รายงานสหกิจศึกษาฉบับสมบูรณ์

ระบบลงเวลาของพนักงานด้วยไลน์บีคอน

Employees Time Tracking System with LINE Beacon

นางสาววรวรรณ นาสมรูป

สาขาวิชาวิศวกรรมสารสนเทศ ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2562

ชื่อโครงงานสหกิจศึกษา ระบบลงเวลาของพนักงานด้วยไลน์บีคอน

ชื่อ-สกุล นักศึกษา นางสาววรวรรณ นาสมรูป

คณะ วิศวกรรมศาสตร์ **ภาควิชา** วิศวกรรมคอมพิวเตอร์ **สาขาวิชา** วิศวกรรมสารสนเทศ

ชื่อ-สกุล อาจารย์นิเทศ ผศ. มยุรี เลิศเวชกุล

ชื่อ-สกุล ผู้นิเทศงาน คุณจิตรลดา สุปรียาพร, คุณธนวัฒน์ กิตติชัยการ

ชื่อสถานประกอบการ บริษัท นิปปอน เทเลกราฟ แอนด์ เทเลโฟน (ประเทศไทย) จำกัด

บทคัดย่อ

ระบบลงเวลาของพนักงานด้วยไลน์บีคอน เป็นโครงงานที่จัดทำขึ้นเพื่อนำมาใช้ในการแก้ปัญหาของระบบตอกบัตรแบบเก่า เพื่อเพิ่มความแม่นยำ ความสะดวกสบาย และความรวดเร็ว โดยสามารถนำข้อมูลที่ได้ไปใช้ในการบริหารจัดการให้เกิดประสิทธิผลสูงสุด

โครงงานระบบลงเวลาของพนักงานด้วยไลน์บีคอน เป็นระบบที่พัฒนาขึ้นด้วย Node JS ซึ่งเป็นภาษาที่ใช้ในการเขียนโปรแกรม ใช้ Raspberry Pi เป็นอุปกรณ์ที่ในการกระจายสัญญาณบลูทูธ โดยมี MongoDB เป็นฐานข้อมูล นอกจากนั้นยังมีการนำ Elasticsearch และ Kibana มาใช้ในการสร้าง Dashboard ทั้งนี้การดำเนินงานจำเป็นต้องใช้ความรู้ทั้งทางด้านซอฟต์แวร์ ฮาร์ดแวร์ และความเข้าใจระบบการทำงานต่าง ๆ ในการวิเคราะห์การทำงานและแก้ไขปัญหาที่เกิดขึ้น

คำสำคัญ: ระบบลงเวลาของพนักงาน, ไลน์บีคอน, ราสเบอร์รี่พาย

Co-operative Title Employee Time Tracking System with LINE Beacon
Student Intern Name Miss.Worawan Nasomroop
Faculty Engineering **Department** Computer Engineering **Major** Information Engineering
Advisor Name Asst. Prof. Mayuree Lertwatechakul
Mentor Name Miss.Jitlada Supreeyaporn, Mr.Thanawat Kitichaikarn
Company Nippon Telegraph and Telephone Corporation

ABSTRACT

Employee time tracking system with LINE Beacon was created to solve the problems of the old clocking system. The objective of developing the employee time tracking system is to gain the accuracy of clocking system. It will be more convenience for the employee to check their working hours and places as well as the manager. Thus, more efficiency and productivity could be achieved.

The system uses Raspberry Pi as Bluetooth transmitter. Node JS was chosen as the programing language cooperate with MongoDB for data operation. Kibana was a tool selected for visualizing data and create dashboards. Developing the system, it is required to have background knowledge in software development and understanding of hardware and system as to analyze and solve the problem.

Keywords: Employee time tracking system, LINE Beacon, Raspberry PI

กิตติกรรมประกาศ

ข้าพเจ้าได้รับผิดชอบและปฏิบัติหน้าที่ในบริษัท นิปปอน เทเลกราฟ แอนด์ เทเลโฟน (ประเทศไทย) จำกัด ระหว่างวันที่ 5 สิงหาคม ถึงวันที่ 22 พฤศจิกายน พ.ศ.2562 ในโครงการวิชาสหกิจศึกษาที่ทางคณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง และบริษัทฯร่วมมือกันจัดตั้งขึ้นในหัวข้อ ระบบลงเวลาของพนักงานด้วยไลน์บีคอน ซึ่งข้าพเจ้าได้รับความรู้ ความเข้าใจ และประสบการณ์ในการทำงานที่เป็นประโยชน์อย่างมาก อีกทั้งการดูแลและการช่วยเหลือต่าง ๆ ตลอดการทำงาน โดยการปฏิบัติงานสหกิจศึกษาในครั้งนี้สำเร็จลุล่วงได้เพราะมีการชี้แนะและได้รับความร่วมมือจากบุคคลต่าง ๆ ดังต่อไปนี้

พนักงานแผนก Consulting-Application Integration

- คุณวรพงศ์ เล็กอุทัยพรรณ
- คุณองอาจ ปรีดาภิรัตน์
- คุณจิตรลดา สุปรียาพร
- คุณภาณุวัชร์ บุญยะทรัพย์
- คุณธนวัฒน์ กิตติชัยการ
- คุณพีรกฤต สกุลสายทองคำ

ข้าพเจ้าขอขอบคุณอาจารย์ที่ปรึกษา ผศ.มยุรี เลิศเวชกุล ที่คอยให้คำแนะนำ คำปรึกษา คอยรับฟังและช่วยเหลือปัญหาต่าง ๆ ในการทำโครงการครั้งนี้ และท้ายที่สุดข้าพเจ้าขอขอบคุณครอบครัวที่คอยให้กำลังใจที่ดีแก่ข้าพเจ้าเสมอมาทำให้ปริญญานิพนธ์ฉบับนี้ให้สำเร็จลุล่วงไปได้ด้วยดี

สารบัญ

สารบัญ (ต่อ)

บทที่ **หน้า**

สารบัญ (ต่อ)

สารบัญตาราง

ตารางที่ **หน้า**

สารบัญภาพ

รูปที่ **หน้า**

สารบัญภาพ (ต่อ)

รูปที่ **หน้า**

สารบัญภาพ (ต่อ)

รูปที่ **หน้า**

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญ

เนื่องจาก บริษัท นิปปอน เทเลกราฟ แอนด์ เทเลโฟน (ประเทศไทย) จำกัด ได้จัดโครงการสหกิจศึกษาร่วมกับ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง โดยในส่วนงานของแผนก Consulting-Application Integration ได้มีความต้องการพัฒนาระบบลงเวลาของพนักงานด้วยไลน์บีคอน เนื่องจากระบบเดิมมีความซับซ้อนในการใช้งาน และไม่สามารถตรวจสอบช่วงเวลาเข้า-ออกงานได้ ดังนั้นจึงได้มีการพัฒนาระบบลงเวลาของพนักงานด้วยไลน์บีคอน เพื่อใช้ในการตรวจสอบจำนวนพนักงาน เวลาเข้า-ออกของพนักงาน และมอบหมายงานให้นักศึกษาทำการศึกษา และหาความรู้เกี่ยวกับไลน์บีคอน (Line Beacon) ซึ่งเป็นเทคโนโลยีที่ใช้ในการส่งข้อมูลผ่านสัญญาณบลูทูธ รวมถึงวิธีการติดตั้งและการออกแบบ เพื่อให้นักศึกษามีความรู้และสามารถออกแบบและติดตั้งระบบ

1.2 วัตถุประสงค์ของโครงงาน

1) เพื่อทำการตรวจสอบจำนวนพนักงาน เวลาเข้า-ออกงานในแต่ละบุคคล
2) เพื่อเก็บข้อมูลการทำงานของพนักงานในแต่ละวัน
3) เพื่อใช้ในการติดตามความก้าวหน้าของโครงงาน
4) เพื่อเพิ่มความสะดวกสบายแก่พนักงานในการลงเวลา

1.3 ขอบเขตของโครงงาน

1) ระบบสามารถรองรับผู้ใช้ได้เพียง 20 คนเท่านั้น
2) ในการลงเวลาผู้ใช้ต้องอยู่ภายในพื้นที่ที่ให้บริการโดยไลน์บีคอนเท่านั้น
3) รัศมีของพื้นที่ที่ให้บริการขึ้นอยู่กับสถานที่และอุปกรณ์ที่ใช้ในการกระจายสัญญาณ

1.4 วิธีการดำเนินโครงการ

1) ศึกษาขั้นตอนและทำความเข้าใจการทำงานของไลน์บีคอน
2) ศึกษาการติดตั้งอุปกรณ์ราสเบอร์รี่พาย (Raspberry Pi Zero W) และเทคโนโลยีการส่งข้อมูลด้วยสัญญาณบลูทูธ
3) กำหนดเป้าหมายและวางแผนการทำงาน
4) ออกแบบโครงสร้างและการทำงานของระบบ
5) พัฒนาระบบ
6) ทดสอบระบบ และแก้ไขปัญหาที่เกิดขึ้น

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1) พนักงานสามารถลงเวลาได้สะดวกและรวดเร็วมากยิ่งขึ้น
2) ผู้จัดการสามารถตรวจสอบและดูความคืบหน้าในการทำงานของพนักงานในแต่ละวันได้
3) สามารถนำความรู้ในการใช้งานเครื่องมือสำหรับพัฒนาระบบไปใช้งานได้ในอนาคต

บทที่ 2
ทฤษฎีที่เกี่ยวข้อง

2.1 Visual Studio Code

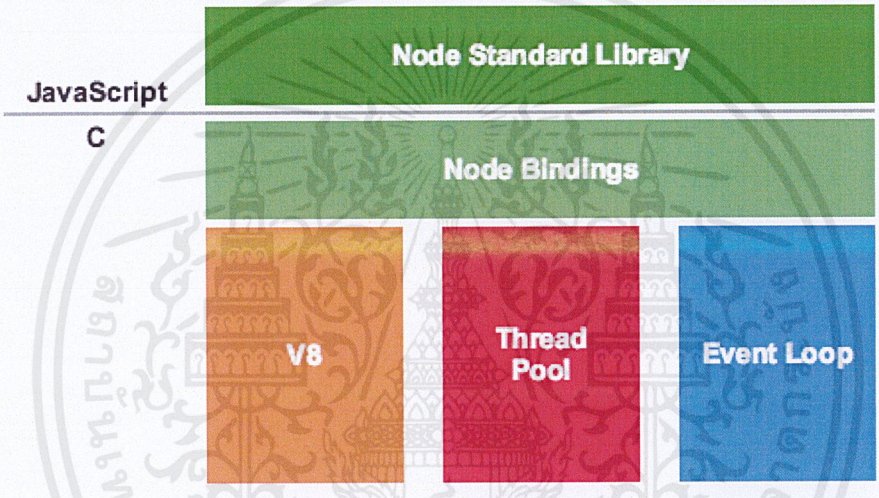
Visual Studio Code เป็นโปรแกรมที่ใช้ในการเขียนและปรับแต่งซอร์สโค้ด (Source-code editor) ในรูปแบบของโอเพนซอร์ส (Open Source) พัฒนาโดยค่ายไมโครซอฟท์คอร์ปอเรชั่น ซึ่ง Visual Studio Code นั้นสามารถใช้งานข้ามแพลตฟอร์มและรองรับการทำงานทั้งWindows, macOS และ Linux สนับสนุนทั้งภาษา JavaScript, TypeScript และ Node.js การใช้งานง่ายไม่ซับซ้อนสามารถเชื่อมต่อกับ Git ได้และยังมีเครื่องมือเสริมอีกมากมายให้เลือกใช้งาน

