บทคัดย่อ		II
สารบัญ		IV
สารบัญรูป		VI
สารบัญตาราง		VIII
บทที่ 1	**บทนำ**	**1**
	1.1 ความเป็นมาและความสำคัญของปัญหา	1
	1.2 วัตถุประสงค์	2
	1.3 ขอบเขตของปริญญานิพนธ์	2
บทที่ 2	**ทฤษฎีและหลักการที่เกี่ยวข้อง**	**4**
	2.1 โรคเรื้อรัง	4
	2.2 ไมโครคอนโทรลเลอร์	6
	2.3 ESP8266	8
	2.4 ภาษาซี (C Programming Language)	19
	2.5 ฐานข้อมูลแบบ NOSQL	20
	2.6 FIREBASE	24
	2.7 วิธีตรวจชีพจรและสัญญาณบอกอันตราย	25
	2.8 GPS	27
	2.9 Flutter	28
	2.10 ภาษา Dart	28
	2.11 แหล่งจ่ายไฟสำหรับไมโครคอนโทรลเลอร์	29
	2.12 IFTTT	32
บทที่ 3	**การออกแบบและการจัดทำปริญญานิพนธ์**	**33**
	3.1 การออกแบบ	33

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ (ต่อ)

		หน้า
	3.2 เครื่องมือที่ใช้ในการทดลอง	50
	3.3 การจัดเก็บผลการทดลอง	62
บทที่ 4	**ผลการทดลอง**	**63**
	4.1 ทดสอบการส่งข้อมูลอัตราการเต้นหัวใจของเซนเซอร์ MAX-30102 ผ่าน NODEMCU ESP12E ESP8266	63
	4.2 การทดลองการส่งการแจ้งเตือนการหกล้มผ่านโมดูลวัดความเร่ง 3 แกน GY-521 MPU6050	65
	4.3 การทดลองการส่งข้อมูลจากเซนเซอร์ไปยังฐานข้อมูล	68
	4.4 การทดลองเปรียบเทียบบริเวณที่เหมาะสมในการติดตั้งเซนเซอร์ MAX30102	68
	4.5 ทดสอบความแม่นยำของอุปกรณ์ในการตรวจจับการหกล้ม	73
	4.6 การทดลองประสิทธิภาพของแบตเตอรี่ลิเทียมไอออนโพลีเมอร์ (LITHIUM-ION POLYMER BATTERY) 3.7 โวลต์ 500 มิลลิแอมแปร์	79
	4.7 ทดสอบการทำงานของแอพลิเคชัน และการแสดงผล	83
บทที่ 5	**สรุปผลและข้อเสนอแนะ**	**88**
	5.1 สรุปผล	88
	5.2 ข้อเสนอแนะ	89
บรรณานุกรม		**90**
ภาคผนวก ก	โค้ดที่ใช้ในการส่งข้อมูลจากไมโครคอนโทรลเลอร์	93
ภาคผนวก ข	โค้ดส่วนของแอปพลิเคชัน	102

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป

รูปที่		หน้า
1.1	บล็อกไดอะแกรมของปริญญานิพนธ์	3
2.1	NODEMCU ESP12E ESP8266	8
2.2	ขาของโมดูล ESP8266	9
2.3	รูปแบบพินสำหรับแพ็คเกจ QFN แบบ 32 PIN	10
2.4	ฟังก์ชันบล็อกไดอะแกรม ESP8266EX	13
2.5	ตัวอย่างการเก็บข้อมูลแบบ WIDE-COLUMN STORE	22
2.6	ตัวอย่างของ GRAPH DATABASE ที่เก็บข้อมูลความสัมพันธ์ระหว่าง อัลบั้ม, ศิลปิน และโพรดิวเซอร์	23
2.7	ตัวอย่างแบตเตอรี่	31
3.1	บล็อกไดอะแกรมของปริญญานิพนธ์	33
3.2	การเชื่อมต่อขอของโมดูล WI-FI NODEMCU ESP12E ESP8266 และ เซนเซอร์ MAX30102	34
3.3	การเชื่อมต่อขาของโมดูล WI-FI NODEMCU ESP12E ESP8266 และ โมดูลGY-521 MPU6050	36
3.4	แผนผังการทำงานของโปรแกรมเซนเซอร์	38
3.5	การเชื่อมต่ออุปกรณ์รวมเซนเซอร์ทุกตัวคือ GY-521 MPU6050, และ MAX30102 ให้สามารถใช้ NODEMCU ESP12E ESP8266 ควบคุมได้	40
3.6	FIREBASE REALTIME DATABASE กับการทำงานบนทุกอุปกรณ์	42
3.7	การตั้งค่า RULE ของ FIREBASE ที่ใช้ในปริญญานิพนธ์นี้	42
3.8	การเชื่อมต่อของชุดอุปกรณ์กับอุปกรณ์จ่ายกระแสไฟ	43
3.9	อุปกรณ์ทั้งหมดลงติดตั้งบนถุงมือ	44
3.10	แผนผังการทำงานของแอปพลิเคชัน	45
3.11	หน้าต่าง login	47
3.12	หน้าต่าง login	47
3.13	หน้าหลักของหน้าต่างแอปพลิเคชัน	48
3.14	หน้าต่างแอปพลิเคชันแสดงหน้าการโทรฉุกเฉิน	49

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป (ต่อ)

รูปที่		หน้า
3.15	หน้าต่างแอปพลิเคชันหน้า GPS	50
3.16	NODEMCU ESP12E ESP8266	51
3.17	PIN ของ NODEMCU ESP12E ESP8266	52
3.18	เซนเซอร์วัดระดับออกซิเจนในเลือดและอัตราการเต้นของหัวใจ MAX30102	54
3.19	แผนผังระบบของเซนเซอร์วัดระดับออกซิเจนในเลือดและอัตราการเต้นของหัวใจ MAX30102	55
3.20	PIN ไดอะแกรมของเซนเซอร์ MAX30102	55
3.21	โมดูล GY-521 MPU6050	58
3.22	PIN OUT ไดอะแกรม GY-521 MPU6050	58
3.23	แบตเตอรี่ลิเทียมไอออนโพลีเมอร์	60
3.24	โมดูลชาร์จแบตเตอรี่ TP4056	61
4.1	ค่าที่แสดงผลจากเซนเซอร์ MAX-30102 ตอนที่ไม่มีนิ้ววางบนเซนเซอร์	64
4.2	ค่าที่แสดงผลจากเซนเซอร์ MAX-30102 ตอนมีนิ้วบนเซนเซอร์	65
4.3	การตั้งค่าการทำงานเว็บไซต์ IFTTT	66
4.4	ผลที่แสดงบน Serial Monitor เมื่อชุดอุปกรณ์หยุดนิ่งบนโต๊ะ	67
4.5	ผลที่แสดงบน Serial Monitor เมื่อแกว่งชุดอุปกรณ์ไปมา	67
4.6	ผลที่แสดงบน Serial Monitor เมื่อเกิดการหกล้ม	68
4.7	ข้อมูลเซนเซอร์ที่เก็บไว้ที่ฐานข้อมูล	69
4.8	กราฟแสดงค่าอัตราการเต้นของหัวใจที่อ่านได้จากปลายนิ้ว	71
4.9	กราฟแสดงค่าอัตราการเต้นของหัวใจที่อ่านได้จากข้อมือด้านใน	72
4.10	กราฟแสดงค่าอัตราการเต้นของหัวใจที่อ่านได้จากข้อมือด้านนอก	73
4.11	ผู้ทดลองในท่ายืนตรง	75
4.12	ผู้ทดลองในท่าหกล้มเข่าลงพื้น	75
4.13	ผู้ทดลองในท่าล้มข้างไหล่ลงพื้น	76

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป (ต่อ)

หน้า

รูปที่

4.14 ผู้ทดลองในท่าลื่นล้มหงายหลัง 76
4.15 ผลจากหน้าจอ SERIAL MONITOR ในท่ายืนตรง 77
4.16 ผลจาก SERIAL MONITOR เมื่อให้ผู้ลองล้มในท่าหกล้มเข่าลงพื้น 78
4.17 ผลจาก SERIAL MONITOR ของการล้มในท่าล้มข้างไหล่ลงพื้น 79
4.18 ผลจาก SERIAL MONITOR ของการล้มในท่าลื่นล้มหงายหลัง 79
4.19 ผลจาก FIREBASE ของการนับค่าการหกล้ม 80
4.20 เริ่มการจับเวลาทดสอบประสิทธิภาพของแบตเตอรี่ 81
4.21 การหยุดการทำงานของเซนเซอร์ GY-521 MPU6050 82
4.22 ผลการจับเวลาในการทดลองประสิทธิภาพของแบตเตอรี่ 82
4.23 การชาร์จแบตเตอรี่ผ่านชาร์จแบตเตอรี่ TP4056 83
4.24 โมดูลชาร์จแบตเตอรี่ TP4056 เมื่อแบตเตอรี่ชาร์จเต็ม 84
4.25 เวลาที่ใช้ในการชาร์จแบตเตอรี่จนเต็ม 85
4.26 การใช้งานแอพลิเคชันคู่กับอุปกรณ์ 86
4.27 การตรวจสอบหน้าต่างการโทรฉุกเฉินและ GPS 87

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญตาราง

หน้า

ตารางที่

2.1 รายการคำจำกัดความและหน้าที่ของพินแต่ละพิน 11
2.2 ข้อมูลจำเพาะของวงจรนาฬิกาความถี่สูง 15
2.3 การอ้างอิงวงจรนาฬิกาภายนอก 15
2.4 ช่องความถี่ 16
3.1 การเชื่อมต่อขาของเซนเซอร์ MAX30102 กับ โมดูล WI-FI NODEMCU ESP12E 35
3.2 การเชื่อมต่อขาของโมดูล WI-FI NODEMCU ESP12E และ โมดูล GY-521 MPU6050 36
3.3 การเชื่อมต่อขาของอุปกรณ์รวมเซนเซอร์ทุกตัวคือ GY-521 MPU6050 และ MAX30102 41
3.4 ข้อมูลทางเทคนิคของ NODEMCU ESP12E ESP8266 52
3.5 การกำหนดค่า PIN ของบอร์ดพัฒนา NODEMCU ESP12E ESP8266 53
3.6 คำอธิบาย PIN ของ MAX30102 56
3.7 คำอธิบาย PIN OUT ของ GY-521 MPU6050 58
3.8 ข้อกำหนดทางไฟฟ้าของแบตเตอรี่ลิเทียมไอออนโพลีเมอร์ 60
3.9 ข้อมูลจำเพาะของโมดูลชาร์จแบตเตอรี่ TP4056 61
4.1 ตารางการเปรียบเทียบอัตราการเต้นของหัวใจที่วัดได้จากอุปกรณ์และ SMART WATCH 70
4.2 เปรียบเทียบค่า RELATIVE ERROR, ERROR และ ACCURACY (ความแม่นยำ) 74

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1

บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ในปัจจุบันเทคโนโลยี Wi-Fi เป็นเทคโนโลยีที่ได้รับความนิยมที่ช่วยให้อุปกรณ์อิเล็กทรอนิกส์ในการแลกเปลี่ยนข้อมูล หรือการเชื่อมต่ออินเทอร์เน็ตแบบไร้สาย การที่อุปกรณ์อิเล็กทรอนิกส์ต่างๆ สามารถเชื่อมโยงหรือส่งข้อมูลถึงกันได้ด้วยอินเทอร์เน็ต โดยไม่ต้องป้อนข้อมูล การเชื่อมโยงนี้ง่ายจนทำให้เราสามารถสั่งการควบคุมการใช้งานอุปกรณ์อิเล็กทรอนิกส์ต่างๆ ผ่านทางเครือข่ายอินเตอร์เน็ตได้ ไปจนถึงการเชื่อมโยงการใช้งานอุปกรณ์อิเล็กทรอนิกส์ต่างๆ ผ่านทางเครือข่ายอินเตอร์เน็ตเข้ากับการใช้งานอื่นๆ จนเกิดเป็น Internet of Things (IoT) ซึ่งปัจจุบันมีการประยุกต์ใช้เทคโนโลยี IoT ในหลาย ๆด้านอย่างเช่น ด้าน Healthcare รวมทั้งมีการนำไปใช้งานร่วมกับแผงควบคุมไมโครคอนโทรลเลอร์อย่างเช่น ไมโครคอนโทรลเลอร์ที่รองรับ Wi-Fi ที่มีขนาดเล็ก คือ ESP8266

เนื่องด้วยโรคเรื้อรัง (chronic disease) คือโรคที่เป็นยาวนานและมีการเปลี่ยนแปลงไปอย่างช้าๆ พบว่า 1 ใน 3 ของคนทั่วโลกต่าง ต้องเผชิญกับความเจ็บป่วยเรื้อรังที่แตกต่างกันออกไป ซึ่งโรคเรื้อรังนับเป็นสาเหตุของการเสียชีวิตลำดับต้น ๆ โดยคิดเป็นอัตราส่วนประมาณ 63% ของการเสียชีวิตของคนทั่วโลกซึ่งพัฒนาการของโรคจะดำเนินไปอย่างช้า ๆ แต่ค่อย ๆ สะสมอาการอย่างต่อเนื่อง และเมื่อมีอาการของโรคเกิดขึ้นแล้ว ก็มักจะเกิดการเรื้อรังของโรคด้วย ผู้ป่วยที่มีอาการเจ็บป่วยเรื้อรังมักมีอาการของโรคดังต่อไปนี้ เช่น โรคหลอดเลือดหัวใจ โรคความดันโลหิต โรคหัวใจเต้นผิดจังหวะเป็นต้น โดยลักษณะขอโรคดังกล่าวเมื่อเกิดขึ้นในกลุ่มผู้สูงอายุ ประเทศไทยมีจำนวนของผู้สูงอายุจำนวนมากที่ป่วยเป็นโรคเรื้อรัง ความก้าวหน้าทางการแพทย์ในปัจจุบันทำให้อัตราการเสียชีวิตของผู้ป่วยเรื้อรังจากภาวะวิกฤตลดลง ผู้ป่วยเรื้อรังที่รอดจากภาวะวิกฤตและไม่สามารถดูแลตัวเองได้มีจำนวนมากขึ้น ผู้ป่วยเหล่านี้ต้องการการดูแลอย่างต่อเนื่องเพราะมีความเสี่ยงสูงต่อการเกิดโรคแซงซ้อน หรือภาวะอาการกำเริบที่สามารถส่งผลต่อชีวิต และต้องการการฟื้นฟูสภาพ ซึ่งถ้าอยู่รักษาในโรงพยาบาลจนทุเลาหรือดีขึ้นจะต้องใช้เวลานานมากและมีค่าใช้จ่ายที่สูง ซึ่งบวกกับนโยบายของโรงพยาบาลในปัจจุบันนี้พยายามที่จะให้ผู้ป่วยเรื้อรังกลับบ้านโดยเร็วที่สุด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์นี้เป็นการออกแบบ และสร้างถุงมือที่สามารถวัดอัตราการเต้นของหัวใจ, แจ้งเตือนการหกล้ม, และระบุตำแหน่งของผู้ป่วยโรคเรื้อรัง และนำข้อมูลทั้งหมดมาแสดงผล และแจ้งเตือนบนแอพลิเคชันมือถือ โดยใช้เซนเซอร์ MAX30102 วัดอัตราการเต้นของหัวใจ, ใช้โมดูล GY-521 MPU6050 ตรวจจับการหกล้ม, รวมถึงการดึงพิกัด GPS ในโทรศัพท์ของผู้ป่วยมาระบุตำแหน่ง โดยนำเซนเซอร์ทั้งหมดเชื่อมต่อกับโมดูล Wi-Fi NodeMCU ESP12E ESP8266 เพื่อทำการควบคุมการทำงานของเซนเซอร์และ เชื่อมต่อ Wi-Fi เพื่อส่งข้อมูลที่ได้จากเซนเซอร์ไปยังฐานข้อมูลบน Firebase เพื่อนำข้อมูลมาแสดงบนแอปพลิเคชันได้

1.2 วัตถุประสงค์

1) เพื่อออกแบบ และสร้างถุงมือรวมถึงออกแบบโปรแกรมสำหรับการวัดอัตราการเต้นของหัวใจ, ระบุตำแหน่งพิกัด และการแจ้งเตือนการหกล้ม

2) เพื่อศึกษา และออกแบบฐานข้อมูลแบบเรียลไทม์เพื่อเก็บข้อมูลที่ได้จากชุดอุปกรณ์สำหรับการวัดอัตราการเต้นของหัวใจ, ระบุตำแหน่งพิกัด และการแจ้งเตือนการหกล้ม

3) เพื่อศึกษา และออกแบบแอปพลิเคชันที่สามารถนำข้อมูลจากชุดอุปกรณ์วัดอัตราการเต้นของหัวใจ, ระบุตำแหน่ง และการแจ้งเตือนการหกล้ม จากฐานข้อมูลมาแสดงได้

1.3 ขอบเขตของปริญญานิพนธ์

ปริญญานิพนธ์นี้ได้ทำการออกแบบ และทดลองชุดอุปกรณ์สำหรับวัดอัตราการเต้นของหัวใจ, ระบุตำแหน่งพิกัด และการแจ้งเตือนการหกล้ม ในรูปแบบของถุงมือโดยชุดอุปกรณ์ประกอบไปด้วย 6 ส่วนคือ

1) ใช้เซนเซอร์ MAX30102 ในการวัดอัตราการเต้นของหัวใจ

2) ใช้โมดูล GY-521 MPU6050 ตรวจจับการหกล้ม และแจ้งเตือนการหกล้มผ่าน LINE แอปพลิเคชัน

3) ใช้โมดูล Wi-Fi NodeMCU ESP12E ESP8266 ซึ่งภายในประกอบด้วยไมโครคอนโทรลเลอร์ ESP8266 เพื่อทำการควบคุมการทำงานของเซนเซอร์ทั้งหมด และเชื่อมต่อ Wi-Fi เพื่อส่งข้อมูลที่ได้จากเซนเซอร์ไปยังฐานข้อมูล

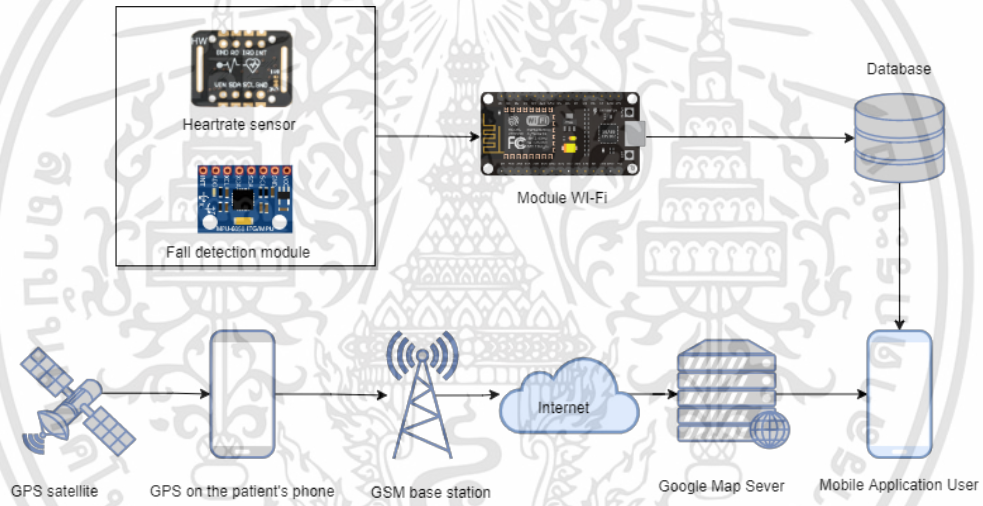
4) ใช้ Firebase เป็นฐานข้อมูลในการเก็บข้อมูลจากชุดอุปกรณ์ทั้งหมด

5) แสดงพิกัดบนแอปพลิเคชันโดยใช้บริการของ Google Map

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6) ออกแบบหน้าต่างของแอปพลิเคชัน ในระบบปฏิบัติการแอนดรอยด์โดยใช้ภาษา Dart ใน Flutter เป็น Framework ในการออกแบบหน้าต่างของแอปพลิเคชัน

ซึ่งบล็อกไดอะแกรมของปริญญานิพนธ์นี้ได้ทำการออกแบบ และทดลองชุดอุปกรณ์สำหรับวัดอัตราการเต้นของหัวใจ, ระบุตำแหน่งพิกัด และการแจ้งเตือนการหกล้ม โดยใช้เซนเซอร์ MAX30102 วัดอัตราการเต้นของหัวใจ, ใช้โมดูล GY-521 MPU6050 ตรวจจับการหกล้ม โดยนำเซนเซอร์ทั้งหมดเชื่อมต่อกับโมดูล Wi-Fi NodeMCU ESP12E ESP8266 เพื่อทำการควบคุมการทำงานของเซนเซอร์และ เชื่อมต่อ Wi-Fi เพื่อส่งข้อมูลที่ได้จากเซนเซอร์ไปยังฐานข้อมูลบน Firebase เพื่อนำข้อมูลมาแสดงบนแอปพลิเคชัน และดึงข้อมูลพิกัดตำแหน่งมาแสดงบนแอปพลิเคชันโดยสามารถแสดงบล็อกไดอะแกรมของปริญญานิพนธ์นี้ได้ดังรูปที่ 1.1



รูปที่ 1.1 บล็อกไดอะแกรมของปริญญานิพนธ์

จากรูปที่ 1.1 แสดงบล็อกไดอะแกรมของปริญญานิพนธ์โดยสามารถอธิบายการทำงานได้ดังนี้คือ ใช้เซนเซอร์ MAX30102 วัดอัตราการเต้นของหัวใจ, ใช้โมดูล GY-521 MPU6050 ตรวจจับการหกล้มโดยนำเซนเซอร์ทั้งหมดเชื่อมต่อกับโมดูล Wi-Fi NodeMCU ESP12E ESP8266 โดยความสามารถของไมโครคอนโทรลเลอร์ ESP8266 สามารถเชื่อมต่อ Wi-Fi เพื่อส่งข้อมูลจากเซนเซอร์คือค่าอัตราการเต้นของหัวใจ และการหกล้ม ไปยังฐานข้อมูลแบบเรียลไทม์บน Firebase จากนั้นทำการดึงข้อมูลจากฐานข้อมูลแบบเรียลไทม์ และพิกัดจาก GPS ไปแสดงบนแอปพลิเคชันได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2

ทฤษฎีและหลักการที่เกี่ยวข้อง

ในการทำปริญญานิพนธ์นี้จะประกอบด้วยส่วนของโปรแกรมบนบอร์ดNodeMCU ซึ่งประกอบไปด้วยไมโครคอนโทรลเลอร์ ESP8266 ที่สามารถเชื่อมต่อสัญญาณ Wi-Fiเพื่อทำการควบคุมการทำงานของเซนเซอร์ MAX30102 วัดอัตราการเต้นของหัวใจ, โมดูล GY-521 MPU6050 ตรวจจับการหกล้ม จากนั้นส่งข้อมูลที่อ่านค่าได้จากเซนเซอร์ทั้งหมดไปเก็บที่ฐานข้อมูลแบบเรียลไทม์บน Firebase และพิกัด GPS แสดงผลข้อมูลทั้งหมดบนแอปพลิเคชัน การเขียนโปรแกรมเพื่อควบคุมการทำงานของเซนเซอร์จำเป็นต้องมีความรู้พื้นฐานเกี่ยวกับหลักการทำงานของอุปกรณ์ต่างๆ ไมโครคอนโทรลเลอร์ รวมไปถึงพื้นฐานทางด้านภาษาในการเขียนโปรแกรมสั่งงานไมโครคอนโทรลเลอร์ และการจัดการฐานข้อมูล ซึ่งจำเป็นต้องศึกษาทฤษฎีที่เกี่ยวข้องดังนี้

2.1 โรคเรื้อรัง (Chronic Disease)

โรคเรื้อรังคือโรคที่คงอยู่เป็นเวลานาน โรคเรื้อรังเกิดขึ้นได้ตั้งแต่ 3 เดือนขึ้นไป ตามคำจำกัดความของศูนย์สถิติสุขภาพแห่งชาติของสหรัฐอเมริกา โรคเรื้อรังโดยทั่วไปไม่สามารถป้องกันได้ด้วยวัคซีนหรือรักษาด้วยยา และไม่หายขาด ร้อยละแปดสิบแปดของชาวอเมริกันที่อายุเกิน 65 ปีมีภาวะสุขภาพเรื้อรังอย่างน้อยหนึ่งรายการ (ณ ปี 2541) พฤติกรรมที่สร้างความเสียหายต่อสุขภาพโดยเฉพาะอย่างยิ่งการสูบบุหรี่ การขาดกิจกรรมทางกาย และนิสัยการกินที่ไม่ดี ล้วนเป็นสาเหตุสำคัญที่ทำให้เกิดโรคเรื้อรัง โรคเรื้อรังนับเป็นสาเหตุของการเสียชีวิตของคนส่วนใหญ่โดยคิดเป็นอัตราส่วนประมาณ 63% ของการเสียชีวิตของคนทั่วโลกผู้ป่วยที่มีอาการเจ็บป่วยเรื้อรังมักมีอาการของโรคดังต่อไปนี้ เช่น โรคหลอดเลือดหัวใจ, โรคเบาหวาน, โรคความดันโลหิตสูง, โรคไขมันในเลือดสูง, โรคหัวใจเต้นผิดจังหวะ และอื่นๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.1.1 โรคหลอดเลือดหัวใจ

โรคหลอดเลือดหัวใจตีบเกิดขึ้นเมื่อหลอดเลือดหลักที่ส่งไปเลี้ยงหัวใจของคุณเสียหายหรือเป็นโรค คอเลสเตอรอลที่มีการสะสม (คราบจุลินทรีย์) ในหลอดเลือดหัวใจตีบและการอักเสบมักจะเป็นสาเหตุของโรคหลอดเลือดหัวใจหลอดเลือดหัวใจจะส่งเลือด ออกซิเจน และสารอาหารไปยังหัวใจของคุณ การสะสมของคราบไขมันอาจทำให้หลอดเลือดแดงตีบตัน ทำให้เลือดไปเลี้ยงหัวใจลดลง ในที่สุด การไหลเวียนของเลือดที่ลดลงอาจทำให้เกิดอาการเจ็บหน้าอก (โรคหลอดเลือดหัวใจตีบ) หายใจถี่ หรืออาการและอาการของโรคหลอดเลือดหัวใจตีบอื่นๆ การอุดตันอย่างสมบูรณ์อาจทำให้หัวใจวายได้เนื่องจากโรคหลอดเลือดหัวใจตีบมักเกิดขึ้นในช่วงหลายทศวรรษที่ผ่านมา คุณอาจไม่สังเกตเห็นปัญหาจนกว่าคุณจะมีการอุดตันอย่างมีนัยสำคัญหรือหัวใจวาย

2.1.2 โรคเบาหวาน

โรคเบาหวานเป็นภาวะสุขภาพเรื้อรัง (ยาวนาน) ที่ส่งผลต่อการที่ร่างกายของคุณเปลี่ยนอาหารให้เป็นพลังงานอาหารส่วนใหญ่ที่คุณกินจะถูกย่อยเป็นน้ำตาล (หรือที่เรียกว่ากลูโคส) และปล่อยเข้าสู่กระแสเลือดของคุณ เมื่อน้ำตาลในเลือดของคุณสูงขึ้น มันจะส่งสัญญาณให้ตับอ่อนของคุณหลั่งอินซูลิน อินซูลินทำหน้าที่เป็นกุญแจสำคัญในการให้น้ำตาลในเลือดเข้าสู่เซลล์ของร่างกายเพื่อใช้เป็นพลังงานหากคุณเป็นเบาหวาน ร่างกายของคุณอาจผลิตอินซูลินไม่เพียงพอหรือไม่สามารถใช้อินซูลินที่ผลิตได้เท่าที่ควร เมื่อมีอินซูลินหรือเซลล์ไม่เพียงพอหยุดตอบสนองต่ออินซูลินน้ำตาลในเลือดก็จะคงอยู่ในกระแสเลือดมากเกินไป เมื่อเวลาผ่านไป อาจทำให้เกิดปัญหาสุขภาพร้ายแรง เช่น โรคหัวใจ สูญเสียการมองเห็น และโรคไต

2.1.3 โรคความดันโลหิตสูง

ความดันโลหิตสูงหรือที่เรียกว่าความดันโลหิตสูงคือความดันโลหิตที่สูงกว่าปกติ ความดันโลหิตของคุณเปลี่ยนแปลงตลอดทั้งวันตามกิจกรรมของคุณ การวัดความดันโลหิตให้สูงกว่าปกติอย่างสม่ำเสมออาจส่งผลให้มีการวินิจฉัยโรคความดันโลหิตสูง (หรือความดันโลหิตสูง) ยิ่งระดับความดันโลหิตของคุณสูงขึ้นเท่าใด คุณก็ยิ่งเสี่ยงต่อปัญหาสุขภาพอื่นๆ เช่น โรคหัวใจ หัวใจวาย และโรคหลอดเลือดสมอง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.1.4 โรคหัวใจเต้นผิดจังหวะ

โรคหัวใจเต้นผิดจังหวะ คือโรคที่หัวใจเต้นผิดจังหวะ (หรือเต้นในรูปแบบที่ผิดปกติ) โดยเรียกจะอาการนี้ว่า หัวใจเต้นผิดจังหวะ เป็นภาวะที่คลื่นไฟฟ้าหัวใจทำงานผิดปกติหรือเกิดไฟฟ้าลัดวงจรในห้องภายในหัวใจ ทำให้หัวใจเต้นผิดจังหวะ เช่นเต้นเร็วหรือช้าเกินไปทำให้การสูบฉีดเลือดไปยังส่วนต่าง ๆ ของสามารถเกิดขึ้นได้จากลักษณะทางพันธุกรรม, โภชนาการและการใช้ชีวิตประจำวัน

2.2 ไมโครคอนโทรลเลอร์

ไมโครคอนโทรลเลอร์ (MCU สำหรับหน่วยไมโครคอนโทรลเลอร์) คือคอมพิวเตอร์ขนาดเล็กบนชิปวงจรรวม VLSI ของวงจรรวม (IC) ของโลหะออกไซด์-เซมิคอนดักเตอร์ (MOS) ไมโครคอนโทรลเลอร์ประกอบด้วยซีพียู (คอร์โปรเซสเซอร์) ตั้งแต่หนึ่งตัวขึ้นไป พร้อมด้วยหน่วยความจำและอุปกรณ์ต่อพ่วงอินพุต/เอาต์พุตที่ตั้งโปรแกรมได้ หน่วยความจำโปรแกรมในรูปแบบของ ferroelectric RAM, NOR flash หรือ OTP ROM มักจะรวมอยู่ในชิปเช่นเดียวกับ RAM จำนวนเล็กน้อย ไมโครคอนโทรลเลอร์ได้รับการออกแบบมาสำหรับการใช้งานแบบฝังตัว ตรงกันข้ามกับไมโครโปรเซสเซอร์ที่ใช้ในคอมพิวเตอร์ส่วนบุคคลหรือการใช้งานทั่วไปอื่นๆ ที่ประกอบด้วยชิปแยกต่างๆ โดยโครงสร้างทั่วไปของไมโครคอนโทรลเลอร์แบ่งออกเป็น 5 ส่วนใหญ่ ๆ ต่อไปนี้

2.2.1 หน่วยประมวลผลกลางหรือซีพียู

หน่วยประมวลผลกลางหรือซีพียู (Central Processing Unit : CPU) โปรเซสเซอร์ (CPU) -- โปรเซสเซอร์ถือได้ว่าเป็นสมองของอุปกรณ์ มันประมวลผลและตอบสนองต่อคำสั่งต่าง ๆ ที่ควบคุมการทำงานของไมโครคอนโทรลเลอร์ สิ่งนี้เกี่ยวข้องกับการดำเนินการทางคณิตศาสตร์ ตรรกะ และ I/O พื้นฐาน นอกจากนี้ยังดำเนินการถ่ายโอนข้อมูล ซึ่งจะสื่อสารคำสั่งไปยังส่วนประกอบอื่นๆ หน่วยประมวลผลกลาง มีหน่วยสำคัญที่ทำงานประสานกัน 2 หน่วย คือ หน่วยควบคุมและ หน่วยคำนวณและตรรกะ

2.2.2 หน่วยความจำ

หน่วยความจำของไมโครคอนโทรลเลอร์ใช้เพื่อจัดเก็บข้อมูลที่โปรเซสเซอร์ได้รับและใช้เพื่อตอบสนองต่อคำสั่งที่ได้รับการตั้งโปรแกรมให้ดำเนินการ ไมโครคอนโทรลเลอร์มีหน่วยความจำหลักสองประเภทข้อมูล (Data Memory) ใช้เป็นเหมือนกับกระดาษทดในการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คำนวณของซีพียู และเป็นที่พักข้อมูลชั่วคราวขณะทำงาน แต่หากไม่มีไฟเลี้ยง ข้อมูลก็จะหายไป คล้ายกับหน่วยความจำแรม (RAM) ในเครื่องคอมพิวเตอร์ทั่วๆไปแต่สำหรับไมโครคอนโทรลเลอร์สมัยใหม่ หน่วยความจำข้อมูลจะมีทั้งที่เป็นหน่วยความจำแรม ซึ่งข้อมูลจะหายไปเมื่อไม่มีไฟเลี้ยง และเป็นอีอีพรอม (EEPROM : Electrically Erasable Programmable Read-Only Memory) ซึ่งสามารถเก็บข้อมูลได้แม้ไม่มีไฟเลี้ยงก็ตาม

2.2.3 พอร์ต (Port)

พอร์ต คืออุปกรณ์ที่ใช้ในการสื่อสารกับอุปกรณ์ภายนอก พอร์ต (Port) จะมี 2 ชนิด คือ พอร์ตรับสัญญาณหรืออินพุตพอร์ต (Input Port) และพอร์ตส่งสัญญาณหรือเอาต์พุตพอร์ต (Output Port) ส่วนนี้จะใช้ในการเชื่อมต่อกับอุปกรณ์ภายนอก ถือว่าเป็นส่วนที่สำคัญมาก ใช้ร่วมกันระหว่างพอร์ตอินพุต เพื่อรับสัญญาณ อาจจะด้วยการกดสวิตช์ เพื่อนำไปประมวลผลและส่งไปพอร์ตเอาต์พุต เพื่อแสดงผลเช่น การติดสว่างของหลอดไฟ เป็นต้น

2.2.4 บัส (BUS)

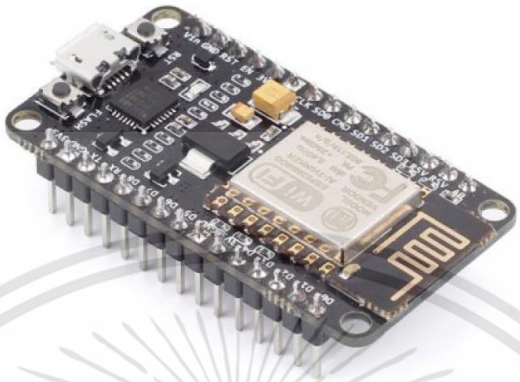
ช่องทางเดินของสัญญาณ หรือบัส (BUS) คือเส้นทางการแลกเปลี่ยนสัญญาณข้อมูลระหว่าง ซีพียู หน่วยความจำและพอร์ต เป็นลักษณะของสายสัญญาณ จำนวนมากอยู่ภายในตัวไมโครคอนโทรลเลอร์ โดยแบ่งเป็นบัสข้อมูล (Data Bus) ,บัสแอดเดรส (Address Bus) และบัสควบคุม (Control Bus)

2.2.5 วงจรกำเนิดสัญญาณนาฬิกา (Clock generator)

วงจรกำเนิดสัญญาณนาฬิกา เป็นองค์ประกอบที่สำคัญมากอีกส่วนหนึ่ง เนื่องจากการทำงานที่เกิดขึ้นในตัวไมโครคอนโทรลเลอร์ จะขึ้นอยู่กับการกำหนดจังหวะ หากสัญญาณนาฬิกามีความถี่สูง จังหวะการทำงานก็จะสามารถทำได้ถี่ขึ้นส่งผลให้ไมโครคอนโทรลเลอร์ตัวนั้น มีความเร็วในการประมวลผลสูงตามไปด้วย

ในปริญญานิพนธ์นี้ได้เลือกใช้บอร์ด NodeMCU ESP8266 บอร์ดคอนโทรลเลอร์ที่มีลักษณะการทำงานตามคำสั่งภาษา C คล้าย Arduino แต่มีลักษณะพิเศษกว่าตรงที่ สามารถเชื่อมต่อกับ Wi-Fi ได้ การควบคุมการทำงานสามารถใช้โปรแกรม Arduino IDE ได้เช่นเดียวกันกับบอร์ด Arduino บอร์ดของ NodeMCU ประกอบไปด้วย ESP8266 (ไมโครคอนโทรลเลอร์ที่สามารถเชื่อมต่อ WiFi ได้) พร้อมอุปกรณ์อำนวยความสะดวกต่างๆ เช่น พอร์ต micro USB สำหรับจ่ายไฟ/อัปโหลดโปรแกรม, ชิปสำหรับอัปโหลดโปรแกรมผ่านสาย USB, ชิปแปลงแรงดันไฟฟ้า และขาสำหรับเชื่อมต่ออุปกรณ์ภายนอกเป็นต้น ดังแสดงได้ในรูปที่ 2.1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.1 NodeMCU ESP12E ESP8266

จากรูปที่ 2.1 แสดง NodeMCU ESP12E ESP8266 โดยมีรายละเอียดข้อมูลทางเทคนิคของ NodeMCU ESP12E ESP8266 ดังนี้คือ

1. ใช้โมดูล ESP8266-12E ที่ภายในมีไมโครคอนโทรลเลอร์ 32 บิต หน่วยความจำแบบแฟลชความจุ 4 เมกะไบต์และโมดูล Wi-Fi ในตัว

2. มีชิป CP2102 สำหรับแปลงสัญญาณพอร์ต USB เป็น UART เพื่อเชื่อมต่อคอมพิวเตอร์สำหรับโปรแกรมเฟิร์มแวร์

3. ใช้ไฟเลี้ยงภายนอก +5V มีวงจรควบคุมแรงดันไฟเลี้ยงสำหรับอุปกรณ์ 3.3 โวลต์ กระแสไฟฟ้าสูงสุด 800 มิลลิแอมแปร์