รูปที่ 2.1 สัญลักษณ์ของ Visual Studio Code

(ที่มา : https://upload.wikimedia.org/wikipedia/commons/thumb/9/9a/Visual_Studio_Code_1.35_icon.svg/1200px-Visual_Studio_Code_1.35_icon.svg.png)

2.2 Node.js

Node.js คือ Cross Platform Runtime Environment สำหรับฝั่งเซิร์ฟเวอร์ (Server) ถูกสร้างบน Chrome's V8 JavaScript Engine มีสถาปัตยกรรมแสดงดังรูปที่ 2.2 เขียนด้วยภาษา JavaScript โดยมี JavaScript Runtime Environment ที่สามารถใช้งาน JavaScript นอกเว็บบราวเซอร์ได้ โดย Node.js จะใช้รูปแบบการทำงานแบบ Event-Driven, Asynchronous input/output (I/O) เพื่อลด overhead และเพิ่มประสิทธิภาพในการทำงาน



รูปที่ 2.2 สถาปัตยกรรมของ Node JS

(ที่มา : http://khan.io/2015/02/25/the-event-loop-and-non-blocking-io-in-node-js/)

2.3 Express.js

Express.js เป็นเว็บแอปพลิเคชันเฟรมเวิร์ก (web application framework) สำหรับ Node.js ซึ่งใช้ในการสร้างเว็บไซต์หรือใช้ทำ Single Page Application โดยใช้ API Server รับ-ส่ง request และ response ผ่าน RESTful โดย Express.js และ Node.js ต่างก็ใช้ภาษา JavaScript ในการพัฒนา

2.3.1 การจัดเส้นทาง (Routing)

Routing คือกำหนดเส้นทางการเข้าถึง (Endpoint) ว่าจะให้แอปพลิเคชันตอบสนองต่อ client request อย่างไร โดยเรียกผ่าน HTTP request method เช่น GET, POST และอื่น ๆ

รูปแบบการกำหนด routing

app.METHOD (PATH, HANDLER)

- app เป็นinstanceของ express
- METHOD เป็น HTTP request method พิมพ์ด้วยตัวพิมพ์เล็ก
- PATH เป็นเส้นทางบนเซิร์ฟเวอร์
- HANDLER เป็นฟังก์ชันที่จะทำงานเมื่อเส้นทางถูกจับคู่

a) Route methods “เป็น method ที่แปลงมาจาก HTTP method นำไปผูกเข้ากับ app ซึ่งเป็น instance ของ express” โดย Http Request Method มี 4 Methods ที่สำคัญ คือ GET PUT POST และ DELETE

ตัวอย่าง: การกำหนดเส้นทางด้วย GET และ POST method

```
app.get('/', function (req, res) {
res.send('GET request to the homepage')
})
app.post('/', function (req, res) {
res.send('POST request to the homepage')
})
```

b) Route paths ถูกนำมาใช้ร่วมกับ request method ใช้ในการกำหนด endpoint ที่จะทำการร้องขอ โดย Route paths สามารถเป็นได้ทั้ง string, string pattern, regular expression

ตัวอย่าง: การกำหนด path ในรูปแบบของ String

```
app.get('/', function (req, res) {
  res.send('GET request to the homepage')
})
app.get('/about', function (req, res) {
  res.send('about')
})
```

ตัวอย่าง: การกำหนด path ในรูปแบบของ String patterns

```
app.get('/ab?cd', function (req, res) {
  res.send('ab?cd')
})
app.get('/ab+cd', function (req, res) {
  res.send('ab+cd')

})
```

ตัวอย่าง: การกำหนด path ในรูปแบบของ Regular expression

```
app.get(/a/, function (req, res) {
  res.send('/a/')
})
app.get(/.*fly$/, function (req, res) {
  res.send('/.*fly$/')
})
```

c) Route parameters หรือ URI segment ใช้สำหรับเก็บค่าที่ต้องการระบุในตำแหน่งต่าง ๆ ของ URL สามารถเรียกค่าเหล่านี้ผ่าน req.params โดยชื่อของ parameter จะกำหนดในรูปแบบของ key เรียงตามลำดับดังตัวอย่างถัดไป

ตัวอย่าง: การกำหนด Route parameter ในรูปแบบ key

```
Route path: /users/:userId/books/:bookId
Request URL: http://localhost:3000/users/34/books/8989
req.params: { "userId": "34", "bookId": "8989" }
app.get('/users/:userId/books/:bookId', function (req, res) {
  res.send(req.params)
})
```

ชื่อของ parameter ต้องกำหนดในรูปแบบของตัวอักษรภาษาอังกฤษ ตัวเลข สัญลักษณ์ underscore ตามรูปแบบ (A-Z, a-z, 0-9, _) และสามารถใช้ hyphen (-) และ dot (.) เพื่อวัตถุประสงค์พิเศษได้ เช่น การเชื่อม parameter 2 ตัวเข้าด้วยกัน

ตัวอย่าง: การใช้ – หรือ . ในการเชื่อม parameter

```
Route path: /flights/:from-:to
Request URL: http://localhost:3000/flights/LAX-SFO
req.params: { "from": "LAX", "to": "SFO" }
Route path: /plantae/:genus.:species
Request URL: http://localhost:3000/plantae/Prunus.persica
req.params: { "genus": "Prunus", "species": "persica" }
```

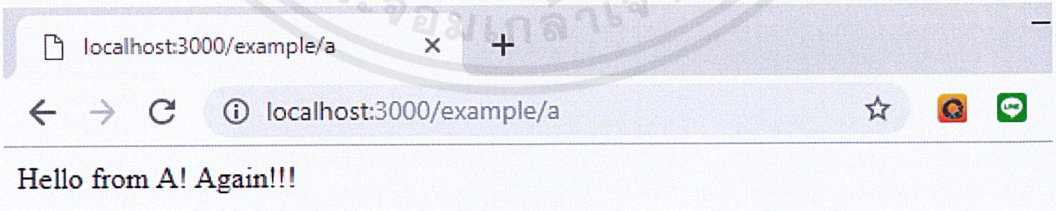
d) Route handlers เป็นฟังก์ชันที่จะถูกเรียกใช้หากตรงตามเงื่อนไข หรือ path ที่ผู้ใช้ทำการร้องขอ request ตรงกับ path ที่ถูกกำหนดไว้ ซึ่งจะทำหน้าที่เป็นเหมือน middleware ใช้ในการจัดการ request

ตัวอย่าง: Route handler แบบฟังก์ชันเดียว

```
app.get('/example/b', function (req, res) {
  res.send('Hello from B!')
}
```

ตัวอย่าง: Route handler แบบหลายฟังก์ชัน

```
app.get('/example/a, function (req, res, next) {
  req.myobj = 'Hello from A!'
  next()
}, function (req, res) {
  req.myobj += ' Again!!!'
  res.send(req.myobj)
})
```



รูปที่ 2.3 ผลลัพธ์การเขียน route handlers ในรูปแบบหลายฟังก์ชัน

(ที่มา : https://i.imgur.com/iwh9r2R.png)

ตัวอย่าง: Route handler แบบ array ของฟังก์ชัน

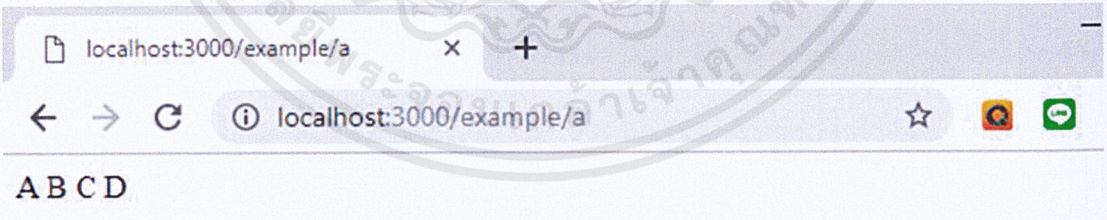
```
var cb0 = function (req, res, next) {
  req.myobj = 'A'
  next()
var cb1 = function (req, res, next) {
  req.myobj += ' B'
  next()
var cb2 = function (req, res) {
    req.myobj += ' C'
    res.send(req.myobj);
app.get('/example/a', [cb0, cb1, cb2])
```

รูปที่ 2.4 ผลลัพธ์การเขียน route handlers ในรูปแบบ array

(ที่มา : https://i.imgur.com/n1cRX2L.png)

ตัวอย่าง: Route handler แบบผสม

```
var cb0 = function (req, res, next) {
  req.myobj = 'A'
  next()
}
var cb1 = function (req, res, next) {
  req.myobj += ' B'
  next()
}
app.get('/example/a', [cb0, cb1]) function (req, res, next) {
    req.myobj += ' C'
    res.send(req.myobj);
} function (req, res, ){
    req.myobj += ' D'
    res. send (req.myobj);
});
```



รูปที่ 2.5 ผลลัพธ์การเขียน route handlers ในรูปแบบผสม

(ที่มา : https://i.imgur.com/54uqmHB.png)

e) Respond methods เป็นส่วนที่ใช้ในการส่ง response ไปยังผู้ใช้ และใช้ในการจบการทำงานของ request-response หากไม่มีการเรียกใช้ method นี้ การร้องขอ request อาจหยุดชะงักได้

ตารางที่ 2.1 คำสั่ง response method

คำสั่ง	ความหมาย
res.download()	ดาวน์โหลดไฟล์ที่ต้องการ
res.end()	จบการทำงาน
res.json()	ส่งกลับข้อมูลในรูปแบบ JSON string
res.jsonp()	ส่งกลับข้อมูลในรูปแบบ JSON string รองรับ JSONP
res.redirect()	ลิ้งค์หรือเปลี่ยน path ไปยัง url ที่กำหนด เช่น res.redirect('http://example.com');
res.render()	สร้าง HTML string จาก view template แล้วส่งออกมาแสดง
res.send()	ส่งข้อมูลกลับมาแสดง โดยสามารถเป็น Buffer Object , String ข้อความ , Object และ Array
res.sendFile()	ส่งไฟล์ หรือนำข้อมูลไฟล์มาแสดง
res.sendStatus()	กำหนด status code และส่งข้อความแจ้งเช่น res.sendStatus(200); // มีค่าเท่ากับใช้คำสั่ง res.status(200).send('OK')

f) app.route() เพื่อลดความซับซ้อนและความพิดผลาดจากการพิมพ์ที่เกิดจากการกำหนด route path ซ้ำ ๆ ให้กับแต่ละ method นั้น สามารถใช้ app.route() ใช้ในการจัดการหรือสร้างการเชื่อมต่อให้กับ route เป็นฟังก์ชันต่อ ๆ กัน โดยการกำหนด route path แล้วจึงกำหนด method แยกย่อย