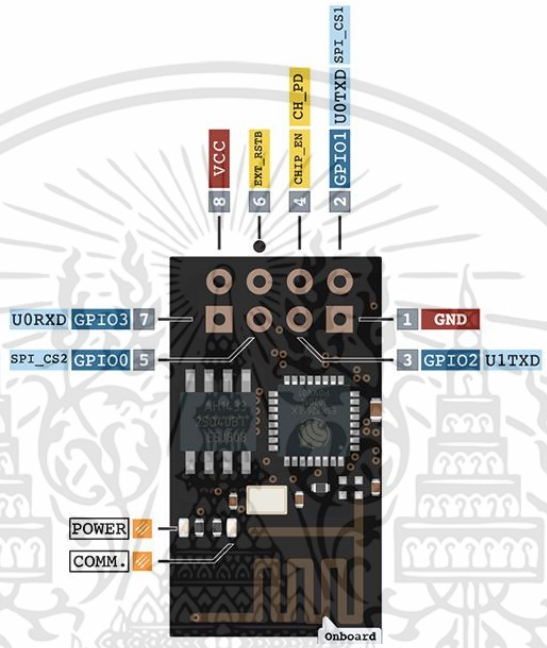
4. มีขาพอร์ต SPI สำหรับติดต่อกับ SD การ์ด มีสวิตช์ RESET และ Flash สำหรับโปรแกรมเฟิร์มแวร์ใหม่ มีอินพุตเอาต์พุตดิจิทัล (ลอจิก 3.3 โวลต์) รวม 16 ขา Arduino ESP8266 มีอินพุตแอนะล็อก 1 ช่อง รับแรงดันไฟตรง 0 ถึง +1 โวลต์ เข้าสู่วงจรแปลงสัญญาณแอนะล็อก เป็นดิจิทัลความละเอียด 10 บิต เสียบลงบนเบรดบอร์ดเพื่อทำการทดลองได้ทันทีหรือนำไปติดตั้งบนแผงวงจรประยุกต์ที่ออกแบบขึ้นเองได้อย่างสะดวก

2.3 ESP8266

ESP8266 เป็นไมโครคอนโทรลเลอร์ที่มีคุณสมบัติเด่นคือการเชื่อมต่อ Wi-Fi ที่มาพร้อมกับ Full TCP/IP Stack ตัวชิปมีราคาถูก อีกทั้งการเป็นไมโครคอนโทรลเลอร์หมายถึงสามารถเขียนโปรแกรมลงไปในตัวมันได้เลย ด้วยข้อดีต่างๆทั้งราคาถูก เขียนโปรแกรมได้ มีฟังก์ชัน Wi-Fi ติดมาพร้อม ทำให้ ESP8266 เป็นสิ่งที่ตอบสนองต่อการมาของยุค Internet of Things จึงทำให้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ได้รับความนิยมอย่างแพร่หลาย ESP8266 ทำงานที่แรงดันไฟฟ้า 3.0- 3.6 โวลต์ การนำไปใช้งานร่วมกับเซนเซอร์อื่นๆที่ใช้แรงดัน 5 โวลต์ ต้องใช้วงจรปรับแรงดันเพื่อไม่ให้โมดูลเสียหาย กระแสที่ใช้งานสูงสุดคือ 200 มิลลิแอมป์ ความถี่คริสตัล 40 เมกะเฮิรตซ์ขาของโมดูล ESP8266 แบ่งได้ตามรูปที่ 2.2



รูปที่ 2.2 ขาของโมดูล ESP8266

จากรูปที่ 2.2 แสดงขาของโมดูล ESP8266 ดังนี้

1) GND ขากราวนด์

2) GPIO1 เป็นขาที่ต้องต่อลง GND เท่านั้นเพื่อให้โมดูลทำงานได้

3) GPIO2 UART TX ระหว่างการเขียนโปรแกรมแฟลช GPIO2

4) ADC เป็นขาแอนะล็อกอินพุตรับแรงดันได้สูงสุดที่ 1 โวลต์ ขนาด 10 บิต (สามารถอ่านค่าที่มีความแตกต่างกันได้ตั้งแต่ 0 - 1023) หากนำไปใช้งานกับแรงดันที่สูงกว่าต้องใช้วงจรแบ่งแรงดันช่วย

5) GPIO0 (จีพีไอโอศูนย์) เป็นขาสำหรับการเลือกโหมดการทำงาน หากต่อลง GND จะเป็นการ Flash โปรแกรม (เขียนโปรแกรมลง ESP8266) หากลอยขาไว้จะเป็นการทำงานปกติ

6) Reset ไว้สำหรับเริ่มการทำงานของ ESP8266 ใหม่ (เหมือนการรีบูตระบบ)

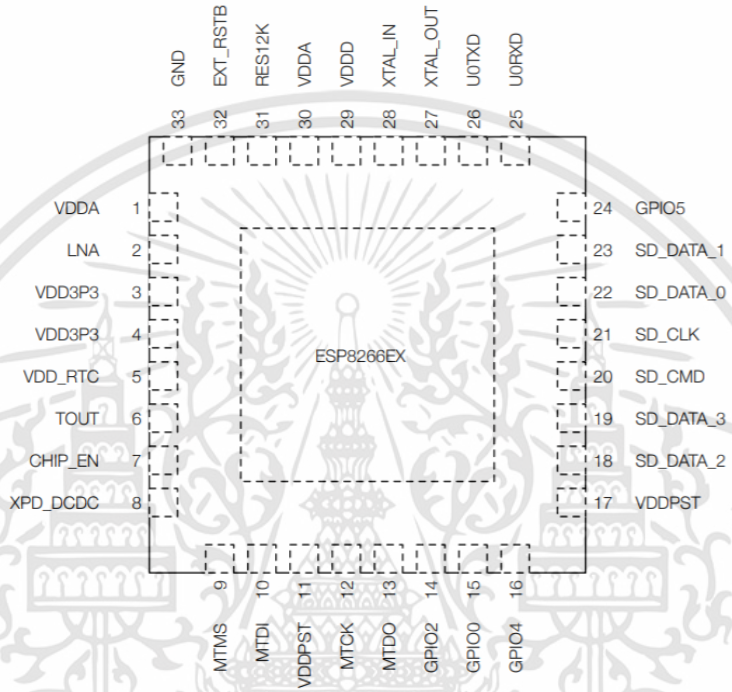
7) GPIO3 เป็นขาดิจิทัลอินพุต / เอาต์พุต ทำงานที่แรงดัน 3.3 โวลต์ UART Rx

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ระหว่างการเขียนโปรแกรมแฟลช GPIO3

8) VCC เป็นขาสำหรับจ่ายไฟเข้าเพื่อให้โมดูลทำงานได้ ซึ่งแรงดันที่ใช้งานได้คือ 3.0 - 3.6 โวลต์

โดยแสดงรูปแบบพินสำหรับแพ็กเกจ QFN แบบ 32 พิน ได้ในรูปที่ 2.3



รูปที่ 2.3 รูปแบบพินสำหรับแพ็กเกจ QFN แบบ 32 พิน

จากรูปที่ 2.3 แสดงรูปแบบพินสำหรับแพ็กเกจ QFN แบบ 32 พิน โดยรายละเอียดจะแสดงได้ในตารางที่ 2.1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 2.1 รายการคำจำกัดความและหน้าที่ของพินแต่ละพิน

PIN	ชื่อ	ประเภท	ฟังก์ชัน
1.	VDDA	P	แอนะล็อกเพาเวอร์ 2.5-3.6 โวลต์
2.	LNA	I/O	อินเทอร์เฟซเสาอากาศ RF อิมพีแดนซ์เอาต์พุตของชิป = 39 + j6 โอห์ม แนะนำให้เก็บเครือข่ายการจับคู่ประเภท π เพื่อให้ตรงกับเสาอากาศ
3.	VDD3P3	P	แอมพลิฟายเออร์เพาเวอร์ 2.5 - 3.6 โวลต์
4.	VDD3P3	P	แอมพลิฟายเออร์เพาเวอร์ 2.5 - 3.6 โวลต์
5.	VDD_RTC	P	NC (1.1 โวลต์)
6.	TOUT	I	พิน ADC สามารถใช้ทดสอบแรงดันไฟฟ้าของแหล่งจ่ายไฟของ VDD3P3 (Pin3 และ Pin4) และแรงดันไฟฟ้าขาเข้าของ TOUT (Pin 6) อย่างไรก็ตาม ฟังก์ชันทั้งสองนี้ใช้ไม่ได้พร้อมกัน
7.	CHIP_EN	I	ชิปเปิดใช้งาน สูง: เปิด ชิปทำงานอย่างถูกต้อง ต่ำ: ปิด ใช้กระแสไฟน้อย
8.	XPD_DCDC	I/O	Deep-sleep wakeup (จำเป็นต้องเชื่อมต่อกับ EXT_RSTB) GPIO16
9.	MTMS	I/O	GPIO 14; HSPI_CLK
10.	MTDI	I/O	GPIO 12; HSPI_MISO
11.	VDDPST	P	แหล่งจ่ายไฟดิจิทัล/IO 1.8 - 3.6 โวลต์
12.	MTCK	I/O	GPIO 13; HSPI_MOSI; UART0_CTS
13.	MTDO	I/O	GPIO 15; HSPI_CS; UART0_RTS
14.	GPIO2	I/O	UART TX ระหว่างการเขียนโปรแกรมแฟลช GPIO2
15.	GPIO0	I/O	GPIO0; SPI_CS2
16.	GPIO4	I/O	GPIO4
17.	VDDPST	P	แหล่งจ่ายไฟดิจิทัล/IO 1.8 - 3.6 โวลต์
18.	SDIO_DATA_2	I/O	เชื่อมต่อกับ SD_D2 (ซีรีส์ R: 20 โอห์ม); SPIHD;

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

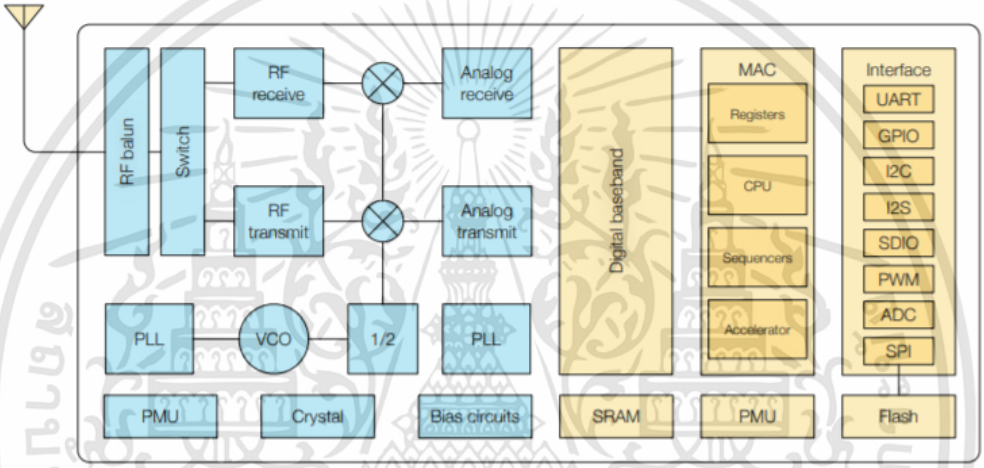
PIN	ชื่อ	ประเภท	ฟังก์ชัน
			HSPIHD; GPIO9
19.	SDIO_DATA_3	I/O	เชื่อมต่อกับ SD_D3 (ซีรีส์ R: 200 โอห์ม); SPIWP; HSPIWP; GPIO10
20.	SDIO_CMD	I/O	เชื่อมต่อกับ SD_CMD (ซีรีส์ R: 200 โอห์ม); SPI_CS0; GPIO11
21.	SDIO_CLK	I/O	เชื่อมต่อกับ SD_CLK (ซีรีส์ R: 200 โอห์ม); SPI_CLK; GPIO6
22.	SDIO_DATA_0	I/O	เชื่อมต่อกับ SD_D0 (ซีรีส์ R: 200 โอห์ม); SPI_MISO; GPIO7
23.	SDIO_DATA_1	I/O	เชื่อมต่อกับ SD_D1 (ซีรีส์ R: 200 โอห์ม); SPI_MOSI; GPIO8
24.	GPIO5	I/O	GPIO5
25.	U0RXD	I/O	UART Rx ระหว่างการเขียนโปรแกรมแฟลช GPIO3
26.	U0TXD	I/O	UART TX ระหว่างการเขียนโปรแกรมแฟลช GPIO1; SPI_CS1
27.	XTAL_OUT	I/O	เชื่อมต่อกับเอาต์พุตคริสตัลออสซิลเลเตอร์ สามารถใช้เพื่อให้ BT clock อินพุต
28.	XTAL_IN	I/O	เชื่อมต่อกับอินพุตคริสตัลออสซิลเลเตอร์
29.	VDDD	P	แอนะล็อกเพาเวอร์ 2.5-3.6 โวลต์
30.	VDDA	P	แอนะล็อกเพาเวอร์ 2.5-3.6 โวลต์
31.	RES12K	I	การเชื่อมต่อแบบอนุกรมกับตัวต้านทาน 12 กิโลโอห์ม และเชื่อมต่อกับกราวนด์
32.	EXT_RSTB	I	สัญญาณรีเซ็ตภายนอก (ระดับแรงดันต่ำ: ใช้งานอยู่)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากตารางที่ 2.1 แสดงรายการคำจำกัดความและหน้าที่ของพินแต่ละพินทั้งหมด 32 PIN โดยพินมีทั้งหมด 3 ประเภทคือ P, I และ I/O หมายเหตุใช้ GPIO2, GPIO0 และ MTDO เพื่อเลือกโหมดการบูสและโหมด SDIO2 ไม่ควรดึง U0TXD จากภายนอกไปยังระดับลอจิกต่ำในระหว่างการเปิดเครื่อง

2.3.1 รายละเอียดการทำงาน

แผนภาพการทำงานของ ESP8266EX โดยฟังก์ชันบล็อกไดอะแกรมสามารถแสดงได้ดังรูปที่ 2.4



รูปที่ 2.4 ฟังก์ชันบล็อกไดอะแกรม ESP8266EX

รูปที่ 2.4 แสดงฟังก์ชันบล็อกไดอะแกรม ESP8266EX โดยมีส่วนประกอบของโครงสร้างดังต่อไปนี้

2.3.1.1 ซีพียู (CPU)

ESP8266EX รวมเอาโปรเซสเซอร์ Ten silica L106 32 บิต RISC เข้าไว้ด้วยกัน ซึ่งบรรลุการสิ้นเปลืองพลังงานที่ต่ำเป็นพิเศษและถึงความเร็วสัญญาณนาฬิกาสูงสุด 160 เมกะเฮิรตซ์ เรียลไทม์ ระบบปฏิบัติการ (RTOS) และสแต็ก Wi-Fi อนุญาตให้ 80% ของพลังการประมวลผลเป็นพร้อมใช้งานสำหรับการเขียนโปรแกรมและการพัฒนาแอปพลิเคชันของผู้ใช้ ซีพียูประกอบด้วย

-อินเทอร์เฟซด้านล่าง:

- อินเตอร์เฟซ RAM/ROM ที่ตั้งโปรแกรมได้ (iBus) ซึ่งสามารถเชื่อมต่อกับหน่วยความจำได้ ตัวควบคุมและยังสามารถใช้เพื่อเยี่ยมชมแฟลช

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- Data RAM interface (dBus) ซึ่งสามารถเชื่อมต่อกับตัวควบคุมหน่วยความจำ

- อินเทอร์เฟซ AHB ซึ่งสามารถใช้เพื่อเยี่ยมชมการลงทะเบียน

2.3.1.2 หน่วยความจำ (Memory)

หน่วยความจำ (Memory) ESP8266EX Wi-Fi SoC รวมตัวควบคุมหน่วยความจำและหน่วยหน่วยความจำรวมถึง SRAM และ ROM MCU สามารถเข้าถึงหน่วยหน่วยความจำผ่านอินเทอร์เฟซ iBus, dBus และ AHB หน่วยหน่วยความจำทั้งหมดสามารถเข้าถึงได้เมื่อมีการร้องขอ ในขณะที่หน่วยความจำจะตัดสินลำดับการทำงานตามเวลาที่โปรเซสเซอร์ได้รับคำขอเหล่านี้ตามSDK เวอร์ชันปัจจุบันของเรา พื้นที่ SRAM ที่พร้อมใช้งานสำหรับผู้ใช้ถูกกำหนดไว้ดังนี้

- ขนาด RAM < 50 kB นั่นคือเมื่อ ESP8266EX ทำงานภายใต้โหมด Stationและเชื่อมต่อกับเราเตอร์ พื้นที่ตั้งโปรแกรมสูงสุดที่เข้าถึงได้ใน Heap ส่วนข้อมูลอยู่ที่ประมาณ 50 กิโลบิต

- ไม่มี ROM ที่ตั้งโปรแกรมได้ใน SoC ดังนั้นจึงต้องเก็บโปรแกรมผู้ใช้ไว้ในแฟลชSPI ภายนอก

2.3.1.3 แฟลชภายนอก (External Flash)

ESP8266EX ใช้แฟลช SPI ภายนอกเพื่อจัดเก็บโปรแกรมผู้ใช้ และรองรับสูงสุด 16 เมกะไบต์ ความจุหน่วยความจำในทางทฤษฎี หน่วยความจำแฟลชขั้นต่ำของ ESP8266EX แสดงไว้ดังนี้

- OTA ถูกปิดใช้งาน: อย่างน้อย 512 กิโลไบต์

- เปิดใช้งาน OTA: อย่างน้อย 1 เมกะไบต์

2.3.1.4 วงจรนาฬิกาความถี่สูง (High Frequency Clock)

นาฬิกาความถี่สูงบน ESP8266EX ใช้เพื่อขับเคลื่อนทั้งการส่งและรับมิกเซอร์ นาฬิกานี้สร้างจากคริสตัลออสซิลเลเตอร์ภายในและ คริสตัลภายนอก ช่วงความถี่ตั้งแต่ 24 เมกะเฮิรตซ์ ถึง 52 เมกะเฮิรตซ์ การสอบเทียบภายในคริสตัลออสซิลเลเตอร์ช่วยให้มั่นใจได้ว่า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สามารถใช้คริสตัลได้หลากหลาย แต่คุณภาพของคริสตัลยังคงเป็นปัจจัยที่ต้องพิจารณาว่ามีสัญญาณรบกวนเฟซที่เหมาะสมและความไวของ Wi-Fi ที่ดี ซึ่งพิจารณาได้ดังตาราง 2.2 เพื่อวัดการชดเชยความถี่

ตารางที่ 2.2 ข้อมูลจำเพาะของวงจรนาฬิกาความถี่สูง

พารามิเตอร์	เครื่องหมาย	ต่ำสุด	สูงสุด	หน่วย
ความถี่	FXO	24	52	MHz
กำลังโหลด	CL	-	32	pF
ความจุของการเคลื่อนที่	CM	2	5	pF
ความต้านทานอนุกรม	RS	0	65	Ω
ความทนทานต่อความถี่	ΔFXO	-15	15	ppm
ความถี่เทียบกับอุณหภูมิ (–25 °C - 75 °C)	ΔFXO,Temp	-15	15	ppm

จากตารางที่ 2.2 แสดงข้อมูลจำเพาะของวงจรนาฬิกาความถี่สูงทั้งหมด 6 พารามิเตอร์ ประกอบด้วยความถี่, กำลังโหลด, ความจุของการเคลื่อนที่, ความต้านทานอนุกรม, ความทนทานต่อความถี่ และความถี่เทียบกับอุณหภูมิ

2.3.1.5 ข้อกำหนดวงจรนาฬิกาภายนอก (External Clock Requirements)

นาฬิกาที่สร้างจากภายนอกสามารถใช้ได้กับความถี่ตั้งแต่ 24 เมกะเฮิรตซ์ ถึง 52 เมกะเฮิรตซ์ ลักษณะดังต่อไปนี้คาดว่าจะเป็นประสิทธิภาพที่ดีของวิทยุ โดยจะแสดงในตารางที่ 2.3

ตารางที่ 2.3 การอ้างอิงวงจรนาฬิกาภายนอก

พารามิเตอร์	เครื่องหมาย	ต่ำสุด	สูงสุด	หน่วย
แอมพลิจูดวงจรนาฬิกา	VXO	0.8	1.5	Vpp
ความแม่นยำของวงจรนาฬิกาภายนอก	ΔFXO, EXT	-15	15	ppm
สัญญาณรบกวนเฟส ออฟเซ็ต 1 กิโลเฮิรตซ์, นาฬิกา 40 เมกะเฮิร์ตซ์	-	-	-120	dBc/Hz

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

พารามิเตอร์	เครื่องหมาย	ต่ำสุด	สูงสุด	หน่วย
สัญญาณรบกวนเฟส ออฟเซ็ต 10 กิโลเฮิรตซ์, นาฬิกา 40 เมกะเฮิร์ตซ์	-	-	-130	dBc/Hz
สัญญาณรบกวนเฟส ออฟเซ็ต 100 กิโลเฮิรตซ์, นาฬิกา 40 เมกะเฮิร์ตซ์	-	-	-138	dBc/Hz

จากตารางที่ 2.3 แสดงการอ้างอิงวงจรนาฬิกาภายนอก ทั้งหมด 6 พารามิเตอร์

2.3.1.6 วิทยุ (Radio)

วิทยุ ESP8266EX ประกอบด้วยบล็อกต่อไปนี้

- ตัวรับ 2.4 GHz
- เครื่องส่ง 2.4 GHz
- เครื่องกำเนิดนาฬิกาความเร็วสูงและออสซิลเลเตอร์คริสตัล
- อคติและหน่วยงานกำกับดูแล
- การจัดการพลังงาน

2.3.1.7 ความถี่ช่องสัญญาณ (Channel Frequencies)

ตัวรับส่งสัญญาณ RF รองรับช่องสัญญาณต่อไปนี้ตามมาตรฐาน IEEE802.11 b/g/n โดยแสดงช่องความถี่ได้ดังตารางที่ 2.4

ตารางที่ 2.4 ช่องความถี่

หมายเลขช่อง	ความถี่ (เมกะเฮิรตซ์)	หมายเลขช่อง	ความถี่ (เมกะเฮิรตซ์)
1	2412	8	2447
2	2417	9	2452
3	2422	10	2457
4	2427	11	2462
5	2432	12	2467
6	2437	13	2472
7	2442	14	2484

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.3.1.8 ตัวรับ 2.4 GHz (2.4 GHz Receiver)

ตัวรับสัญญาณ 2.4 GHz ลงแปลงสัญญาณ RF เป็นสัญญาณเบสแบนด์สี่เหลี่ยมจัตุรัสและ แปลงเป็นโดเมนดิจิทัลด้วย ADC ความเร็วสูงที่มีความละเอียดสูง 2 ตัว เพื่อปรับตัวให้เข้ากับเงื่อนไขช่องสัญญาณที่แตกต่างกัน, ตัวกรอง RF, การควบคุมอัตราขยายอัตโนมัติ (AGC), DC offsetวงจรการยกเลิกและตัวกรองเบสแบนด์รวมอยู่ใน ESP8266EX

2.3.1.9 เครื่องส่งสัญญาณ 2.4 GHz (2.4 GHz Transmitter)

เครื่องส่งสัญญาณ 2.4 GHz ขึ้นแปลงสัญญาณเบสแบนด์แบบสี่เหลี่ยมจัตุรัสเป็น 2.4 GHz และขับเคลื่อนเสาอากาศด้วยเครื่องขยายสัญญาณ CMOS กำลังแรงสูง ฟังก์ชันการสอบเทียบแบบดิจิทัลช่วยปรับปรุงความเป็นเส้นตรงของเพาเวอร์แอมป์ ทำให้ประสิทธิภาพอันล้ำหน้าในการส่งพลังงาน TX เฉลี่ย +19.5 dBm สำหรับการส่งสัญญาณ 802.11b และ +18 dBm สำหรับการส่ง 802.11n (MSC0) มีการปรับเทียบเพิ่มเติมเพื่อชดเชยความไม่สมบูรณ์ของวิทยุ เช่น การรั่วไหลของผู้ให้บริการ, การจับคู่เฟซ I/Q, ความไม่เชิงเส้นของเบสแบนด์, ฟังก์ชันการสอบเทียบในตัวเหล่านี้ช่วยลดเวลาการทดสอบผลิตภัณฑ์และทำการทดสอบอุปกรณ์ที่ไม่จำเป็น

2.3.1.10 เครื่องกำเนิดนาฬิกา (Clock Generator)

เครื่องกำเนิดสัญญาณนาฬิกาสร้างสัญญาณนาฬิกาสี่เหลี่ยมจัตุรัส 2.4 GHz สำหรับเครื่องรับและ เครื่องส่งสัญญาณ ส่วนประกอบทั้งหมดของตัวสร้างสัญญาณนาฬิกาถูกรวมเข้ากับชิปรวมถึงทั้งหมดตัวเหนี่ยวนำ วาแรคเตอร์ ตัวกรองลูป ตัวควบคุมแรงดันไฟฟ้าเชิงเส้น และ ตัวแบ่งเครื่องกำเนิดสัญญาณนาฬิกามีวงจรสอบเทียบและทดสอบตัวเองในตัว เฟซนาฬิกาสี่เหลี่ยมจัตุรัสและสัญญาณรบกวนเฟซได้รับการปรับให้เหมาะสมบนชิปด้วยอัลกอริทึมการสอบเทียบที่ได้รับการจดสิทธิบัตรเพื่อให้แน่ใจว่า ประสิทธิภาพที่ดีที่สุดของเครื่องรับและเครื่องส่งสัญญาณ

2.3.1.11 Wi-Fi

Wi-Fi ESP8266EX ใช้ TCP/IP และโปรโตคอล 802.11 b/g/n WLAN MAC เต็มรูปแบบ รองรับการทำงาน Basic Service Set (BSS) STA และ Soft AP ภายใต้ Distributed Control Function (DCF) การจัดการพลังงานได้รับการจัดการโดยมีการโต้ตอบกับโฮสต์ขั้นต่ำเพื่อลดระยะเวลาการใช้งาน

2.3.1.12 วิทยุ Wi-Fi และเบสแบนด์ (Wi-Fi Radio and Baseband)

วิทยุ Wi-Fi และเบสแบนด์ วิทยุ ESP8266EX Wi-Fi และเบสแบนด์รองรับคุณสมบัติดังต่อไปนี้:

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- 802.11 b และ 802.11 g
- 802.11 n MCS0-7 ในแบนด์วิดท์ 20 MHz
- ช่วงเวลาป้องกัน 802.11 n 0.4 μs
- อัตราข้อมูลสูงสุด 72.2 Mbps
- รับ STBC 2 x 1
- กำลังส่งสูงสุด 20.5 dBm
- กำลังส่งที่ปรับได้

2.3.1.13 Wi-Fi MAC

ESP8266EX Wi-Fi MAC ใช้ฟังก์ชันโปรโตคอลระดับต่ำโดยอัตโนมัติดังนี้:

- 2 × อินเทอร์เฟซ Wi-Fi เสมือน
- โหมดสถานีโครงสร้างพื้นฐาน BSS/โหมด Soft AP/โหมดสำส่อน
- ขอส่ง (RTS), ล้างเพื่อส่ง (CTS) และบล็อกทันที ACK
- การจัดเรียงข้อมูล
- CCMP (CBC-MAC, โหมดตัวนับ), TKIP (MIC, RC4), WEP (RC4) และ CRC
- การตรวจสอบบีคอนอัตโนมัติ (ฮาร์ดแวร์ TSF)
- รองรับบลูทูธแบบเสาอากาศคู่และเดี่ยวพร้อมอุปกรณ์เสริมพร้อมกัน

2.3.1.14 การจัดการพลังงาน

ESP8266EX ได้รับการออกแบบด้วยเทคโนโลยีการจัดการพลังงานขั้นสูงและมีไว้สำหรับอุปกรณ์พกพา อุปกรณ์อิเล็กทรอนิกส์ที่สวมใส่ได้ และแอปพลิเคชัน Internet of Thingsสถาปัตยกรรมพลังงานต่ำทำงานในโหมดต่อไปนี้:

- โหมดแอคทีฟ: เปิดวิทยุชิปแล้ว ชิปสามารถรับ ส่ง หรือฟังได้
- โหมด Modem-sleep: CPU กำลังทำงาน Wi-Fi และวิทยุถูกปิดใช้งาน
- โหมด Light-sleep: CPU และอุปกรณ์ต่อพ่วงทั้งหมดจะหยุดชั่วคราว เหตุการณ์ปลุกใดๆ (MAC, โฮสต์, ตัวจับเวลา RTC หรือการขัดจังหวะภายนอก) จะทำให้ชิปทำงาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- โหมดดีปสลีป: เฉพาะ RTC เท่านั้นที่ใช้งานได้และส่วนอื่นๆ ของชิปทั้งหมดปิดเครื่อง

2.4 ภาษาซี (C programming language)

ภาษาซี (C programming language) เป็นภาษาโปรแกรมคอมพิวเตอร์ทั่วไป สร้างขึ้นในปี 1970 โดย Dennis Ritchie และยังคงใช้กันอย่างแพร่หลายและมีอิทธิพลอย่างมาก จากการออกแบบ คุณสมบัติของ C สะท้อนถึงความสามารถของ CPU เป้าหมายอย่างชัดเจน พบการใช้งานที่ยาวนานในระบบปฏิบัติการ ไดรเวอร์อุปกรณ์ สแต็คโปรโตคอล แม้ว่าจะลดลงสำหรับซอฟต์แวร์แอปพลิเคชัน และพบได้ทั่วไปในสถาปัตยกรรมคอมพิวเตอร์ที่มีตั้งแต่ซูเปอร์คอมพิวเตอร์ที่ใหญ่ที่สุดไปจนถึงไมโครคอนโทรลเลอร์ที่เล็กที่สุดและระบบฝังตัว C เป็นภาษาขั้นตอนที่จำเป็นซึ่งสนับสนุนการเขียนโปรแกรมที่มีโครงสร้าง ขอบเขตของตัวแปรคำศัพท์ และการเรียกซ้ำ ด้วยระบบประเภทสแตติก ได้รับการออกแบบมาให้คอมไพล์เพื่อให้เข้าถึงหน่วยความจำและโครงสร้างภาษาในระดับต่ำที่แมปกับคำสั่งเครื่องได้อย่างมีประสิทธิภาพ ทั้งหมดนี้รองรับรันไทม์น้อยที่สุด แม้จะมีความสามารถระดับต่ำ แต่ภาษาก็ได้รับการออกแบบมาเพื่อสนับสนุนการเขียนโปรแกรมข้ามแพลตฟอร์ม โปรแกรม C ที่ได้มาตรฐานซึ่งเขียนขึ้นโดยคำนึงถึงการพกพาสามารถคอมไพล์สำหรับแพลตฟอร์มคอมพิวเตอร์และระบบปฏิบัติการที่หลากหลาย โดยมีการเปลี่ยนแปลงเล็กน้อยในซอร์สโค้ดและตัวแปลโปรแกรมของภาษาซีมีให้ใช้งานได้สำหรับสถาปัตยกรรมคอมพิวเตอร์และระบบปฏิบัติการต่าง ๆ เป็นส่วนมากภาษาหลายภาษาในยุคหลังได้หยิบยืมภาษาซีไปใช้ทั้งทางตรงและทางอ้อม ตัวอย่างเช่น ภาษาดี ภาษาโก ภาษารัสต์ ภาษาจาวา จาวาสคริปต์ ภาษาลิมโบ ภาษาแอลพีซี ภาษาซีชาร์ป ภาษาอ็อบเจกทีฟ-ซี ภาษาเพิร์ล ภาษาพีเอชพี ภาษาไพทอน ภาษาเวอริล็อก (ภาษาพรรณนาฮาร์ดแวร์) และซีเชลล์ของยูนิกซ์ ภาษาเหล่านี้ได้ดึงโครงสร้างการควบคุมและคุณลักษณะพื้นฐานอื่น ๆ มาจากภาษาซี ส่วนใหญ่มีวากยสัมพันธ์คล้ายคลึงกับภาษาซีเป็นอย่างมากโดยรวม (ยกเว้นภาษาไพทอนที่ต่างออกไปอย่างสิ้นเชิง) และตั้งใจที่จะผสานนิพจน์และข้อความสั่งที่จำแนกได้ของวากยสัมพันธ์ของภาษาซี ด้วยระบบชนิดตัวแปร ตัวแบบข้อมูล และอรรถศาสตร์ที่อาจแตกต่างกันโดยมูลฐาน ภาษาซีพลัสพลัสและภาษาอ็อบเจกทีฟ-ซีเดิมเกิดขึ้นในฐานะตัวแปลโปรแกรมที่สร้างรหัสภาษาซี ตัวต่อจากภาษาโปรแกรม B, C ได้รับการพัฒนาขึ้นที่ Bell Labs โดย Dennis Ritchie ระหว่างปี 1972 และ 1973 เพื่อสร้างยูทิลิตี้ที่ทำงานบน Unix มันถูกนำไปใช้กับการนำเคอร์เนลของระบบปฏิบัติการ Unix ไปใช้อีกครั้ง ในช่วงปี 1980 C ค่อยๆ ได้รับความนิยม มันได้กลายเป็นหนึ่งในภาษาโปรแกรมที่ใช้กันอย่างแพร่หลายมากที่สุดโดยมีคอมไพเลอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

C พร้อมใช้งานสำหรับสถาปัตยกรรมคอมพิวเตอร์และระบบปฏิบัติการที่ทันสมัยเกือบทั้งหมด C ได้รับมาตรฐานโดย ANSI ตั้งแต่ปี 1989 (ANSI C) และโดยองค์การระหว่างประเทศเพื่อการมาตรฐาน (ISO)

C เป็นภาษาขั้นตอนที่จำเป็นซึ่งสนับสนุนการเขียนโปรแกรมที่มีโครงสร้าง ขอบเขตของตัวแปรคำศัพท์ และการเรียกซ้ำ ด้วยระบบประเภทสแตติก ได้รับการออกแบบมาให้คอมไพล์เพื่อให้เข้าถึงหน่วยความจำและโครงสร้างภาษาในระดับต่ำที่แมปกับคำสั่งเครื่องได้อย่างมีประสิทธิภาพ ทั้งหมดนี้รองรับรันไทม์น้อยที่สุด แม้จะมีความสามารถระดับต่ำ แต่ภาษาก็ได้รับการออกแบบมาเพื่อสนับสนุนการเขียนโปรแกรมข้ามแพลตฟอร์ม โปรแกรม C ที่ได้มาตรฐานซึ่งเขียนขึ้นโดยคำนึงถึงการพกพาสามารถคอมไพล์สำหรับแพลตฟอร์มคอมพิวเตอร์และระบบปฏิบัติการที่หลากหลาย โดยมีการเปลี่ยนแปลงเล็กน้อยในซอร์สโค้ด

ตั้งแต่ปี 2000 C ได้รับการจัดอันดับให้เป็นหนึ่งในสองภาษาชั้นนำอย่างสม่ำเสมอในดัชนี TIOBE ซึ่งเป็นหน่วยวัดความนิยมของภาษาโปรแกรมมิ่ง

2.5 ฐานข้อมูลแบบ NoSQL

ฐานข้อมูลแบบ NoSQL ใช้โมเดลข้อมูลที่หลากหลายสำหรับการเข้าถึงและจัดการข้อมูล ฐานข้อมูลประเภทนี้ได้รับการปรับปรุงประสิทธิภาพสำหรับแอปพลิเคชันที่ต้องใช้ข้อมูลปริมาณมาก มีเวลาแฝงต่ำ และมีโมเดลข้อมูลที่ยืดหยุ่นโดยเฉพาะ ซึ่งเกิดขึ้นโดยการผ่อนปรนข้อจำกัดความสม่ำเสมอข้อมูลของฐานข้อมูลอื่นๆ NoSQL นั้นย่อมาจาก “Non SQL” และต่อมาในภายหลังเพี้ยนเป็น “Not only SQL” ซึ่งสื่อความหมายเป็นนัยว่า มันเป็นอะไรที่มากกว่า SQL ในสมัยก่อนที่ปริมาณข้อมูลมีจำนวนจำกัด และยังไม่มากในระดับที่เรียกว่า Big Data นั้น ข้อมูลแบบ structured data (หรือข้อมูลที่มีรูปแบบชัดเจน) ส่วนมากจะถูกจัดเก็บในระบบฐานข้อมูลแบบ Relational Database ซึ่งความนิยมในตัว RDBMS (Relational Database Management System) นี้เองที่ทำให้ภาษา SQL ได้ถูกนำไปใช้กันอย่างแพร่หลาย เนื่องจากเป็นภาษาที่ถูกคิดขึ้นมาเพื่อดึงข้อมูลออกจาก RDBMS โดยเฉพาะ

2.5.1 คุณสมบัติของ NoSQL Database

2.5.1.1 Dynamic Schemas

การจัดเก็บข้อมูลต่างๆ ในฐานข้อมูลแบบ Relational Database จำเป็นจะต้องมีการสร้าง Schema หรือรูปแบบของโครงสร้างตาราง เพื่อจะจัดเก็บข้อมูลใดบ้าง เมื่อต้องการจัดเก็บข้อมูลเพิ่มเติมต้องเปลี่ยน Schema ภายหลัง (Alter-Table) ก่อนจะจัดเก็บข้อมูลรูปแบบใหม่ได้ แต่ในปัจจุบันการจัดเก็บข้อมูลมีการเปลี่ยนแปลงตลอดเวลา เพราะความต้องการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จัดเก็บข้อมูลต่างๆ มีหลากหลายมากขึ้นเรื่อย ๆ การต้องกำหนดโครงสร้างของตารางฐานข้อมูลเป็นอย่างไร เป็นเรื่องค่อนข้างยาก และการต้องเปลี่ยนโครงสร้างฐานข้อมูลบ่อยๆ โดยที่ข้อมูลมีอยู่แล้วเป็นเรื่องที่ยากหรือไม่ก็ทำไม่ได้เลย วิธีการคือเราต้องแยกออกเป็นตารางใหม่ ซึ่งเป็นวิธีแก้ปัญหาเพียงชั่วคราว เพราะระบบฐานข้อมูลแบบ NoSQL ไม่จำเป็นต้องมี Schema ที่ตายตัว หรือไม่ต้องมี Schema ก่อนที่จะจัดเก็บข้อมูล ข้อมูลแต่ละแถว (Row) สามารถจัดเก็บได้ตามต้องการ จะเพิ่มหรือลด ก็ไม่มีปัญหากับระบบ ทำให้เราสามารถจัดเก็บข้อมูลได้ตามที่ต้องการ เปลี่ยนแปลงได้ตลอดเวลา สะดวกรวดเร็วมาก