ตัวอย่าง: การใช้งาน app.route()

```
app.route('/book')
  .get(function (req, res) {
    res.send('Get a random book')
  })
  .post(function (req, res) {
    res.send('Add a book')
  })
  .put(function (req, res) {
    res.send('Update the book')
  })
```

g) express.Router ใช้ในการจับคู่ API และ middleware เข้าด้วยกัน และยังสามารถนำมาประกอบเป็นโมดูลใหญ่ๆเพื่อนำไปใช้งานได้

ตัวอย่าง: การใช้งาน express.Router

```
var express = require('express')
var router = express.Router()
// middleware that is specific to this router
router.use(function timeLog (req, res, next) {
  console.log('Time: ', Date.now())
  next()
})
```

```
// define the home page route
router.get('/', function (req, res) {
  res.send('Birds home page')
})
// define the about route
router.get('/about', function (req, res) {
  res.send('About birds')
})
module.exports = router
```

2.3.2 Middleware

ฟังก์ชัน Middleware คือฟังก์ชันที่มีการเข้าถึง request-response และสามารถเรียกใช้ middleware ในลำดับถัดไปด้วย next function

Middleware มีหน้าที่ดังต่อไปนี้

- สั่งให้โปรแกรมทำงานต่างคำสั่งต่าง ๆ
- แก้ไขเปลี่ยนแปลง request และ response
- สิ้นสุดการทำงานของ request-response
- เรียกใช้ middleware ลำดับถัดไป

ประเภทของ Middleware ฟังก์ชัน

a) Application-level middleware สามารถผูกเข้ากับ instance ของ app ได้โดยใช้ app.use() และ app.METHOD() โดย METHOD คือ HTTP request method ใช้จัดการฟังก์ชัน เช่น GET, PUT, POST เขียนอยู่ในรูปของตัวพิมพ์เล็ก

b) Router-level middleware ทำหน้าที่เหมือนกับ Application-level middleware แต่ผูกเข้ากับ instance ของ express.Router()

c) Error-handling middleware การกำหนดฟังก์ชันมีรูปแบบคล้ายกับฟังก์ชันอื่น ๆ มีข้อแตกต่างคือในฟังก์ชันจะมี 4 arguments คือ (err, req, res, next) ในขณะที่ฟังก์ชันอื่นมีเพียง 3 arguments

d) Built-in middleware ในเวอร์ชัน 4.x หรือสูงกว่านั้นจะไม่ขึ้นอยู่กับการเชื่อมต่อ

e) Third-party middleware ใช้ในการเพิ่มฟังก์ชันการทำงานต่าง ๆ ซึ่งถูกพัฒนาโดยผู้อื่นไม่ได้เป็นของ express

2.4 mongoDB

mongoDB คือ cross-platform document-oriented database พัฒนาโดย mongoDB Inc. เป็นฐานข้อมูลแบบ NoSQL ข้อมูลจะถูกจัดในรูปแบบของ key-value ลักษณะคล้าย JSON ไว้ใน collection โดยที่ไม่จำเป็นจะต้องมี schema เหมือนกัน ข้อมูล document ที่ถูกเก็บใน collection จะประกอบด้วย _id ทำหน้าที่เปรียบเสมือนเป็น primary key อยู่ด้วย

```
{
  firstName: "Jamie",
  lastName: "Munro"
}
```

รูปที่ 2.6 สัญลักษณ์ของ mongoDB

(ที่มา : https://worldvectorlogo.com/logo/mongodb)

2.5 Kibana

Kibana เป็นแพลตฟอร์มที่ใช้ในการวิเคราะห์และสร้าง visualize ถูกออกแบบมาให้ทำงานร่วมกับ Elasticsearch โดย Kibana ยังสามารใช้ในการค้นหา ดู ปฏิสัมพันธ์กับข้อมูลที่ถูกเก็บไว้ภายใน Elasticsearch และสามารถแสดงผลออกมาในรูปแบบต่าง ๆ เช่น กราฟ ตาราง แผนที่และอื่น ๆ

รูปที่ 2.7 สัญลักษณ์ของ Kibana

(ที่มา : https://os2.eu/sites/default/files/oss-projects-logo/kibana-logo-color-v.png)

2.6 Elasticsearch

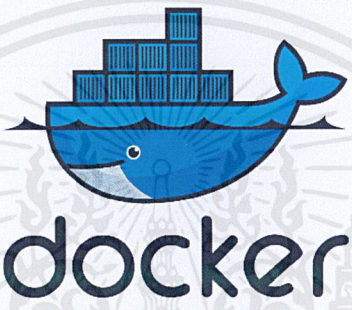
Elasticsearch คือ search และ analytics engine มีความสามารถในการค้นหาได้อย่างรวดเร็ว เมื่อเทียบกับข้อมูลที่มีจำนวนมหาศาล เพราะข้อมูลทั้งหมดจะถูกทำเป็น index ไว้ และยังสามารถติดต่อกับ data store ได้โดยผ่าน RESTful API

รูปที่ 2.8 สัญลักษณ์ของ Elasticsearch

(ที่มา : https://oliverveits.files.wordpress.com/2016/11/elasticsearch_logo.png)

2.7 Docker

Docker คือ platform ที่ใช้ในการจำลองสภาพแวดล้อมขึ้นมาบนเครื่อง Server มีการทำงานคล้ายกับ Virtual Machine โดย Docker นั้นจะใช้ Container ในการจำลองระบบปฏิบัติการของเซิร์ฟเวอร์โดยไม่ต้องมีส่วนของ OS (Operating system) ไม่เหมือนกับ Virtual Machine ที่ต้องมี Hypervisor เพื่อใช้ในการลง OS



รูปที่ 2.9 สัญลักษณ์ของ Docker

(ที่มา : https://cdn.shadowandy.net/wp/wp-content/uploads/docker.png)

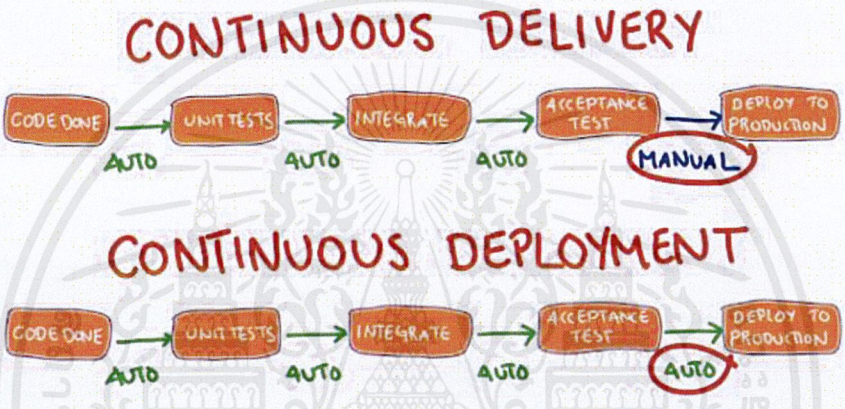
2.8 Jenkins

Jenkins เป็น automation tool พัฒนาด้วยภาษา Java มาพร้อมกับปลั๊กอิน (Plug in) ที่ถูกสร้างขึ้นเพื่อใช้ในการทำ CI/CD ทำให้ง่ายต่อการ integrate การเปลี่ยนแปลงต่าง ๆ

a) Continuous Integration (CI) เป็นหลักการที่นักพัฒนาควรที่จะปฏิบัติ คือการรวมการเปลี่ยนแปลงของโค้ดไปยัง Version Control System โดยในการรวมกันนั้นจะต้องมีการทำ automate test เพื่อที่จะได้พบข้อผิดพลาดและแก้ไขได้ทันท่วงที

b) Continuous Delivery (CD) คือ การทำงานต่าง ๆ ไม่ว่าจะเป็นการ compile, build หรือการทดสอบโดยอัตโนมัติ ไม่มีการแทรกแซงจากมนุษย์ แต่ในส่วนของขั้นตอนการ deployment ขึ้น production นั้นเป็นการทำงานแบบ manual หรือต้องได้รับการอนุญาตจากผู้พัฒนา เช่นการคลิกปุ่มต่าง ๆ

c) Continuous Deployment (CD) มีความแตกต่างจาก Continuous Delivery คือในทุก ๆ กระบวนการแม้แต่การ deployment ขึ้น production นั้น จะทำงานโดยอัตโนมัติ



รูปที่ 2.10 ความแตกต่างของ Continuous Delivery และ Continuous Delivery

(ที่มา : https://sdtimes.com/automation/guest-view-continuous-delivery-vs-continuous-deployment-whats-difference/)

บทที่ 3
ขั้นตอนการดำเนินงาน

3.1 ศึกษาการทำงานของ LINE official account, LINE Beacon, Messaging API และ LINE Front-end Framework

บัญชีทางการ หรือไลน์โอเอ หรือ LINE OA (LINE Official Account) เป็นบัญชีสำหรับใช้ในทางธุรกิจหรือใช้ในการเชื่อมต่อกับลูกค้า เพื่อส่งเสริมกิจกรรมทางธุรกิจหรือการส่งข้อมูลข่าวสาร สามารถเชื่อมต่อข้อมูลหรือระบบภายนอกด้วย Messaging API รวมทั้งยังสามารถส่งข้อความไปยังผู้ใช้รายบุคคลหรือเฉพาะกลุ่มได้อีกด้วย