2.5.1.2 Auto-Shading

เมื่อข้อมูลมีขนาดใหญ่ หรือเราต้องการเพิ่มประสิทธิภาพการ อ่าน-เขียนข้อมูลปริมาณมากๆ การทำการแบ่งส่วน (Shading) ในระบบ NoSQL Database จะทำการกระจายข้อมูลไปยังเซิร์ฟเวอร์ (Server) ต่างๆอัตโนมัติ (Auto-Sharding) ผู้พัฒนา (Developer) ไม่ต้องเขียนโปรแกรมในการกระจายข้อมูลเองเหมือน Relational Database การกระจายข้อมูลออกไปหลายๆเซิร์ฟเวอร์นี้ยังทำให้มีข้อดีคือ ประหยัดต้นทุนในการขยายระบบ เพราะเป็นการขยายแบบแนวราบ (Scale Out) ซึ่งสามารถนำเซิร์ฟเวอร์ปกติทั่วไปมาใช้งานได้ ไม่จำเป็นต้องเป็นเซิร์ฟเวอร์องค์กร (Enterprise Server)

2.5.1.3 Replication

การสำเนาข้อมูลจากเครื่องหนึ่งไปยังอีกเครื่องหนึ่ง (Replication) เมื่อเซิร์ฟเวอร์หนึ่งเสียหาย อีกเครื่องหนึ่งจะขึ้นมาทำงานแทนทันที โดยข้อมูลของแต่ละเครื่องจะมีข้อมูลเหมือนกัน ดังนั้น Replication เป็นหนึ่งคุณสมบัติที่ตอบสนองต่อการใช้งานที่ต้องการความต่อเนื่องได้ตลอดเวลา (High Availability)

2.5.1.4 Integrated Caching

การจัดเก็บข้อมูลที่ใช้งานบ่อยๆ เข้าไว้ใน Memory (RAM) ซึ่งเป็นคุณสมบัติเด่นของ NoSQL ที่รวม Caching ไว้ในตัวเองอยู่แล้ว เราไม่จำเป็นต้องทำ Cache Layer เหมือน Relational Database อีกต่อไป ที่ต้องทำ Cache Layer แยกต่างหากและต้องดูแลรักษาแยกออกไปต่างหากอีกด้วย

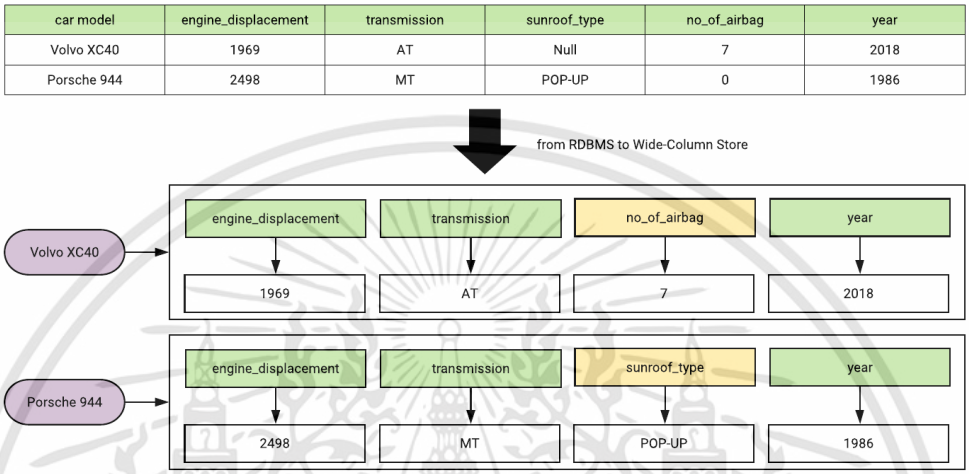
2.5.2 ประเภทของฐานข้อมูล NoSQL Database

2.5.2.1 Wide-Column Store

ฐานข้อมูลแบบ RDBMS นั้นมีการเก็บข้อมูลในรูปแบบของ แถว (row) ที่มี คอลัมน์ (column) ที่แน่นอน แต่ wide-column store นั้นมีการเก็บข้อมูลโดยแต่ละ จะมีคู่ของ key column และ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

value column ที่อาจจะเหมือนหรือแตกต่างกันก็ได้ ตัวอย่างของฐานข้อมูลที่มีรูปแบบการเก็บข้อมูลแบบนี้เช่น HBase และ Cassandra เป็นต้น โดยตัวอย่างการเก็บข้อมูลแบบ wide-column store จะแสดงได้ดังรูปที่ 2.5



รูปที่ 2.5 ตัวอย่างการเก็บข้อมูลแบบ wide-column store

จากรูปที่ 2.5 แสดงตัวอย่างการเก็บข้อมูลแบบ wide-column store จะเห็นว่าแต่ละแถว (row) ที่ถูกเก็บจะแบ่งออกเป็นแต่ละคีย์ (key) และไม่จำเป็นต้องมีคอลัมน์ (column) ที่เหมือนกันก็ได้

2.5.2.2 Key-value stores

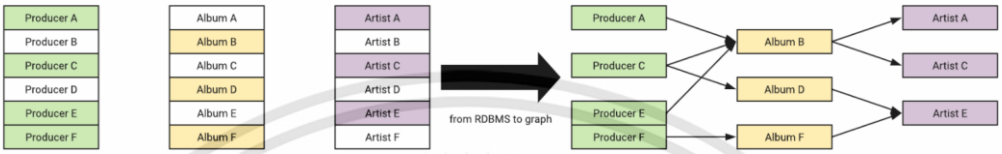
Key-value stores เป็นฐานข้อมูลแบบเน้นความเรียบง่าย เพราะแต่ละการบันทึก(record) จะมีเพียงแค่ คีย์ (key) และ value เท่านั้น สามารถเข้าถึงข้อมูลได้อย่างรวดเร็วด้วยการระบุค่า คีย์ (key) ลงไปโดยไม่จำเป็นต้องใช้ SQL ในการดึงข้อมูล ตัวอย่างของฐานข้อมูลที่มีรูปแบบการเก็บข้อมูลแบบนี้เช่น Redis และ Oracle NoSQL Database (OND) เป็นต้น การเก็บข้อมูลแบบนี้ถูกนำไปใช้กับทวิตเตอร์ (Twitter)

2.5.2.3 Graph Database

หากความสำคัญของข้อมูลที่เราจะเก็บอยู่ที่ความสัมพันธ์กันของข้อมูล Graph Database จะเป็นหนึ่งในตัวเลือกที่ดีที่สุด ยกตัวอย่างเช่น เดิมหากเราจะเก็บข้อมูลใน RDBMS ว่า โปรดิวเซอร์เพลง (music producer) คนไหนผลิต (produce) อัลบั้มไหนของศิลปินคนใด เราอาจจะต้องแบ่งข้อมูลออกเป็น 3 ตารางก่อน (อัลบั้ม, ศิลปิน และโปรดิวเซอร์) แล้วจึงค่อย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

นำข้อมูลในแต่ละตารางมาร่วมกันทีหลังผ่าน SQL ในทางกลับกันถ้าเราเก็บข้อมูลแบบ Graph Database ซึ่งเป็นการเก็บความสัมพันธ์ของข้อมูลตั้งแต่แรกอยู่แล้ว เราก็จะไม่จำเป็นต้องนำข้อมูลมารวมกันโดยตัวอย่างจะแสดงได้ในรูปที่ 2.6



รูปที่ 2.6 ตัวอย่างของ graph database ที่เก็บข้อมูลความสัมพันธ์ระหว่าง อัลบั้ม, ศิลปิน และโปรดิวเซอร์

จากรูปที่ 2.6 แสดงตัวอย่างของ graph database ที่เก็บข้อมูลความสัมพันธ์ระหว่างอัลบั้ม, ศิลปิน และโปรดิวเซอร์เก็บข้อมูลแบบ graph Database ซึ่งเป็นการเก็บความสัมพันธ์ของข้อมูลตั้งแต่แรกอยู่แล้ว เราก็จะไม่จำเป็นต้องนำข้อมูลมารวมกัน

2.5.2.4 Document Based

Document Based เป็นเก็บข้อมูลในรูปแบบของไฟล์ semi-structured data ต่างๆ เช่น JSON หรือ XML โดยภายในไฟล์ที่เก็บในแต่ละ table อาจจะไม่ต้องมีรูปแบบ (structure) ที่เหมือนกันก็ได้ ซึ่งเราจะสามารถค้นหาข้อมูลในแต่ละแท็ก (tag) ได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.6 Firebase

Firebase เป็นแพลตฟอร์มที่พัฒนาโดย Google สำหรับสร้างแอปพลิเคชันมือถือและเว็บ เดิมเป็นบริษัทอิสระที่ก่อตั้งขึ้นในปี 2554 ในปี 2557 Google ได้ซื้อแพลตฟอร์มและ ปัจจุบันเป็นข้อเสนอหลักสำหรับการพัฒนาแอป ซึ่ง Firebase สามารถนำมาเป็นตัวกลางในการเชื่อมต่ออุปกรณ์เข้าด้วยกัน เช่น NodeMCU ESP8266 ส่งข้อมูลไปยัง แอปพลิเคชันบนระบบปฏิบัติการ Android

Firebase เป็น Backend-as-a-Service (Baas) มีเครื่องมือและบริการที่หลากหลายแก่นักพัฒนาเพื่อช่วยในการพัฒนาแอปที่มีคุณภาพ เพิ่มฐานผู้ใช้ และรับผลกำไร สร้างขึ้นบนโครงสร้างพื้นฐานของ Google Firebase จัดอยู่ในประเภทโปรแกรมฐานข้อมูล NoSQL ซึ่งจัดเก็บข้อมูลในเอกสารที่คล้ายกับ JSON Google เป็นแพลตฟอร์มการพัฒนาแอปพลิเคชันบนมือถือจาก Google ที่มีคุณสมบัติอันทรงพลังสำหรับการพัฒนา จัดการ และเพิ่มประสิทธิภาพแอปพลิเคชัน Firebase เป็นแพลตฟอร์มแบ็กเอนด์สำหรับสร้างเว็บและ แอปพลิเคชันมือถือ โดยพื้นฐานแล้ว Firebase คือชุดเครื่องมือที่นักพัฒนาสามารถไว้วางใจได้ สร้างแอปพลิเคชันและขยายตามความต้องการ Firebase มีเป้าหมายเพื่อแก้ปัญหาหลักสามประการสำหรับนักพัฒนา สร้างแอปอย่างรวดเร็ว, ปล่อยและตรวจสอบแอพด้วยความมั่นใจ, ดึงดูดผู้ใช้ และนักพัฒนาที่ใช้แพลตฟอร์มนี้สามารถเข้าถึงบริการที่พวกเขาจะต้องพัฒนาตนเอง และช่วยให้พวกเขาสามารถมุ่งเน้นไปที่การส่งมอบประสบการณ์การใช้งานที่มีประสิทธิภาพ คุณลักษณะเด่นบางอย่างของแพลตฟอร์ม Google Firebase ได้แก่ ฐานข้อมูล การตรวจสอบสิทธิ์ ข้อความพุช การวิเคราะห์ พื้นที่จัดเก็บไฟล์ และอื่นๆ อีกมากมาย เนื่องจากบริการนี้โฮสต์บนคลาวด์ นักพัฒนาจึงสามารถดำเนินการปรับขนาดตามต้องการได้อย่างราบรื่นโดยไม่ต้องยุ่งยาก ปัจจุบัน Firebase เป็นหนึ่งในแพลตฟอร์มการพัฒนาแอปชั้นนำที่นักพัฒนาทั่วโลกไว้วางใจ

2.6.1 Firebase Realtime Database

Firebase Realtime Database เป็น NoSQL cloud database ฐานข้อมูลเรียลไทม์ของ Firebase เป็นฐานข้อมูลที่โฮสต์บนคลาวด์ ข้อมูลจะถูกจัดเก็บเป็น JSON และซิงโครไนซ์แบบเรียลไทม์กับทุกไคลเอนต์ที่เชื่อมต่อ เมื่อคุณสร้างแอปข้ามแพลตฟอร์ม Flutter & Firebase ลูกค้าทั้งหมดของคุณสามารถแชร์อินสแตนซ์ฐานข้อมูลเรียลไทม์และได้รับการอัปเดตด้วยข้อมูลใหม่ล่าสุดโดยอัตโนมัติ

2.6.2 Firebase Cloud Firestore

Firebase Cloud Firestore เป็นฐานข้อมูล NoSQL ที่โฮสต์บนคลาวด์ ซึ่ง Apple,

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Android และเว็บแอปของคุณสามารถเข้าถึงได้โดยตรงผ่าน SDK ดั้งเดิม Cloud Firestore ยังมีให้บริการใน Native Node.js, Java, Python, Unity, C++ และ Go SDK นอกเหนือจาก REST และ RPC API

ตามโมเดลข้อมูล NoSQL ของ Cloud Firestore คุณจะจัดเก็บข้อมูลในเอกสารที่มีการแมปฟิลด์กับค่าต่างๆ เอกสารเหล่านี้จัดเก็บไว้ในคอลเลกชั่น ซึ่งเป็นคอนเทนเนอร์สำหรับเอกสารของคุณ ซึ่งคุณสามารถใช้จัดระเบียบข้อมูลและสร้างคิวรีได้ เอกสารรองรับข้อมูลหลายประเภท ตั้งแต่สตริงและตัวเลขอย่างง่าย ไปจนถึงอ็อบเจ็กต์ที่ซับซ้อนและซ้อนกัน คุณยังสามารถสร้างคอลเลกชั่นย่อยภายในเอกสารและสร้างโครงสร้างข้อมูลแบบลำดับชั้นที่ปรับขนาดเมื่อฐานข้อมูลของคุณเติบโตขึ้น โมเดลข้อมูล Cloud Firestore รองรับโครงสร้างข้อมูลแบบใดก็ได้ที่เหมาะกับแอปของคุณมากที่สุด

นอกจากนี้ การสืบค้นใน Cloud Firestore ยังแสดงออกถึงความรู้สึก มีประสิทธิภาพและยืดหยุ่นอีกด้วย สร้างการสืบค้นข้อมูลแบบตื้นเพื่อดึงข้อมูลที่ระดับเอกสารโดยไม่จำเป็นต้องเรียกค้นทั้งคอลเล็กชันหรือคอลเล็กชันย่อยที่ซ้อนกัน เพิ่มการจัดเรียง การกรอง และการจำกัดการสืบค้นข้อมูลหรือเคอร์เซอร์เพื่อจัดหน้าผลลัพธ์ หากต้องการให้ข้อมูลในแอปของคุณเป็นปัจจุบันโดยไม่ต้องดึงฐานข้อมูลทั้งหมดทุกครั้งที่มีการอัปเดต ให้เพิ่มฟังแบบเรียลไทม์ การเพิ่มผู้ฟังแบบเรียลไทม์ให้กับแอปของคุณจะแจ้งเตือนคุณด้วยสแนปชอตข้อมูลเมื่อใดก็ตามที่ข้อมูลที่แอปไคลเอ็นต์ของคุณรับฟังการเปลี่ยนแปลง โดยจะดึงเฉพาะการเปลี่ยนแปลงใหม่เท่านั้น

ปกป้องการเข้าถึงข้อมูลของคุณ Cloud Firestore ด้วย Firebase Authentication และ Cloud Firestore Security Rules สำหรับ Android, แพลตฟอร์ม Apple และ JavaScript หรือ Identity and Access Management (IAM) สำหรับภาษาฝั่งเซิร์ฟเวอร์

2.7 วิธีตรวจชีพจร และสัญญานบอกอันตราย

ชีพจรคือการเต้นของหัวใจถ้าชีพจรเต้นเบา, แรงหรือไม่เป็นจังหวะมีความผิดปกติอาจแสดงถึงความไม่ปกติของระบบการเต้นของหัวใจและระบบไหลเวียนของเลือดในร่างกาย ซึ่งเป็นสัญญาณของการเจ็บป่วยร้ายแรงได้ ทำให้การสังเกตระดับชีพจรช่วยให้รับรู้ถึงปัญหาสุขภาพและช่วยให้หาวิธีการรับมือและช่วยแก้ไขปัญหาที่จะเกิดขึ้นได้ วิธีการวัดชีพจรสามารถทำได้หลากหลายวิธีโดยมีวิธีการดังนี้

2.7.1 วิธีการตรวจจับชีพจร

สามารถตรวจชีพจรได้ด้วยตนเองโดยการจับตามบริเวณต่าง ๆ ที่มีหลอดเลือดแดงอยู่ใกล้กับผิวหนัง เช่น ข้อมือ, ข้อเท้าหรือลำคอ ซึ่งมีวิธีการตรวจจับชีพจรดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.7.1.1 ข้อมือ

ยื่นมือข้างที่จะทำการวัดชีพจรไปข้างหน้างอศอกเข้าหาตัวเล็กน้อยและหงายฝ่ามือขึ้นจากนั้นวางนิ้วชี้และนิ้วกลางของมืออีกข้างหนึ่งลงเพื่อจับชีพจรที่ข้อมือบริเวณโคนนิ้วโป้ง กดนิ้วชี้และนิ้วกลางลงบนผิวหนังจนกว่าจะรู้สึกถึงการเต้นของชีพจร หรือขยับตำแหน่งนิ้วทั้ง 2 เล็กน้อยจนกว่าจะจับชีพจรได้แล้วเริ่มจับเวลานับการเต้นของชีพจรใน 1 นาทีได้ทั้งหมดกี่ครั้

2.7.1.2 ลำคอ

วางนิ้วชี้และนิ้วกลางลงบนลำคอบริเวณใต้กรามใกล้กับหลอดลม ซึ่งเป็นการจับชีพจรบริเวณเส้นเลือดแดงที่จะนำเลือดไปเลี้ยงสมอง แต่จะทำการวัดชีพจรได้ยากกว่าที่ข้อมือ โดยต้องไม่จับชีพจรที่คอพร้อมกันสองด้านเพราะจะทำให้หมดสติหรือเป็นอันตรายได้ และเมื่อรู้สึกถึงการเต้นของชีพจรได้แล้วเริ่มจับเวลานับการเต้นของชีพจรใน 1 นาทีได้ทั้งหมดกี่ครั้ง

2.7.2 ชีพจรที่ปกติ

ปกติผู้ใหญ่จะมีชีพจรเต้น 60-100 ครั้ง/นาที แต่ผู้ที่สุขภาพแข็งแรงมักมี ชีพจรต่ำ โดยอาจมีอัตราการเต้นของหัวใจขณะพักอยู่ที่ 40-60 ครั้ง/นาที

2.7.3 ชีพจรที่ผิดปกติ

ในการใช้ชีวิตประจำวัน มีหลายอย่างที่ส่งผลต่อการเต้นขิงหัวใจส่งผลต่อ ชีพจร เช่น การทำกิจกรรมสันธนาการ, การอยู่ในท่าทางต่าง ๆ เช่น การนอน, การยืนหรือการนั่ง เป็นต้น, ขนาดของร่างกาย, อุณหภูมิของบริเวณที่อยู่ในขณะนั้น, การใช้ยารักษาโรคบางชิด ดังนั้นในบางครั้งชีพจรอาจเต้นผิดปกติหรือขาดหายไปก็เป็นเรื่องปกติที่อาจจะเกิดขึ้นได้ แต่ถ้าหากชีพจรหยุด, หาย, เต้นเร็วแรง หรือเต้นเบาอ่อนผิดปกติเป็นเวลานานอย่างต่อเนื่องเป็นการแสดงถึงปัญหาสุขภาพที่เกิดขึ้น จึงควรได้รับการเฝ้าสังเกตความผิดปกติที่เกิดขึ้น และไปปรึกษาแพทย์เกี่ยวกับปัญหาที่เกิดขึ้น หากพบปัญหาดังต่อไปนี้เช่น ชีพจรมากกว่า 100 ครั้ง/นาที หรือน้อยกว่า 60 ครั้ง/นาทีโดยที่ไม่ได้เป็นนักกีฬาโดยเฉพาะอย่างยิ่งเมื่อมีอาการป่วยอื่น ๆ เกิดขึ้นร่วมด้วย เช่น วิงเวียนศีรษะ หายใจไม่ทั่วท้อง หน้ามืดและเป็นลม เป็นต้น

2.7.4 ปัญหาสุขภาพที่เกิดจากชีพจรเต้นผิดปกติ

2.7.4.1 ภาวะหัวใจเต้นเร็ว

ภาวะหัวใจเต้นเร็วคือสภาวะที่ชีพจรจะเต้นเร็วผิดปกติสูงเกินกว่า 100 ครั้ง/นาที โดยอาจทำให้มีอาการใจสั่น หายใจหอบเหนื่อย อ่อนล้า เวียนศีรษะหรือเป็นลมหมดสติได้ หากมีอาการร้ายแรงอาจทำให้เกิดลิ่มเลือดอุดตัน หัวใจวาย หรือหัวใจหยุดเต้นและเสียชีวิตได้

2.7.4.2 ภาวะหัวใจเต้นช้า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สภาวะหัวใจเต้นช้า ชีพจรเต้นช้ากว่าปกติ โดยมีอัตราการเต้นของหัวใจที่ต่ำกว่า 60 ครั้ง/นาที ซึ่งอาจก่อให้เกิดปัญหาสุขภาพได้ โดยเฉพาะหากหัวใจเต้นช้าจนไม่สามารถสูบฉีดเลือดและนำออกซิเจนไปเลี้ยงอวัยวะส่วนต่าง ๆ ตามร่างกายได้อย่างเพียงพอ ซึ่งอาจทำให้มีอาการ เช่น รู้สึกเหนื่อย อ่อนเพลีย เหนื่อยเร็วเมื่อออกกำลังกาย วิงเวียนศีรษะ สับสนมึนงง ไม่มีสมาธิ หายใจไม่สุด ใจสั่น เป็นลม วูบ หากมีอาการร้ายแรงอาจทำให้ความดันโลหิตผิดปกติ เป็นลมบ่อย หัวใจวาย หรือหัวใจหยุดเต้นได้

2.7.4.3 ภาวะหัวใจเต้นผิดจังหวะ

ภาวะหัวใจเต้นผิดจังหวะ จะมีลักษณะดังต่อไปนี้ชีพจรเต้นจะในลักษณะผิดปกติ โดยสามารถเกิดขึ้นร่วมกับอัตราการเต้นของชีพจรที่สูงกว่าปกติหรือต่ำกว่าปกติ ซึ่งอาจเป็นผลมาจากปัญหาสุขภาพ เช่น ขาดสมดุลของแร่ธาตุในเลือด กล้ามเนื้อหัวใจเกิดการเปลี่ยนแปลง โรคหลอดเลือดหัวใจ หรือเป็นการบาดเจ็บจากภาวะกล้ามเนื้อหัวใจขาดเลือด โดยอาการที่เกิดขึ้นอาจแตกต่างกันไปตามตำแหน่งของหัวใจที่เกิดความเสียหาย ซึ่งมีหลายประเภทด้วยกัน เช่น ภาวะหัวใจห้องบนเต้นสั่นพลิ้ว (Atrial Fibrillation) ภาวะหัวใจห้องล่างเต้นแผ่วระรัว (Ventricular Fibrillation) เป็นต้น

2.8 GPS

Global Positioning System (GPS) ซึ่งเดิมเรียกว่า Navstar GPS คือระบบนำทางด้วยคลื่นวิทยุบนดาวเทียมของรัฐบาลสหรัฐอเมริกาและดำเนินการโดยกองกำลังอวกาศแห่งสหรัฐอเมริกา เป็นหนึ่งในระบบดาวเทียมนำทางทั่วโลก (GNSS) ที่ให้ข้อมูลตำแหน่งทางภูมิศาสตร์และเวลาแก่เครื่องรับ GPS ทุกที่ในหรือใกล้โลกซึ่งมีเส้นสายตาที่ปราศจากสิ่งกีดขวางสำหรับดาวเทียม GPS สี่ดวงขึ้นไปอุปสรรคเช่นภูเขาและอาคารสามารถปิดกั้นสัญญาณ GPS ที่ค่อนข้างอ่อนได้

2.8.1 การทำงานของ GPS

Global Positioning System (GPS) คือระบบนำทางที่ใช้ดาวเทียม เครื่องรับ และอัลกอริธึมในการซิงโครไนซ์ข้อมูลตำแหน่ง ความเร็ว และเวลาสำหรับการเดินทางทางอากาศ ทางทะเล และทางบก

ระบบดาวเทียมประกอบด้วยกลุ่มดาวบริวาร 24 ดวงในเครื่องบินโคจรที่มีศูนย์กลางโลก 6 ลำ แต่ละลำมีดาวเทียมสี่ดวง โคจรอยู่ที่ 13,000 ไมล์ (20,000 กม.) เหนือพื้นโลก และเดินทางด้วยความเร็ว 8,700 ไมล์ต่อชั่วโมง (14,000 กม./ชม.) แม้ว่าเราต้องการดาวเทียมเพียงสามดวงเพื่อสร้างตำแหน่งบนพื้นผิวโลก แต่ดาวเทียมดวงที่สี่มักถูกใช้เพื่อตรวจสอบความถูกต้องของ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ข้อมูลจากอีกสามดวง ดาวเทียมดวงที่สี่ยังย้ายเราไปสู่มิติที่สามและช่วยให้เราสามารถคำนวณความสูงของอุปกรณ์ได้GPS ทำงานโดยใช้เทคนิคที่เรียกว่า trilateration ใช้ในการคำนวณตำแหน่ง ความเร็ว และระดับความสูง Trilateration จะรวบรวมสัญญาณจากดาวเทียมไปยังข้อมูลตำแหน่งที่ส่งออก มักถูกเข้าใจผิดว่าเป็นสามเหลี่ยม ซึ่งใช้ในการวัดมุม ไม่ใช่ระยะทาง ดาวเทียมที่โคจรรอบโลกจะส่งสัญญาณให้อุปกรณ์ GPS อ่านและตีความได้ ซึ่งตั้งอยู่บนหรือใกล้พื้นผิวโลก ในการคำนวณตำแหน่ง อุปกรณ์ GPS จะต้องสามารถอ่านสัญญาณจากดาวเทียมอย่างน้อยสี่ดวง

ดาวเทียมแต่ละดวงในเครือข่ายโคจรรอบโลกวันละสองครั้ง และดาวเทียมแต่ละดวงจะส่งสัญญาณ พารามิเตอร์การโคจร และเวลาที่ไม่ซ้ำกัน ในช่วงเวลาใดก็ตาม อุปกรณ์ GPS สามารถอ่านสัญญาณจากดาวเทียมหกดวงขึ้นไป ดาวเทียมดวงเดียวออกอากาศสัญญาณไมโครเวฟที่อุปกรณ์ GPS หยิบขึ้นมาและใช้ในการคำนวณระยะทางจากอุปกรณ์ GPS ไปยังดาวเทียม

2.9 Flutter

Flutter เป็นชุดเครื่องมือ UI ข้ามแพลตฟอร์มที่ออกแบบมาเพื่ออนุญาตให้ใช้รหัสซ้ำในระบบปฏิบัติการต่างๆ เช่น iOS และ Android ในขณะที่ยังช่วยให้แอปพลิเคชันสามารถเชื่อมต่อโดยตรงกับบริการแพลตฟอร์มพื้นฐาน เป้าหมายคือการช่วยให้นักพัฒนาสามารถนำเสนอแอพประสิทธิภาพสูงที่ให้ความรู้สึกเป็นธรรมชาติบนแพลตฟอร์มที่แตกต่างกัน โดยยอมรับความแตกต่างที่มีอยู่ในขณะที่แชร์โค้ดให้มากที่สุด ในระหว่างการพัฒนา แอป Flutter จะทำงานใน VM ที่มีการรีโหลดการเปลี่ยนแปลงอย่างรวดเร็วโดยไม่ต้องคอมไพล์ใหม่ทั้งหมด สำหรับการเปิดตัว แอป Flutter จะได้รับการคอมไพล์โดยตรงไปยังรหัสเครื่อง ไม่ว่าจะเป็นคำสั่ง Intel x64 หรือ ARM หรือ JavaScript หากกำหนดเป้าหมายไปที่เว็บ เฟรมเวิร์กนี้เป็นโอเพ่นซอร์ส พร้อมใบอนุญาต BSD ที่อนุญาต และมีระบบนิเวศที่เฟื่องฟูของแพ็คเกจบุคคลที่สามที่เสริมการทำงานของไลบรารีหลัก

2.10 ภาษา Dart

Dart เป็นภาษาที่ปรับให้เหมาะกับไคลเอ็นต์สำหรับการพัฒนาแอปที่รวดเร็วบนทุกแพลตฟอร์ม เป้าหมายของมันคือการนำเสนอภาษาการเขียนโปรแกรมที่มีประสิทธิผลมากที่สุดสำหรับการพัฒนาหลายแพลตฟอร์ม จับคู่กับแพลตฟอร์มรันไทม์การดำเนินการที่ยืดหยุ่นสำหรับเฟรมเวิร์กของแอปพลิเคชัน ภาษาถูกกำหนดโดยขอบเขตทางเทคนิคตัวเลือกที่เกิดขึ้นระหว่างการพัฒนาซึ่งกำหนดความสามารถและจุดแข็งของภาษา Dart ได้รับการออกแบบมาสำหรับซองจดหมายทางเทคนิคที่เหมาะสมอย่างยิ่งกับการพัฒนาไคลเอนต์ โดยจัดลำดับความสำคัญของทั้งการพัฒนา (การรีโหลดสถานะแบบด่วนย่อยที่สอง) และประสบการณ์การผลิตคุณภาพสูงในเป้าหมายการรวบรวมที่หลากหลาย (เว็บ มือถือ และเดสก์ท็อป)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาษา Dart เป็นประเภทที่ปลอดภัย มันใช้การตรวจสอบประเภทคงที่เพื่อให้แน่ใจว่าค่าของตัวแปรตรงกับประเภทคงที่ของตัวแปรเสมอ บางครั้งสิ่งนี้เรียกว่าการพิมพ์ด้วยเสียง แม้ว่าประเภทจะบังคับ แต่คำอธิบายประกอบประเภทก็เป็นทางเลือกเนื่องจากการอนุมานประเภทระบบการพิมพ์ Dart ยังมีความยืดหยุ่น ทำให้สามารถใช้ประเภทไดนามิกร่วมกับการตรวจสอบรันไทม์ ซึ่งจะมีประโยชน์ระหว่างการทดสอบหรือสำหรับโค้ดที่ต้องการไดนามิกเป็นพิเศษ Dart เสนอความปลอดภัยที่เป็นโมฆะ หมายความว่าค่าต่างๆ จะเป็นค่าว่างไม่ได้ เว้นแต่คุณจะบอกว่าเป็นได้ด้วยความปลอดภัยที่เป็นโมฆะ Dart สามารถปกป้องคุณจากข้อยกเว้นที่เป็นโมฆะในขณะใช้งานจริงผ่านการวิเคราะห์โค้ดแบบคงที่ ต่างจากภาษาอื่นๆ ที่ปลอดภัยสำหรับค่า null เมื่อ Dart กำหนดว่าตัวแปรไม่เป็นค่าว่าง ตัวแปรนั้นจะไม่เป็นค่าว่างเสมอ หากคุณตรวจสอบโค้ดที่รันอยู่ในโปรแกรมดีบั๊ก คุณจะเห็นว่าการไม่มีค่าเป็นโมฆะยังคงอยู่ที่รันไทม์ (เพราะฉะนั้นจึงปลอดภัยไร้ค่า)

Dart เป็นภาษากลุ่ม Compiler นั่นคือจำต้อง Compile ก่อนเอาโปรแกรมไปรัน ไม่เหมือนภาษากลุ่ม Script ที่ใช้ interpreter ในการรันตัว source code ตรงๆ ตัวโปรแกรมจำเริ่มทำงานที่ฟังก์ชัน main เป็นหลัก เราไม่สามารถเขียน statement นอกฟังก์ชันได้ การแสดงผลมาตราฐานจะใช้คำสั่ง print (คำสั่งนี้ auto-newline เสมอ) เรื่องหนึ่งที่ควรจำคือภาษา Dart นั้นการเขียน; (semi-colon) ไม่ใช่ optional คือจำเป็นต้องใส่; ทุกครั้งหลังจบ statement ไม่สามารถละ; ได้แบบภาษาตระกูล C ยุคใหม่ๆ เช่น JavaScript หรือ Kotlin

2.11 แหล่งจ่ายไฟสำหรับไมโครคอนโทรลเลอร์

ในการใช้งานไมโครคอนโทรลเลอร์ในงานด้าน IoT จำเป็นต้องมีการใช้แบตเตอรี่เป็นแหล่งจ่ายไฟในการทำให้อุปกรณ์สามารถใช้งานได้ยาวนานโดยจะมีปัจจัยคือ Hardware Design ,Power Source และ Coding

2.11.1 Hardware Design

ถ้าเราใช้บอร์ดสำเร็จรูปตระกูล ESP8266 อย่างเช่น WeMos D1 mini หรือ Nodemcu จะสะดวกต่อการใช้งานเพราะเสียบสาย USB ก็สามารถโปรแกรมด้วย Arduino IDE ได้แล้วแถมมี Regulator on board มาให้ด้วย ทำให้การต่อ sensor ต่างๆที่ใช้ไฟ 3.3v สะดวกขึ้น มี led แสดงสถานะของบอร์ด มี WiFi ในตัว แต่ถ้านำไปใช้งานทางด้าน iot ที่ต้องทำงานด้วย battery หรืองานที่มีข้อจำกัดทางด้านการจ่ายไฟเลี้ยงให้ อาจจะไม่เหมาะนัก เพราะอุปกรณ์เสริมที่ว่ามาทั้งหมดนั้นมีการใช้กระแสในการทำงาน ถ้าจะเอาไปใช้จริงอาจต้องบัดกรีหรือมีการตัดลายวงจรบางส่วนที่ไม่จำเป็นออก ซึ่ง IC Regulator ที่ยอดฮิตก็มีหลายตัวด้วยกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ยกตัวอย่างเช่น

- MCP16251: StepUp Switching Regulator
- MCP1700: LDO Regulator
- AMS1117-5: LDO Regulator

โดยที่การเลือก regulator มาใช้กับงานที่แหล่งจ่ายไฟเป็นแบตเตอรี่หรือต้องการประหยัดพลังงาน จะมี term ตัวหนึ่งที่ผู้ออกแบบอุปกรณ์ให้ความสำคัญก็คือ Quiescent Current นั่นก็คืออัตราการกินกระแสในสภาวะปกติ อย่างที่ได้กล่าวไปข้างต้น ดังนั้นถึงแม้เราจะให้ MCU ของเรา Deep Sleep ขนาดไหน เจ้า Regulator ที่ต่อไว้ก็ต้องการกระแสเพื่อใช้ในสถานะปกติอยู่ดี ฉะนั้นเมื่อเปิด Datasheet ดู Quiescent Current ยิ่งน้อยเท่าไหร่ ก็ยิ่งประหยัดไฟได้ดีขึ้น แต่ก็ต้องดูว่าไม่ใช่เลือกแต่ IC ที่ Quiescent Current น้อยๆ แต่สุดท้ายจ่ายกระแสให้กับ MCU และวงจรการทำงานที่เหลือของอุปกรณ์ไม่ได้

2.11.2 Power Source

แบตเตอรี่ที่มีขนาดความจุยิ่งเยอะยิ่งดี แต่ก็ต้องคำนึงถึงความเหมาะสมด้วย เพราะความจุเยอะก็มักจะตามมาด้วยขนาด น้ำหนัก และราคา ที่สูงขึ้น ส่วนจะเลือกแบตเตอรี่ประเภทไหนนั้น ก็ขึ้นอยู่กับว่าเราออกแบบ Hardware ไว้อย่างไร บางอุปกรณ์อาจจะใช้ StepUp Regulator เพื่อที่จะใช้ไฟให้คุ้มค่าจากแบตเตอรี่ที่สุด เรียกได้ว่าความจุแบตเตอรี่เหลือน้อยๆ เมื่อผ่าน Regulator แล้วตัว MCU ของเราก็ยังทำงานได้ หรือกรณีของ Stepdown Regulator ก็จะจ่ายไฟที่มากกว่าแรงดันที่กำหนดไว้ เช่น 5-12v แล้วปรับค่าลงมาเหลือ 5 หรือ 3.3v คราวนี้ก็ซัดแบตเตอรี่ก้อนใหญ่ความจุเยอะได้ ส่วนแบบไหนจะดีกว่ากันก็ขึ้นอยู่กับเทคนิคของแต่ละท่าน หรือจะตัดในส่วนของ Regulator ไปเลยแล้วจ่ายตรงให้กับ MCU ก็ได้ ซึ่งก็ต้องเลือกในส่วนของชนิดแบตเตอรี่ให้เหมาะสมและดูงานของเราด้วยว่าสามารถทำงานในแรงดันที่จ่ายให้ได้ถูกต้องหรือไม่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 2.7 ตัวอย่างแบตเตอรี่

จากรูปที่ 2.7 แบตเตอรี่ในรูปมีแค่สองประเภทเท่านั้นเอง คือ Li-ion และ LiFePo4 จะต่างกันก็แค่ขนาดและความจุเท่านั้น โดยคุณสมบัติหลักๆที่แตกต่างกันของแบตเตอรี่สองประเภทนี้ก็คือแรงดันที่จ่าย

2.11.3 Coding

การเขียนทุกบรรทัด ทุกคำสั่งในการทำงานแต่ละบรรทัดของโปรแกรมทีเราเขียน หมายถึงรอบการทำงานของ MCU ของอุปกรณ์ของเรา และนั่นย่อมหมายถึงพลังงานที่ใช้จากแหล่งที่จ่ายให้กับอุปกรณ์ ยิ่ง Code ของเรายาวและมีการทำงานที่นานขึ้น นั่นก็จะทำให้อุปกรณ์ของเราใช้พลังงานนานขึ้นด้วย ฉะนั้นบางครั้งเมื่อออกแบบวงจรทางด้าน Hardware เสร็จแล้ว เขียน Code ที่สามารถทำงานเสร็จ ได้ผลอย่างที่ต้องการแล้ว เราควรทำการ Optimize Code ของเราด้วย หรือในกรณีของการทำ Deep Sleep อันนี้ขึ้นกับตระกูลของ MCU ที่ใช้ ซึ่งก็จะมีการ Deep Sleep ได้หลายแบบ บางแบบยังสามารถเชื่อมต่อ Wi-Fi ได้ บางแบบตัดทุกอย่างรอ External Interrupt อย่างเดียวเพื่อปลุกขึ้นมาให้ทำงานต่อ ก็จะประหยัดการใช้พลังงานไปได้อีก

2.11.4 วิธีคำนวณระยะเวลาที่เหลืออยู่ของแบตเตอรี่

วิธีคำนวณระยะเวลาที่เหลืออยู่ของแบตเตอรี่ จากการใช้ แรงดัน กระแส หรือกำลัง เพื่อหาแบตเตอรี่ที่เหมาะสมกับงานมีข้อควรรู้เบื้องต้นดังนี้

1) A = Ampere

2) ความจุแบตเตอรี่ มีหน่วยเป็น mAh คือ มิลลิแอมป์ชั่วโมง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หมายถึงประมาณกระแสหน่วยมิลลิแอมป์ที่ใช้ไปแล้วจำนวณเวลาหน่วยชั่วโมง เช่น แบตเตอรี่ก้อนนี้หากจ่ายไฟกระแส 5 mA จะจ่ายได้ 10 h (ชั่วโมง) แสดงว่าแบตเตอรี่ก้อนนี้มีความจุ 5(mA) x 10(h) = 50 mAh เป็นต้นหรืออาจเป็นหน่วย Ah คือ แอมป์ชั่วโมง เช่น แบตเตอรี่ก้อนนี้หากจ่ายกระแส 0.5 A จะจ่ายได้ 5 h แสดงว่าแบตเตอรี่ก้อนนี้มีความจุ 0.5(A) x 5(h) = 2.5Ah หรือเท่ากับ 2500 mAhหรืออาจเป็นหน่วย Wh คือ วัตต์ชั่วโมง เช่น แบตเตอรี่ก้อนนี้หากจ่ายกำลังให้อุปกรณ์ 3 W จะจ่ายได้ 20 h แสดงว่าแบตเตอรี่ก้อนนี้มีความจุ 3 (W) x 20 (h) = 60 Wh แต่หากจะแปลงให้เป็น mAh เราต้องทราบแรงดันของแบตเตอรี่ที่ใช้ เช่น 5V แสดงว่าแบตเตอรี่ก้อนนี้จ่ายกระแสให้อุปกรณ์เท่ากับ (จาก I=P/V) 3W / 5V = 0.6 A = 600 mA

สูตรการคำนวณที่ควรรู้

รู้กระแส รู้ระยะเวลาสามาถหาปริมาณความจุที่ใช้ไป

$$(A)*(h)*1000=(mAh) \tag{2.1}$$

รู้ความจุ ใช้กระแส สามารถหาเวลาที่เหลือที่แบตเตอรี่จ่ายได้

$$(mAh)/(Amps*1000)=(hours) \tag{2.2}$$

รู้ความจุ ใช้แรงดัน กับกำลัง คำนวณหาเวลาที่เหลือที่แบตเตอรี่จ่ายได้

$$(mAh*Volt)/watts*1000=(hours) \tag{2.3}$$

2.12 IFTTT

IFTTT หรือ IF This Then That มาจากชื่อคำสั่งเงื่อนไขการเขียนโปรแกรม "ถ้าเป็นอย่างนั้น" สิ่งที่บริษัทจัดหาให้คือแพลตฟอร์มซอฟต์แวร์ที่เชื่อมต่อแอป อุปกรณ์ และบริการจากนักพัฒนาต่างๆ เพื่อกระตุ้นการทำงานอัตโนมัติอย่างน้อยหนึ่งอย่างที่เกี่ยวข้องกับแอป อุปกรณ์ และบริการเหล่านั้นคือบริการออนไลน์ที่ช่วยให้ผู้ใช้สาามารถตั้งค่าการททำงานได้อย่างยืดหยุ่น ตามเงื่อนไขที่เราต้องการยกตัวอย่างเช่น ให้เก็บข้อมูลที่อ่านได้จาก เซนเซอร์ ไว้ที่ Google Sheets หรือถ้ามีเหตุการณ์เกิดขึ้นให้แจ้งไปที่ Line Notify กล่าวคือ IFTTT เป็นเว็บและแอปพลิเคชัน ที่นำเอา API ของ Service หลายเจ้าในโลกใบนี้เข้ามาใช้ด้วยกันได้ สามารถสร้างสูตร (Recipe) ขึ้นมาได้อย่างอิสระและนำไปแชร์ให้บุคคลอื่นใช้ได้อีกด้วย IFTTT มีทั้งบนเว็บไซต์ และบนแอป iOS, Android

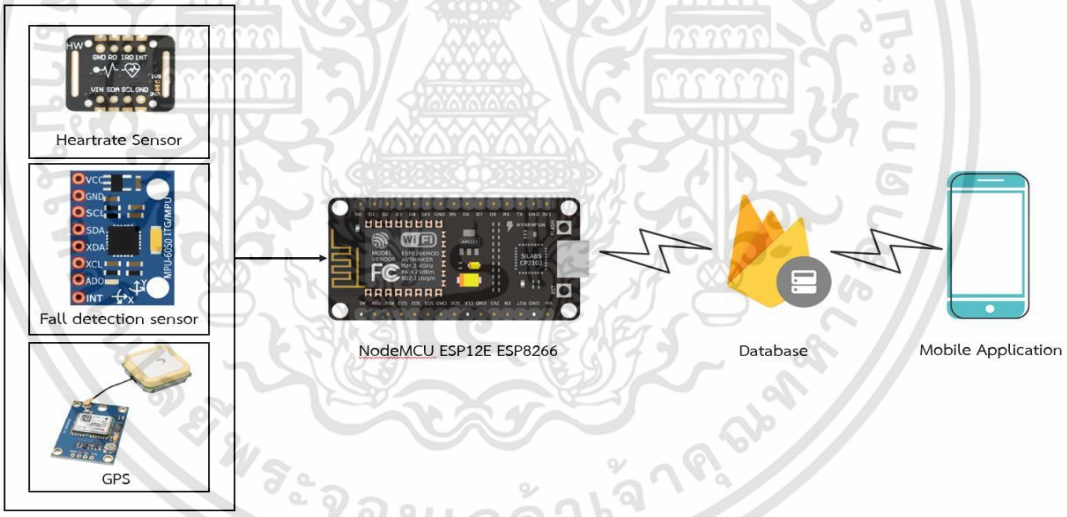
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3

การออกแบบและการจัดทำปริญญานิพนธ์

3.1 การออกแบบ

ปริญญานิพนธ์นี้เป็นการออกแบบอุปกรณ์ที่สามารถวัดอัตราการเต้นของหัวใจ, แจ้งเตือนการหกล้ม, และระบุตำแหน่งของผู้ป่วยโรคเรื้อรัง โดยนำข้อมูลทั้งหมดมาแสดงผล และแจ้งเตือนบนแอพลิเคชันมือถือโดยใช้เซนเซอร์ MAX30102 วัดอัตราการเต้นของหัวใจ, ใช้โมดูล GY-521 MPU6050 ตรวจจับการหกล้ม, ใช้โมดูล GPS GY-NEO6MV2 ในการระบุตำแหน่ง โดยนำข้อมูลทั้งหมดเชื่อมต่อกับโมดูล Wi-Fi NodeMCU ESP12E ESP8266 เพื่อทำการควบคุมการทำงานของเซนเซอร์และ เชื่อมต่อ Wi-Fi เพื่อส่งข้อมูลที่ได้จากเซนเซอร์ไปยังเซิร์ฟเวอร์เพื่อเก็บเป็นฐานข้อมูลเพื่อนำข้อมูลมาแสดงบน Mobile Application ได้ต่อไป แสดงบล็อกไดอะแกรมของปริญญานิพนธ์ได้ดังรูปที่ 3.1



รูปที่ 3.1 แสดงบล็อกไดอะแกรมของปริญญานิพนธ์

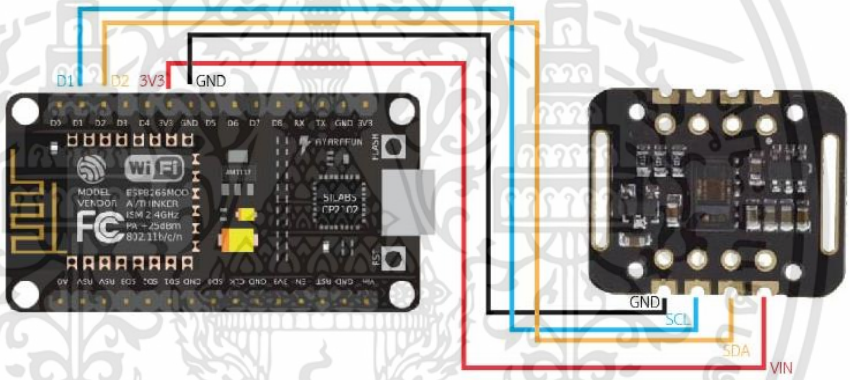
จากบล็อกไดอะแกรมในรูปที่ 3.1 แสดงถึงภาพรวมของปริญญานิพนธ์โดยสามารถอธิบายการทำงานได้ดังนี้คือใช้เซนเซอร์ MAX30102 วัดอัตราการเต้นของหัวใจ, ใช้โมดูล GY-521 MPU6050 ตรวจจับการหกล้ม, ใช้โมดูล GPS GY-NEO6MV2 ในการระบุตำแหน่ง โดยนำข้อมูลทั้งหมดเชื่อมต่อกับโมดูล Wi-Fi NodeMCU ESP12E ESP8266 โดยความสามารถของ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ไมโครคอนโทรลเลอร์ ESP8266 สามารถเชื่อมต่อ Wi-Fi เพื่อส่งข้อมูลจากเซนเซอร์ทั้งสามตัว ไปยังฐานข้อมูลแบบเรียลไทม์ (Real-time Database) บน Firebase จากนั้นทำการดึงข้อมูลจากฐานข้อมูลแบบเรียลไทม์ไปแสดงบน Mobile Application ได้

3.1.1 ออกแบบการเชื่อมต่อโมดูล Wi-Fi NodeMCU ESP12E ESP8266 กับเซนเซอร์ MAX30102

ทำการออกแบบการเชื่อมต่อโมดูลWi-Fi NodeMCU ESP12E ESP8266 กับเซนเซอร์ MAX30102 เพื่อทำการวัดอัตราการเต้นของหัวใจ และเขียนคำสั่งในโปรแกรม Arduino IDE เพื่อให้เซนเซอร์อ่านอัตราการเต้นของหัวใจ โดยทำการเชื่อมต่อขาของอุปกรณ์ดังรูปที่ 3.2



รูปที่ 3.2 แสดงการเชื่อมต่อขาของโมดูล Wi-Fi NodeMCU ESP12E ESP8266 และเซนเซอร์ MAX30102

จากรูปที่ 3.2 แสดงการเชื่อมต่อขาของเซนเซอร์ MAX30102 และ โมดูลWi-Fi NodeMCU ESP12E ESP8266 โดยขา D1 ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา SCL ของเซนเซอร์ MAX30102, ขา D2 ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา SDA ของเซนเซอร์ MAX30102, ขา 3V3 ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา VIN ของเซนเซอร์ MAX30102 และขา GND ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา GND ของเซนเซอร์ MAX301021 แสดงความสัมพันธ์ได้ดังตารางที่ 3.1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 3.1 แสดงการเชื่อมต่อขาของเซนเซอร์ MAX30102 และ โมดูล Wi-Fi NodeMCU ESP12E ESP8266

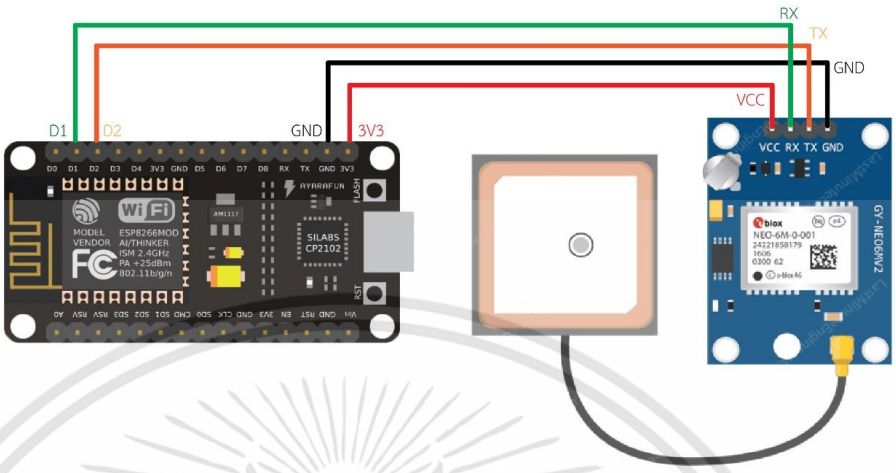
โมดูล Wi-Fi NodeMCU ESP12E ESP8266	**เซนเซอร์ MAX30102**
D1	SCL
D2	SDA
3V3	VIN
GND	GND

จากตารางที่ 3.1 แสดงการเชื่อมต่อขาของเซนเซอร์ MAX30102 และ NodeMCU ESP12E ESP8266 โดยขา D1 ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา SCL ของเซนเซอร์ MAX30102, ขา D2 ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา SDA ของเซนเซอร์ MAX30102, ขา 3V3 ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา VIN ของเซนเซอร์ MAX30102 และขา GND ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา GND ของเซนเซอร์ MAX301021

3.1.2 ออกแบบการเชื่อมต่อโมดูล Wi-Fi NodeMCU ESP12E ESP8266 กับโมดูล GPS GY-NEO-6MV2

ทำการออกแบบการเชื่อมต่อโมดูล Wi-Fi NodeMCU ESP12E ESP8266 กับ โมดูล GPS GY-NEO-6MV2 เพื่อทำการอ่านค่าพิกัดตำแหน่งละติจูด ลองจิจูด และเขียนคำสั่งในโปรแกรม Arduino IDE เพื่อให้เซนเซอร์อ่านค่าพิกัดตำแหน่งละติจูด ลองจิจูด โดยทำการเชื่อมต่อขาของอุปกรณ์ดังรูปที่ 3.3

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.3 แสดงการเชื่อมต่อขาของโมดูล Wi-Fi NodeMCU ESP12E ESP8266 และโมดูล GPS GY-NEO-6MV2

จากรูปที่ 3.3 แสดงการเชื่อมต่อขาโมดูล Wi-Fi NodeMCU ESP12E ESP8266 และโมดูล GPS GY-NEO-6MV2 โดยขา D1 ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา RX ของ GPS GY-NEO6MV2, ขา D2 ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา TX ของ GPS GY-NEO6MV2, ขา GND ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา GND ของ GPS GY-NEO6MV2 และ ขา 3V3 ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา VCC ของ GPS GY-NEO6MV2 ซึ่งแสดงความสัมพันธ์ได้ดังตารางที่ 3.2

ตารางที่ 3.2 แสดงการเชื่อมต่อขาของ GPS GY-NEO6MV2 กับ NodeMCU ESP12E ESP8266

โมดูล Wi-Fi NodeMCU ESP12E ESP8266	**โมดูล GY-521 MPU6050**
D1	RX
D2	TX
3V3	VCC

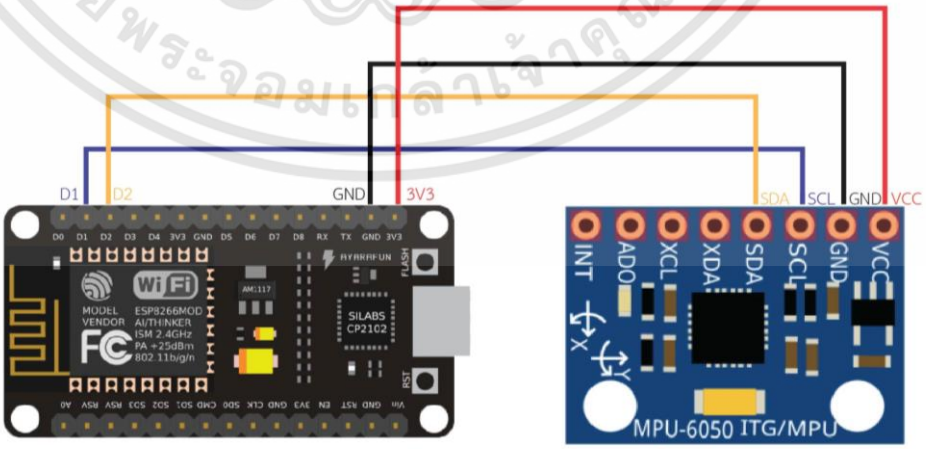
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โมดูล Wi-Fi NodeMCU ESP12E ESP8266	โมดูล GY-521 MPU6050
GND	GND

จากตารางที่ 3.2 แสดงการเชื่อมต่อขาของโมดูล GPS GY-NEO6MV2 กับ NodeMCU ESP12E ESP8266 โดยขา D1 ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา RX ของ GPS GY-NEO6MV2, ขา D2 ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา TX ของ GPS GY-NEO6MV2, ขา GND ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา GND ของ GPS GY-NEO6MV2 และ ขา 3V3 ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา VCC ของ GPS GY-NEO6MV2

3.1.3 ออกแบบการเชื่อมต่อโมดูล Wi-Fi NodeMCU ESP12E ESP8266 กับโมดูล GY-521 MPU6050

ทำการออกแบบการเชื่อมต่อโมดูล Wi-Fi NodeMCU ESP12E ESP8266 กับโมดูล GY-521 MPU6050 และเขียนคำสั่งในโปรแกรม Arduino IDE เพื่อให้โมดูลล GY-521 MPU6050 ตรวจจับการตกลงมาด้วยความเร็วระดับหนึ่งจะทำการส่งสัญญาณผ่าน Wi-Fi ไปยังเว็บไซต์ IFTTT เพื่อทำการส่งสัญญาณแจ้งเตือนไปยังโทรศัพท์มือถือผ่านแอพลิเคชัน Line โดยเชื่อมต่อบอร์ดดังรูปที่ 3.4



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 3.4 แสดงการเชื่อมต่อขาของโมดูล Wi-Fi NodeMCU ESP12E ESP8266 และโมดูลGY-521 MPU6050

จากรูปที่ 3.4 แสดงการเชื่อมต่อขาของเซนเซอร์ GY-521 MPU6050 และ NodeMCU ESP12E ESP8266 โดยขา D1 ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา SCL เซนเซอร์ GY-521 MPU6050,ขา D2 ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา SDA เซนเซอร์ GY-521 MPU6050, ขา 3V3 ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา VCC เซนเซอร์ GY-521 MPU6050 และขา GND ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา GND เซนเซอร์ GY-521 MPU6050 แสดงความสัมพันธ์ได้ดังตารางที่ 3.3

ตารางที่ 3.3 แสดงการเชื่อมต่อขาของโมดูล Wi-Fi NodeMCU ESP12E ESP8266 และ โมดูลGY-521 MPU6050

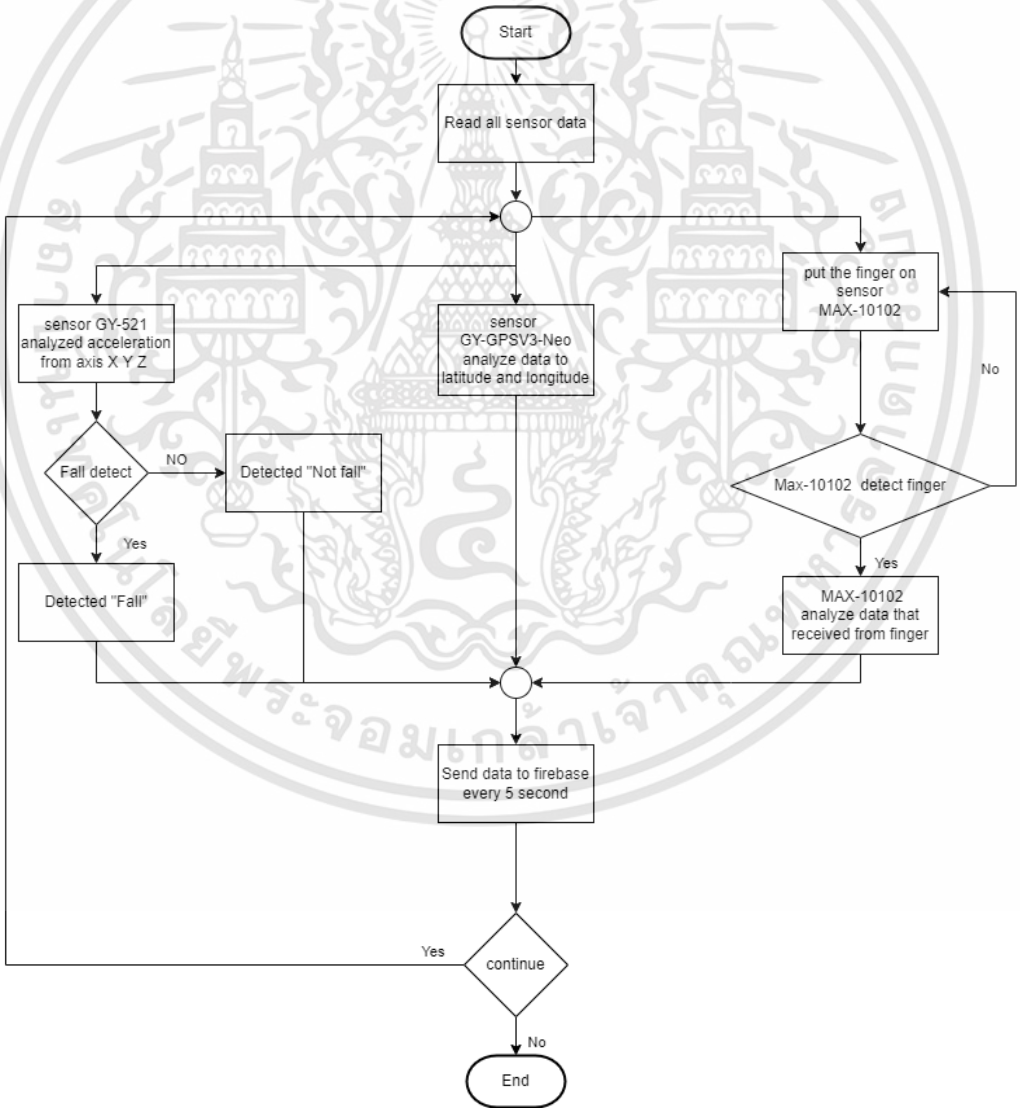
โมดูล Wi-Fi NodeMCU ESP12E ESP8266	**โมดูล GY-521 MPU6050**
D1	SCL
D2	SDA
3V3	VIN
GND	GND

จากตารางที่ 3.3 แสดงการเชื่อมต่อขาของโมดูล GY-521 MPU6050 และ โมดูล Wi-Fi NodeMCU ESP12E ESP8266โดยขา D1 ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา SCL เซนเซอร์ GY-521 MPU6050,ขา D2 ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา SDA เซนเซอร์ GY-521 MPU6050, ขา 3V3 ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา VCC เซนเซอร์ GY-521 MPU6050 และขา GND ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา GND เซนเซอร์ GY-521 MPU6050

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.1.4 ออกแบบโปรแกรม Arduino IDE รวมเซนเซอร์ทุกตัวคือ MAX30102, GPS GY-NEO6MV2 และ GY-521 MPU6050 ให้สามารถใช้ NodeMCU ESP12E ESP8266 ควบคุมได้ และส่งข้อมูลทั้งหมดเก็บบนฐานข้อมูลบน Firebase

หลังจากทำการทดลองเชื่อมต่ออุปกรณ์แต่ละตัวเข้ากับ NodeMCU ESP12E ESP8266 ทีละตัว หลังทำการทดลอง และทดสอบการทำงานทั้งหมด จึงได้ทำการออกแบบโปรแกรม Arduino IDE รวมเซนเซอร์ทุกตัวคือ GY-521 MPU6050, GPS GY-NEO6MV2 และ MAX30102 ให้สามารถใช้ NodeMCU ESP12E ESP8266 ควบคุมได้ โดยแผนผังการทำงานของโปรแกรมจะแสดงในรูปที่ 3.5



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากรูปที่ 3.5 แสดงแผนผังการทำงานของโปรแกรมเซนเซอร์ MAX30102, GY-521 MPU6050 และGPS GY-NEO6MV2 ที่ควบคุมโดย NodeMCU ESP12E ESP8266 โดยแผนผังการทำงานเริ่มต้นด้วยการเก็บข้อมูลจากเซนเซอร์ทุกตัวประกอบไปด้วย 3 เซนเซอร์คือ MAX30102, GY-521 MPU6050 และ GPS GY-NEO6MV2 โดยจะนำไปประมวลผลที่ NodeMCU ESP12E ESP8266 โดยมีเซนเซอร์มีดังนี้

1. GY-521 MPU6050 จะทำการประมวลผลข้อมูลที่ได้เป็นค่าความเร่งในแกน X Y Z โดยค่าความเร่งในแกน X Y Z เปลี่ยนเป็นตัวเลขเพื่อให้ง่ายต่อการตัดสินใจของระบบค่าความเร่งที่เซนเซอร์อ่านได้ตอนไม่เกิดความเร่งจะอยู่ที่ 9 และจะแจ้งเตือนว่า “ไม่เกิดการล้ม” เมื่อค่าความเร่งที่อ่านได้เพิ่มขึ้นจนมากกว่าหรือเท่ากับ 12 จะทำการแจ้งเตือนว่า “เกิดการล้ม” โดยประมวลผลที่ NodeMCU

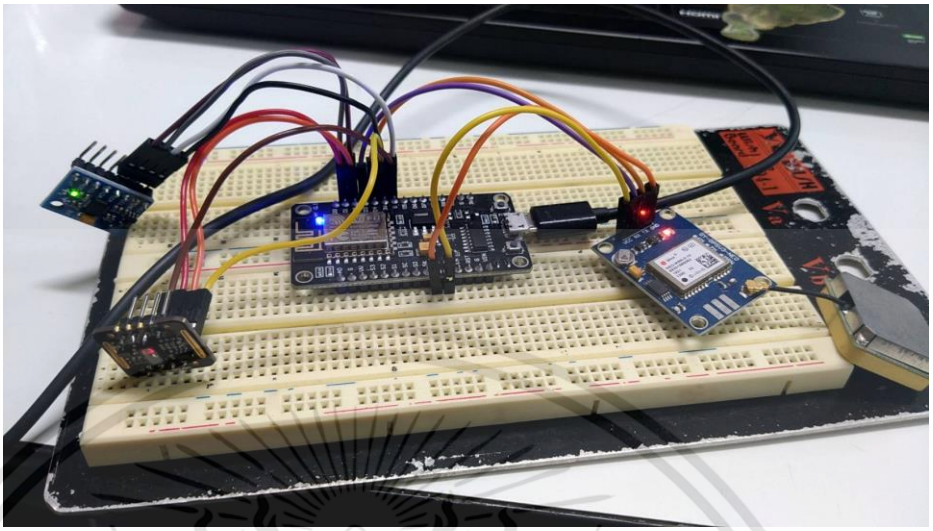
2. GPS GY-NEO6MV2 จะทำการอ่านข้อมูลออกมาเป็นตำแหน่งละติจูดและลองจิจูดเพื่อนำค่าไปแสดงผลต่อไป

3 .MAX-10102 จะทำการอ่านค่าอัตราการเต้นของหัวใจเมื่อมีนิ้วมือวางอยู่บนตัวเซนเซอร์หากไม่มีนิ้วมือบนเซนเซอร์จะไม่สามารถอ่านค่าออกมาได้ เมื่อได้ค่าอัตราการเต้นของหัวใจแล้วจะทำการประมวลผลเป็นอัตราการเต้นของหัวใจต่อนาทีเพื่อการแสดงผลต่อไป

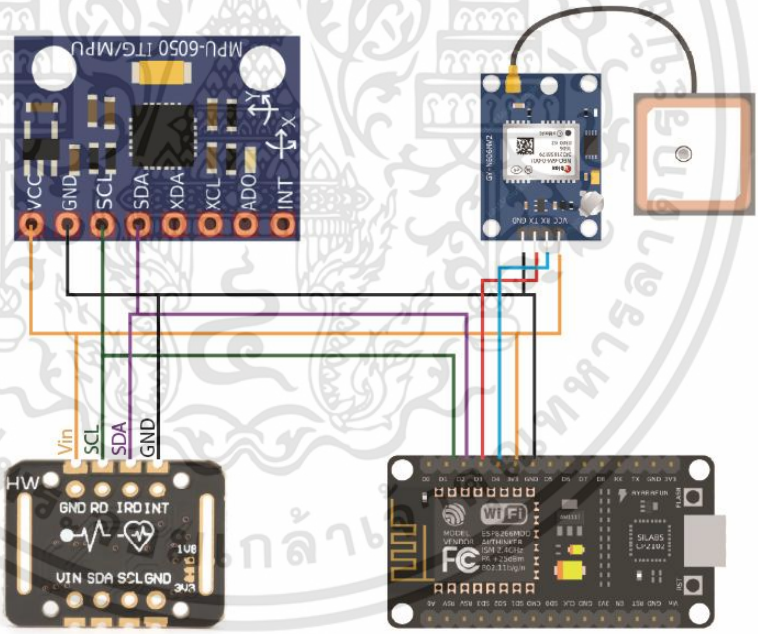
หลังจากนั้นจะทำการนำข้อมูลจากเซนเซอร์ทั้ง3ตัวที่ประมวลผลแล้วส่งผ่าน WI-FI ของ NodeMCU ESP12E ESP8266 ทุกๆ 5 วินาทีทำซ้ำจนกว่าจะหยุดการทำงาน

เมื่อทำการออกแบบโปรแกรม Arduino IDE รวมเซนเซอร์ทุกตัวคือ GY-521 MPU6050, GPS GY-NEO6MV2 และ MAX30102 ให้สามารถใช้ NodeMCU ESP12E ESP8266 ควบคุมได้ จากนั้นทำการเชื่อมต่ออุปกรณ์ทั้งหมดคือ GY-521 MPU6050, GPS GY-NEO6MV2 และ MAX30102 ให้สามารถใช้ NodeMCU ESP12E ESP8266 ควบคุมเข้าด้วยกัน โดยแสดงการเชื่อมต่ออุปกรณ์ได้ดังรูปที่ 3.6 และ 3.7

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.6 แสดงการเชื่อมต่ออุปกรณ์รวมเซนเซอร์ทุกตัวคือ GY-521 MPU6050, GPS GY-NEO6MV2 และ MAX30102 ให้สามารถใช้ NodeMCU ESP12E ESP8266 ควบคุมได้



รูปที่ 3.7 แสดงการเชื่อมต่ออุปกรณ์รวมเซนเซอร์ทุกตัวคือ GY-521 MPU6050, GPS GY-NEO6MV2 และ MAX30102 ให้สามารถใช้ NodeMCU ESP12E ESP8266 ควบคุมได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากรูปที่ 3.6 และ 3.7 แสดงการเชื่อมต่ออุปกรณ์รวมเซนเซอร์ทุกตัวคือ GY-521 MPU6050, GPS GY-NEO6MV2 และ MAX30102 ให้สามารถใช้ NodeMCU ESP12E ESP8266 ควบคุมได้ โดยที่ขา D1 ของ NodeMCU ESP12E ESP8266 เชื่อมกับขา SCL ของ MAX30102 และ GY-521 MPU6050, ขา D2 ของ NodeMCU ESP12E ESP8266 เชื่อมกับขา SDA ของ MAX30102 และ GY-521 MPU6050, ขา D3 ของ NodeMCU ESP12E ESP8266 เชื่อมกับขา RX ของ GPS GY-NEO6MV2, ขา D4 ของ NodeMCU ESP12E ESP8266 เชื่อมกับขา TX ของ GPS GY-NEO6MV2, ขา 3V3 ของ NodeMCU ESP12E ESP8266 เชื่อมต่อขา VCC ของ GY-521 MPU6050 และ GPS GY-NEO6MV2 และขา VIN ของ MAX30102, ขา GND ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา GND ของเซนเซอร์ทุกตัวคือ GY-521 MPU6050, GPS GY-NEO6MV2 และ MAX30102 โดยสามารถแสดงการเชื่อมต่อได้ดังตารางที่ 3.4

ตารางที่ 3.4 แสดงการเชื่อมต่อขาของอุปกรณ์รวมเซนเซอร์ทุกตัวคือ GY-521 MPU6050, GPS GY-NEO6MV2 และ MAX30102

NodeMCU ESP12E ESP8266	GY-521 MPU6050	GPS GY-NEO6MV2	MAX30102
D1	SCL	-	SCL
D2	SDA	-	SDA
D3	-	RX	-
D4	-	TX	-
3V3	VCC	VCC	VIN
GND	GND	GND	GND

จากตารางที่ 3.4 แสดงการเชื่อมต่ออุปกรณ์รวมเซนเซอร์ทุกตัวคือ GY-521 MPU6050, GPS GY-NEO6MV2 และ MAX30102 ให้สามารถใช้ NodeMCU ESP12E ESP8266 ควบคุมได้ โดยที่ขา D1 ของ NodeMCU ESP12E ESP8266 เชื่อมกับขา SCL ของ MAX30102 และ GY-521 MPU6050, ขา D2 ของ NodeMCU ESP12E ESP8266 เชื่อมกับขา SDA ของ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MAX30102 และ GY-521 MPU6050, ขา D3 ของ NodeMCU ESP12E ESP8266 เชื่อมกับขา RX ของ GPS GY-NEO6MV2, ขา D4 ของ NodeMCU ESP12E ESP8266 เชื่อมกับขา TX ของ GPS GY-NEO6MV2, ขา 3V3 ของ NodeMCU ESP12E ESP8266 เชื่อมต่อขา VCC ของ GY-521 MPU6050 และ GPS GY-NEO6MV2 และขา VIN ของ MAX30102, ขา GND ของ NodeMCU ESP12E ESP8266 เชื่อมต่อกับขา GND ของเซนเซอร์ทุกตัวคือ GY-521 MPU6050, GPS GY-NEO6MV2 และ MAX30102

3.1.5 การออกแบบฐานข้อมูลสำหรับการเก็บข้อมูล

สำหรับการออกแบบฐานข้อมูลผู้จัดทำปริญญานิพนธ์ได้ใช้งาน Firebase Realtime Database เป็นฐานข้อมูลแบบ NoSQL cloud ที่เก็บข้อมูลในรูปแบบของ JSON และมีการ Sync ข้อมูลแบบ Realtime กับทุก Device ที่เชื่อมต่อแบบอัตโนมัติในเสี้ยววินาที รองรับการทำงานเมื่อ Offline (ข้อมูลจะถูกเก็บไว้ใน Local จนกระทั่งกลับมา online ก็จะทำการ sync ข้อมูลให้อัตโนมัติ) รวมถึงมี Security Rules ให้เราสามารถออกแบบเงื่อนไขการเข้าถึงข้อมูลการ read และ write ได้ดั่งใจทั้ง Android, iOS และ Web รูปที่ 3.8 แสดง Firebase Realtime Database กับการทำงานบนทุกอุปกรณ์

รูปที่ 3.8 Firebase Realtime Database กับการทำงานบนทุกอุปกรณ์

จากรูปที่ 3.8 สำหรับปริญญานิพนธ์นี้ได้ใช้ Firebase ในการเก็บข้อมูลแบบ Realtime สำหรับข้อมูลเช่น ตำแหน่งที่ตั้ง ,อัตราการเต้นของหัวใจเฉลี่ย และการแจ้งเตือนการหกล้มทำให้ การเข้าถึงข้อมูลสำหรับ Firebase Realtime Database ทั้ง read และ write โดยปกติ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เราจะต้องทำการ Authentication ผ่าน Firebase Authentication ก่อน และหลังจากนั้นการตั้งค่า Rule ของ Firebase ดังแสดงในรูปที่ 3.9

รูปที่ 3.9 การตั้งค่า Rule ของ Firebase ที่ใช้ในปริญญานิพนธ์นี้

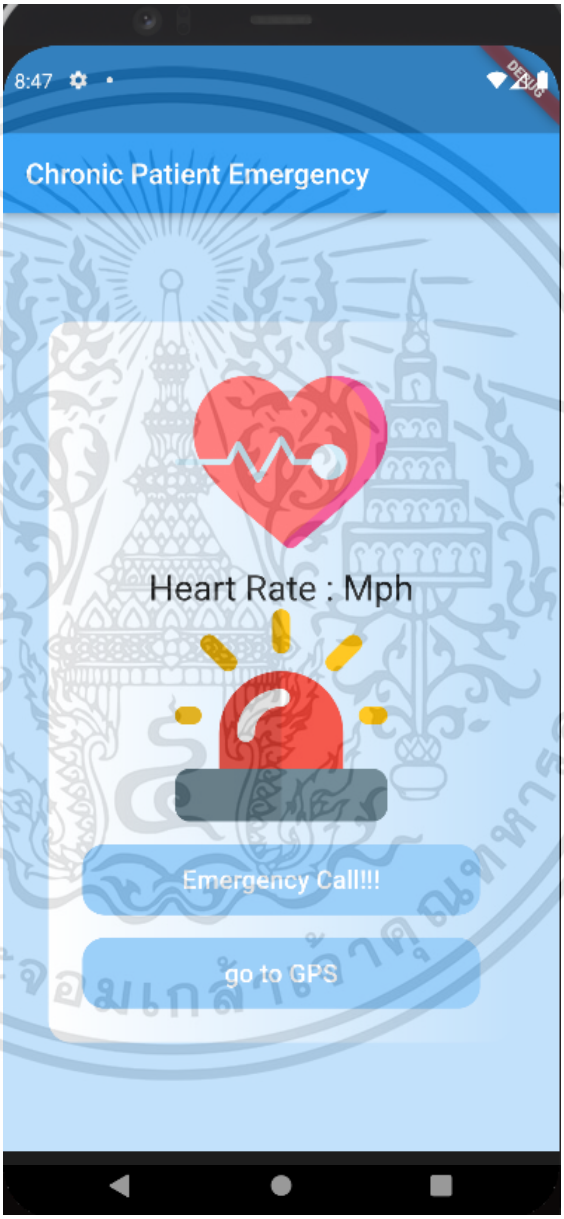
จากรูปที่ 3.9 จากการตั้งค่า Rule โดยจะตั้งไว้เป็น public ทำให้ไม่ต้อง Authentication ก็สามรถเข้าถึงข้อมูลได้โดยเมื่อเราต้องการนำข้อมูลมาเก็บไว้ที่ฐานข้อมูลจะเป็นการ write และเมื่อต้องการนำข้อมูลออกไปจะเป็นการ read

3.1.6 การออกแบบหน้าต่างของแอปพลิเคชันที่ใช้แสดงผล

ทำการออกแบบหน้าต่างแอปพลิเคชันด้วยโปรแกรม VS code โดยใช้ภาษา Dart เริ่มด้วยการสร้างโปรเจคขึ้นมาใหม่ ตั้งชื่อโปรเจค จากนั้น run คำสั่งเพื่อเริ่มต้นการสร้างแอปพลิเคชัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ขึ้นมา ชื่อแอปพลิเคชัน “Chronic Patient Emergency” เป็นแอพที่ใช้แสดงตำแหน่งของผู้ป่วย อัตราการเต้นของหัวใจ รายงานกรณีผู้ป่วยเกิดการหกล้ม ในหน้าแรกของแอปพลิเคชันจะแสดงหน้า Heart Rate, ปุ่มกดEmergency Call และปุ่มกดไปหน้าต่าง GPS โดยแสดงหน้าต่างแอปพลิเคชันหน้าแรกได้ดังรูปที่ 3.10