รูปที่ 3.1 สัญลักษณ์ LINE Official Account

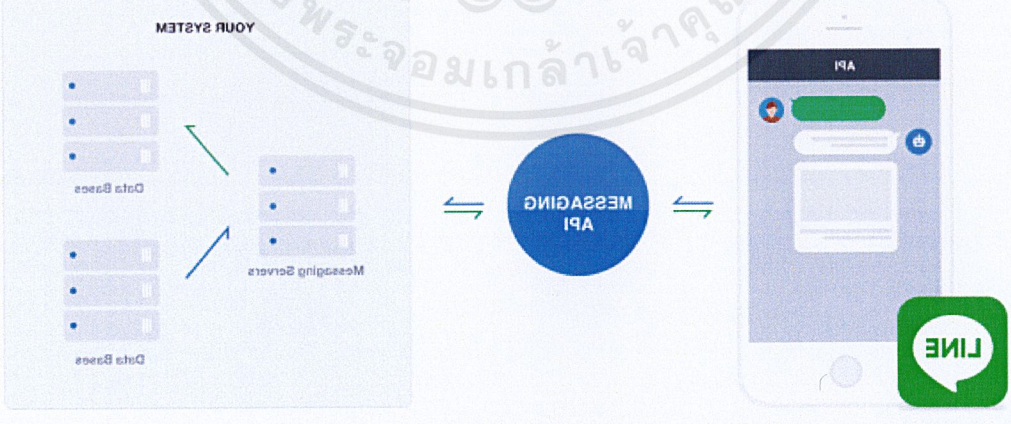
(ที่มา : https://apps.apple.com/us/app/line-official-account/id1450599059)

ไลน์บีคอน (LINE Beacon) เป็นเทคโนโลยีที่ใช้ในการรับ-ส่งข้อมูลระยะสั้นผ่านสัญญาณบลูทูธที่ใช้พลังงานต่ำ หรือ BLE (Bluetooth Low Energy) หากผู้ใช้อยู่ภายในรัศมีของไลน์บีคอนจะได้รับการแจ้งเตือนหรือข้อมูลข่าวสารต่าง ๆ ผ่านแอปพลิเคชันไลน์

รูปที่ 3.2 สัญลักษณ์ LINE Beacon

(ที่มา : https://www.iphoneapptube.com/line-beacon/)

Messaging API เป็นทำหน้าที่เป็นตัวกลางที่ในการเชื่อมต่อระหว่างระบบและผู้ใช้ผ่านไลน์แพลตฟอร์ม ซึ่ง Request จะถูกส่งผ่าน HTTPS ในรูปของ JSON format เมื่อผู้ใช้ส่งข้อความมายังไลน์โอเอ Webhook event จะถูกส่งไปยัง LINE Bot server ที่ได้ทำการลงทะเบียนไว้ หลังจากนั้น LINE Bot server จะตอบสนองผู้ใช้ผ่านไลน์แพลตฟอร์ม



รูปที่ 3.3 การทำงานของ Messaging API

(ที่มา : https://developers.line.biz/en/docs/messaging-api/overview/)

LIFF (LINE Front-end Framework) แพลตฟอร์มสำหรับเว็บแอปพลิเคชัน (Web application) โดยมีไลน์เป็นผู้ให้บริการ เว็บแอปพลิเคชันที่ทำงานบนแพลตฟอร์มนี้จะถูกเรียกว่า LIFF apps. LIFF apps สามารถดึงข้อมูลจากไลน์แพลตฟอร์มได้ เช่น user ID, display Name, user's profile อีกทั้งยังสามารถนำข้อมูลเหล่านี้ไปใช้ในการให้บริการต่าง ๆ และทำให้การใช้งานของผู้ใช้ง่ายยิ่งขึ้น

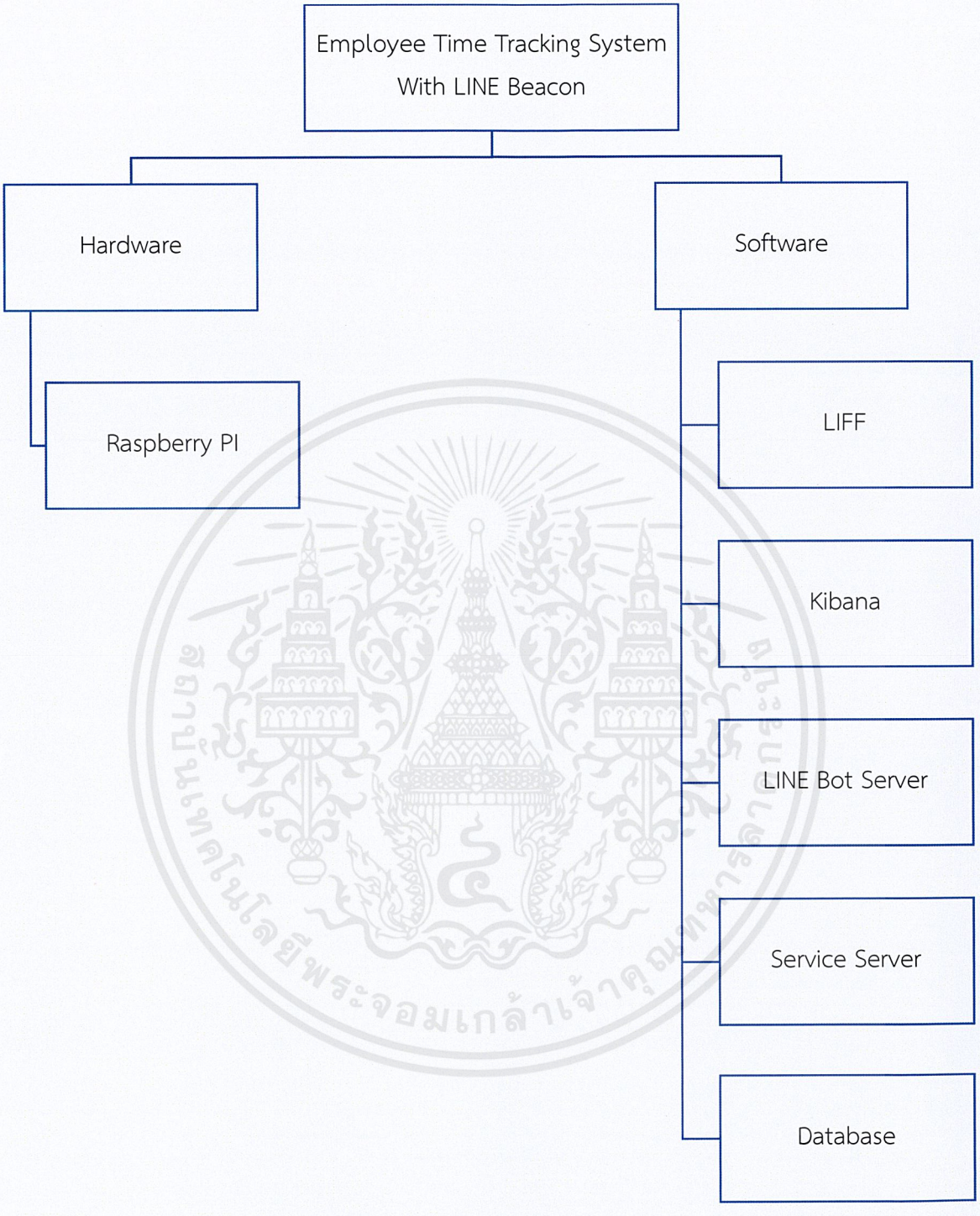
3.2 ศึกษาข้อกำหนดของอุปกรณ์ไลน์บีคอน (LINE Beacon device specification)

อุปกรณ์ที่ใช้ในการกระจายสัญญาณบลูทูธต้องรองรับ Bluetooth® Low Energy เวอร์ชัน 4.0 และ iBeacon ของ Apple และอุปกรณ์ต้องมีคุณสมบัติตรงตามข้อกำหนดต่อไปนี้:

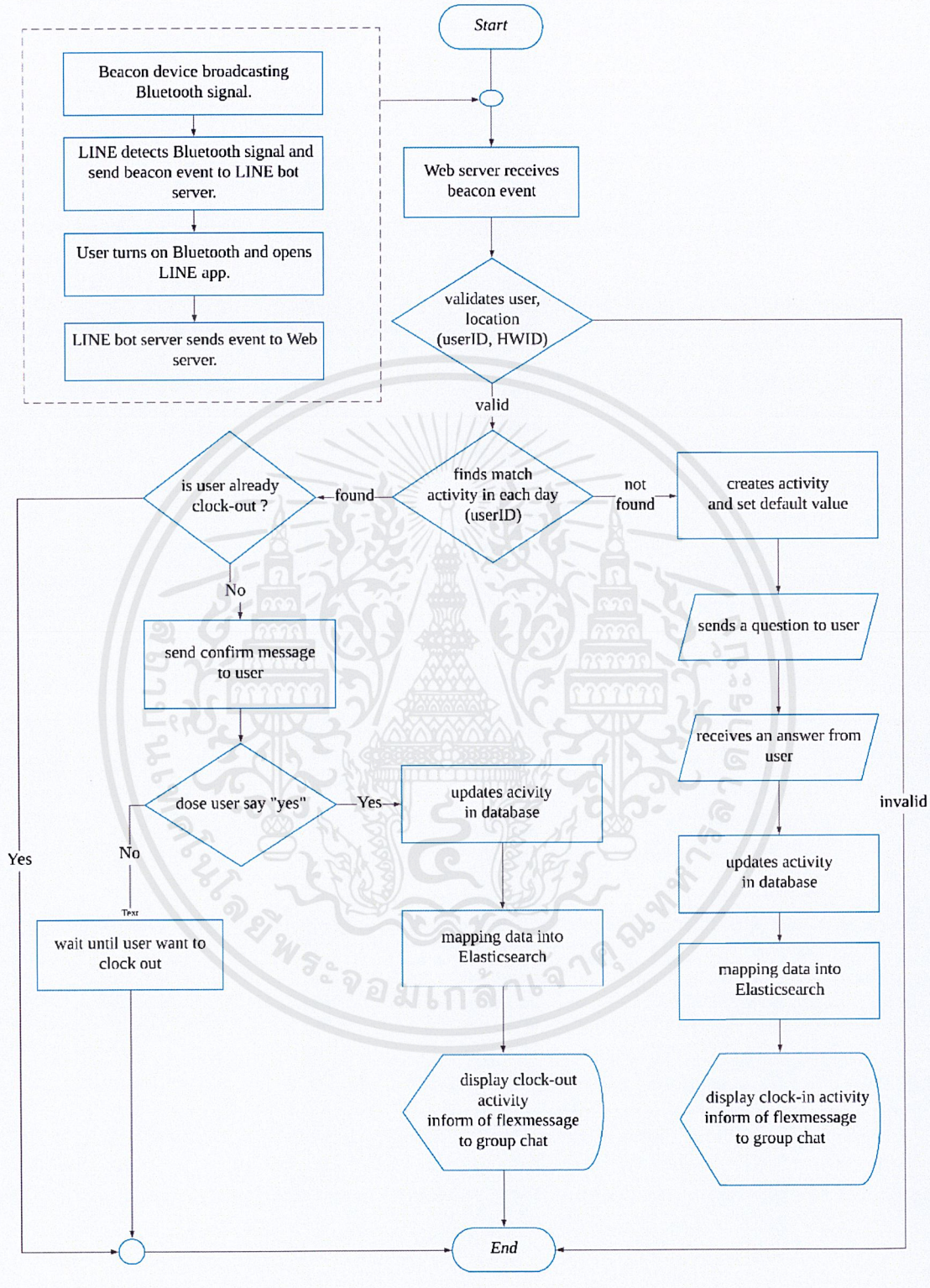
- advertise LINE Beacon packets ได้
- สร้าง secure message โดยการเข้ารหัสแบบ SHA-256 และ XOR (exclusive OR) operations.
- อัปเดต secure message ทุก ๆ 15 วินาที
- มี unique Hardware ID และแสดง Hardware ID นั้นบน body