รูปที่ 3.10 แสดงหน้าแรกของหน้าต่างแอปพลิเคชัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

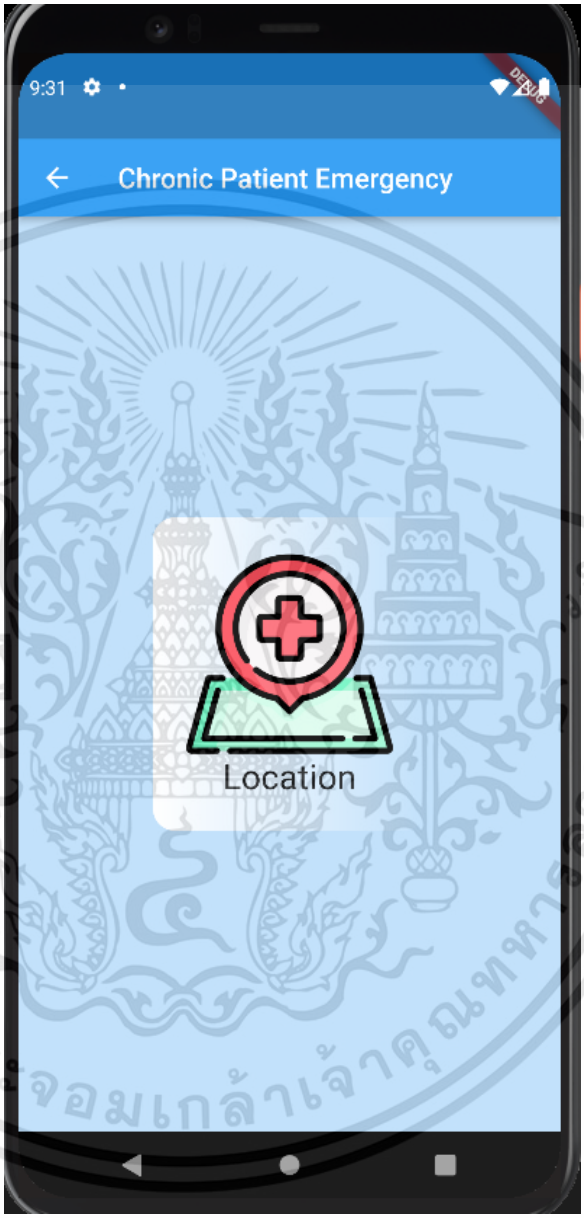
จากรูปที่ 3.10 แสดงหน้าแรกของหน้าต่างแอปพลิเคชัน ในหน้าแรกของแอปพลิเคชันจะแสดงหน้า Heart Rate, ปุ่มกด Emergency Call และปุ่มกดไปหน้าต่าง GPS เมื่อเราเลือกกดเลือกปุ่ม Emergency Call แอปพลิเคชันจะแสดงหน้าต่างดังรูปที่ 3.11

รูปที่ 3.11 หน้าต่างแอปพลิเคชันแสดงหน้าการโทรฉุกเฉิน

จากรูป 3.11 หน้าต่างแอปพลิเคชันหน้านี้แสดงหน้าถึงการโทรฉุกเฉิน เมื่อเรากดที่ปุ่มข้อความ Emergency call จะทำให้แสดงผลที่หน้าจอการโทรออกของโทรศัพท์ โดยจะแสดงเบอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โทรฉุกเฉินของโรงพยาบาลเอาไว้ เพื่อให้สามารถกดโทรออกได้ในทันที และเมื่อกดที่ปุ่มข้อความ go to GPS จะทำการแสดงผลที่หน้า GPS ดังรูปที่ 3.12



รูปที่ 3.12 หน้าต่างแอปพลิเคชันหน้า GPS

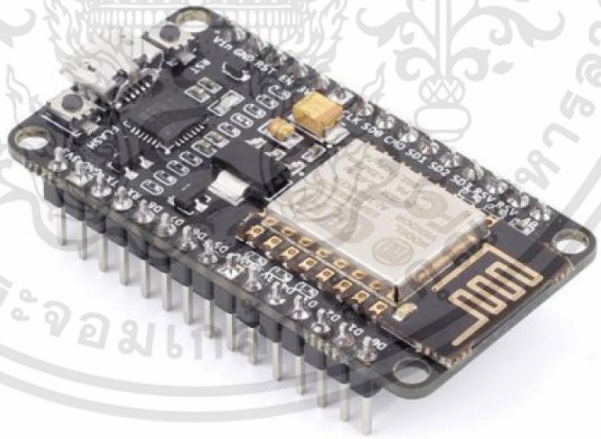
จากรูปที่ 3.12 เป็นการแสดงหน้าต่างแอปพลิเคชันที่เป็นหน้า GPS เมื่อกดที่ปุ่มข้อความ go to GPS จะทำการแสดงผลที่หน้า GPS เพื่อใช้ระบุตำแหน่งของผู้ป่วยให้ได้รับรู้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2 เครื่องมือที่ใช้ในการทดลอง

3.2.1 NodeMCU ESP12E ESP8266

NodeMCU คือ บอร์ดคล้าย Arduino ที่สามารถเชื่อมต่อกับ Wi-Fi ได้, สามารถเขียนโปรแกรมด้วย Arduino IDE ได้เช่นเดียวกับ Arduino และบอร์ดก็มีราคาถูก ESP8266 NodeMCU เป็นบอร์ดพัฒนาที่ได้รับความนิยมและใช้กันอย่างแพร่หลายโดยอิงจากโมดูล ESP-12E WiFi ที่รวมองค์ประกอบของการเขียนโปรแกรมอย่างง่ายเข้ากับ Arduino IDE (C\C++) และความสามารถ WiFi ผ่านโปรแกรมเมอร์ในตัวและชิป USB-to-Serial CH340G การแฟลช ESP8266 และเอาต์พุตอนุกรมบนพีซี การพัฒนาและโครงการสร้างต้นแบบทำได้อย่างง่ายดาย เช่นเดียวกับบอร์ด Arduino นั้น ESP8266 NodeMcu มีพิน GPIO, ตัวควบคุมแรงดันไฟฟ้า, ADC, พอร์ต Micro-USB (สำหรับการกะพริบและเอาต์พุตแบบอนุกรม) ทั้งหมดนี้รวมอยู่ในบอร์ดเดียว ยิ่งไปกว่านั้น ESP8266 NodeMcu ยังมี WiFi เต็มรูปแบบที่ดูแลการสื่อสาร WiFi ไปยังเซิร์ฟเวอร์หรือไคลเอนต์ โดยสามารถแสดงข้อมูลทางเทคนิคได้ดังตารางที่ 3.5 และแสดงรูปของ NodeMCU ESP12E ESP8266 ได้ดังรูปที่ 3.13



จากรูปที่ 3.13 แสดง NodeMCU ESP12E ESP8266

จากรูปที่ 3.13 แสดง NodeMCU ESP12E ESP8266 ภายในบอร์ดของ NodeMCU ประกอบไปด้วย ESP8266 ซึ่งเป็นไมโครคอนโทรลเลอร์ที่สามารถเชื่อมต่อ Wi-Fi ได้ พร้อมอุปกรณ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

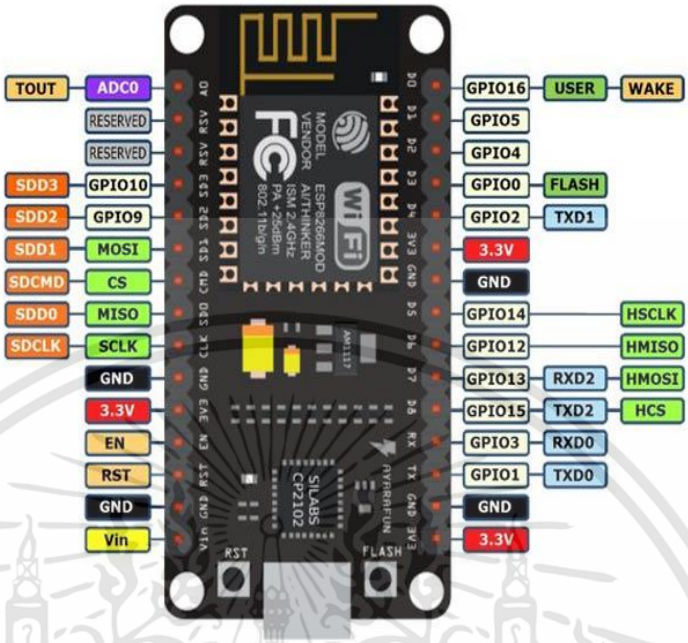
อำนวยความสะดวกต่างๆ เช่น พอร์ต micro-USB สำหรับจ่ายไฟ/อัปโหลดโปรแกรม, ชิพสำหรับอัปโหลดโปรแกรมผ่านสาย USB, ชิพแปลงแรงดันไฟฟ้า และขาสำหรับเชื่อมต่ออุปกรณ์ภายนอก

ตารางที่ 3.5 แสดงข้อมูลทางเทคนิคของ NodeMCU ESP12E ESP8266

ชื่อ	คุณสมบัติ
ไมโครคอนโทรลเลอร์	ESP8266 / 32 บิต
ความเร็วสัญญาณนาฬิกา	80 เมกะเฮิรตซ์
ตัวแปลง USB	CP2102
ตัวเชื่อมต่อ USB	Micro USB
แรงดันใช้งาน	3.3 โวลต์
หน่วยความจำแฟลช	4 เมกะไบต์
ดิจิทัล I/O	11
แอนะล็อกอินพุต	1
การสื่อสาร	ซีเรียล, เอสพีไอ. I2C และ 1-Wire ผ่านไลบรารีซอฟต์แวร์
Wi-Fi	802.11 b/g/n

จากตารางที่ 3.5 แสดงข้อมูลทางเทคนิคพื้นฐานของ NodeMCU ESP12E ESP8266 ซึ่งบอร์ด NodeMCU ESP12E ESP8266 ภายในมีไมโครคอนโทรลเลอร์ ESP8266 ซึ่งสามารถเชื่อมต่อ Wi-Fi ได้มาตรฐาน 802.11 b/g/n มีความเร็วสัญญาณนาฬิกาอยู่ที่ 80 เมกะเฮิรตซ์ สามารถแสดง PIN ของ NodeMCU ESP12E ESP8266 ได้ดังรูปที่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.14 แสดง PIN ของ NodeMCU ESP12E ESP8266

จากรูปที่ 3.14 แสดง PIN ของ NodeMCU ESP12E ESP8266 โดยสามารถแจกแจงได้ดังตารางที่ 3.6 ดังต่อไปนี้

ตารางที่ 3.6 แสดงการกำหนดค่า PIN ของบอร์ดพัฒนา NodeMCU ESP12E ESP8266

ประเภทของ PIN	ชื่อ	คำอธิบาย
Power	Micro-USB, 3.3 โวลต์, GND, Vin	Micro-USB: NodeMCU สามารถขับเคลื่อนผ่านพอร์ต USB ได้ 3.3 โวลต์: สามารถจ่ายไฟ 3.3 โวลต์ แบบควบคุมให้กับพินนี้เพื่อจ่ายไฟให้กับบอร์ด GND: หมุดกราวด์ Vin: แหล่งจ่ายไฟภายนอก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ประเภทของ PIN	ชื่อ	คำอธิบาย
Control Pins	EN, RST	พินและปุ่มรีเซ็ตไมโครคอนโทรลเลอร์
Analog Pin	A0	ใช้สำหรับวัดแรงดันอนาล็อกในช่วง 0-3.3 โวลต์
GPIO Pins	GPIO1 to GPIO16	NodeMCU มีพินอินพุต-เอาต์พุต เอนกประสงค์ 16 อันบนบอร์ด
SPI Pins	SD1, CMD, SD0, CLK	NodeMCU มีสี่พินสำหรับการสื่อสาร SPI
UART Pins	TXD0, RXD0, TXD2, RXD2	NodeMCU มีอินเทอร์เฟซ UART สอง อินเทอร์เฟซ UART0 (RXD0 & TXD0) และ UART1 (RXD1 & TXD1) UART1 ใช้เพื่ออัปโหลดเฟิร์มแวร์/โปรแกรม
I2C Pins	-	NodeMCU มีการรองรับฟังก์ชัน I2C แต่เนื่องจากการทำงานภายในของพินเหล่านี้ คุณต้องค้นหาว่าพินใดเป็น I2C

จากตารางที่ 3.6 แสดงการกำหนดค่า PIN ของบอร์ดพัฒนา NodeMCU ESP12E ESP8266 NodeMCU สามารถขับเคลื่อนโดยใช้แจ็ค Micro USB และพิน VIN (พินซัพพลายภายนอก) รองรับอินเทอร์เฟซ UART, SPI และ I2C

3.2.2 เซนเซอร์ MAX30102

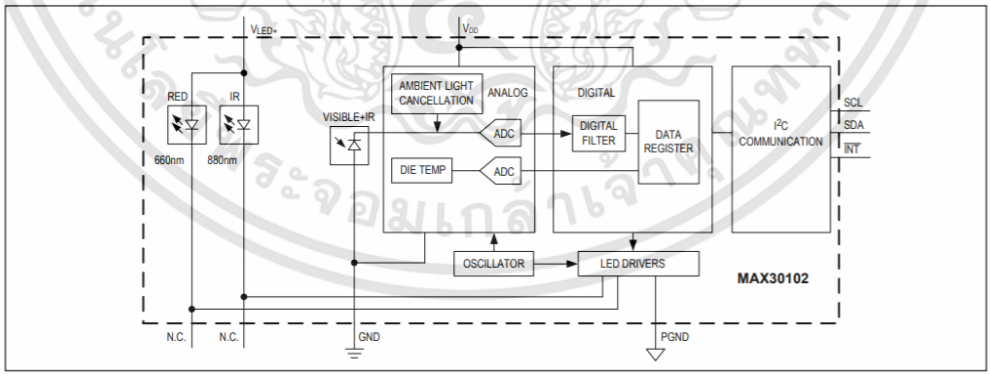
MAX30102 เป็นเซนแซอร์วัดค่าออกซิเจนในเลือดแบบบูรณาการและโมดูลตรวจสอบอัตราการเต้นของหัวใจประกอบด้วยไฟ LED ภายในเครื่อง เพื่อตรวจจับแสงและ องค์ประกอบทางแสง เป็นอุปกรณ์อิเล็กทรอนิกส์ที่มีสัญญาณรบกวนต่ำด้วยการตัดแสงโดยรอบ MAX30102 ให้โซลูชันระบบที่สมบูรณ์เพื่อให้ขั้นตอนการออกแบบง่ายขึ้นสำหรับอุปกรณ์พกพาและอุปกรณ์สวมใส่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MAX30102 ทำงานบนแหล่งจ่ายไฟ 1.8 โวลต์ ตัวเดียวและแหล่งจ่ายไฟ 3.3 โวลต์ แยกต่างหาก สำหรับ LED ภายในการสื่อสารผ่านมาตรฐาน I2C ที่รองรับอินเตอร์เฟซ โมดูลสามารถปิดได้โดยใช้ซอฟต์แวร์ด้วยกระแสไฟสแตนด์บายเป็นศูนย์ ทำให้รางไฟฟ้าสามารถยังคงขับเคลื่อนอยู่ตลอดเวลา โดยลักษณะของเซนเซอร์วัดระดับออกซิเจนในเลือดและอัตราการเต้นของหัวใจ MAX30102 แสดงได้ในรูปที่ 3.15

รูปที่ 3.15 แสดงเซนเซอร์วัดระดับออกซิเจนในเลือดและอัตราการเต้นของหัวใจ MAX30102

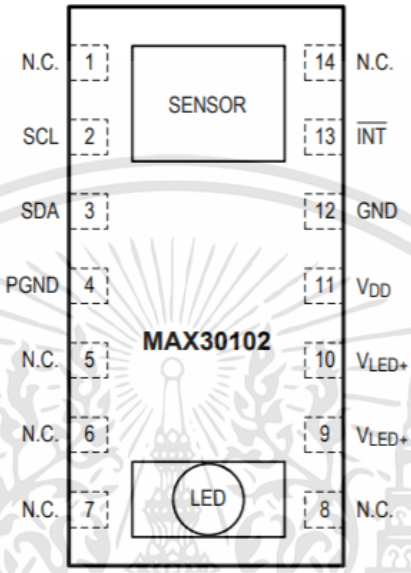
จากรูปที่ 3.15 แสดงเซนเซอร์วัดระดับออกซิเจนในเลือดและอัตราการเต้นของหัวใจ MAX30102 และสามารถแสดงแผนผังระบบได้ดังรูปที่ 3.16



รูปที่ 3.16 แสดงแผนผังระบบของเซนเซอร์วัดระดับออกซิเจนในเลือดและอัตราการเต้นของหัวใจ MAX30102

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากรูปที่ 3.16 แสดงแผนผังระบบของเซนเซอร์วัดระดับออกซิเจนในเลือดและอัตราการเต้นของหัวใจ MAX30102 นอกจากนี้ยังสามารถแสดง PIN ของ MAX30102 ได้ดังรูปที่ 3.17



จากรูปที่ 3.17 แสดง PIN ไดอะแกรมของเซนเซอร์ MAX30102 โดยสามารถแจกแจงรายละเอียดได้ดังตารางที่ 3.7

ตารางที่ 3.7 แสดงคำอธิบาย PIN

PIN	ชื่อ	ฟังก์ชัน
1, 5, 6, 7, 8, 14	N.C.	ไม่มีการเชื่อมต่อแต่เชื่อมต่อกับแผ่น PCB เพื่อความมั่นคงทางกล
2	SCL	อินพุตนาฬิกา I2C
3	SDA	ข้อมูล I2C แบบสองทิศทาง (Open-Drain)
4	PGND	กราวด์ไฟของบล็อกไดรเวอร์LED
9	VLED+	แหล่งจ่ายไฟ LED (การเชื่อมต่อขั้วบวก) ใช้ตัวเก็บประจุบายพาสกับ PGND เพื่อประสิทธิภาพที่ดีที่สุด
10	VLED+	แหล่งจ่ายไฟ LED (การเชื่อมต่อขั้วบวก) ใช้ตัวเก็บประจุบายพาสกับ PGND เพื่อประสิทธิภาพที่ดีที่สุด
11	VDD	อินพุตเพาเวอร์ซัปพลายแบบแอนะล็อก ใช้ตัวเก็บประจุ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

PIN	ชื่อ	ฟังก์ชัน
		บายพาสไปยัง GND เพื่อประสิทธิภาพที่ดีที่สุด
12	GND	แอนะล็อกกราวด์
13	INT	Active-Low Interrupt (เปิด-Drain) เชื่อมต่อกับแรงดันไฟฟ้าภายนอกด้วยตัวต้านทานแบบดึงขึ้น

จากตารางที่ 3.7 แสดงคำอธิบาย PIN โดยมีทั้งหมด 14 PIN OUT มีชื่อ PIN ดังนี้ N.C., SCL, SDA, PGND, VLED+, VLED+, VDD, GND และ INT

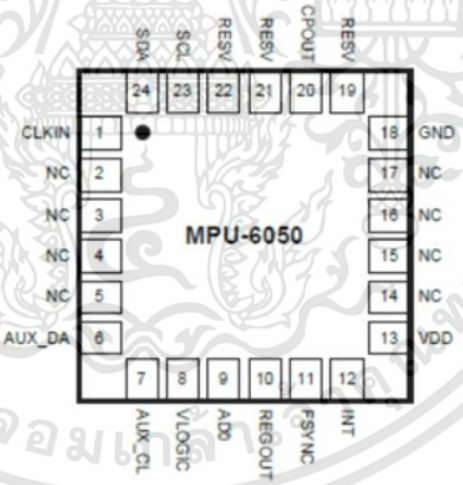
3.2.2 โมดูล GY-521 MPU6050

โมดูล GY-521 MPU6050 Accelerometer/Gyro Module เป็นโมดูลวัดความเร่ง วัดได้ 3 แกน x, y และ z เมื่อเราเอนโมดูล ค่าของอัตราเร่งเชิงมุม (วัดได้โดย accelerometer) และ อัตราเร็วเชิงมุม (วัดได้โดย gyroscope) จะเปลี่ยนแปลง โมดูลนี้มีอินเตอร์เฟซแบบ I2C ใช้สายสัญญาณ 2 เส้น ใช้ไฟ 3.3 โวลต์ Accelerometer ใช้สำหรับวัดความเร่ง หรือความเอียง เมื่อเราจับเซนเซอร์วางขนานกับพื้นโลก ค่าที่ได้จาก Accelerometer ในแกน z จะประมาณ 10 เนื่องจากถูกแรงโน้มถ่วงของโลกดึงไว้ แต่เมื่อเราเอียงเซนเซอร์ไปในทิศทางต่าง ๆ ค่าในแต่ละแกนจะเปลี่ยนไป เช่น เอียงเซนเซอร์ไปทางซ้ายที่ 90 องศา และหยุดนิ่ง ค่าที่ได้แกน z จะเป็น 0 แต่แกน x หรือ y จะเป็น 10 เนื่องจากแรงโน้มถ่วงดึงให้ค่าในแกน x หรือ y ลงด้านล่าง Gyroscope ใช้วัดความเร่งในการเคลื่อนที่ ความสามารถของ Accelerometer จะให้ค่าที่คงที่อยู่เสมอ แตกต่างจาก Gyroscope ที่จะให้ค่าเมื่อมีการเคลื่อนที่เท่านั้น หน่วยวัดเป็น องศาเรเดียนต่อวินาที (rad/s) การใช้งานวัดความเอียง การหมุนต่าง ๆ นิยมใช้งานทั้ง Accelerometer และ Gyroscope คู่กัน เนื่องจาก Accelerometer ไม่สามารถวัดการหมุนในทิศทางที่เซนเซอร์วางขนานกับพื้นโลกได้ (เมื่อหมุนเซนเซอร์ที่วางขนานกับพื้นโลก ค่า x y และ z จะไม่เปลี่ยนแปลง) จึงจำเป็นต้องใช้ Gyroscope มาใช้วัดร่วมด้วยเซนเซอร์นี้ สามารถนำไปประยุกต์ใช้ร่วมกับโดรนเพื่อควบคุมความสมดุล หรือนำไปใช้กับบาลานซ์โรบอท เพื่อให้หุ่นยนต์สามารถตั้งตรงได้อยู่เสมอ นอกจากนี้ยังนำไปวัดความลาดเอียดเพื่อประยุกต์ใช้อื่น ๆ โดยลักษณะของโมดูลวัดความเร่ง3แกน GY-521 MPU6050 แสดงได้ดังรูปที่ 3.18

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 3.18 แสดงโมดูล GY-521 MPU6050

จากรูปที่ 3.18 แสดงโมดูล GY-521 MPU6050 แสดงโมดูลวัดความเร่ง 3 แกน GY-521 MPU6050 ซึ่งประกอบด้วยขา ทั้งหมด 8 ขาคือ VCC, GND, SCL, SDA, XDA, XCL, ADD และ INT และสามารถแสดง PIN OUT ไดอะแกรมได้ดังรูปที่ 3.19



รูปที่ 3.19 PIN OUT ไดอะแกรม

จากรูปที่ 3.19 PIN OUT ไดอะแกรมของ GY-521 MPU6050 โดนสามารถแสดงได้ดังตารางที่ 3.8

ตารางที่ 3.8 แสดงคำอธิบาย PIN OUT

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

PIN	ชื่อ	คำอธิบาย
1	CLKIN	อินพุตนาฬิกาอ้างอิงภายนอกเสริม เชื่อมต่อกับ GND หากไม่ได้ใช้
6	AUX_DA	ข้อมูลอนุกรมหลัก I2C สำหรับเชื่อมต่อกับเซนเซอร์ภายนอก
7	AUX_CL	ข้อมูลอนุกรมหลัก I2C สำหรับเชื่อมต่อกับเซนเซอร์ภายนอก
8	/CS	เลือกชิป SPI (0=โหมด SPI)
8	VLOGIC	แรงดันไฟจ่ายดิจิตอล I/O
9	AD0 / SDO	ที่อยู่ I2C LSB (AD0); เอาต์พุตข้อมูลอนุกรม SPI (SDO)
9	AD0	I2C ที่อยู่ LSB (AD0)
10	REGOUT	การเชื่อมต่อตัวเก็บประจุตัวกรองเรกูเลเตอร์
11	FSYNC	อินพุตดิจิตอลซิงโครไนซ์เฟรม เชื่อมต่อกับ GND หากไม่ได้ใช้
12	INT	อินเตอร์รัปต์เอาต์พุตดิจิตอล
13	VDD	แรงดันไฟฟ้าของแหล่งจ่ายไฟและแรงดันไฟฟ้าของแหล่งจ่ายไฟดิจิตอล I/O
18	GND	กราวด์ของแหล่งจ่ายไฟ
19,21	RESV	ที่สงวนไว้ อย่าเชื่อมต่อ
20	CPOUT	การเชื่อมต่อตัวเก็บประจุปั๊มชาร์จ
22	CLKOUT	สัญญาณนาฬิกาของระบบ
23	SCL / SCLK	นาฬิกาอนุกรม (SCLK)
23	SCL	นาฬิกาอนุกรม (SCL)
24	SDA / SDI	ข้อมูลอนุกรม 2C (SDA) อินพุตข้อมูลอนุกรม SPI (SDI)
24	SDA	ข้อมูลอนุกรม (SDA)
2, 3, 4, 5, 14, 15, 16, 17	NC	ไม่ได้เชื่อมต่อภายใน อาจใช้สำหรับการกำหนดเส้นทางการติดตาม PCB

จากตารางที่ 3.8 แสดง PIN OUT ไดอะแกรม โดยมีทั้งหมด 24 PIN OUT

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

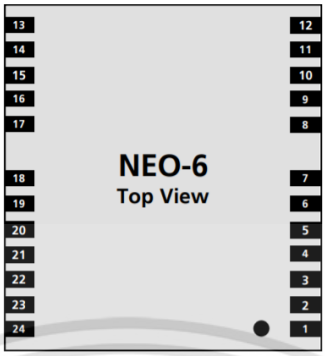
3.2.3 โมดูล GPS GY-NEO6MV2

ชุดโมดูล NEO-6 เป็นตระกูลเครื่องรับ GPS แบบสแตนด์อโลนที่มี u-blox 6 ประสิทธิภาพสูง เครื่องรับ GPS ราคาประหยัดแต่ทรงพลังนี้ใช้โมดูล GPS u-Blox Neo-6M ที่มีชื่อเสียงและคุณภาพสูง มันมาพร้อมกับแบตเตอรี่ขนาดเล็กสำหรับการสตาร์ทแบบร้อน และมี EEPROM ในตัวด้วย เพื่อให้รับสัญญาณได้ดีขึ้น มีเสาอากาศเซรามิกภายนอกที่เชื่อมต่อกับบอร์ดผ่านขั้วต่อ U.FL แบบทึบ สามารถทำงานได้ตั้งแต่ระบบ 3.3V ถึง 5V ดังนั้นบอร์ด Arduino 5V ทั้งหมด (Maker uno, arduino uno , Mega, Leonardo และ อื่นๆ อีกมากมาย) และตัวควบคุม 3.3V ที่มี Arduino และ Raspberry Pi ทำงานได้อย่างสมบูรณ์แบบกับโมดูล GPS นี้ แม้ว่ามันจะเข้ากันได้กับ 5V คุณจะต้องใช้ตัวเปลี่ยนระดับที่เหมาะสมเพื่อเชื่อมต่อกับสายซีเรียล (UART) โปรดใช้ตัวเปลี่ยนระดับที่เหมาะสมเพื่อทำ และโปรดทราบว่าไม่ใช่ Arduino shield หรือ Raspberry Pi HAT form factor คุณจะต้องเดินสายด้วยตนเองเพื่อเชื่อมต่อกับบอร์ดควบคุม โดยลักษณะของโมดูล GPS GY-NEO6MV2 แสดงได้ดังรูปที่ 3.20

รูปที่ 3.20 แสดงโมดูล GPS GY-NEO6MV2

จากรูปที่ 3.20 แสดงโมดูล GPS GY-NEO6MV2 ประกอบด้วยขา PIN OUT 4 ขา คือ GND, TX, RX และ VCC โดยแสดง PIN OUT ไดอะแกรมของ GPS GY-NEO6MV2 ได้ดังรูปที่ 3.21

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.21 PIN OUT ไดอะแกรมของ GPS GY-NEO6MV2

จากรูปที่ 3.21 แสดง PIN OUT ไดอะแกรมของ GPS GY-NEO6MV2 และสามารถอธิบายได้ดังตารางที่ 3.9

ตารางที่ 3.9 แสดงคำอธิบาย PIN OUT

PIN	ชื่อ	I/O	คำอธิบาย
1	Reserved	I	ที่สงวนไว้
2	SS_N	I	SPI Select
3	TIMEPULSE	O	ไทม์พัลส์ (1PPS)
4	EXTINT0	I	พินขัดจังหวะภายนอก
5	USB_DM	I/O	USB Data
6	USB_DP	I/O	USB Data
7	VDDUSB	I	อุปกรณ์จ่าย USB
8	Reserved	O	ที่สงวนไว้
9	VCC_RF	I	แรงดันไฟขาออก ส่วน RF ขา 8 และ 9 ต้องเชื่อมต่อเข้าด้วยกัน
10	GND	I	กราวนด์
11	RF_IN	I	อินพุตสัญญาณ GPS
12	GND	I	กราวนด์
13	GND	O/I	กราวนด์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

PIN	ชื่อ	I/O	คำอธิบาย
14	MOSI/CFG_COM0	I	SPI MOSI
15	MISO/CFG_COM1	I	SPI MISO / การกำหนดค่า Pin เปิดทิ้งไว้หากไม่ได้ใช้
16	CFG_GPS0/SCK	I	พินการกำหนดค่าโหมดพลังงาน / นาฬิกา SPI เปิดทิ้งไว้ถ้าไม่ได้ใช้
17	Reserved	I	ที่สงวนไว้
18	SDA2	I/O	ข้อมูล DDC
19	SCL2	I/O	สัญญาณนาฬิกา DDC
20	TxD1	O	พอร์ตอนุกรม 1
21	RxD1	I	พอร์ตอนุกรม 1
22	V_BCKP	I	แหล่งจ่ายไฟสำรอง
23	VCC	I	แรงดันไฟจ่าย
24	GND	I	กราวนด์

จากตารางที่ 3.9 แสดงคำอธิบาย PIN OUT ทั้งหมด 24 PIN OUT ของ GPS GY-NEO6MV2

3.3 การจัดเก็บผลการทดลอง

การจัดเก็บผลการทดลองการออกแบบชุดอุปกรณ์ทดสอบการส่งข้อมูลผ่าน Wi-Fi แบ่งเป็น 4 ส่วนดังนี้

3.3.1 ทดสอบการส่งข้อมูลอัตราการเต้นหัวใจของเซนเซอร์ MAX-30102 ไปยังฐานข้อมูลผ่าน NodeMCU ESP12E ESP8266

เป็นการทดสอบการส่งข้อมูลอัตราการเต้นหัวใจจากเซนเซอร์ MAX-30102 ที่เชื่อมต่อกับ NodeMCU ESP12E ESP8266 เพื่อทำการส่งข้อมูลที่ได้ไปเก็บยังฐานข้อมูลเพื่อการวิเคราะห์สถานะการเต้นของหัวใจ

3.3.2 ทดสอบการส่งข้อมูลตำแหน่งที่อยู่ของโมดูล GPS GY-NEO-6MV2 ไปยังฐานข้อมูลผ่าน NodeMCU ESP12E ESP8266

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เป็นการทดลองการส่งข้อมูลตำแหน่งที่อยู่จากโมดูล GPS GY-NEO-6MV2 ที่เชื่อมต่อกับ NodeMCU ESP12E ESP8266 เพื่อทำการส่งข้อมูลที่ได้ไปเก็บยังฐานข้อมูลเพื่อนำไปแสดงผลบน Mobile Application

3.3.3 ทดสอบการส่งแจ้งเตือนการหกล้มจากโมดูลวัดความเร่ง 3 แกน GY-521 MPU6050

เป็นการทดลองการส่งการแจ้งเตือนการหกล้มจากโมดูลวัดความเร่ง 3 แกน GY-521 MPU6050 ที่เชื่อมต่อกับ NodeMCU ESP12E ESP8266 เพื่อทำการส่งข้อมูลทำการแจ้งเตือนว่าเกิดการหกล้มไปยังบัญชีผู้ใช้ Application Line ตามที่ตั้งค่าไว้

3.3.4 ทดสอบการส่งข้อมุลจากเซนเซอร์ทุกตัวไปยังฐานข้อมูลผ่าน NodeMCU ESP12E ESP8266

เป็นการทดสอบการส่งข้อมูลอัตราการเต้นหัวใจ, ข้อมูลตำแหน่งที่อยู่ และ การแจ้งเตือนการหกล้มจากเซนเซอร์ MAX-30102 , GPS GY-NEO-6MV2 และ GY-521 MPU6050 ที่เชื่อมต่อกับ NodeMCU ESP12E ESP8266 เพื่อทำการส่งข้อมูลที่ได้ไปเก็บยังฐานข้อมูลเพื่อนำไปแสดงผลบน Mobile Application

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4

ผลการทดลอง

ผู้จัดทำได้ทำการเก็บผลการทำงานของระบบโดยแบ่งการทดลองและจัดเก็บผลการทดลองเป็นส่วน ๆ ดังต่อไปนี้

4.1 ทดสอบการส่งข้อมูลอัตราการเต้นหัวใจของเซนเซอร์ MAX-30102 ผ่าน NodeMCU ESP12E ESP8266

ชุดอุปกรณ์การส่งข้อมูลอัตราการเต้นของหัวใจที่เซนเซอร์ MAX-30102 ที่เชื่อมต่อกับ NodeMCU ESP12E ESP8266 เพื่อทำการประมวลผลข้อมูลและทำการส่งข้อมูลไปยังฐานข้อมูลที่ได้เตรียมไว้สำหรับการจัดเก็บข้อมูลที่จะใช้ในการแสดงผลบน Mobile Application โดยการทดลองจะเป็นการอ่านข้อมูลจากหน้าจอ Serial Monitor ที่แสดงผลค่าอัตราการเต้นของหัวใจผู้ที่นำนิ้วมือวางไว้บนเซนเซอร์ MAX-30102 โดยค่าที่ได้จะเป็นค่าที่ผ่านการประมวลผลผ่าน NodeMCU ESP12E ESP8266 เป็นตัวเลขสำหรับการการแสดงผลในค่าต่าง ๆ เช่น อัตราการเต้นหัวใจต่อนาทีและอัตราการเต้นหัวใจเฉลี่ยดังแสดงในรูปที่ 4.1 และ 4.2

```
14:28:44.501 -> IR=915, BPM=0.00, Avg BPM=0 No finger?
14:28:44.501 -> IR=911, BPM=0.00, Avg BPM=0 No finger?
14:28:44.548 -> IR=896, BPM=0.00, Avg BPM=0 No finger?
14:28:44.548 -> IR=887, BPM=0.00, Avg BPM=0 No finger?
14:28:44.596 -> IR=905, BPM=0.00, Avg BPM=0 No finger?
14:28:44.596 -> IR=893, BPM=0.00, Avg BPM=0 No finger?
14:28:44.596 -> IR=915, BPM=0.00, Avg BPM=0 No finger?
14:28:44.644 -> IR=900, BPM=0.00, Avg BPM=0 No finger?
14:28:44.644 -> IR=886, BPM=0.00, Avg BPM=0 No finger?
14:28:44.691 -> IR=903, BPM=0.00, Avg BPM=0 No finger?
14:28:44.691 -> IR=895, BPM=0.00, Avg BPM=0 No finger?
14:28:44.739 -> IR=917, BPM=0.00, Avg BPM=0 No finger?
14:28:44.739 -> IR=905, BPM=0.00, Avg BPM=0 No finger?
14:28:44.739 -> IR=885, BPM=0.00, Avg BPM=0 No finger?
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 4.1 ค่าที่แสดงผลจากเซนเซอร์ MAX-30102 ตอนที่ไม่มีนิ้ววางบนเซนเซอร์