3.3 วิเคราะห์และออกแบบ

หัวข้อนี้เป็นการวิเคราะห์และทำการเก็บรวบรวมข้อมูลภาพรวมของระบบ เพื่อทำการจัดเตรียมอุปกรณ์ต่าง ๆ การออกแบบโครงสร้างของระบบสามารถแยกได้ดังแผนภาพในรูปที่ 3.4 ซึ่งประกอบไปด้วยฮาร์ดแวร์และซอฟต์แวร์ ในส่วนประกอบของฮาร์ดแวร์คืออุปกรณ์ที่ใช้ในการกระจายสัญญาณบลูทูธ และซอฟท์แวร์จะประกอบไปด้วย LIFF, Kibana, LINE Bot server, Service Server และ Database



รูปที่ 3.4 ส่วนประกอบของระบบ



รูปที่ 3.5 Flowchart แสดงการทำงานของระบบ

3.3.1 การออกแบบฐานข้อมูล

ในส่วนของฐานข้อมูลนั้นเลือกใช้ mongoDB เนื่องจากเป็นฐานข้อมูลแบบ NOSQL เก็บข้อมูลในรูปแบบของ JSON โดยค่าที่เก็บจะอยู่ในรูป key : value มีคุณสมบัติเป็น schema less โดยมีโครงสร้างดังต่อไปนี้

โครงสร้างของ user model : รูปแบบในการเก็บข้อมูลของผู้ใช้

userId: String,

displayName: String,

firstName: String,

lastName: String,

nickName: String

โครงสร้างของ activity model : รูปแบบในการเก็บข้อมูลของผู้ใช้เมื่อเข้ามาในรัศมีของไลน์บีคอน

userId: String,
displayName: String,
type: String,
clockin: Date,
clockout: Date,
location: Object,
askstate: Boolean,
dialogs:Boolean,
plan: String,
url: String

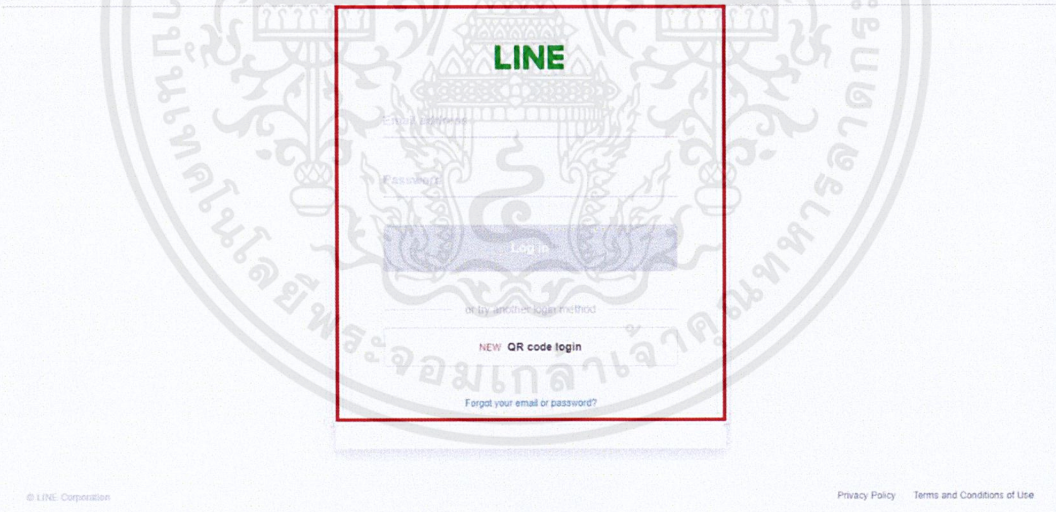
โครงสร้างของ location model : รูปแบบในการเก็บข้อมูลพิกัดของสถานที่ทำงาน

```
hardwareID: String,
locationName: String,
point: {
  coordinates: [Number]
}
```

3.4 จัดเตรียมเครื่องมือและอุปกรณ์ที่ใช้

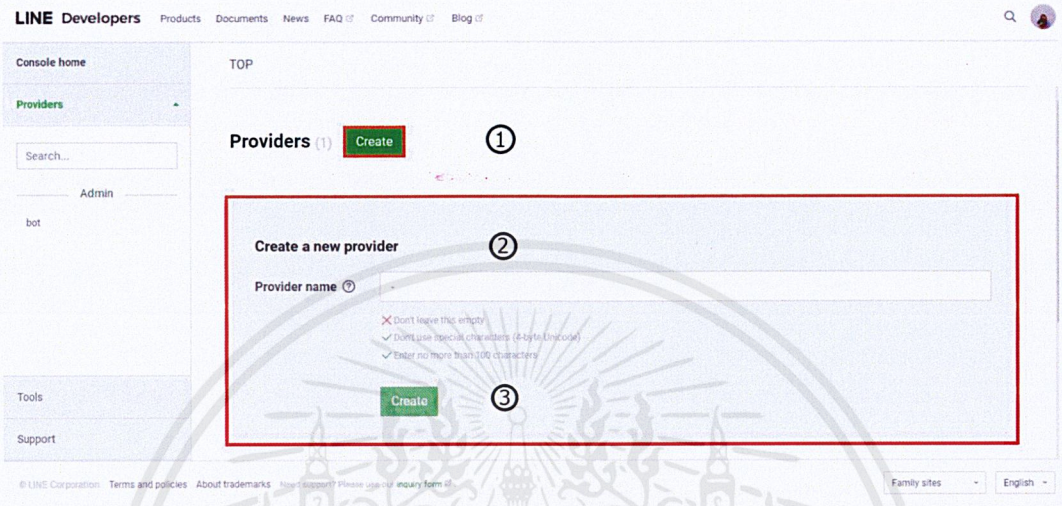
3.4.1 สร้างบัญชีทางการ (LINE official account)

ทำการสร้างบัญชีทางการสำหรับใช้ในการขอ Hardware ID และแจ้งเตือนผ่านแอปพลิเคชันไลน์ โดยเข้าไปยังเว็บไซต์ https://developers.line.me/ ในหน้าเว็บจะให้ทำการกรอกข้อมูล Username และ Password ที่ใช้เชื่อมต่อกับ LINE account ดังรูปที่ 3.6



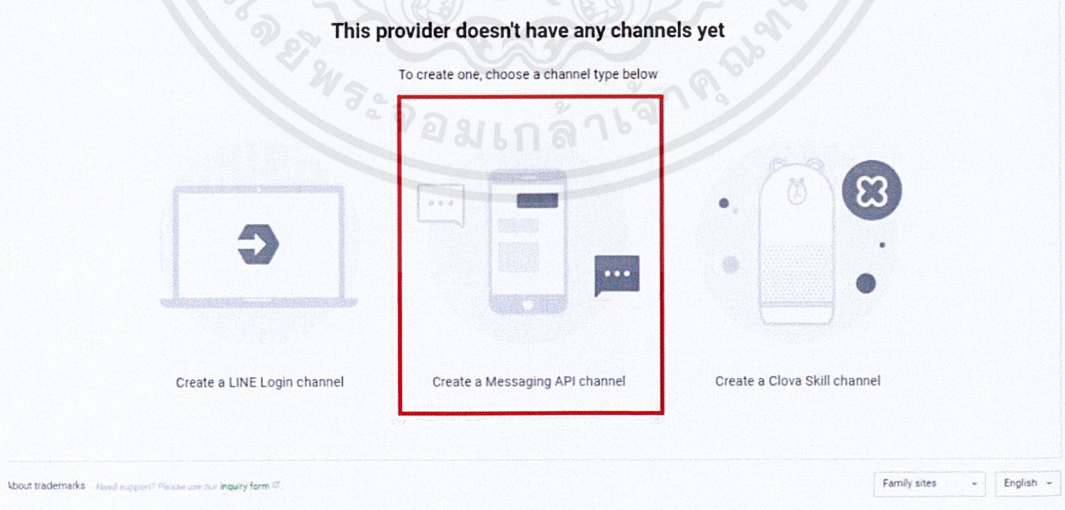
รูปที่ 3.6 เว็บไซต์สำหรับสร้าง LINE official account

เมื่อทำการกรอก Username และ Password เพื่อเข้าถึงบัญชี จากนั้นจะปรากฏหน้า Provider List ขึ้นทำการสร้าง Provider ด้วยการกดไปที่ Create New Provider ดังรูปที่ 3.7

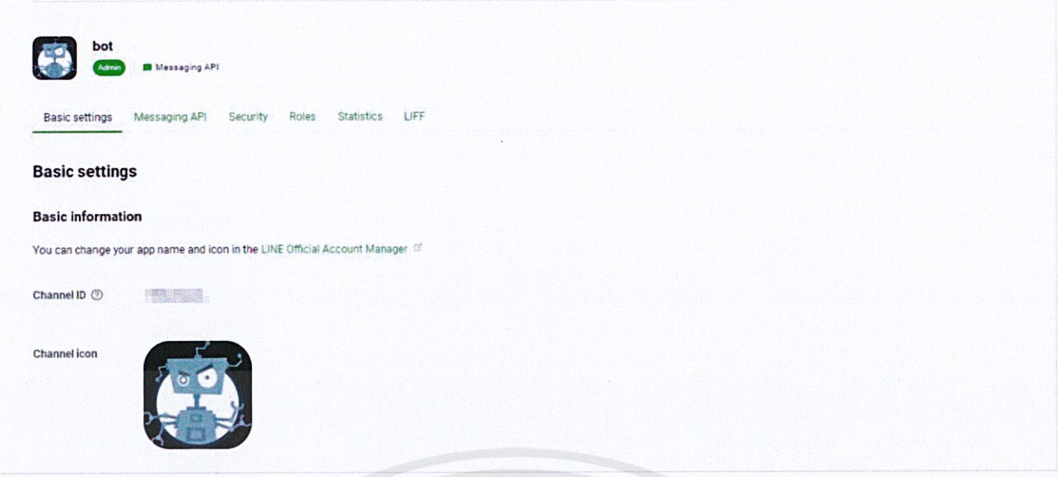


รูปที่ 3.7 การสร้าง Provider

หลังจากสร้าง Provider เสร็จเรียบร้อยแล้วนั้น ในขั้นตอนต่อมาจะทำการสร้าง Bot Channel เพื่อใช้ในการเชื่อมต่อและส่งข้อความโดย LINE Message API โดยไปที่ Create a Messaging API channel ดังรูปที่ 3.8 เมื่อดำเนินการเสร็จสิ้นจะได้ผลลัพธ์ดังรูปที่ 3.9

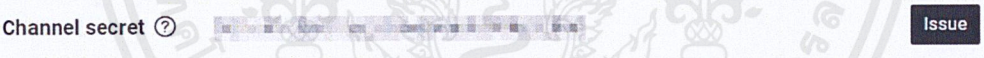


รูปที่ 3.8 การสร้าง Bot Channel

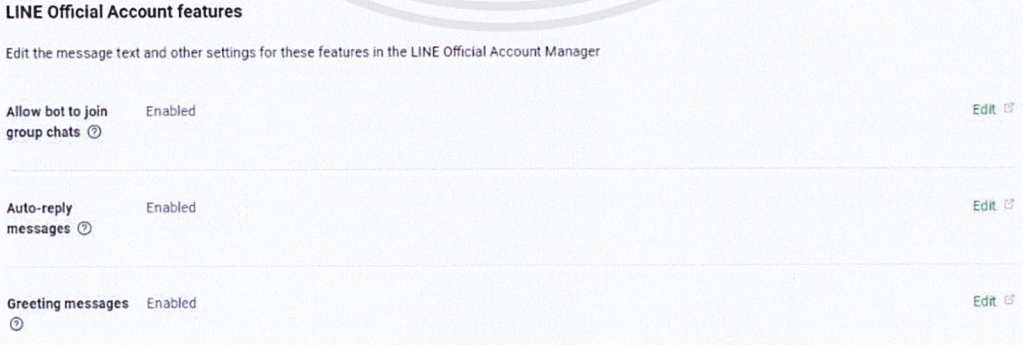


รูปที่ 3.9 การสร้าง Bot Channel

เมื่อเลื่อนเว็บไซต์ลงมาด้านล่าง จะพบกับหัวข้อ Channel secret ใช้เป็น secret key ดังรูปที่ 3.10 และไปยังหัวข้อ Messaging API ทำการตั้งค่าดังรูปที่ 3.11 หลังจากนั้นจะพบ Channel access token ที่ใช้ในการเรียก Messaging API ดังรูปที่ 3.12 ค่าเหล่านี้ต้องทำการเก็บไว้อย่างดีเพราะหากผู้ใช้อื่นมีรหัส Token จะสามารถส่งข้อความและเข้ามาใช้งานได้ และ Webhook URL เป็นการระบุ endpoint ที่จะใช้รับ-ส่ง request จากไลน์แพลตฟอร์ม ดังรูปที่ 3.13

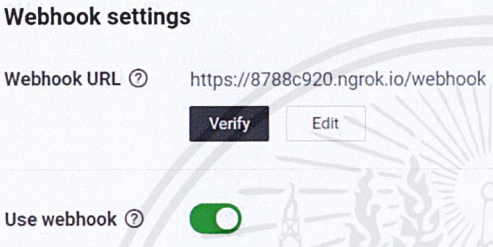


รูปที่ 3.10 Channel secret ของ Bot channel



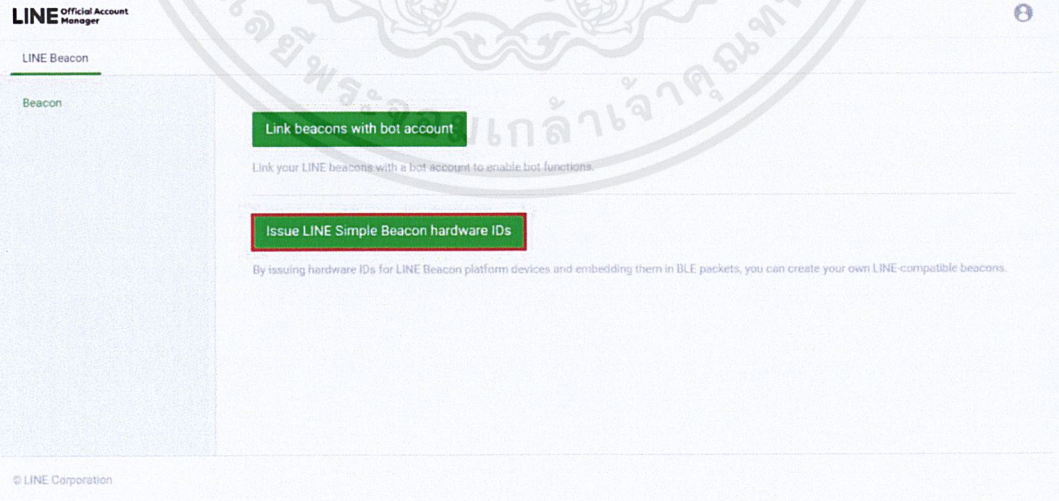
รูปที่ 3.11 ตั้งค่าให้กับ LINE official account

รูปที่ 3.12 Channel access token ของ Bot channel



รูปที่ 3.13 Webhook URL ที่ใช้ในการเชื่อมต่อระหว่าง LINE Bot server และ Service server

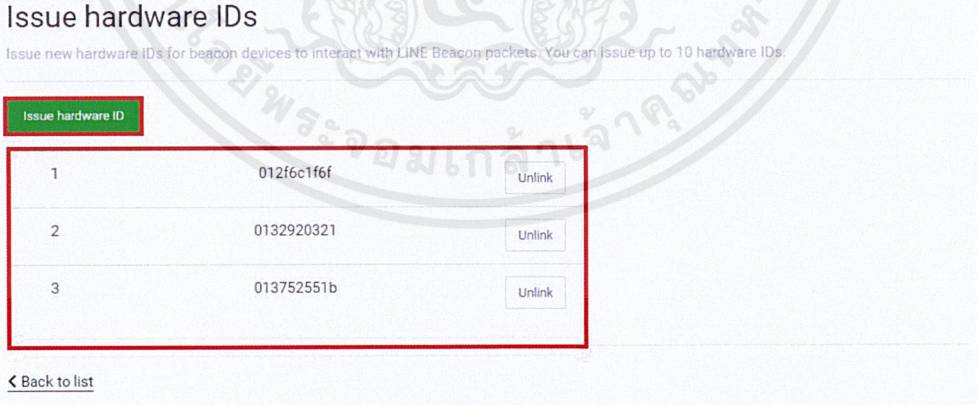
ในขั้นตอนถัดมาจะเป็นการสร้าง Hardware ID เพื่อนำไปใช้กับอุปกรณ์กระจายสัญญาณบลูทูธ โดยเข้าไปยัง https://manager.line.biz/beacon/register จากนั้นเลือก Issue LINE Simple Beacon Hardware ID ดังรูปที่ 3.14 จะพบบัญชีทางการที่ได้ทำการสร้างไว้ ดังรูปที่ 3.15



รูปที่ 3.14 การตั้งค่าHardware ID ให้อุปกรณ์

รูปที่ 3.15 บัญชีทางการที่ได้ทำการสร้างไว้

เข้าไปยังบัญชีที่ต้องการและกด Issue Hardware ID ระบบจะทำการสร้าง Hardware ID มาให้โดยอัตโนมัติ ซึ่ง Hardware ID เป็นเลขฐาน 16 จำนวน 10 หลัก ดังรูปที่ 3.16 หลังจากนั้นนำ Hardware ID ที่ได้ไปใช้ในการ config อุปกรณ์ที่ใช้ในการกระจายสัญญาณบลูทูธ



รูปที่ 3.16 ระบบสร้าง Hardware ID

หลังจากนั้นนำรหัส Channel access token และ Channel secret มาเขียนไว้ในไฟล์ config โดยในที่นี้จะเก็บค่าไว้ในรูปแบบของ environment variable ดังรูปที่ 3.17 และทำการเขียนโปรแกรมสั่งการให้ดำเนินการตามที่ต้องการ

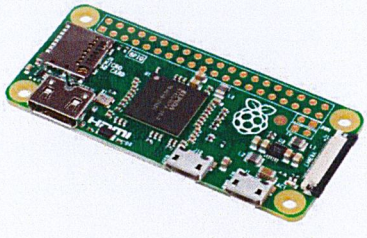
```
config > JS index.js > ...
1
2   // module variables
3   const config = require('./config.json');
4   const environment = process.env.NODE_ENV || 'development';
5   const environmentConfig = config[environment];
6
7   const finalConfig = {
8       ...environmentConfig,
9       channelAccessToken: process.env.channelAccessToken,
10      channelSecret: process.env.channelSecret,
11      reportGroupId: process.env.reportGroupId,
12      elasticConfig:process.env.elasticConfig,
13      AnswerAlertDuration:3600000
14  }
15  export {finalConfig};
16
17
```

รูปที่ 3.17 การกำหนดค่า Channel access token
และ Channel secret ในรูปแบบ environment variable

3.4.2 อุปกรณ์และโปรแกรมที่ใช้ในการติดตั้งไลน์บีคอน

1) ราสเบอร์รี่พาย (Raspberry Pi)

Raspberry Pi เป็นบอร์ดคอมพิวเตอร์ขนาดเล็ก ถูกพัฒนาขึ้นโดย Raspberry Pi Foundation สามารถต่อเข้ากับจอคอมพิวเตอร์หรือทีวี เมาส์ คีย์บอร์ด และอุปกรณ์อื่น ๆ อีกทั้งยังสามารถเชื่อมต่อระบบเครือข่ายแบบใช้สายหรือไร้สายได้



รูปที่ 3.18 Raspberry Pi

(ที่มา : https://upload.wikimedia.org/wikipedia/commons/7/7e/Raspberry-Pi-Zero-FL.jpg)

2) USB micro power supply สำหรับจ่ายไฟให้กับราสเบอร์รี่พาย

รูปที่ 3.19 Power supply

(ที่มา : https://inex.co.th/shop/media/catalog/product/cache/1 /image/9df78eab33525d08d6e5fb8d27136e95/r/p/rpi-adaptor-type-c01.png)

3) Micro SD-Card ใช้ร่วมกับบอร์ดกับราสเบอร์รี่พาย

รูปที่ 3.120 Micro SD-Card

(ที่มา : https://cdn.sparkfun.com//assets/parts/1/1/6/3/5/13945-01.jpg)

4) โปรแกรม Win32 Disk Imager

Win32 Disk Imager เป็นเครื่องมือที่ใช้ในการอ่าน เขียน ตรวจสอบ ดิสก์อิมเมจ (Disk Image) ลงบนแฟลชไดร์ฟ (USB Flash Drive) หรือการ์ดเก็บข้อมูลต่าง ๆ สามารถดาวน์โหลดโปรแกรมได้ที่ https://sourceforge.net/projects/win32diskimager/