```
14:30:03.600 -> IR=101333, BPM=83.33, Avg BPM=87
14:30:03.600 -> IR=101327, BPM=83.33, Avg BPM=87
14:30:03.600 -> IR=101332, BPM=83.33, Avg BPM=87
14:30:03.647 -> IR=101309, BPM=83.33, Avg BPM=87
14:30:03.647 -> IR=101328, BPM=83.33, Avg BPM=87
14:30:03.695 -> IR=101347, BPM=83.33, Avg BPM=87
14:30:03.695 -> IR=101322, BPM=83.33, Avg BPM=87
14:30:03.743 -> IR=101310, BPM=83.33, Avg BPM=87
14:30:03.743 -> IR=101252, BPM=83.33, Avg BPM=87
14:30:03.743 -> IR=101181, BPM=83.33, Avg BPM=87
14:30:03.791 -> IR=101137, BPM=83.33, Avg BPM=87
14:30:03.791 -> IR=101128, BPM=83.33, Avg BPM=87
14:30:03.839 -> IR=101161, BPM=83.33, Avg BPM=87
14:30:03.839 -> IR=101195, BPM=83.33, Avg BPM=87
14:30:03.839 -> IR=101212, BPM=83.33, Avg BPM=87
```

a

รูปที่ 4.2 ค่าที่แสดงผลจากเซนเซอร์ MAX-30102 ตอนมีนิ้วบนเซนเซอร์

จากรูปที่ 4.1 จะเป็นการแสดงผลจาก Serial Monitor ที่แสดงผลค่า IR คือ ค่าตัวแปรแทน Check for Heartbeat หรือเช็คการเต้นของหัวใจหากเซนเซอร์ไม่สามารถตรวจจับการเต้นของหัวใจได้ค่าสถานะที่ปรากฏจะเป็นตัวเลขหลักร้อยตามรูปภาพ BPM คือค่าอัตราการเต้นหัวใจต่อนาที Avg BPM คือ ค่าเฉลี่ยอัตราการเต้นของหัวใจ No finger? คือ สถานะแจ้งเตือน จะปรากฏขึ้นมาเมื่อเซนเซอร์ไม่พบนิ้วมือที่วางอยู่บนเซนเซอร์ โดยในรูปถัดมาจะเป็นรูปที่ 4.2 เมื่อทำการวางนิ้วมือลงบนเซนเซอร์แล้วจะเห็นได้ว่าการแจ้งเตือน No finger? หายไปและตัวแปร IR เปลี่ยนจากหลักร้อยเป็นหลักแสนตามรูปแต่ในส่วนค่าของ BPM และ Avg BPM จะเปลี่ยนไปเป็นค่าที่ประมวลผลที่ได้จากเซนเซอร์ MAX-30102 โดยการประมวลผลจะใช้สูตรการคำนวณเป็นการอ่านค่า rate แล้วนำไปบันทึกใน array แล้วทำการหาค่าเฉลี่ยออกมาจากข้อมูลที่เป็นผลลัพธ์ แล้ว NodeMCU ESP12E ESP8266 จะนำไปแสดงผลบน Serial Monitor

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2 การทดลองการส่งการแจ้งเตือนการหกล้มผ่านโมดูลวัดความเร่ง 3 แกน GY-521 MPU6050

การส่งการแจ้งเตือนจากชุดอุปกรณ์โมดูลวัดความเร่ง 3 แกน GY-521 MPU6080 โดยโมดูลวัดความเร่ง 3 แกนจะทำการอ่านข้อมูลออกมาทั้ง 3 แกนและมุมเพื่อนำไปประมวลผลออกมาโดยการประมวลผลจะใช้สูตรการคำนวณคือการนำข้อมูลที่ได้จากเซนเซอร์โดยจะมีข้อมูลคือ AcX ,AcY ,AcZ ,GyX ,GyY.และ Gy Z การคำนวณจะเป็นการนำค่า AcX ,AcY ,AcZ มาหาค่าความเร่งและนำค่า GyX ,GyY.และ Gy Z มาหาค่ามุมที่เปลี่ยนแปลงไปหลังจากนั้นจะทำการส่งการแจ้งเตือนเมื่อค่าความเร่งและมุมที่วัดได้เกินเงื่อนไขที่กำหนดโดยจะมี 3 เงื่อนไขคือความเร่งน้อยกว่า 2 เป็นเงื่อนไขที่ 1 ต่อมาเมื่อค่าความเร่งเกิน 12 จะเป็นเงื่อนไขที่ 2 ต่อมาเมื่อค่ามุมเปลี่ยนไปเกิน 30 องศาจะเป็นเงื่อนไขที่ 3 เมื่อผ่านครบ3เงื่อนไขแล้วจะทำการส่งการแจ้งเตือนไปยังเว็บไซต์ IFTTT และทำการส่งการแจ้งเตือนไปยัง Application Line ดังรูปที่ 4.3 ,4.4 ,4.5 และ 4.6

รูปที่ 4.3 การตั้งค่าการทำงานเว็บไซต์ IFTTT

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
14:35:59.727 -> 9
14:35:59.870 -> 9
14:35:59.966 -> 9
14:36:00.062 -> 9
14:36:00.158 -> 9
14:36:00.253 -> 9
14:36:00.349 -> 9
14:36:00.445 -> 9
14:36:00.589 -> 9
14:36:00.685 -> 9
14:36:00.781 -> 9
14:36:00.883 -> 9
14:36:00.977 -> 9
14:36:01.069 -> 9
14:36:01.164 -> 9
```

รูปที่ 4.4 ผลที่แสดงบน Serial Monitor เมื่อชุดอุปกรณ์หยุดนิ่งบนโต๊ะ

```
14:36:45.547 -> 11
14:36:45.690 -> 16
14:36:45.785 -> 17
14:36:45.881 -> 12
14:36:45.976 -> 7
14:36:46.071 -> 8
14:36:46.167 -> 13
14:36:46.263 -> 18
14:36:46.407 -> 15
14:36:46.503 -> 11
14:36:46.599 -> 7
14:36:46.693 -> 8
14:36:46.788 -> 11
14:36:46.882 -> 14
14:36:46.978 -> 16
```

รูปที่ 4.5 ผลที่แสดงบน Serial Monitor เมื่อแกว่งชุดอุปกรณ์ไปมา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
15:44:33.749 -> 9
15:44:33.845 -> 9
15:44:33.941 -> 8
15:44:34.037 -> 7
15:44:34.181 -> 10
15:44:34.277 -> 11
15:44:34.373 -> 22
15:44:34.469 -> 9
15:44:34.565 -> 9
15:44:34.661 -> 8
15:44:34.756 -> 7
15:44:34.899 -> 2
15:44:34.899 -> TRIGGER 1 ACTIVATED
15:44:34.993 -> 18
15:44:34.993 -> TRIGGER 2 ACTIVATED
15:44:34.993 -> 279
15:44:34.993 -> 279
15:44:34.993 -> TRIGGER 3 ACTIVATED
15:44:35.089 -> 9
15:44:35.184 -> 10
15:44:35.280 -> 7
15:44:35.376 -> 9
15:44:35.471 -> 9
15:44:35.615 -> 9
15:44:35.709 -> 9
15:44:35.805 -> 9
15:44:35.901 -> 9
15:44:35.901 -> Connecting to maker.ifttt.com
15:44:35.997 -> Requesting URL: /trigger/fall_detect/with/key/k-q5QiPz839C_KFJWgbTSxx6ndQo953XlKsVtsjTMsE
15:44:37.237 ->
15:44:37.237 -> closing connection
```

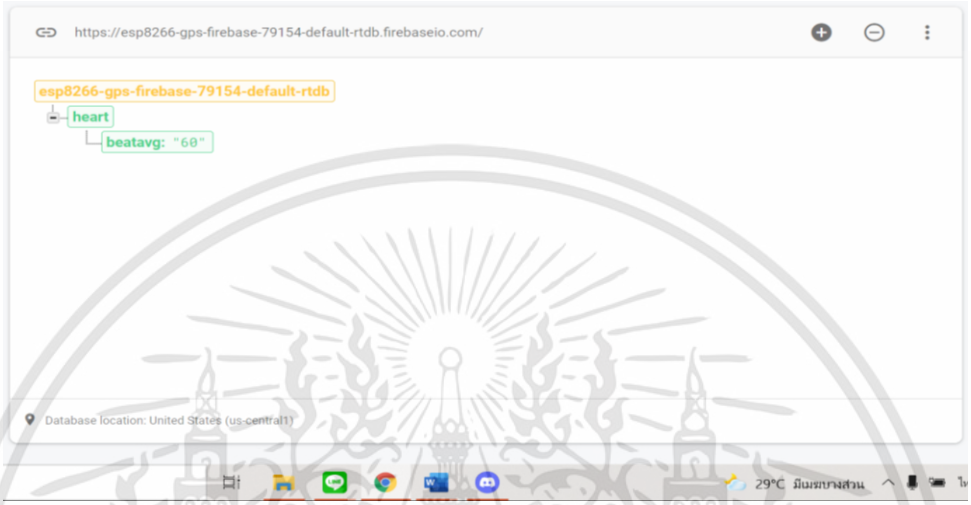
รูปที่ 4.6 ผลที่แสดงบน Serial Monitor เมื่อเกิดการหกล้ม

จากรูปที่ 4.3 เป็นรูปการณ์ตั้งค่าบนหน้าเว็บไซต์ให้ทำการส่งการแจ้งเตือน Line ไปยังแอคเคาท์ที่ได้ทำการตั้งค่าไว้เมื่อมีสัญญาณจากชุดการทดลองโมดูลวัดความเร่ง 3 แกน GY-521 MPU6050 ที่เชื่อมต่อกับ NodeMCU ESP12E ESP8266 โดยสัญญาณจะส่งออกมาเมื่อค่าความเร่งที่คำนวณได้เกิดการเปลี่ยนแปลงกะทันหันโดยในรูปที่ 4.4 จะเป็นการแสดงผลบน Serial Monitor เมื่อวางชุดอุปกรณ์ไว้บนโต๊ะโดยจะแสดงค่าความเร่งที่ได้เป็นค่า 9 คงที่ต่อมาภาพที่ 4.5 จะเป็นค่าที่แสดงผลบน Serial Monitor เมื่อทำการแกว่งชุดอุปกรณ์โดยค่าที่ได้จะมีค่าไม่คงที่ตั้งแต่ 7-18 ตามความเร่งที่เกิดขึ้นต่อมาในรูปที่ 4.6เป็นการแสดงผลบน Serial Monitor เมื่อเกิดการล้มโดยชุดการทดลองจะมีเงื่อนไขตามที่กล่าวไว้ข้างบนจากรูปจะเห็นได้ว่าเมื่อผ่านเงื่อนไขทั้ง 3 คือ 1).เกิดการเปลี่ยนแปลงของ Amp น้อยกว่า 2 เงื่อนไข 1 จะมีค่าเป็นจริง 2)เมื่อเงื่อนไข 1 มีค่าเป็นจริงแล้วค่า Amp เพิ่มขึ้นมากกว่า 12 เงื่อนไข 2 จะแสดงค่าเป็นจริง 3) เมื่อเงื่อนไข 2 มีค่าเป็นจริงแล้วให้คำนวณค่ามุมที่เปลี่ยนไปเมื่อมุมที่เปลี่ยนไปอยู่ระหว่าง 100-300 แล้วเงื่อนไข 3 จะมีค่าเป็นจริงแล้วเมื่อเงื่อนไข 3 เป็นจริงแล้วมุมไม่เปลี่ยนแปลงไปมีค่า 0-100.แล้วจะทำการส่งการแจ้งเตือนไปยังเว็บไซต์ IFTTT เป็นทั้งหมดของชุดการทดลอง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.3 การทดลองการส่งข้อมูลจากเซนเซอร์ไปยังฐานข้อมูล

การทดลองนี้จะเป็นการนำข้อมูลจากเซนเซอร์ส่งไปยังฐานข้อมูลตามรูปที่ 4.7



รูปที่ 4.7 ข้อมูลเซนเซอร์ที่เก็บไว้ที่ฐานข้อมูล

การทดลองนี้เป็นการนำข้อมูลจากเซนเซอร์ MAX30102 ส่งไปยังฐานข้อมูลเพื่อเตรียมไว้สำหรับนำไปแสดงผลบน Mobile Application โดยฐานข้อมูลที่ใช้จะเป็น Firebase Realtime Database เป็น NoSQL cloud database ที่เก็บข้อมูลในรูปแบบของ JSON ดังนั้นเมื่อจะทำการเก็บข้อมูลจะต้องเปลี่ยนข้อมูลเป็นรูปแบบ JSON ก่อนโดยมีข้อมูลจากเซนเซอร์คือ MAX30102 โดยในฐานข้อมูล Heart rate จะเป็นได้เมื่อ NodeMCU ESP12E ESP8266 ประมวลผลข้อมูลที่ได้ จะเป็นข้อมูลที่ได้จากเซนเซอร์ MAX-30102 โดยจะเป็นข้อมูลอัตราการเต้นของหัวใจเฉลี่ยที่ได้ประมวลผลมาแล้วโดยข้อมูลทั้งหมดจะส่งพร้อมกันทุก ๆ 5 วินาทีตามที่ได้ตั้งค่าไว้

4.4 การทดลองเปรียบเทียบบริเวณที่เหมาะสมในการติดตั้งเซนเซอร์ MAX30102

ขั้นตอนการดำเนินการออกแบบการสร้างถุงมือพร้อมด้วยชุดอุปกรณ์ที่สามารถวัดอัตราการเต้นของหัวใจ, แจ้งเตือนการหกล้ม, และระบุตำแหน่งของผู้ป่วยโรคเรื้อรัง ในที่นี้เซนเซอร์วัดอัตราการเต้นของหัวใจ MAX30102 คณะผู้จัดทำได้ทำการออกทดลองวัด และอ่านค่าอัตราการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

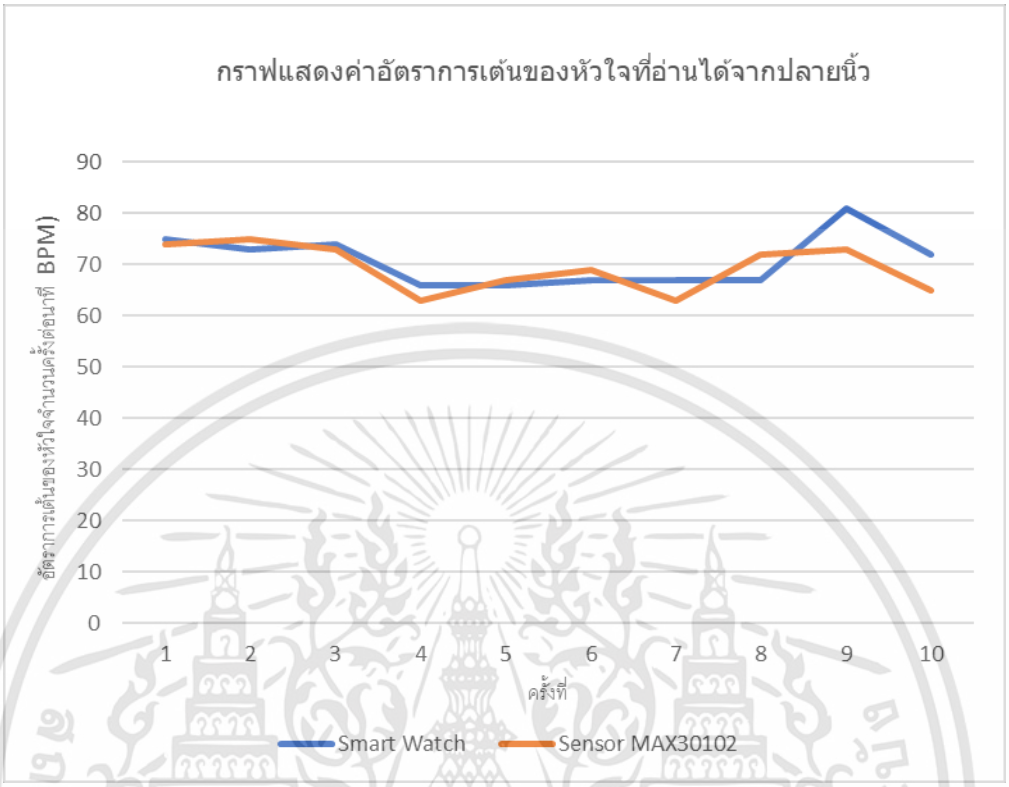
เต้นของหัวใจที่ได้จากเซนเซอร์ MAX30102 ในตำแหน่งต่างๆของมือดังนี้คือ ปลายนิ้ว, ข้อมือภายใน และข้อมือภายนอก เปรียบเทียบกับอุปกรณ์มาตราฐานในที่นี้คือ Smart Watch ที่ใช้หลักการวัดอัตราการเต้นของหัวใจเดียวกันกับเซนเซอร์ MAX30102 ใช้วิธีการที่เรียกว่า photoplethysmography เพื่อวัดอัตราการเต้นของหัวใจของบุคคล โดยวิธีนี้ส่องแสงบนผิวหนังและวัดการไหลเวียนของเลือด โดยผลการทดลองวัดทั้งหมด 10 ครั้ง สามารถแสดงได้ดังตารางที่ 2 ตารางที่ 2 แสดงค่าอัตราการเต้นของหัวใจที่ได้อ่านจากเซนเซอร์ MAX30102 ในตำแหน่งปลายนิ้ว, ข้อมือภายใน และข้อมือภายนอก เปรียบเทียบกับอุปกรณ์ Smart Watch

ตารางที่ 4.1 ตารางการเปรียบเทียบอัตราการเต้นของหัวใจที่วัดได้จากอุปกรณ์และ smart watch

ครั้งที่	Smart Watch			เซนเซอร์ MAX30102		
	ปลายนิ้ว	ข้อมือด้านใน	ข้อมือด้านนอก	ปลายนิ้ว	ข้อมือด้านใน	ข้อมือด้านนอก
1	75	82	86	74	87	65
2	73	86	86	75	91	56
3	74	85	90	73	88	71
4	66	85	86	63	81	70
5	66	84	87	67	84	56
6	67	84	84	69	82	48
7	67	91	82	63	89	74
8	67	84	90	72	84	61
9	81	84	91	73	83	80
10	72	67	88	65	65	70

จากตารางที่ 4.1 แสดงค่าอัตราการเต้นของหัวใจที่ได้อ่านจากเซนเซอร์ MAX30102 ในตำแหน่งปลายนิ้ว, ข้อมือภายใน และข้อมือภายนอก เปรียบเทียบกับอุปกรณ์ Smart Watch เมื่อทำการเปรียบเทียบเริ่มจากตำแหน่งปลายนิ้วจาก Smart Watch และ เซนเซอร์ MAX30102 สามารถแสดงได้ดังรูปที่ 4.8

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

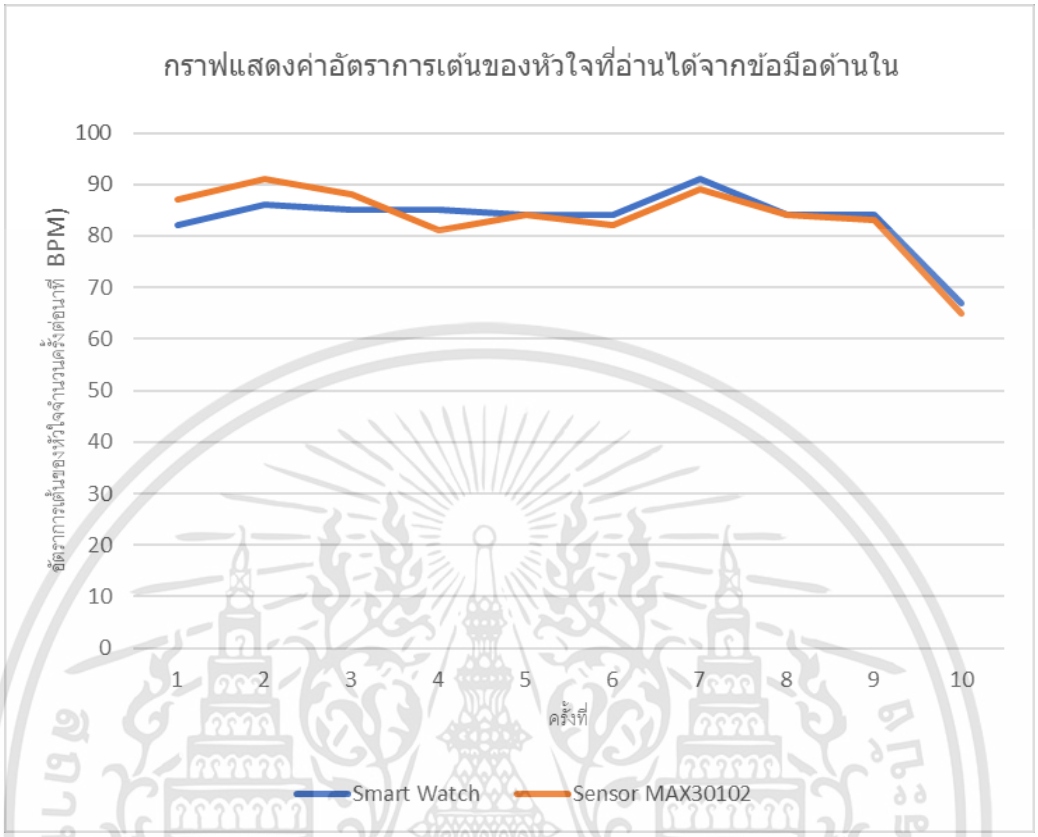


รูปที่ 4.8 กราฟแสดงค่าอัตราการเต้นของหัวใจที่อ่านได้จากปลายนิ้ว

จากรูปที่ 4.8 แสดงกราฟแสดงค่าอัตราการเต้นของหัวใจที่อ่านได้จากปลายนิ้ว เมื่อทำการคำนวณหาค่า Relative error จะได้ 0.047501693 นั่นคือมีค่า error คือ 4.750169265 จากนั้นคำนวณหาค่า Accuracy (ความแม่นยำ) มีค่า 95.24983073

จากตารางที่ 4.1 แสดงค่าอัตราการเต้นของหัวใจที่ได้อ่านจากเซนเซอร์ MAX30102 ในตำแหน่งปลายนิ้ว, ข้อมือภายใน และข้อมือภายนอก เปรียบเทียบกับอุปกรณ์ Smart Watch เมื่อทำการเปรียบเทียบตำแหน่งข้อมือด้านในจาก Smart Watch และ เซนเซอร์ MAX30102 สามารถแสดงได้ดังรูปที่ 4.9

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

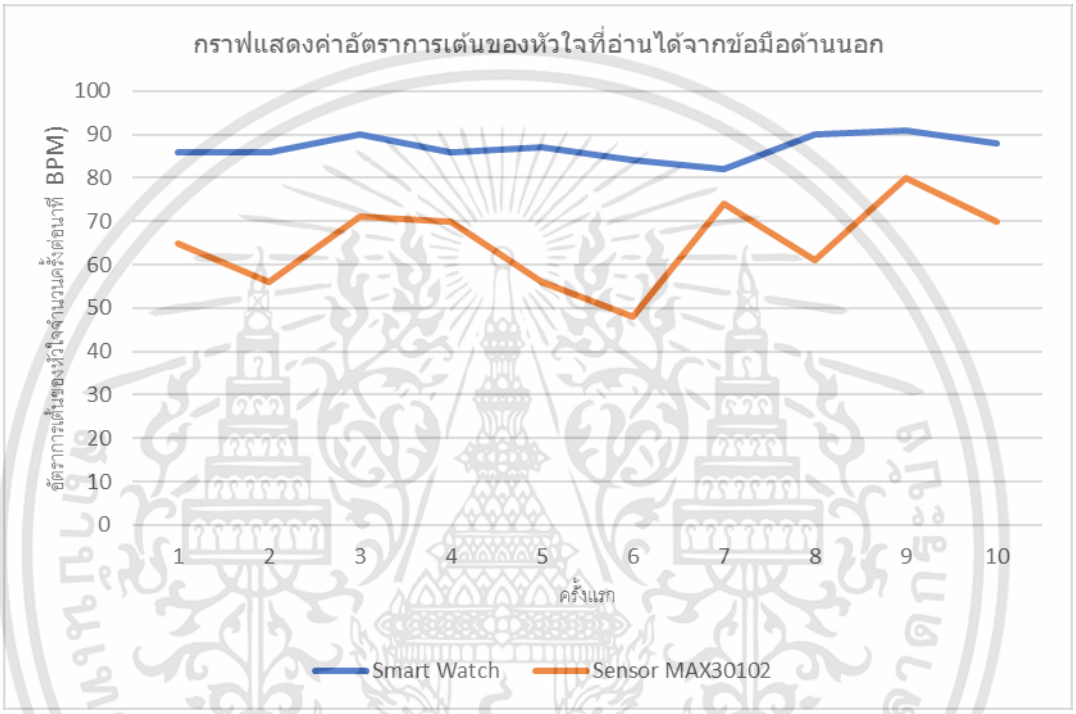


รูปที่ 4.9 กราฟแสดงค่าอัตราการเต้นของหัวใจที่อ่านได้จากข้อมือด้านใน

จากรูปที่ 4.9 แสดงกราฟแสดงค่าอัตราการเต้นของหัวใจที่อ่านได้จากข้อมือด้านใน เมื่อทำการคำนวณหาค่า Relative error จะได้ 0.028901114 นั่นคือมีค่า error คือ 2.890111398 จากนั้นคำนวณหาค่า Accuracy (ความแม่นยำ) มีค่า 97.1098886

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากตารางที่ 4.1 แสดงค่าอัตราการเต้นของหัวใจที่ได้อ่านจากเซนเซอร์ MAX30102 ในตำแหน่งปลายนิ้ว, ข้อมือภายใน และข้อมือภายนอก เปรียบเทียบกับอุปกรณ์ Smart Watch เมื่อทำการเปรียบเทียบตำแหน่งข้อมือด้านนอกจาก Smart Watch และ เซนเซอร์ MAX30102 สามารถแสดงได้ดังรูปที่ 4.10



รูปที่ 4.10 กราฟแสดงค่าอัตราการเต้นของหัวใจที่อ่านได้จากข้อมือด้านนอก

จากรูปที่ 4.10 แสดงกราฟแสดงค่าอัตราการเต้นของหัวใจที่อ่านได้จากข้อมือด้านนอกเมื่อทำการคำนวณหาค่า Relative error จะได้ 0.252028192 นั่นคือมีค่า error คือ 25.20281919 จากนั้นคำนวณหาค่า Accuracy (ความแม่นยำ) มีค่า 74.79718081

คณะผู้จัดทำได้ทำการออกทดลองวัด และอ่านค่าอัตราการเต้นของหัวใจที่ได้จากเซนเซอร์ MAX30102 ในตำแหน่งต่างๆของมือดังนี้คือ ปลายนิ้ว, ข้อมือภายใน และข้อมือภายนอก เปรียบเทียบกับอุปกรณ์มาตราฐานในที่นี้คือ Smart Watch จากการคำนวณเพื่อหาข้อสรุปสามารถเปรียบเทียบได้ดังตารางที่ 4.3

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 4.2 เปรียบเทียบค่า Relative error, error และ Accuracy (ความแม่นยำ)

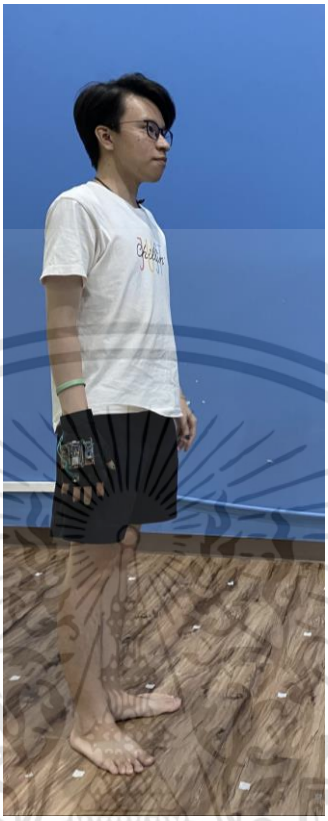
ตำแหน่ง	Relative error	error	Accuracy (ความแม่นยำ)
ปลายนิ้ว	0.047501693	4.750169265	95.24983073
ข้อมือด้านใน	0.028901114	2.890111398	97.1098886
ข้อมือด้านนอก	0.252028192	25.20281919	74.79718081

จากตารางที่ 4.2 แสดงการเปรียบเทียบค่า Relative error, error และ Accuracy (ความแม่นยำ) จากอ่านค่าอัตราการเต้นของหัวใจที่ได้จากเซนเซอร์ MAX30102 ในตำแหน่งต่างๆ ของมือดังนี้คือ ปลายนิ้ว, ข้อมือภายใน และข้อมือภายนอก เปรียบเทียบกับอุปกรณ์มาตราฐานในที่นี้คือ Smart Watch ที่ใช้หลักการวัดอัตราการเต้นของหัวใจเดียวกันกับเซนเซอร์ MAX30102 ใช้วิธีการที่เรียกว่าphotoplethysmography เพื่อวัดอัตราการเต้นของหัวใจของบุคคล โดยวิธีนี้ส่องแสงบนผิวหนังและวัดการไหลเวียนของเลือด โดยผลการทดลองวัดทั้งหมด 10 ครั้ง โดยพบว่าตำแหน่งข้อมือภายนอกมีความแม่นยำต่ำที่สุดคือ 74.79718081 และตำแหน่งข้อมือภายในมีความแม่นยำสูงที่สุดคือ 97.1098886 ซึ่งผลสรุปในครั้งนี้ทำให้ผู้จัดทำสามารถใช้ประกอบการตัดสินใจเพื่อออกแบบตำแหน่งของเซนเซอร์บนถุงมือในขั้นตอนการออกแบบถุงมือต่อไป

4.5 ทดสอบความแม่นยำของอุปกรณ์ในการตรวจจับการหกล้ม

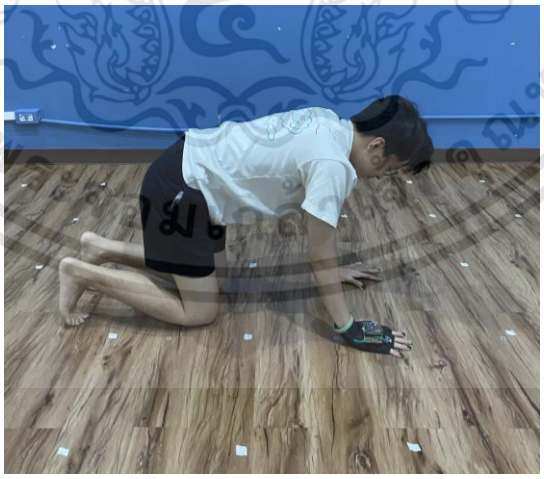
ในขั้นตอนนี้เป็นการออกแบบการทดสอบความแม่นยำของอุปกรณ์ในการตรวจจับการหกล้มโดยจะกำหนดให้การล้มในการทดลองจะให้ผู้ทดลองทำการสวมใส่อุปกรณ์แล้วทำทำการหกล้มในท่าต่าง ๆ โดยการล้มจะมีท่าดังรูปที่ 4.11, รูปที่ 4.12, รูปที่ 4.13 และรูปที่ 4.14

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.11 ผู้ทดลองในท่ายืนตรง

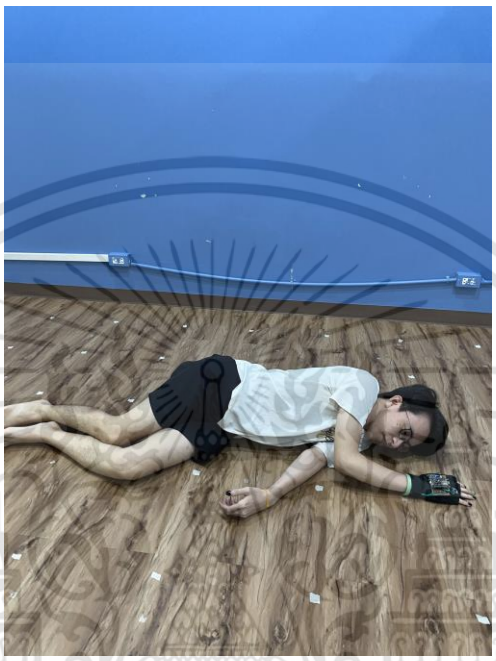
จากรูปที่ 4.11 คือลักษณะผู้ทดลองยืนในท่าตรง และจะแสดงรูปของผู้ทดลองในท่าหกล้มเข่าลงพื้นได้ดังรูปที่ 4.12



รูปที่ 4.12 ผู้ทดลองในท่าหกล้มเข่าลงพื้น

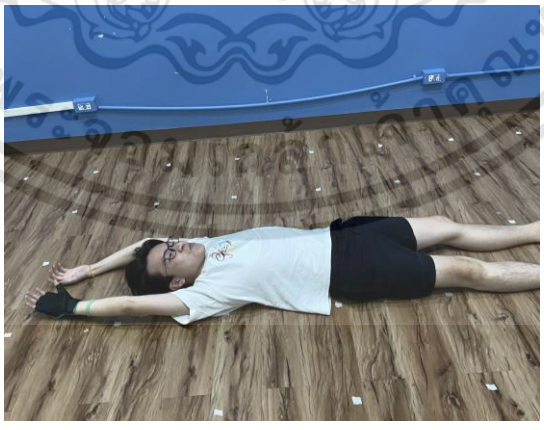
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากรูปที่ 4.12 คือลักษณะผู้ทดลองยืนในท่าหกล้มเข่าลงพื้น และจะแสดงรูปของผู้ทดลองในท่าล้มข้างไหล่ลงพื้นได้ดังรูปที่ 4.13



รูปที่ 4.13 ผู้ทดลองในท่าล้มข้างไหล่ลงพื้น

จากรูปที่ 4.13 คือลักษณะผู้ทดลองในท่าล้มข้างไหล่ลงพื้น และจะแสดงรูปผู้ทดลองในท่าลื่นล้มหงายหลังดังรูปที่ 4.14

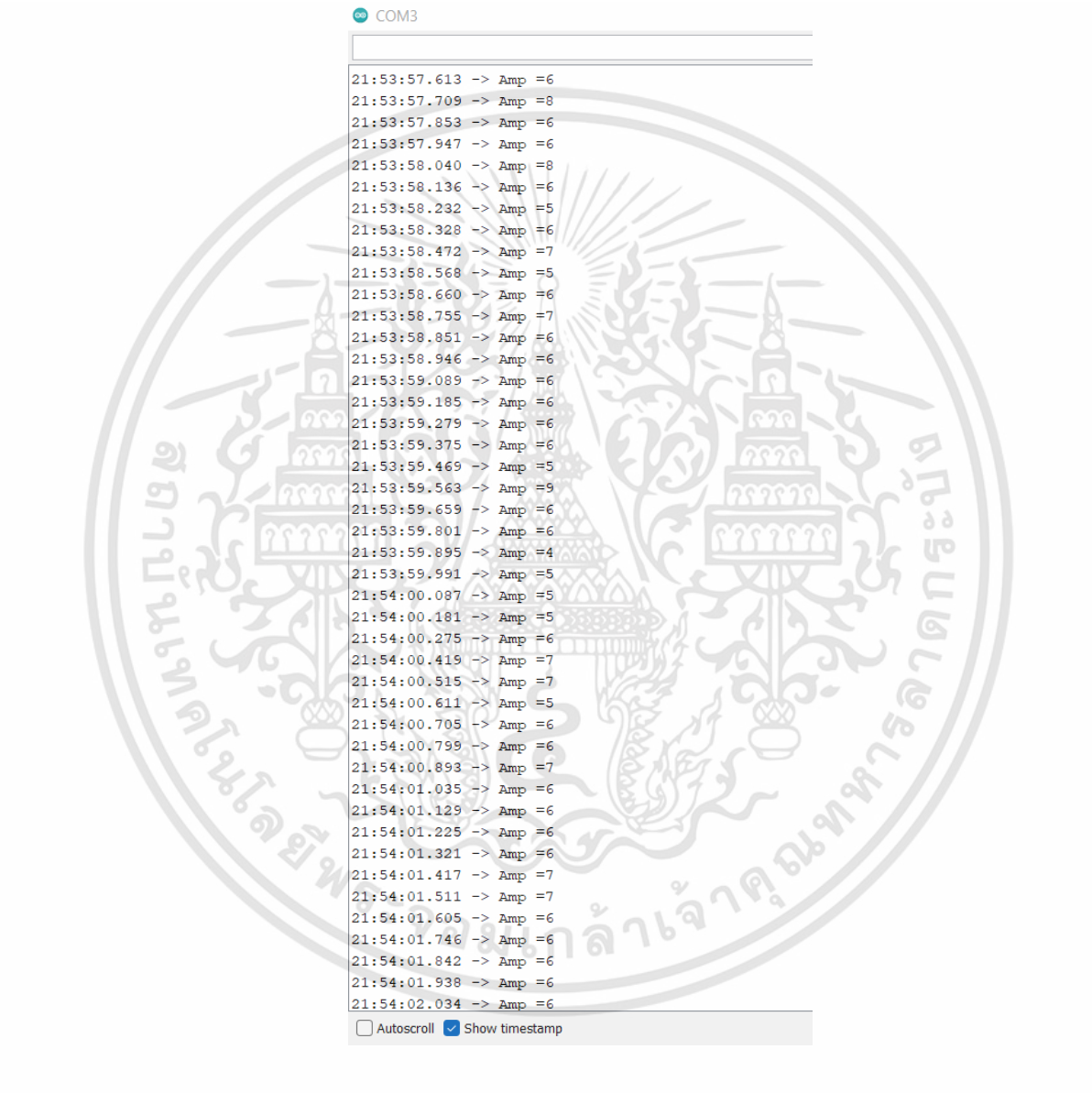


รูปที่ 4.14 ผู้ทดลองในท่าลื่นล้มหงายหลัง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากรูปที่ 4.14 คือลักษณะผู้ทดลองในท่าลื่นล้มหงายหลัง

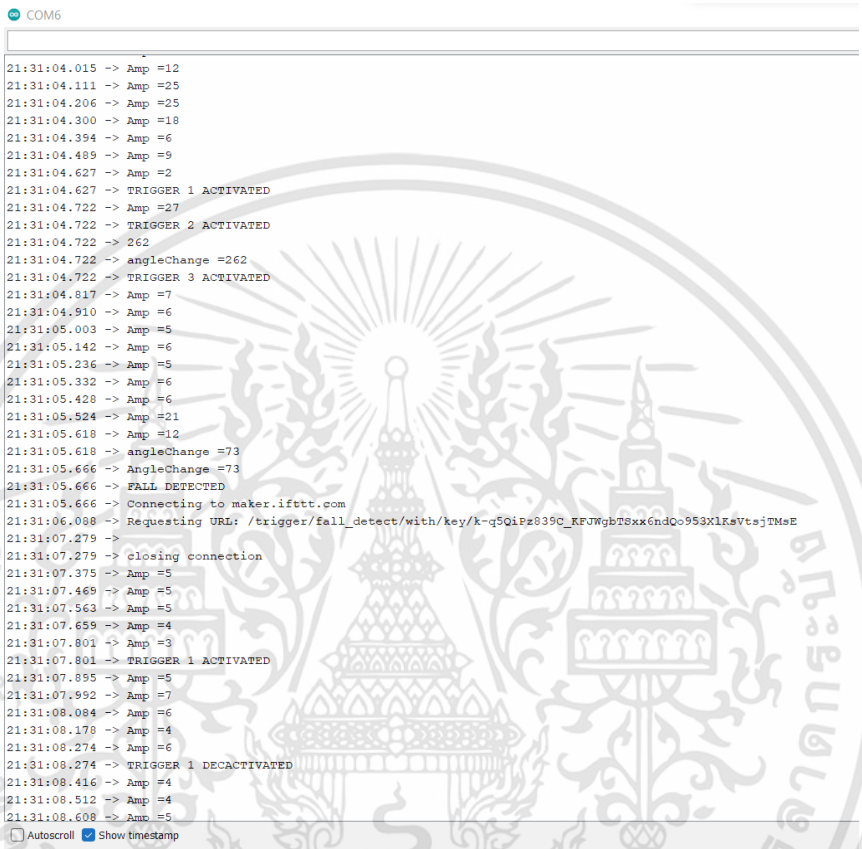
โดยในแต่ล่ะท่าจะเป็นการหกล้มที่สามารถเกิดขึ้นได้ในชีวิตประจำวันโดยผลที่ได้จากการใส่อุปกรณ์และทำการเก็บข้อมูลจากเซนเซอร์ได้ผลดังรูป 4.15



รูปที่ 4.15 ผลจากหน้าจอ Serial monitor ในท่ายืนตรง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากรูปที่ 4.15 จะแสดงผลของหน้าจอ Serial Monitor ในท่ายืนตรงโดยจะมีค่า Ampระหว่าง 5-7 โดยผลจาก Serial monitor เมื่อให้ผู้ลองล้มในท่าหกล้มเข่าลงพื้นจะแสดงในรูปที่ 4.16



รูปที่ 4.16 ผลจาก Serial monitor เมื่อให้ผู้ลองล้มในท่าหกล้มเข่าลงพื้น

จากรูปที่ 4.16 จะเห็นได้ว่าอุปกรณ์สามารถตรวจจับการหกล้มในท่าเข้าลงพื้นได้โดยผลการทดลองการล้มในท่าหกล้มเข่าลงพื้นจะมีค่าการเปลี่ยนแปลงของมุมที่ 262 แล้วลดลงมาที่ 73 โดยผลจาก Serial monitor ของการล้มในท่าล้มข้างไหล่ลงพื้นจะแสดงได้ดังรูปที่ 4.17

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
21:53:34.231 -> Amp =8
21:53:34.372 -> Amp =3
21:53:34.372 -> TRIGGER 1 ACTIVATED
21:53:34.467 -> Amp =5
21:53:34.467 -> angleChange =168
21:53:34.467 -> TRIGGER 3 DEACTIVATED
21:53:34.563 -> Amp =6
21:53:34.658 -> Amp =9
21:53:34.753 -> Amp =8
21:53:34.892 -> Amp =13
21:53:34.892 -> TRIGGER 2 ACTIVATED
21:53:34.892 -> 273
21:53:34.892 -> angleChange =273
21:53:34.892 -> TRIGGER 3 ACTIVATED
21:53:34.988 -> Amp =28
21:53:35.084 -> Amp =24
21:53:35.180 -> Amp =27
21:53:35.276 -> Amp =22
21:53:35.372 -> Amp =22
21:53:35.468 -> Amp =22
21:53:35.610 -> Amp =8
21:53:35.706 -> Amp =12
21:53:35.802 -> Amp =11
21:53:35.802 -> angleChange =70
21:53:35.802 -> AngleChange =70
21:53:35.802 -> FALL DETECTED
21:53:35.802 -> Connecting to maker.ifttt.com
21:53:36.322 -> Requesting URL: /trigger/fall_detect/with/key/k-q5QiPz839C_KFJWgbTSxx6ndQo953XlKsVtsjTMsE
21:53:37.119 ->
21:53:37.119 -> closing connection
```

รูปที่ 4.17 ผลจาก Serial monitor ของการล้มในท่าล้มข้างไหล่ลงพื้น

จากผลที่ได้จากรูปที่ 4.17 เป็นผลจากการล้มในท่าล้มข้างไหล่ลงพื้นจะเห็นได้ว่ามุมที่เปลี่ยนไปของท่าล้มนี้จะมีค่าอยู่ที่ 273 และลดลงไปอยู่ที่ 70 โดยผลจาก Serial Monitor ของการล้มในท่าลื่นล้มหงายหลังดังรูปที่ 4.18

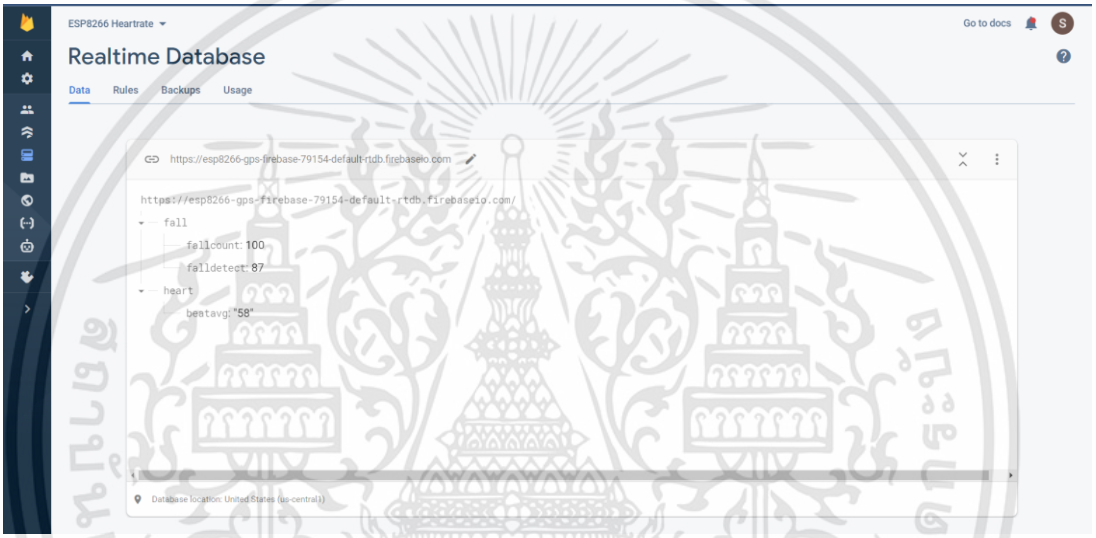
```
21:54:19.668 -> Amp =6
21:54:19.764 -> Amp =7
21:54:19.858 -> Amp =10
21:54:20.000 -> Amp =11
21:54:20.095 -> Amp =3
21:54:20.095 -> TRIGGER 1 ACTIVATED
21:54:20.191 -> Amp =8
21:54:20.287 -> Amp =18
21:54:20.287 -> TRIGGER 2 ACTIVATED
21:54:20.287 -> 262
21:54:20.287 -> angleChange =262
21:54:20.287 -> TRIGGER 3 ACTIVATED
21:54:20.383 -> Amp =9
21:54:20.479 -> Amp =5
21:54:20.621 -> Amp =3
21:54:20.621 -> TRIGGER 1 ACTIVATED
21:54:20.714 -> Amp =6
21:54:20.809 -> Amp =1
21:54:20.809 -> TRIGGER 1 ACTIVATED
21:54:20.904 -> Amp =16
21:54:20.904 -> TRIGGER 2 ACTIVATED
21:54:20.904 -> 116
21:54:21.000 -> Amp =4
21:54:21.000 -> 119
21:54:21.094 -> Amp =10
21:54:21.094 -> 97
21:54:21.190 -> Amp =7
21:54:21.190 -> 83
21:54:21.190 -> angleChange =83
21:54:21.238 -> AngleChange =83
21:54:21.238 -> FALL DETECTED
21:54:21.238 -> Connecting to maker.ifttt.com
21:54:21.284 -> Requesting URL: /trigger/fall_detect/with/key/k-q5QiPz839C_KFJWgbTSxx6ndQo953XlKsVtsjTMsE
21:54:22.184 ->
21:54:22.184 -> closing connection
```

รูปที่ 4.18 ผลจาก Serial Monitor ของการล้มในท่าลื่นล้มหงายหลัง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โดยจากรูปที่ 4.18 เป็นผลจากหน้าจอ Serial Monitor จากการล้มในท่าลื่นล้มหงายหลังโดยค่ามุมที่เปลี่ยนไปจะมีค่าอยู่ที่ 262 แล้วลดลงมาอยู่ที่ 83

ต่อมาได้ทำการทดลองหาความแม่นยำของอุปกรณ์โดยให้ผู้ทดลองล้มลงบนเบาะรองกันกระแทกเพื่อลดอาการบาดเจ็บจากการล้มและทำการล้มเป็นจำนวน 100 ครั้งนับโดยการให้อุปกรณ์ส่งค่าครั้งของการหกล้มจากการผ่านเงื่อนไขที่ 1 และทำการนับครั้งการล้มจากการตรวจจับการหกล้มได้ของอุปกรณ์โดยจำนวณทั้งหมดจะส่งขึ้นไปยัง Database เพื่อความสะดวกในการเช็คความแม่นยำได้ผลดังรูปที่ 4.19



รูปที่ 4.19 ผลจาก FIREBASE ของการนับค่าการหกล้ม

โดยจากรูปที่ 4.19 เป็นผลจาก Firebase ที่ได้เก็บค่าการนับการหกล้มและจำนวณครั้งที่สามารถตรวจจับการหกล้มได้ 87 ครั้งจาก 100 ครั้งคิดเป็นความแม่นยำในการตรวจจับการหกล้มได้ 87%

4.6 การทดลองประสิทธิภาพของแบตเตอรี่ลิเทียมไอออนโพลีเมอร์ (Lithium-Ion Polymer Battery) 3.7 โวลต์ 500 มิลลิแอมแปร์

ในการทดลองประสิทธิภาพของแบตเตอรี่ลิเทียมไอออนโพลีเมอร์ได้ทำการทดลองโดยการใช้แบตเตอรี่จ่ายไฟอุปกรณ์โดยจะมี NodeMCU ESP12E ESP8266 , MAX-30102 และ GY-521 MPU6050 โดยจะทำการปล่อยไฟให้กับอุปกรณ์และทำการจับเวลาจนไฟแสดงการทำงาน

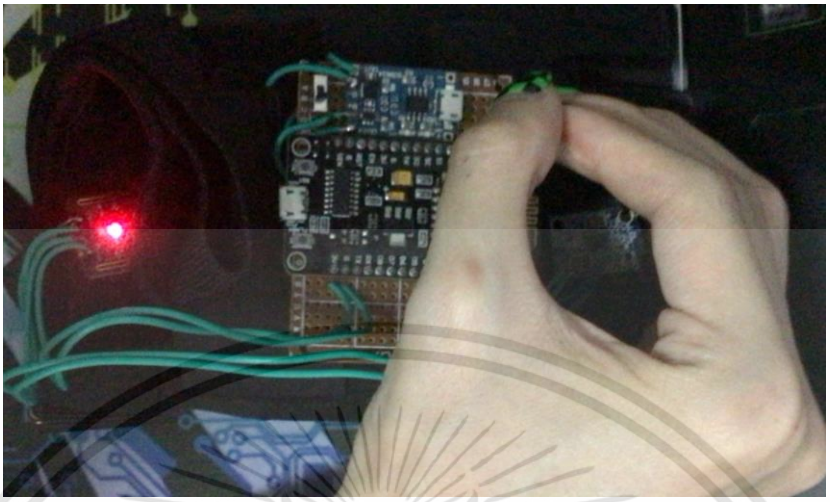
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

อุปกรณ์ไม่สามารถปล่อยแสงออกมาได้ทำให้ไม่สามารถใช้อุปกรณ์ในการส่งข้อมูลได้อีกโดยการจับเวลาเริ่มขึ้นเมื่อเปิดสวิตช์ให้แบตเตอรี่จ่ายไฟให้อุปกรณ์ตามรูป 4.19

รูปที่ 4.20 เริ่มการจับเวลาทดสอบประสิทธิภาพของแบตเตอรี่

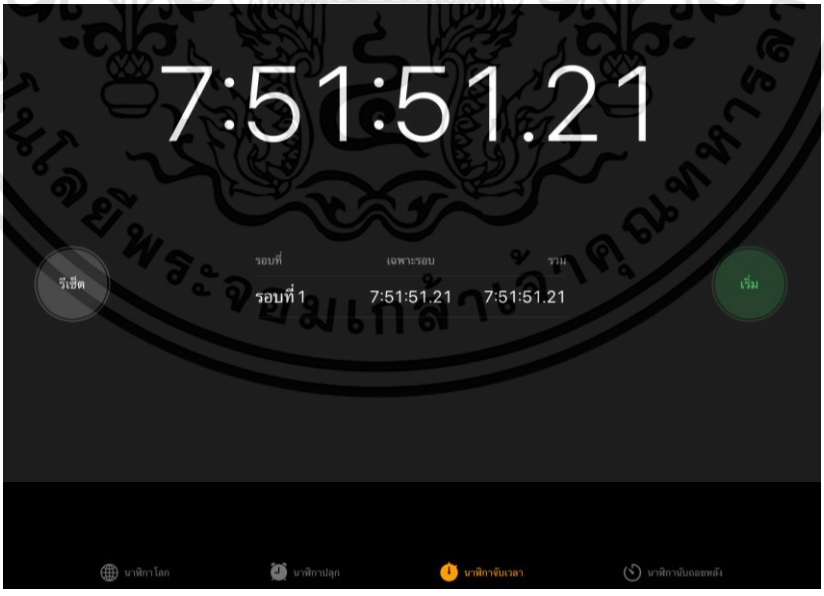
โดยในรูปที่ 4.20 จะแสดงการเริ่มจับเวลาในการทดสอบประสิทธิภาพของแบตเตอรี่ โดยจะมีไฟแสดงการทำงานของเซนเซอร์ GY-521 MPU6050 สว่างตามรูปแล้วทำการจับเวลารอจนกว่าไฟแสดงการทำงานจะดับลงตามรูปที่ 4.21

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.21 การหยุดการทำงานของเซนเซอร์ GY-521 MPU6050

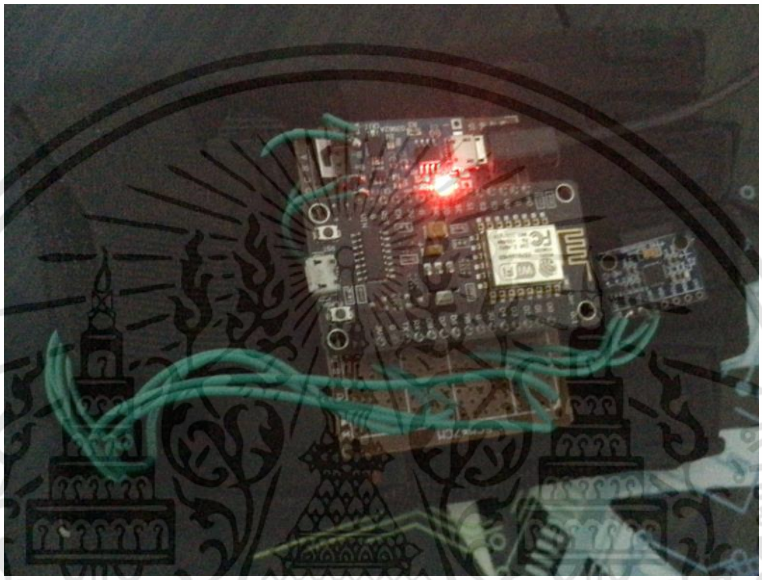
จากรูปที่ 4.21 จะทำการหยุดการจับเวลาในการทดสอบประสิทธิภาพของแบตเตอรี่ เมื่อทำการจ่ายไฟให้กับอุปกรณ์จนไฟแสดงการทำงานของเซนเซอร์ GY-521 MPU6050 ดับลงเป็นการหยุดการทำงานของอุปกรณ์โดยสามารถจ่ายไฟเป็นเวลานาน 7 ชั่วโมง 51 นาที 51 วินาทีตามรูปที่ 4.22



รูปที่ 4.22 ผลการจับเวลาในการทดลองประสิทธิภาพของแบตเตอรี่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

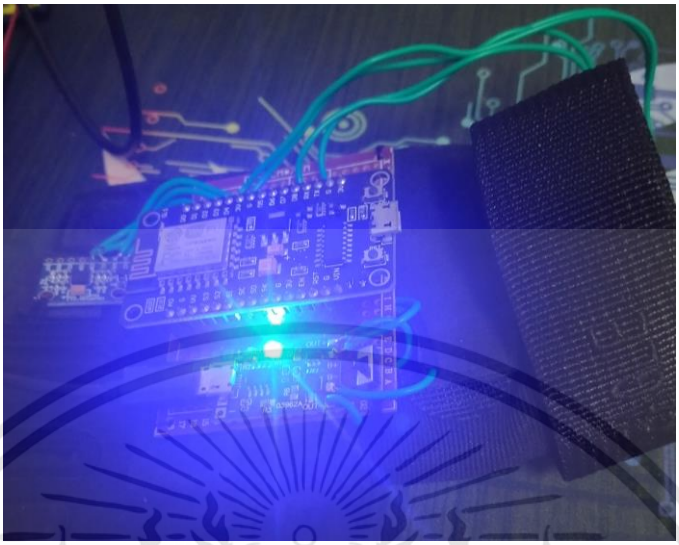
จากรูปที่ 4.22 ผลการจับเวลาในการทดลองประสิทธิภาพของแบตเตอรี่ โดยต่อมาจะทำการจับเวลาในการชาร์จแบตเตอรี่โยเมื่อทำการเสียบสายชาร์จไปยังโมดูลชาร์จแบตเตอรี่ TP4056 โดยทำการเสียบสาย USB ไปยังที่เสียบของโมดูลโดยจะเป็นไปดังรูปที่ 4.22



รูปที่ 4.23 การชาร์จแบตเตอรี่ผ่านชาร์จแบตเตอรี่ TP4056

จากรูปที่ 4.23 แสดงการทำงานของโมดูลชาร์จแบตเตอรี่ TP4056 โดยเมื่อกำลังทำการชาร์จจะแสดงไฟสีแดงในการทำงานของโมดูลและในขณะที่เริ่มทำการชาร์จแบตเตอรี่จะทำการจับเวลาเพื่อบันทึกว่าต้องใช้เวลาเท่าไรในการชาร์จแบตเตอรี่จนเต็มโดยเมื่อแบตเตอรี่เต็มจะแสดงผลการทำงานของโมดูลชาร์จแบตเตอรี่ตามรูปที่ 4.23

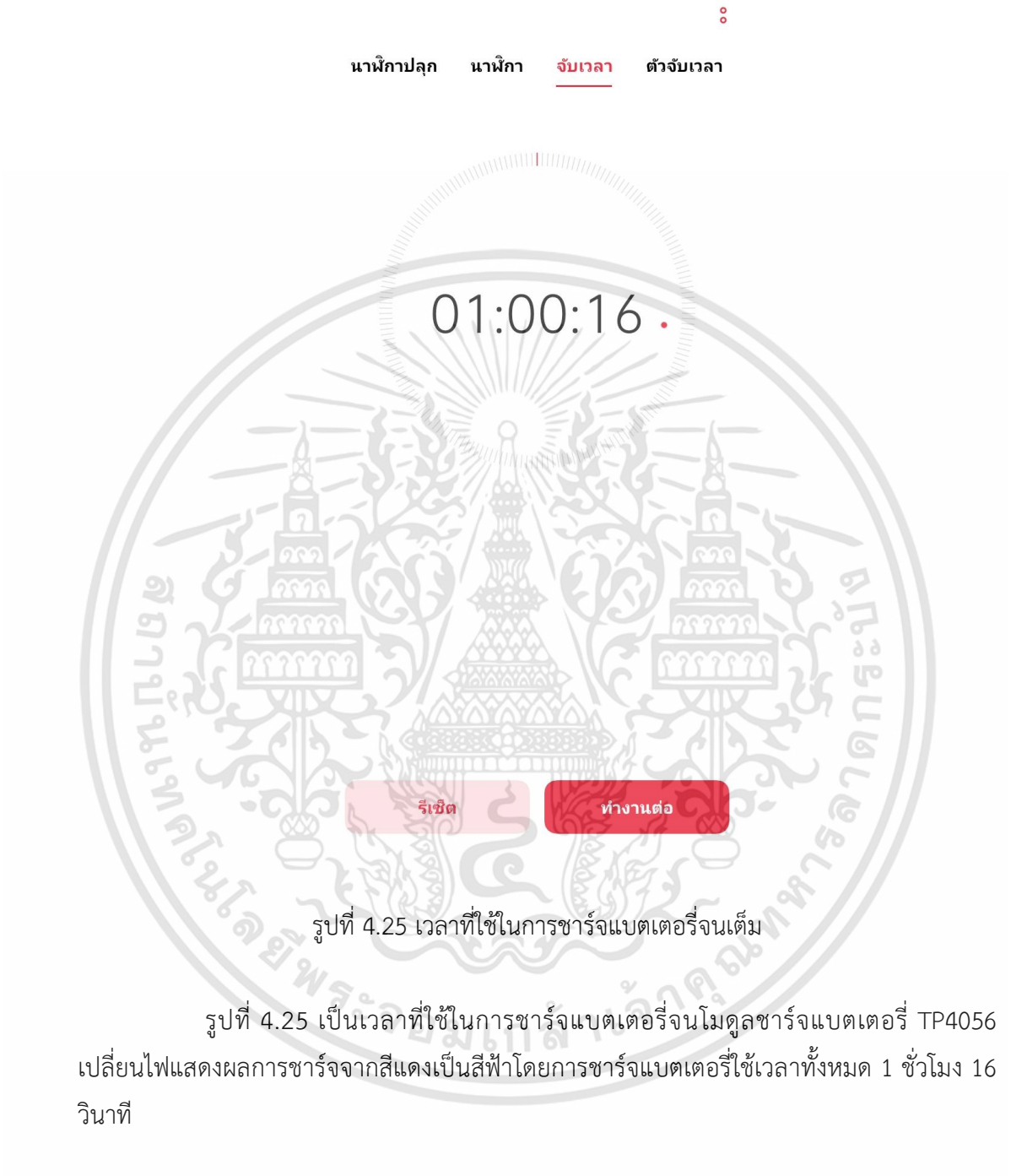
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.24 โมดูลชาร์จแบตเตอรี่ TP4056 เมื่อแบตเตอรี่ชาร์จเต็ม

ในรูปที่ 4.24 เป็นการแสดงผลของโมดูลชาร์จแบตเตอรี่ TP4056 เมื่อทำการชาร์จเต็มจะแสดงไฟสีฟ้าออกมาแสดงว่าแบตเตอรี่ได้ชาร์จเต็มแล้วโดยเวลาที่ใช้ในการชาร์จแบตเตอรี่จนเต็มใช้เวลาทั้งสิ้นตามรูปที่ 4.25

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.25 เวลาที่ใช้ในการชาร์จแบตเตอรี่จนเต็ม

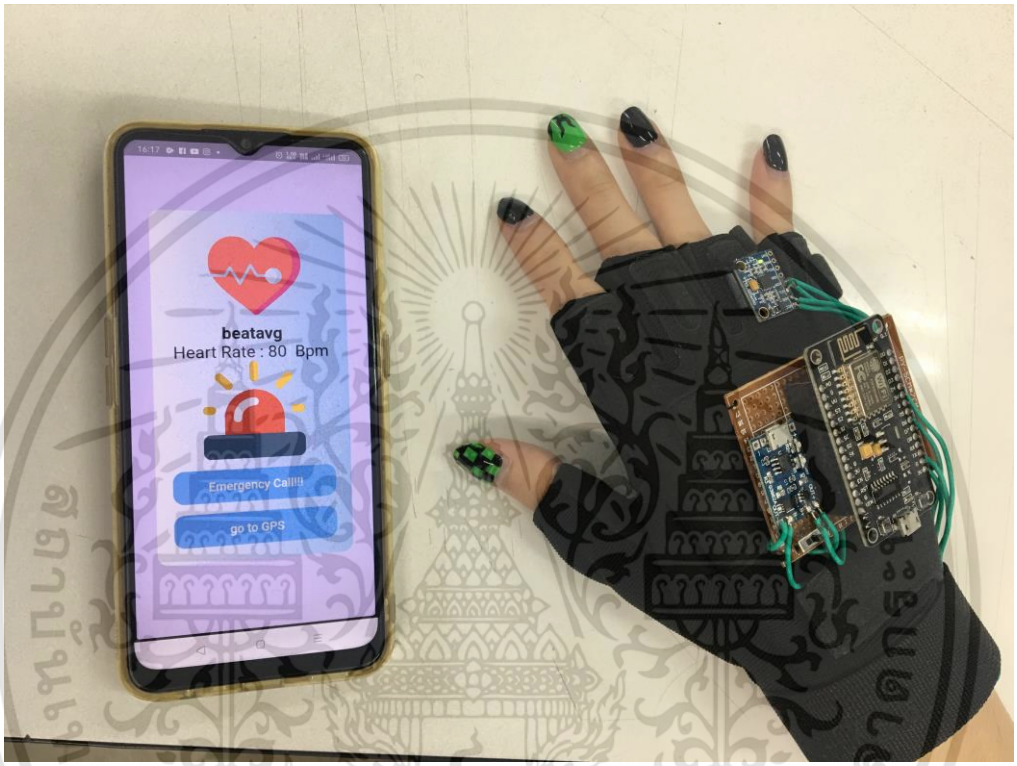
รูปที่ 4.25 เป็นเวลาที่ใช้ในการชาร์จแบตเตอรี่จนโมดูลชาร์จแบตเตอรี่ TP4056 เปลี่ยนไฟแสดงผลการชาร์จจากสีแดงเป็นสีฟ้าโดยการชาร์จแบตเตอรี่ใช้เวลาทั้งหมด 1 ชั่วโมง 16 วินาที

4.7 ทดสอบการทำงานของแอพลิเคชัน และการแสดงผล

เมื่อเรากดเข้าใช้งานแอพลิเคชันบนหน้าจอโทรศัพท์มือถือจะเจอหน้าต่าง login เป็นหน้าแรก โดยจะมีปุ่ม sign up และ sign in ให้เลือก หากไม่เคยทำการลงทะเบียนมาก่อนหรือเข้า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ใช้งานเป็นครั้งแรกให้กดปุ่มsign up เพื่อทำการลงทะเบียนเข้าใช้งาน เมื่อกรอก Email และ password เรียบร้อยแล้ว จะกลับมาที่หน้าจอ log in อีกครั้งจากนั้นกดปุ่มsign in กรอก Email และ password ที่ลงทะเบียนไว้ จะสามารถเข้าสู่หน้าต่างใช้งานได้ดังรูปที่ 4.25

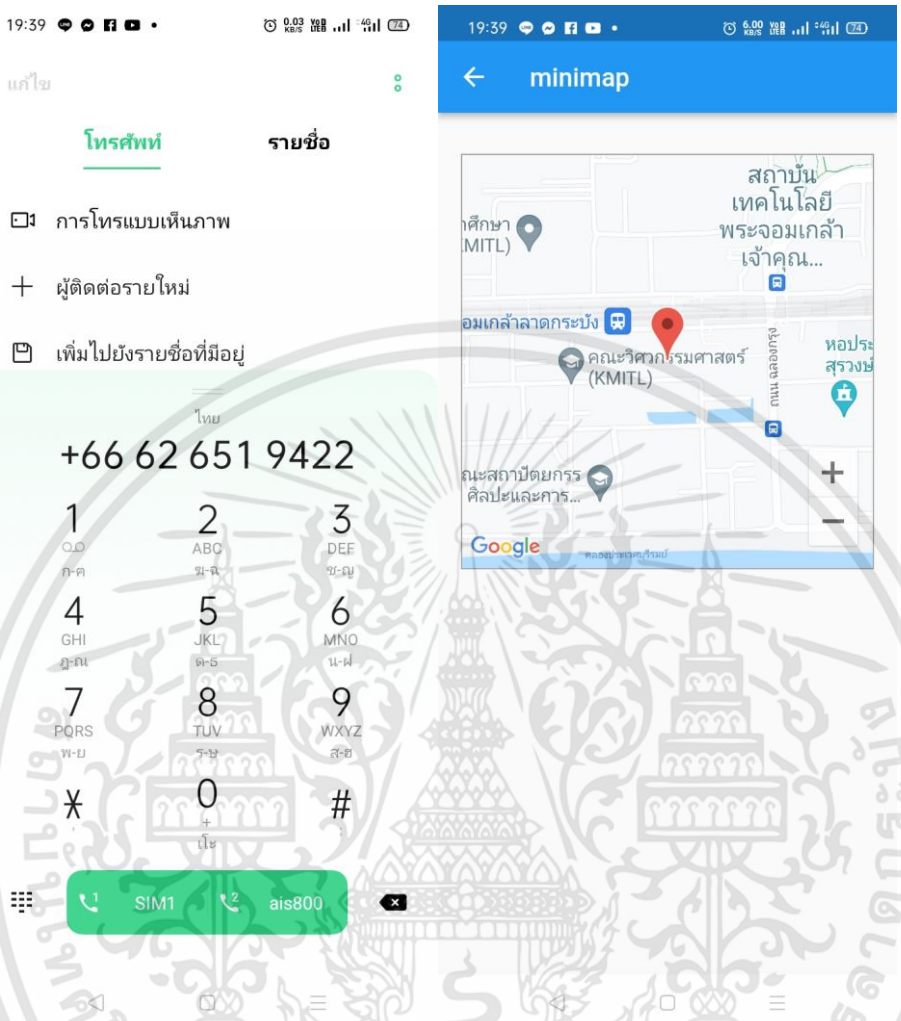


รูปที่ 4.26 การใช้งานแอพลิเคชันคู่กับอุปกรณ์

จากรูปที่ 4.26 เป็นการใช้งานแอพลิเคชันควบคู่ไปกับอุปกรณ์ถุงมือที่สร้างขึ้น จะปรากฏค่า Heart rate ปุ่ม Emergency call และปุ่ม GPS

ต่อมาได้ทำการตรวจสอบ หน้าต่างการโทรออกหรือ ปุ่ม Emergency call ว่าสามารถใช้งานได้ปกติหรือไม่ รวบไปถึงการตรวจสอบความแม่นยำของ GPS ว่าสามารถใช้งานได้และตรงจุดหรือไม่แสดงได้ดังรูปที่ 4.27

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.27 การตรวจสอบหน้าต่างการโทรฉุกเฉินและ GPS

จากรูปที่ 4.27 จากการตรวจสอบหน้าต่างแสดงผลทั้งสองพบว่าสามารถทำงานได้อย่างถูกต้องไม่ผิดพลาด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5

สรุปผลและข้อเสนอแนะ

5.1 สรุปผล

“แอปพลิเคชันแจ้งเหตุฉุกเฉินสำหรับผู้ป่วยโรคเรื้อรัง” นั้นได้ทำการมุ่งเน้นการพัฒนาชุดอุปกรณ์ที่ใช้ในการแจ้งเหตุฉุกเฉินสำหรับผู้ป่วยโรคเรื้อรังซึ่งจัดทำในลักษณะของถุงมือ โดยใช้เซนเซอร์ MAX30102 สำหรับวัดอัตราการเต้นของหัวใจ และใช้โมดูล GY-521 MPU6050 ในการตรวจกับการหกล้มของผู้ป่วย รวมถึงการดึงพิกัด GPS ในโทรศัพท์ผู้ป่วยเพื่อระบุตำแหน่ง โดยนำเซนเซอร์ทั้งหมดเชื่อมต่อกับโมดูล Wi-Fi NodeMCU ESP12E ESP8266 เพื่อทำการควบคุมการทำงานของเซนเซอร์ให้เชื่อมต่อกับ Wi-Fi เพื่อส่งข้อมูลที่อ่านค่าได้จากเซนเซอร์ไปยังฐานข้อมูลแบบเรียลไทม์บน Firebase โดยใช้ Flutter ซึ่งเป็น Cross-Platform Framework ที่ใช้ในการพัฒนา Mobile Application ซึ่งเชื่อมต่อ Firebase กับ Flutter เข้าด้วยกัน ซึ่งจะเชื่อมไปที่ Service ชื่อ Realtime Database ซึ่งเป็น Database ของ Firebase

ในส่วนของชุดอุปกรณ์ถุงมือซึ่งได้ทำการทดลองหาตำแหน่งที่เหมาะสมในการติดตั้งเซนเซอร์ MAX30102 สำหรับวัดอัตราการเต้นของหัวใจ ซึ่งตำแหน่งที่เหมาะสมที่สุดคือบริเวณข้อมือด้านในให้ประสิทธิภาพใกล้เคียงกับ Smart watch ส่วนโมดูล GY-521 MPU6050 ที่ใช้ในการตรวจกับการหกล้มของผู้ป่วยได้ทำการทดลอง สามารถตรวจจับการหกล้มของผู้ป่วย และสามารถส่งแจ้งเตือนผู้ดูแลผู้ป่วยผ่าน Line รวมไปถึงตรวจจับการล้มได้หลายท่าล้มขึ้นอยู่กับการล้มขอผู้ป่วยต้องเกิดการเปลี่ยนมุมให้เซนเซอร์ GY-521 MPU6050 สามารถอ่านค่ามุมที่เปลี่ยนไปได้ และต้องยกขึ้นสูงกว่ามือในตำแหน่งยืนนิ่งก่อนเกิดการหกล้มเช่นการยื่นมือออกไปข้างหน้าป้องกันการบาดเจ็บตามสัญชาติญาณ ให้เซนเซอร์สามารถอ่านค่า amp ให้ลดลงได้และมีความแม่นยำในการตรวจจับการหกล้มที่ 87% ในส่วนของอุปกรณ์แบตเตอรี่ และอุปกรณ์การชาร์จ อุปกรณ์สามารถสามารถจ่ายไฟเป็นเวลานาน 7 ชั่วโมง 51 นาที 51 วินาที และใช้เวลาชาร์จใช้เวลาทั้งหมด 1 ชั่วโมง 16 วินาที

ในส่วนของแอปพลิเคชันซึ่งเป็น Mobile Application ซึ่งเป็นระบบปฏิบัติการ Android สามารถเชื่อมต่อ Realtime Database ซึ่งเป็น Database ของ Firebase ได้ โดยดึงข้อมูลมาแสดงบนหน้าจอแอปพลิเคชันหลักได้ตลอดเวลา โดยค่าอัตราการเต้นของหัวใจจะแสดงทุกๆ 5 วินาที และพิกัด GPS เมื่อทำการขอ API Key สามารถแสดงพิกัดได้แบบเรียลไทม์บนบริการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ของ Google Map รวมไปถึงฟังก์ชัน emergency call สามารถตั้งค่าเบอร์โทรของโรงพยาบาลที่ต้องการ และสามารถกดโทรออกได้ทันที

5.2 ข้อเสนอแนะ

1. ถุงมือควรใช้วัสดุที่ระบายความร้อนได้ดีทำให้ไม่รู้สึกอึดอัดเวลาสวมใส่และช่วยลดแรงกระแทกเพื่อความปลอดภัยของผู้ป่วยด้วย

2. แบตเตอรี่มีอายุการใช้งานน้อยเนื่องจากความจุ และขนาดของแบตเตอรี่ที่เล็ก ทำให้มีอายุการใช้งานที่น้อย หากแบตเตอรี่ที่มีความจุเพิ่มขึ้นจะมีขนาด และน้ำหนักเพิ่มตามไปด้วย อาจไม่เหมาะกับอุปกรณ์ถุงมือ

3. แอปพลิเคชันยังจำกัดการใช้ได้เพียงระบบระบบปฏิบัติการ Android

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บรรณานุกรม

[1] โรงพยาบาลบำรุงราษฎร์. “โรคเรื้อรัง”
https://www.vitallifeintegratedhealth.com/th/concerns/health/chronic-disease-history

[2] วิกิพีเดียสาราณุกรมเสรี. “ไมโครคอนโทรลเลอร์”
https://shorturl.asia/qWRdA

[3] cybertice. “สอนใช้งาน NodeMCU ESP8266 v3”
https://shorturl.asia/isyIo

[4] thiti. “ESP8266 คืออะไร?”
https://thiti.dev/blog/5222/

[5] fitrox. “รู้จักกับ ESP8266”
https://shorturl.asia/IGK2o

[6] PoundXI. “NodeMCU คืออะไร”
https://shorturl.asia/ZbJ6i

[7] mindphp. “ภาษาซี (C programming language)”
https://shorturl.asia/ext8g

[8] วิกิพีเดีย สารานุกรมเสรี. “ซี (ภาษาโปรแกรม)”
https://shorturl.asia/OSrZ7

[9] Todspol Wonhchomphu. “SQL คืออะไร? และ NOSQL คืออะไร?”
https://shorturl.asia/rZJE6

[10] aws.amazon. “ฐานข้อมูล NoSQL คืออะไร”
https://aws.amazon.com/th/nosql/

[11] techtalkthai. “ฐานข้อมูล NoSQL 4 ประเภทเบื้องต้นที่คุณควรรู้จัก”
https://www.techtalkthai.com/4-basic-nosql-databases-you-should-know/

[12] 4 Xtreme Co., Ltd. “Firebase คืออะไร”
https://shorturl.asia/I6Q8f

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

[13] Jirawatee. “รู้จัก Firebase Realtime Database ตั้งแต่ Zero จนเป็น Hero”
https://shorturl.asia/6Vjml

[14] ผศ.ดร.พิมพ์เพ็ญ พรเฉลิมพงศ์. “Precision / ความเที่ยงตรง”
https://shorturl.asia/B490G

[15] พบแพทย์. “วิธีตรวจชีพจร และสัญญานบอกอันตราย”
https://shorturl.asia/QgtWD

[16] Wikipedia, the free encyclopedia. “Geographic coordinate system”
https://en.wikipedia.org/wiki/Geographic_coordinate_system

[17] Hizokad. “Flutter คืออะไร?”
https://shorturl.asia/eAj4c

[18] สมาคมโปรแกรมเมอร์ไทย. “Flutter Framework เครื่องมือที่ประสิทธิภาพในการสร้าง native app แบบหลาย platform”
https://www.thaiprogrammer.org/2019/11/flutter-framework/

[19] Tanapoj Chaivanichanan. “ทำความรู้จักภาษา Dart ฉบับโปรแกรมเมอร์”
https://www.centrilliontech.co.th/blog/2570/dart-101-introduction-for-programmers/

[20] Akenarin Komkoon . “สรุปพื้นฐานภาษา Dart ก่อนเขียน Flutter”
https://shorturl.asia/CIx70

[21] AB-Maker Sensor Module Shield. “แนะนำการต่อแหล่งจ่ายให้กับ Arduino”
https://shorturl.asia/i06Wv

[22] modulemore. “วิธีคำนวณระยะเวลาที่เหลืออยู่ของแบตเตอรี่”
https://shorturl.asia/a9pjn

[23] components101. “NodeMCU ESP8266”
https://components101.com/development-boards/nodemcu-esp8266-pinout-features-and-datasheet

[24] InvenSense Inc. “MPU-6000 and MPU-6050 Product Specification.”
https://www.yumpu.com/en/document/view/45556991/mpu-6000-and-mpu-6050-product-specification-revision-10

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

[25] modulemore. “เซนเซอร์วัดอัตราการเต้นหัวใจ MAX30102”
https://shorturl.asia/3pXwe

[26] addicore. “TP4056 / TC4056A Lithium Battery Charger and Protection Module”
https://www.addicore.com/TP4056-Charger-and-Protection-Module-p/ad310.htm

[27] lipobattery.us. “LiPo Battery LP902030 3.7V 500mAh”
https://www.lipobattery.us/lipo-battery-lp902030-3-7v-500mah/

[28] Natcha Sukulna. “การติดตั้ง Google Maps for Flutter”
https://www.centrilliontech.co.th/blog/2873/how-to-setup-google-maps-for-flutter/

[29] Siratee K. “Flutter x Firebase (Cloud Firestore)”
https://medium.com/firebasethailand/flutter-x-firebase-cloud-firestore-29275799f6e9