5) โปรแกรม Advanced IP Scanner

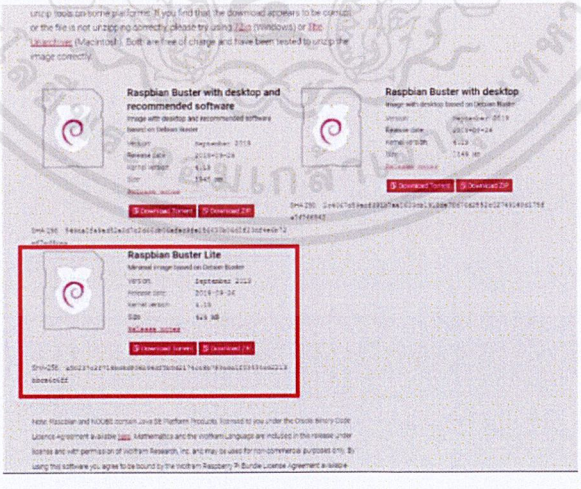
Advanced IP Scanner ใช้สำหรับ Scan หา IP Address ของอุปกรณ์ที่อยู่ในวง LAN เดียวกัน สามารถดาวน์โหลดโปรแกรมได้ที่ http://www.advanced-ip-scanner.com

6) โปรแกรม PuTTY

PuTTY ใช้สำหรับเป็น SSH Client ในการเข้าไปควบคุมการทำงานของราสเบอร์รี่พาย สามารถดาวน์โหลดได้ที่ https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html

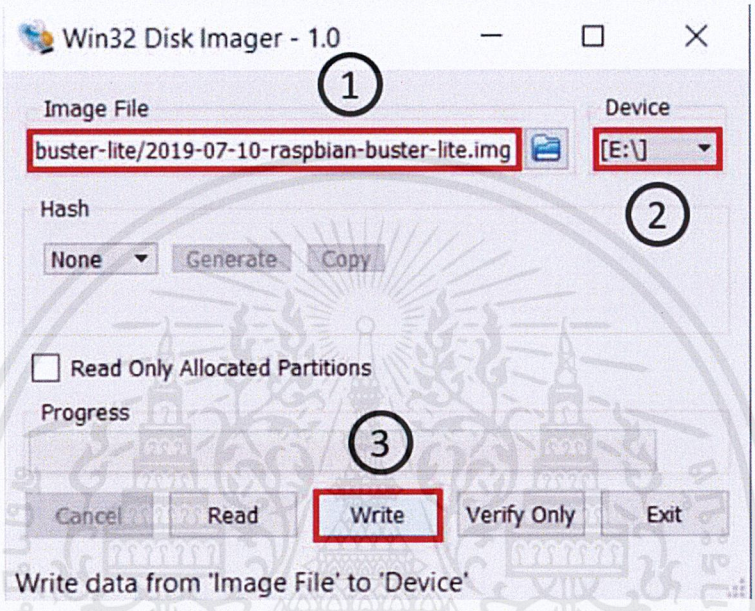
3.4.3 ขั้นตอนการติดตั้งอุปกรณ์

ก่อนที่จะใช้งานราสเบอร์รี่พายต้องทำการติดตั้ง OS ให้กับอุปกรณ์ โดยเขียนไฟล์อิมเมจ (images) ลงใน Micro SD Card ในที่นี้ใช้ Raspbian Buster Lite สามารถดาวน์โหลดไฟล์ image ได้ที่ https://www.raspberrypi.org/downloads/raspbian/ หลังจากนั้นทำการแตกไฟล์ที่ได้ทำการดาวน์โหลดไปเมื่อสักครู่

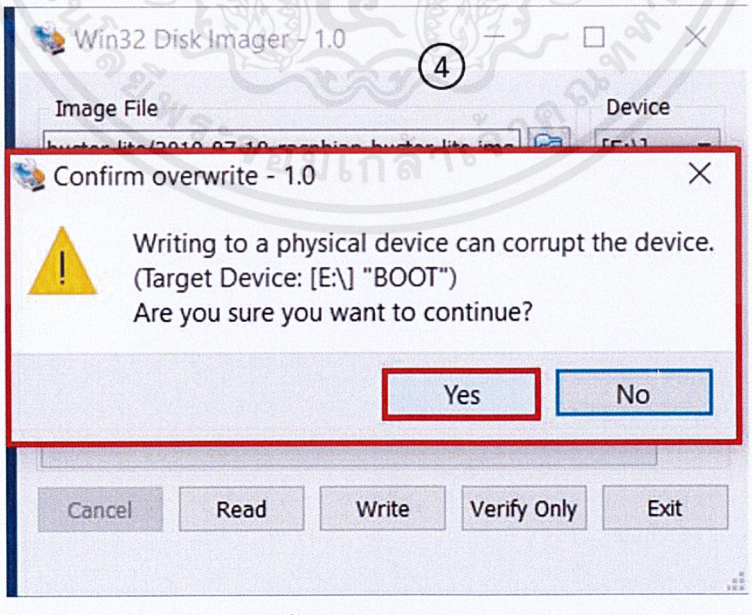


รูปที่ 3.21 หน้าเว็บเพจสำหรับการดาวน์โหลดไฟล์ image

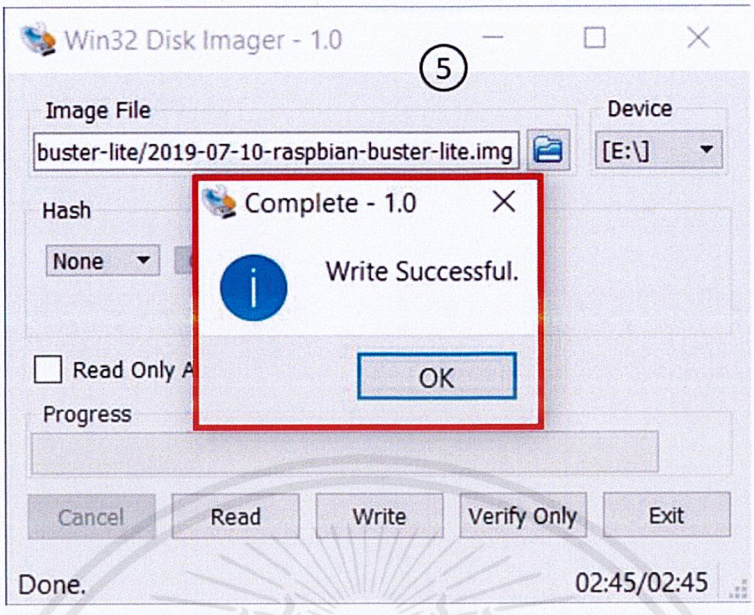
หลังจากนั้นเชื่อม Micro SD-Card เข้ากับคอมพิวเตอร์แล้วเปิดโปรแกรม Win32 Disk Imager (1) ทำการ Brows ไฟล์ image (2) เลือกอุกรณ์ที่ต้องการ (3) คลิกปุ่ม Write (4) จะมีหน้าต่างปรากฏขึ้นเพื่อให้ยืนยันในการเขียน image ลงในไดร์ฟ (5) เมื่อโปรแกรมดำเนินการเสร็จสิ้นหน้าต่าง “Write Successful.” จะปรากฏขึ้นมาตามลำดับ



รูปที่ 3.22 ขั้นตอนการเขียนไฟล์ image



รูปที่ 3.23 ขั้นตอนการเขียนไฟล์ image



รูปที่ 3.24 ขั้นตอนการเขียนไฟล์ image

ทำการเปิด SSH Service (Secure Shell service) บน Raspbian OS เพื่อให้สามารถรีโมทควบคุมการตั้งค่าเริ่มต้นผ่าน PC หรือ Mac โดยเข้าไปยัง Micro SD-Card จากนั้นทำการสร้างไฟล์เปล่าชื่อ ssh โดยไม่มีนามสกุลไฟล์ ดังรูปที่ 3.25

รูปที่ 3.25 เปิดโหมด Secure Shell

ทำการตั้งค่าการเข้าใช้งานผ่าน Wireless ดังรูปที่ 3.26 โดยสร้างไฟล์ wpa_supplicant.conf บน Partition เดียวกันกับไฟล์ ssh และกำหนดค่าต่าง ๆ ดังนี้ ssid คือการกำหนดชื่อ Wi-Fi, psk คือรหัสที่ใช้ในการเชื่อมต่อ Wi-Fi