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ก
โค้ดที่ใช้ในการส่งข้อมูลจากไมโครคอนโทลเลอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

allsensro_firebase

```
#include <Wire.h>
#include <FirebaseESP8266.h>  //https://github.com/mobizt/Firebase-ESP8266
#include "MAX30105.h"
#include "heartRate.h"
#include <ESP8266WiFi.h>
#include <SoftwareSerial.h>
#include <TinyGPS++.h> //https://github.com/mikalhart/TinyGPSPlus
MAX30105 particleSensor;
#define FIREBASE_HOST "esp8266-gps-firebase-79154-default-rtdb.firebaseio.com"
#define FIREBASE_AUTH "g2ZRe2t6IJsixSm85ly41H7sTAghpLGH111eayEZ"
#define WIFI_SSID "Mai"
#define WIFI_PASSWORD "11111111"
//Define FirebaseESP8266 data object
FirebaseData firebaseData;
unsigned long period = 5000; //ระยะเวลาที่ต้องการรอ
unsigned long period2 = 100;
unsigned long last_time = 0; //ประกาศตัวแปรเป็น global เพื่อเก็บค่าไว้ไม่ให้ reset จากการวนloop
FirebaseJson json;
const int MPU_addr=0x68;  // I2C address of the MPU-6050
int16_t AcX,AcY,AcZ,Tmp,GyX,GyY,GyZ;
float ax=0, ay=0, az=0, gx=0, gy=0, gz=0;
boolean fall = false; //stores if a fall has occurred
boolean trigger1=false; //stores if first trigger (lower threshold) has occurred
boolean trigger2=false; //stores if second trigger (upper threshold) has occurred
boolean trigger3=false; //stores if third trigger (orientation change) has occurred
byte trigger1count=0; //stores the counts past since trigger 1 was set true
byte trigger2count=0; //stores the counts past since trigger 2 was set true
byte trigger3count=0; //stores the counts past since trigger 3 was set true
int angleChange=0;
void send_event(const char *event);
const char *host = "maker.ifttt.com";
const char *privateKey = "k-q5QiPz839C_KFJWgbTSxx6ndQo953XlKsVtsjTMsE";
const byte RATE_SIZE = 4; //Increase this for more averaging. 4 is good.
byte rates[RATE_SIZE]; //Array of heart rates
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

allsensro_firebase

```
byte rates[RATE_SIZE]; //Array of heart rates
byte rateSpot = 0;
long lastBeat = 0; //Time at which the last beat occurred
int fallcount=0;
float beatsPerMinute;
String beatavg ;
int beatAvg;

void setup()
{

 Wire.begin();
 Wire.beginTransmission(MPU_addr);
 Wire.write(0x6B);  // PWR_MGMT_1 register
 Wire.write(0);     // set to zero (wakes up the MPU-6050)
 Wire.endTransmission(true);
  Serial.begin(115200);
  Serial.println("");
  wifiConnect();
  Serial.println("Connecting Firebase.....");
  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
  Firebase.reconnectWiFi(true);
  Serial.println("Firebase OK.");
if (!particleSensor.begin(Wire, I2C_SPEED_FAST)) //Use default I2C port, 400kHz speed
{
Serial.println("MAX30105 was not found. Please check wiring/power. ");
while (1);
}
Serial.println("Place your index finger on the sensor with steady pressure.");

particleSensor.setup(); //Configure sensor with default settings
particleSensor.setPulseAmplitudeRed(0x0A); //Turn Red LED to low to indicate sensor is running
particleSensor.setPulseAmplitudeGreen(0); //Turn off Green LED
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

allsensro_firebase

```
68
69 void loop()
70 {
71 long irValue = particleSensor.getIR();
72  if (checkForBeat(irValue) == true)
73   {
74   //We sensed a beat!
75     long delta = millis() - lastBeat;
76     lastBeat = millis();
77
78     beatsPerMinute = 60 / (delta / 1000.0);
79
80   if (beatsPerMinute < 255 && beatsPerMinute > 20)
81   {
82     rates[rateSpot++] = (byte)beatsPerMinute; //Store this reading in the array
83     rateSpot %= RATE_SIZE; //Wrap variable
84
85     //Take average of readings
86     beatAvg = 0;
87     for (byte x = 0 ; x < RATE_SIZE ; x++)
88     beatAvg += rates[x];
89     beatAvg /= RATE_SIZE;
90     }
91   }
92  mpu_read();
93  ax = (AcX-2050)/16384.00;
94  ay = (AcY-77)/16384.00;
95  az = (AcZ-1947)/16384.00;
96  gx = (GyX+270)/131.07;
97  gy = (GyY-351)/131.07;
98  gz = (GyZ+136)/131.07;
99  // calculating Amplitute vactor for 3 axis
100  float Raw_Amp = pow(pow(ax,2)+pow(ay,2)+pow(az,2),0.5);
101  int Amp = Raw_Amp * 10;  // Mulitiplied by 10 bcz values are between 0 to 1
102  Serial.print("Amp =");
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

allsensro_firebase §

```
Serial.println(Amp);
 if (Amp<=3 && trigger2==false){
  trigger1=true;
  }
if (trigger1==true){
  trigger1count++;
  if (Amp>=12){
    trigger2=true;
    trigger1=false; trigger1count=0;
    }
}
if (trigger2==true){
  trigger2count++;
  angleChange = pow(pow(gx,2)+pow(gy,2)+pow(gz,2),0.5); Serial.println(angleChange);
  if (angleChange>=200 && angleChange<=300){
    trigger3=true; trigger2=false; trigger2count=0;
       Serial.print("angleChange =");
      Serial.println(angleChange);
       }
  }
if (trigger3==true){
   trigger3count++;

   if (trigger3count>=10){
      angleChange = pow(pow(gx,2)+pow(gy,2)+pow(gz,2),0.5);
       Serial.print("angleChange =");
      Serial.println(angleChange);
      //delay(10);
      if ((angleChange>=0) && (angleChange<=100)){
         fall=true; trigger3=false; trigger3count=0;
        Serial.print("AngleChange =");
         Serial.println(angleChange);
          }
      else{
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

allsensro_firebase §

```
136         else{
137            trigger3=false; trigger3count=0;
138         }
139       }
140   }
141  if (fall==true){
142    Serial.println("FALL DETECTED");
143      send_event("fall_detect");
144    fall=false;
145    }
146  if (trigger2count>=6){
147    trigger2=false; trigger2count=0;
148    }
149  if (trigger1count>=6){
150    trigger1=false; trigger1count=0;
151    }
152
153 if(millis() - last_time > period) {
154   beatavg = String(beatAvg);
155   Firebase.setString(firebaseData,"/heart/beatavg", beatavg );
156
157    last_time = millis();
158   }
159 }
160 void wifiConnect()
161 {
162   WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
163   Serial.print("Connecting to Wi-Fi");
164   while (WiFi.status() != WL_CONNECTED)
165   {
166     Serial.print(".");
167     delay(300);
168   }
169   Serial.println();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

allsensro_firebase §

```
                "Host: " + host + "\r\n" +
                "Connection: close\r\n\r\n");
  while(client.connected())
  {
    if(client.available())
    {
      String line = client.readStringUntil('\r');
      Serial.print(line);
    } else {
      // No data yet, wait a bit
      delay(50);
    };
  }
  Serial.println();
  Serial.println("closing connection");
  client.stop();
}
void firebaseReconnect()
{
  Serial.println("Trying to reconnect");
  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
}
void mpu_read(){
 Wire.beginTransmission(MPU_addr);
 Wire.write(0x3B);
 Wire.endTransmission(false);
 Wire.requestFrom(MPU_addr,14,true);  // request a total of 14 registers
 AcX=Wire.read()<<8|Wire.read();  // 0x3B (ACCEL_XOUT_H) & 0x3C (ACCEL_XOUT_L)
 AcY=Wire.read()<<8|Wire.read();  // 0x3D (ACCEL_YOUT_H) & 0x3E (ACCEL_YOUT_L)
 AcZ=Wire.read()<<8|Wire.read();  // 0x3F (ACCEL_ZOUT_H) & 0x40 (ACCEL_ZOUT_L)
 Tmp=Wire.read()<<8|Wire.read();  // 0x41 (TEMP_OUT_H) & 0x42 (TEMP_OUT_L)
 GyX=Wire.read()<<8|Wire.read();  // 0x43 (GYRO_XOUT_H) & 0x44 (GYRO_XOUT_L)
 GyY=Wire.read()<<8|Wire.read();  // 0x45 (GYRO_YOUT_H) & 0x46 (GYRO_YOUT_L)
 GyZ=Wire.read()<<8|Wire.read();  // 0x47 (GYRO_ZOUT_H) & 0x48 (GYRO_ZOUT_L)
 }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ข
โค้ดในส่วนแอปพลิเคชัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1. main.dart

```dart
import 'package:flutter/material.dart';
import 'package:flutter_application_more/1.dart';
import 'package:flutter_application_more/class.dart';
import 'package:flutter_application_more/register.dart';
void main() => runApp(const MyApp());
class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Chronic Patient Emergency',
      home: login(),
    );
  }
}
```

2. หน้าหลักและส่วนการดึงข้อมูลจาก Firebase

```dart
import 'package:firebase_core/firebase_core.dart';
import 'package:firebase_database/firebase_database.dart';
import 'package:firebase_database/firebase_database.dart';
import 'package:flutter/material.dart';
import 'package:flutter_application_more/Data.dart';
import 'package:flutter_application_more/long.dart';
import 'package:url_launcher/url_launcher.dart';
import 'Data.dart';

class first extends StatefulWidget {
  const first({Key? key}) : super(key: key);

  @override
  _firstState createState() => _firstState();
}

class Activity extends State<first> {
  late Cubic static;

  @override
  Widget build(BuildContext context) {
    throw UnimplementedError();
  }
}

class _firstState extends State<first> {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```dart
List<data> dataList = [];

late DatabaseReference _dbref;
String databasejson = '';

late String heart = "beatavg";

@override
void initState() {
  super.initState();

  _dbref = FirebaseDatabase.instance.reference();
  _readdb_onechild();
}

@override
Widget build(BuildContext context) {
  return Center(
    child: buildMain(),
  );
}

Widget buildMain() {
  return Container(
    child: Card(
      color: Color.fromARGB(255, 236, 228, 255),
      color: Color.fromARGB(255, 236, 228, 255),
      child: Center(
        child: SingleChildScrollView(
          child: Container(
            decoration: BoxDecoration(
              borderRadius: BorderRadius.circular(16),
              gradient: LinearGradient(colors: [Colors.white, Colors.blue.shade100])),
            margin: EdgeInsets.all(32),
            padding: EdgeInsets.all(24),
            child: Column(
              mainAxisSize: MainAxisSize.min,
              // mainAxisAlignment: MainAxisAlignment.center,
              children: <Widget>[
                const Image(
                  image: AssetImage(
                    'image/heart-beat.png',
                  ),
                  width: 150,
                  height: 150,
                ),
                Text(
                  'beatavg',
                  style: TextStyle(color: Colors.black, fontSize: 25, fontWeight: FontWeight.bold
                ),
                StreamBuilder(
                    stream: _dbref.onValue,
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    stream: _dbref.onValue,
                    builder: (BuildContext context, AsyncSnapshot<Event> snap) {
                      if (snap.hasData && !snap.hasError && snap.data?.snapshot.value != null) {
                        Map data = snap.data?.snapshot.value;

                        data.forEach((key, value) {
                          // beatavg = data['beatavg'];
                          heart = value["beatavg"];
                          print("heart data : ${value["beatavg"]}");
                        });

                        //  var dd = data.entries.elementAt(0).value['key'];

                        // data.forEach((index, data) => item.add({"key": index, ...data}));
                        return Text(
                          "Heart Rate : ${heart}  Bpm",
                          // "Heart Rate : ${data.entries.elementAt(0).value['beatavg']}  Mph",
                          style: TextStyle(fontSize: 25),
                        );
                      } else {
                        return Center(child: Text("No data"));
                      }
                    }),
                const Image(
                  image: AssetImage(
                    'image/alert.png',
                    ),
                    width: 150,
                    height: 150,
                  ),
                  Buttoncall(),
                  Buttongps(),
                ],
              )),
            ),
          ),
        ),
      );
    }

    _readdb_onechild() {
      _dbref.child("heart").child("beatavg").once().then((DataSnapshot dataSnapshot) {
        print(" read once - " + dataSnapshot.value.toString());

        setState(() {
          heart = dataSnapshot.value.toString();
          databasejson = dataSnapshot.value.toString();
        });
      });
    }

    Container Buttoncall() {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Container Buttoncall() {
  return Container(
      constraints: BoxConstraints.expand(height: 50),
      decoration: BoxDecoration(borderRadius: BorderRadius.circular(16), color: Colors.blue[200]),
      child: FlatButton(
        child: Text("Emergency Call!!!",
            textAlign: TextAlign.center, style: TextStyle(fontSize: 18, color: Colors.white)),
        textColor: Colors.white,
        color: Colors.transparent,
        shape: RoundedRectangleBorder(borderRadius: BorderRadius.circular(30.0)),
        onPressed: () => {launch('tel:+${0626519422.toString()}')},
      ),
      margin: EdgeInsets.only(top: 16),
      padding: EdgeInsets.all(12));
}

Container Buttongps() {
  return Container(
      constraints: BoxConstraints.expand(height: 50),
      decoration: BoxDecoration(borderRadius: BorderRadius.circular(16), color: Colors.blue[200]),
      child: FlatButton(
          child: Text("go to GPS", textAlign: TextAlign.center, style: TextStyle(fontSize: 18, colo
          textColor: Colors.white,
          color: Colors.transparent,
          shape: RoundedRectangleBorder(borderRadius: BorderRadius.circular(30.0)),
          onPressed: () {
            Navigator.push(context, MaterialPageRoute(builder: (context) {
              return EditProfileSaler();
            }));
          }),
      margin: EdgeInsets.only(top: 16),
      padding: EdgeInsets.all(12));
}
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

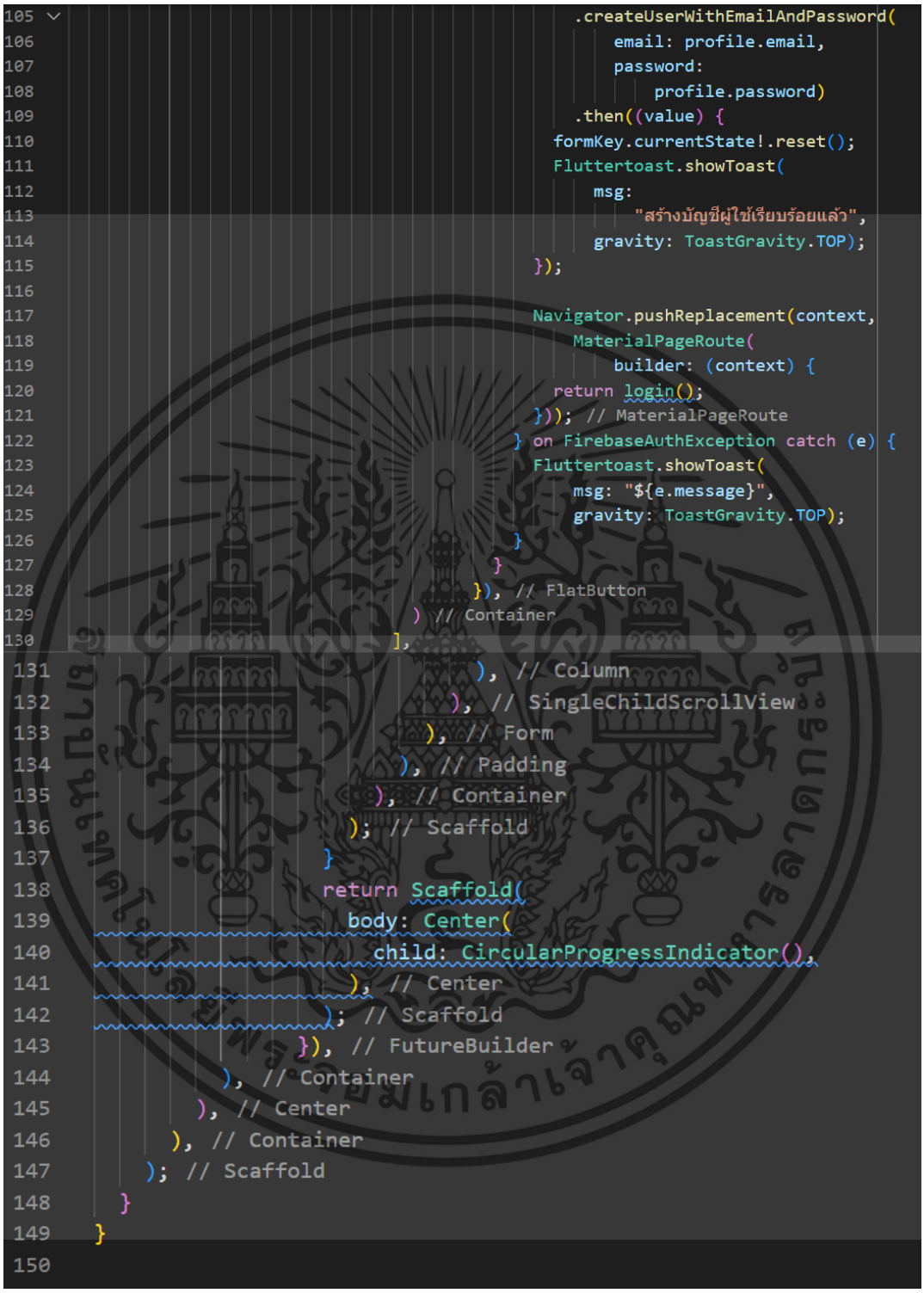
3. account.dart

```dart
import 'package:firebase_auth/firebase_auth.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:flutter/material.dart';
import 'package:flutter_application_more/1.dart';
import 'package:flutter_application_more/profile.dart';
import 'package:flutter_application_more/register.dart';
import 'package:fluttertoast/fluttertoast.dart';
import 'package:form_field_validator/form_field_validator.dart';

class newacc extends StatefulWidget {
  const newacc({Key? key}) : super(key: key);

  @override
  _newaccState createState() => _newaccState();
}

class _newaccState extends State<newacc> {
  final formKey = GlobalKey<FormState>();
  Profile profile = Profile(email: '', password: '');
  final Future<FirebaseApp> firebase = Firebase.initializeApp();

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("New User", style: TextStyle(color: Colors.white)),
      ), // AppBar
      body: Container(
        color: Colors.green[50],
        child: Center(
          child: Container(
            decoration: BoxDecoration(
                borderRadius: BorderRadius.circular(16),
                gradient: LinearGradient(
                    colors: [Colors.yellow.shade100, Colors.green.shade100])), //
            margin: EdgeInsets.all(32),
            padding: EdgeInsets.all(24),
            child: FutureBuilder(
                future: firebase,
                builder: (context, snapshot) {
                  if (snapshot.hasError) {
                    return Scaffold(
                      body: Center(
                        child: Text("${snapshot.error}"),
                      ), // Center
                    ); // Scaffold
                  }
                  if (snapshot.connectionState == ConnectionState.done) {
                    return Scaffold(
                      body: Container(
                        child: Padding(
                          padding: const EdgeInsets.all(8.0),
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
child: Form(
  key: formKey,
  child: SingleChildScrollView(
    child: Column(
      crossAxisAlignment: CrossAxisAlignment.center,
      children: [
        Text("E-mail",
            style: TextStyle(fontSize: 20)), // Text
        TextFormField(
          validator: MultiValidator([
            EmailValidator(
                errorText:
                    "กรุณาป้อนอีเมล์ที่ถูกต้อง"), // EmailVali
            RequiredValidator(
                errorText: "กรุณาป้อนอีเมล์"), // Required
          ]), // MultiValidator
          keyboardType: TextInputType.emailAddress,
          onSaved: (String? email) {
            profile.email = email!;
          },
        ), // TextFormField
        SizedBox(
          height: 15,
        ), // SizedBox
        Text("password",
            style: TextStyle(fontSize: 20)), // Text
        TextFormField(
          validator: RequiredValidator(
              errorText: "กรุณาป้อนรหัสผ่าน"), // Required
          obscureText: true,
          onSaved: (String? password) {
            profile.password = password!;
          },
        ), // TextFormField
        SizedBox(
          height: 20,
        ), // SizedBox
        Container(
          constraints:
              BoxConstraints.expand(height: 50),
          decoration: BoxDecoration(
              borderRadius: BorderRadius.circular(16),
              color: Colors.blue[200]), // BoxDecoratio
          child: FlatButton(
              child: Text("Submit",
                  style: TextStyle(fontSize: 20)), // T
              onPressed: () {
                if (formKey.currentState!
                    .validate()) {
                  formKey.currentState!.save();
                  try {
                    FirebaseAuth.instance
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



```
                .createUserWithEmailAndPassword(
                    email: profile.email,
                    password:
                        profile.password)
                .then((value) {
              formKey.currentState!.reset();
              Fluttertoast.showToast(
                  msg:
                      "สร้างบัญชีผู้ใช้เรียบร้อยแล้ว",
                  gravity: ToastGravity.TOP);
            });

            Navigator.pushReplacement(context,
                MaterialPageRoute(
                    builder: (context) {
                  return login();
                })); // MaterialPageRoute
            } on FirebaseAuthException catch (e) {
              Fluttertoast.showToast(
                  msg: "${e.message}",
                  gravity: ToastGravity.TOP);
            }
          }
        }), // FlatButton
      ) // Container
    ],
                ), // Column
              ), // SingleChildScrollView
            ), // Form
          ), // Padding
        ), // Container
      ); // Scaffold
    }
    return Scaffold(
      body: Center(
        child: CircularProgressIndicator(),
      ), // Center
    ); // Scaffold
  }), // FutureBuilder
), // Container
), // Center
), // Container
); // Scaffold
}
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.Class.dart

```dart
import 'package:firebase_database/firebase_database.dart';
import 'package:flutter/material.dart';
import 'package:flutter_application_more/Data.dart';

class readdata extends StatefulWidget {
  const readdata({Key? key}) : super(key: key);

  @override
  State<readdata> createState() => _readdataState();
}

class _readdataState extends State<readdata> {
  FirebaseDatabase firebaseDatabase = FirebaseDatabase.instance;
  @override
  void initState() {
    // TODO: implement build
    super.initState();
  }

  void readData() async {
    DatabaseReference databaseReference = FirebaseDatabase.instance.reference().child('heart');
    await databaseReference.once().then((DataSnapshot heart) {
      Map<dynamic, dynamic> values = heart.value;
      values.forEach((key, value) {
        print(value['beatavg']);
      });
    });
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("eiei"),
      ), // AppBar
      body: Text('beatavg1234'),
    ); // Scaffold
  }
}

```

5. Data.dart

```dart
class data {
  String beatavgpc;
  String heartpc;

  data({required this.beatavgpc, required this.heartpc});
}

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6. Generated_plugin_registrant.dart

```dart
import 'package:cloud_firestore_web/cloud_firestore_web.dart';
import 'package:firebase_auth_web/firebase_auth_web.dart';
import 'package:firebase_core_web/firebase_core_web.dart';
import 'package:firebase_database_web/firebase_database_web.dart';
import 'package:fluttertoast/fluttertoast_web.dart';
import 'package:geolocator_web/geolocator_web.dart';
import 'package:image_picker_for_web/image_picker_for_web.dart';
import 'package:shared_preferences_web/shared_preferences_web.dart';
import 'package:url_launcher_web/url_launcher_web.dart';

import 'package:flutter_web_plugins/flutter_web_plugins.dart';

// ignore: public_member_api_docs
void registerPlugins(Registrar registrar) {
  FirebaseFirestoreWeb.registerWith(registrar);
  FirebaseAuthWeb.registerWith(registrar);
  FirebaseCoreWeb.registerWith(registrar);
  FirebaseDatabaseWeb.registerWith(registrar);
  FluttertoastWebPlugin.registerWith(registrar);
  GeolocatorPlugin.registerWith(registrar);
  ImagePickerPlugin.registerWith(registrar);
  SharedPreferencesPlugin.registerWith(registrar);
  UrlLauncherPlugin.registerWith(registrar);
  registrar.registerMessageHandler();
}
```

7. GPS.dart

```dart
import 'package:flutter/material.dart';

class gps extends StatefulWidget {
  const gps({Key? key}) : super(key: key);

  @override
  _gpsState createState() => _gpsState();
}

class _gpsState extends State<gps> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
        appBar: AppBar(
          title: Text('Chronic Patient Emergency'),
        ),
        body: Container(
          color: Colors.blue[100],
          child: Center(
            child: Container(
                decoration: BoxDecoration(
                    borderRadius: BorderRadius.circular(16),
                    gradient: LinearGradient(
                        colors: [Colors.white, Colors.blue.shade100])),
                margin: EdgeInsets.all(32),
                padding: EdgeInsets.all(24),
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                child: Column(
                  mainAxisSize: MainAxisSize.min,
                  mainAxisAlignment: MainAxisAlignment.center,
                  children: <Widget>[
                    Image(
                      image: AssetImage(
                        'image/hospital.png',
                      ),
                      width: 150,
                      height: 150,
                    ),
                    Text(
                      'Location',
                      style: TextStyle(fontSize: 25),
                    ),
                  ],
                )),
            ),
          ));
    }
  }

```

8. long.dart

```
import 'dart:convert';
import 'dart:io';
import 'dart:math';

import 'package:cached_network_image/cached_network_image.dart';
import 'package:dio/dio.dart';
import 'package:flutter/material.dart';
import 'package:flutter_application_more/showprogress.dart';
import 'package:geolocator/geolocator.dart';
import 'package:google_maps_flutter/google_maps_flutter.dart';
import 'package:image_picker/image_picker.dart';
import 'package:shared_preferences/shared_preferences.dart';

class EditProfileSaler extends StatefulWidget {
  const EditProfileSaler({Key? key}) : super(key: key);

  @override
  _EditProfileSalerState createState() => _EditProfileSalerState();
}

class _EditProfileSalerState extends State<EditProfileSaler> {
  LatLng? latLng;
  final formKey = GlobalKey<FormState>();
  File? file;

  @override
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```dart
  void initState() {
    // TODO: implement initState
    super.initState();

    findLatLng();
  }

  Future<Null> findLatLng() async {
    Position? position = await findPosition();
    if (position != null) {
      setState(() {
        latLng = LatLng(position.latitude, position.longitude);
        print('lat = ${latLng!.latitude}');
        print('lng = ${latLng!.longitude}');
      });
    }
  }

  Future<Position?> findPosition() async {
    Position? position;
    try {
      position = await Geolocator.getCurrentPosition();
    } catch (e) {
      position = null;
    }
    return position;
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('minimap'),
      ), // AppBar
      body: LayoutBuilder(
        builder: (context, constraints) => GestureDetector(
          onTap: () => FocusScope.of(context).requestFocus(FocusNode()),
          behavior: HitTestBehavior.opaque,
          child: Form(
            key: formKey,
            child: ListView(
              padding: EdgeInsets.all(16),
              children: [
                buildMap(constraints),
              ],
            ), // ListView
          ), // Form
        ), // GestureDetector
      ), // LayoutBuilder
    ); // Scaffold
  }

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Row buildMap(BoxConstraints constraints) {
  return Row(
    mainAxisAlignment: MainAxisAlignment.center,
    children: [
      Container(
        decoration: BoxDecoration(
          border: Border.all(color: Colors.grey),
        ),
        margin: EdgeInsets.symmetric(vertical: 16),
        width: constraints.maxWidth * 0.75,
        height: constraints.maxWidth * 0.5,
        child: latLng == null
            ? ShowProgress()
            : GoogleMap(
                initialCameraPosition: CameraPosition(
                  target: latLng!,
                  zoom: 16,
                ),
                onMapCreated: (controller) {},
                markers: <Marker>[
                  Marker(
                    markerId: MarkerId('id'),
                    position: latLng!,
                    infoWindow: InfoWindow(
                        title: 'You Location',
                        snippet:
                            'lat = ${latLng!.latitude}, lng = ${latLng!.longitude}'),
                  ),
                ].toSet(),
              ),
      ),
    ],
  );
  }
}

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

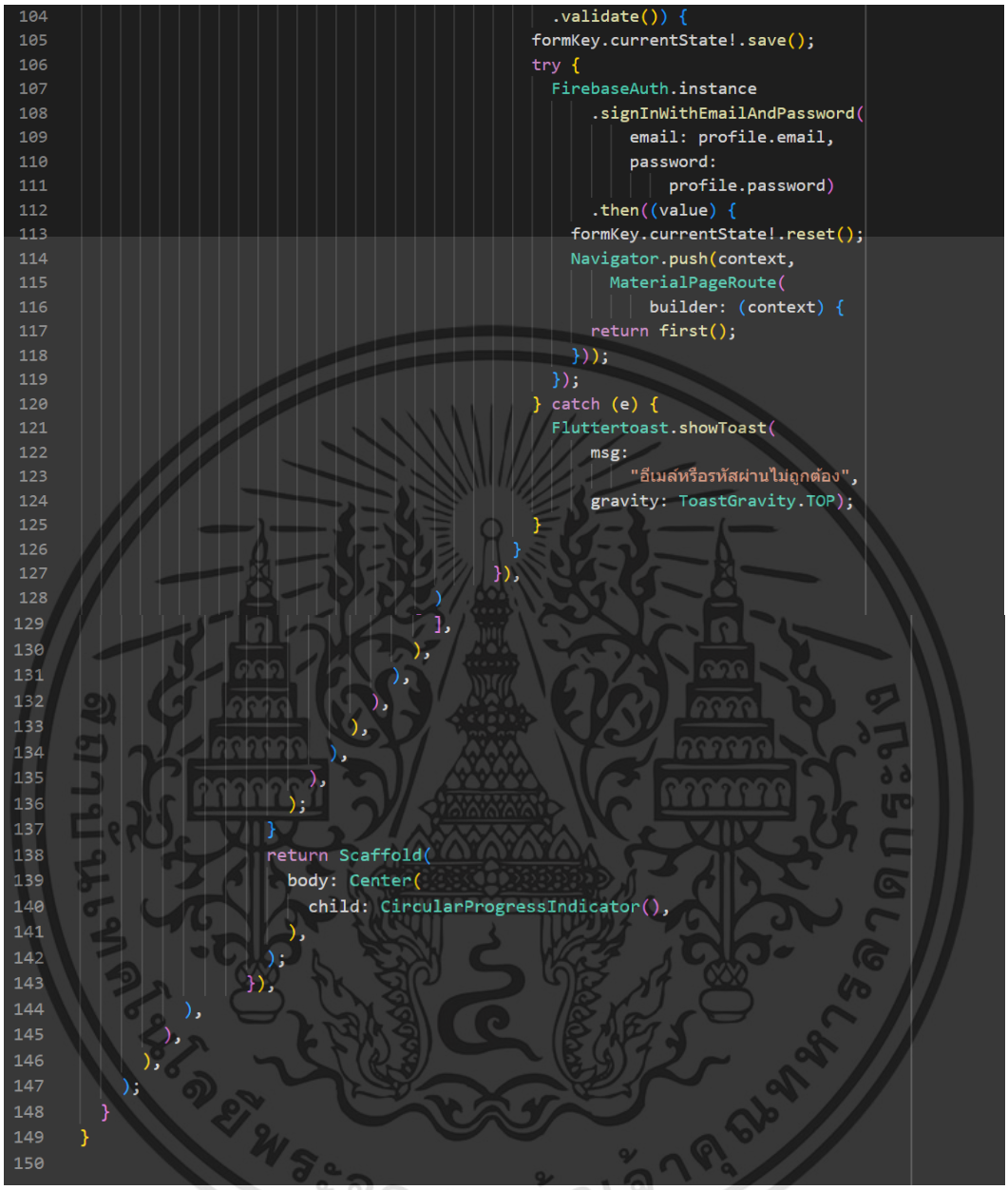
9. oldaccount.dart

```dart
import 'package:firebase_auth/firebase_auth.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:flutter/material.dart';
import 'package:flutter_application_more/1.dart';
import 'package:flutter_application_more/profile.dart';
import 'package:flutter_application_more/register.dart';
import 'package:fluttertoast/fluttertoast.dart';
import 'package:form_field_validator/form_field_validator.dart';

class oldacc extends StatefulWidget {
  const oldacc({Key? key}) : super(key: key);

  @override
  _oldaccState createState() => _oldaccState();
}

class _oldaccState extends State<oldacc> {
  final formKey = GlobalKey<FormState>();
  Profile profile = Profile(email: '', password: '');
  final Future<FirebaseApp> firebase = Firebase.initializeApp();

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("Chronic Patient Wmergency",
            style: TextStyle(color: Colors.white)),
      ),
      body: Container(
        color: Colors.green[50],
        child: Center(
          child: Container(
            decoration: BoxDecoration(
                borderRadius: BorderRadius.circular(16),
                gradient: LinearGradient(
                    colors: [Colors.yellow.shade100, Colors.green.shade100])),
            margin: EdgeInsets.all(32),
            padding: EdgeInsets.all(24),
            child: FutureBuilder(
                future: firebase,
                builder: (context, snapshot) {
                  if (snapshot.hasError) {
                    return Scaffold(
                      body: Center(
                        child: Text("${snapshot.error}"),
                      ),
                    );
                  }
                  if (snapshot.connectionState == ConnectionState.done) {
                    return Scaffold(
                      body: SingleChildScrollView(
                        child: Container(
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
child: Padding(
  padding: const EdgeInsets.all(8.0),
  child: Form(
    key: formKey,
    child: SingleChildScrollView(
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          Text("E-mail",
              style: TextStyle(fontSize: 20)),
          TextFormField(
            validator: MultiValidator([
              EmailValidator(
                  errorText:
                      "กรุณาป้อนอีเมล์ที่ถูกต้อง"),
              RequiredValidator(
                  errorText: "กรุณาป้อนอีเมล์"),
            ]),
            keyboardType: TextInputType.emailAddress,
            onSaved: (String? email) {
              profile.email = email!;
            },
          ),
          SizedBox(
            height: 15,
            ),
            Text("password",
                style: TextStyle(fontSize: 20)),
            TextFormField(
              validator: RequiredValidator(
                  errorText: "กรุณาป้อนรหัสผ่าน"),
              obscureText: true,
              onSaved: (String? password) {
                profile.password = password!;
              },
            ),
            SizedBox(
              height: 20,
            ),
            Container(
              constraints:
                  BoxConstraints.expand(height: 50),
              decoration: BoxDecoration(
                  borderRadius:
                      BorderRadius.circular(16),
                  color: Colors.blue[200]),
              child: ElevatedButton(
                  child: Text("เข้าสู่ระบบ",
                      style: TextStyle(fontSize: 16)),
                  onPressed: () {
                    if (formKey.currentState!
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



```
                .validate()) {
              formKey.currentState!.save();
              try {
                FirebaseAuth.instance
                    .signInWithEmailAndPassword(
                        email: profile.email,
                        password:
                            profile.password)
                    .then((value) {
                  formKey.currentState!.reset();
                  Navigator.push(context,
                      MaterialPageRoute(
                          builder: (context) {
                        return first();
                      }));
                });
              } catch (e) {
                Fluttertoast.showToast(
                    msg:
                        "อีเมล์หรือรหัสผ่านไม่ถูกต้อง",
                    gravity: ToastGravity.TOP);
              }
            }
          }),
        )
      ],
    ),
  ),
),
),
),
),
),
),
);
}
return Scaffold(
  body: Center(
    child: CircularProgressIndicator(),
  ),
);
}),
),
),
),
);
}
}
```

10. profile.dart

```
class Profile {
  String email;
  String password;

  Profile({required this.email, required this.password});
}

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

11. register.dart

```dart
import 'package:flutter/material.dart';
import 'package:flutter_application_more/account.dart';
import 'package:flutter_application_more/oldaccount.dart';

class login extends StatelessWidget {
  const login({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Scaffold(
        appBar: AppBar(
          title: Text("Chronic Patient Emergency",
              style: TextStyle(color: Colors.white)),
        ),
        body: Container(
            color: Colors.green[50],
            child: Center(
              child: Container(
                  decoration: BoxDecoration(
                    borderRadius: BorderRadius.circular(16),
                    gradient: LinearGradient(colors: [
                      Colors.yellow.shade100,
                      Colors.green.shade100
                    ])),
                  margin: EdgeInsets.all(32),
                  padding: EdgeInsets.all(24),
                  child: Column(
                    mainAxisSize: MainAxisSize.min,
                    mainAxisAlignment: MainAxisAlignment.center,
                    children: <Widget>[
                      Image(
                        image: AssetImage('image/hospital.png'),
                        height: 120,
                        width: 120,
                      ),
                      buildButtonSignin(context),
                      buildButtonregister(context),
                    ],
                  )),
            )));
  }
}

Container buildButtonregister(BuildContext context) {
  return Container(
      constraints: BoxConstraints.expand(height: 50),
      decoration: BoxDecoration(
          borderRadius: BorderRadius.circular(16), color: Colors.blue[300]),
      child: FlatButton(
        child: Text("Sign Up",
            textAlign: TextAlign.center,
            style: TextStyle(fontSize: 18, color: Colors.white)),
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            textColor: Colors.white,
            color: Colors.transparent,
            shape:
                RoundedRectangleBorder(borderRadius: BorderRadius.circular(30.0)),
            onPressed: () => {
              Navigator.push(
                  context, MaterialPageRoute(builder: (context) => newacc()))
            },
          ),
          margin: EdgeInsets.only(top: 16),
          padding: EdgeInsets.all(12));
    }

    Container buildButtonSignin(BuildContext context) {
      return Container(
          constraints: BoxConstraints.expand(height: 50),
          decoration: BoxDecoration(
              borderRadius: BorderRadius.circular(16), color: Colors.blue[300]),
          child: FlatButton(
            child: Text("Sign In",
                textAlign: TextAlign.center,
                style: TextStyle(fontSize: 18, color: Colors.white)),
            textColor: Colors.white,
            color: Colors.transparent,
            shape:
                RoundedRectangleBorder(borderRadius: BorderRadius.circular(30.0)),
                RoundedRectangleBorder(borderRadius: BorderRadius.circular(30.0)),
            onPressed: () => {
              Navigator.push(
                  context, MaterialPageRoute(builder: (context) => oldacc()))
            },
          ), // FlatButton
          margin: EdgeInsets.only(top: 16),
          padding: EdgeInsets.all(12)); // Container
    }

```

12. showprogress.dart

```
import 'package:flutter/material.dart';

class ShowProgress extends StatelessWidget {
  const ShowProgress({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Center(
      child: CircularProgressIndicator(),
    );
  }
}

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้