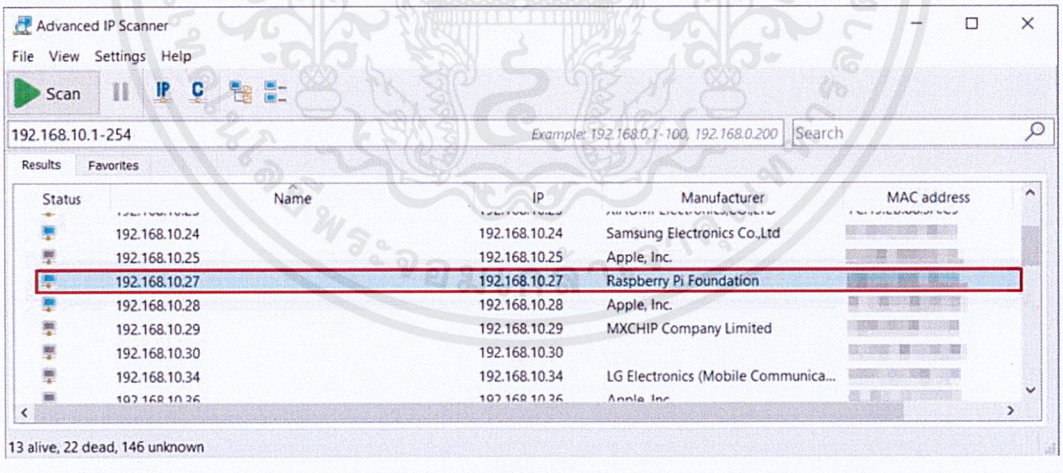
wpa_supplicant.conf - Notepad

File Edit Format View Help

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=TH
network={
        ssid="WiFi"
        psk="Password"
        key_mgmt=WPA-PSK
}
```

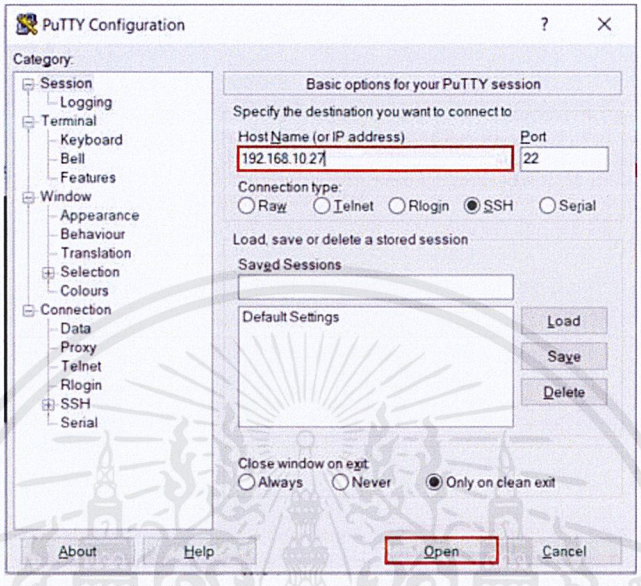
รูปที่ 3.26 ตั้งค่าการใช้งานผ่าน Wi-Fi

หลังจากนั้นให้นำ Micro SD-Card ที่ทำการตั้งค่าเรียบร้อยแล้วใส่เข้าไปในราสเบอร์รี่พาย และทำการต่อสาย Power เพื่อ boot ราสเบอร์รี่พาย จากนั้นเปิดโปรแกรม Advance IP Scanner เพื่อทำการสแกนหา IP Address ที่ราสเบอร์รี่พายใช้อยู่ โดยคอมพิวเตอร์ที่ใช้ต้องเชื่อมต่อเครือข่ายเดียวกันกับที่อุปกรณ์เชื่อมต่ออยู่



รูปที่ 3.27 สแกน IP Address ของราสเบอร์รี่พาย

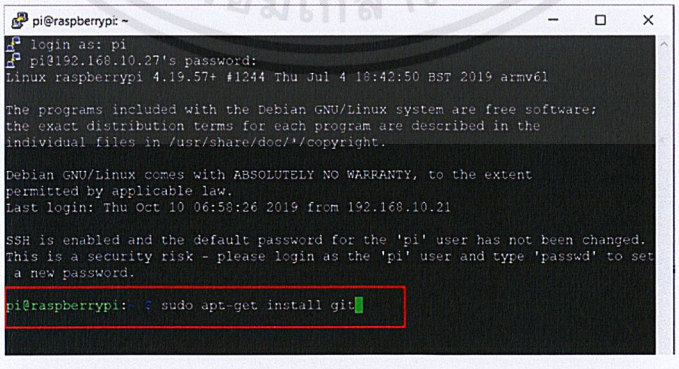
เปิดโปรแกรม PuTTY สำหรับเชื่อมต่อ SSH Client ต่อมากรอก IP Address ของราสเบอร์รี่พาย ที่ช่อง Host Names แล้วคลิกปุ่ม open



รูปที่ 3.28 Putty เพื่อที่จะเข้าไปตั้งค่าราสเบอร์รี่พาย

หน้าต่าง command line จะปรากฏขึ้น ให้กรอก Username และ Password โดย Username คือ pi และ Password คือ raspberry จากนั้นติดตั้ง Git command สำหรับใช้ clone Beacon repository ด้วยคำสั่งต่อไปนี้

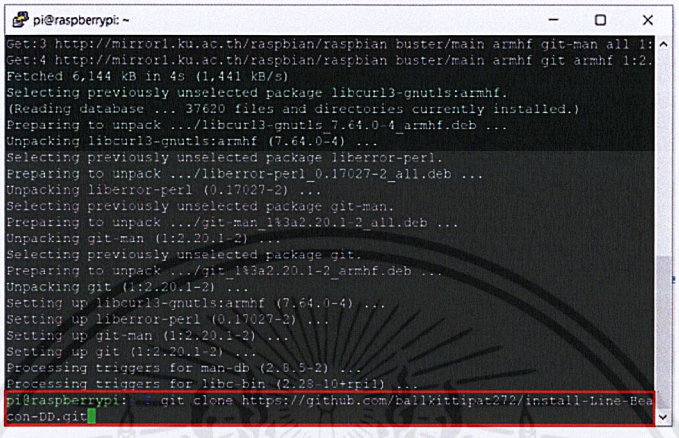
```
$ sudo apt-get install git
```



รูปที่ 3.29 ติดตั้ง Git command

หลังจากนั้นทำการ clone shell script ด้วยคำสั่ง

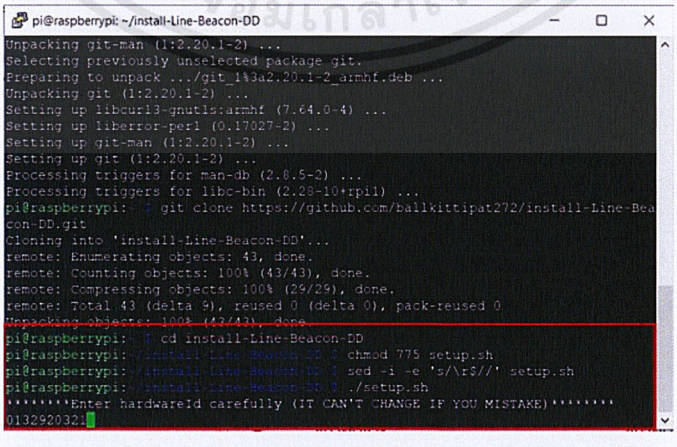
```
$ git clone https://github.com/ballkittipat272/install-Line-Beacon-DD.git
```



รูปที่ 3.30 clone line simple beacon

จากนั้นตั้งค่าอุปกรณ์ให้สามารถทำงานได้โดยอัตโนมัติด้วยคำสั่งเหล่านี้ แล้วทำการกรอก Hardware ID ที่ได้สร้างไว้

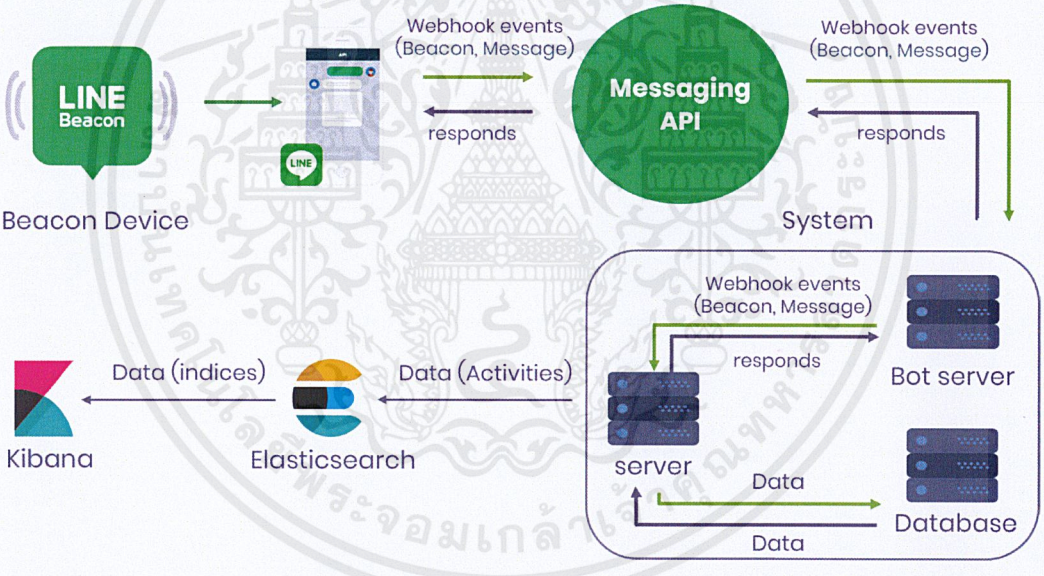
```
$ cd install-Line-Beacon-DD
$ chmod 775 setup.sh
$ sed -i -e 's/\r$//' setup.sh
$ ./setup.sh
```



รูปที่ 3.31 ตั้งค่าอุปกรณ์ให้ automate run เมื่อมีการจ่ายไฟ

บทที่ 4
การทำงานของระบบ

จากรูปที่ 4.1 แอปพลิเคชันไลน์จะทำการตรวจจับสัญญาณที่ถูกส่งจาก Beacon device เมื่อพนักงานเข้ามาในพื้นที่สัญญาบีคอน ต่อมาแอปพลิเคชันไลน์จะทำการส่ง Webhook event ไปยัง LINE bot sever และ LINE bot server จะส่งต่อ event เหล่านั้นไปยัง Service server หลังจากนั้น Service server จะทำการประมวลผล LINE request ด้วยข้อมูลที่ได้รับจากฐานข้อมูล (Database) และทำการบันทึกข้อมูลการลงเวลาลงในฐานข้อมูล เมื่อประมวลผลเสร็จสิ้น Service server จะทำการส่ง response ไปยังพนักงานผ่าน Messaging API นอกจากนั้น ผู้จัดการสามารถดูข้อมูลการลงเวลาที่แสดงบนแดชบอร์ดได้



รูปที่ 4.1 ภาพรวมการทำงานของระบบ

4.1 ขั้นตอนการทำงาน

แอปพลิเคชันไลน์ทำการส่ง Webhook event ที่เป็นประเภท Beacon event ประกอบด้วย type, replyToken, source, timestamp, mode และ beacon ไปยัง LINE Bot server และ LINE Bot server ทำการส่งต่อไปยัง Service server โดย

type คือ ประเภทของ event

replyToken คือ token ที่ใช้ในการตอบกลับevent

source คือ ข้อมูลของผู้ใช้ หรือ กลุ่ม หรือ ห้อง อยู่ในรูปของออบเจกค์

timestamp คือ เวลาที่อยู่ในหน่วยมิลลิวินาที

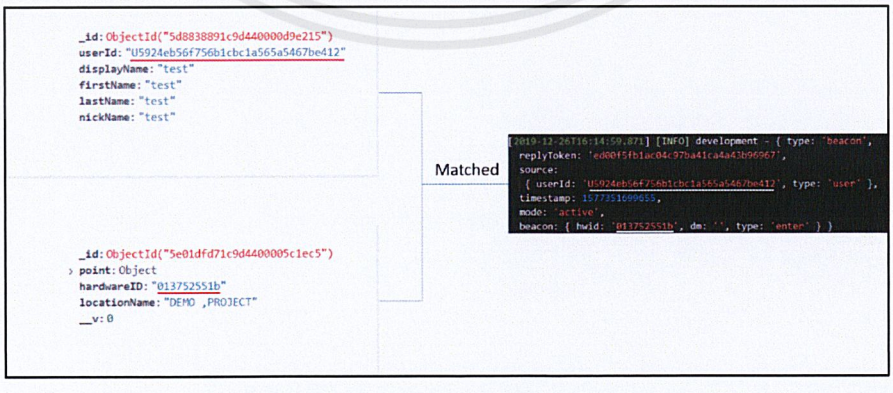
beacon.hwid คือ Hardware ID ของบีคอนที่ตรวจพบ

beacon.type คือ ประเภทของ beacon event

```
[2019-12-26T16:14:59.871] [INFO] development - { type: 'beacon',
  replyToken: 'ed00f5fb1ac04c97ba41ca4a43b96967',
  source:
   { userId: 'U5924eb56f756b1cbc1a565a5467be412', type: 'user' },
  timestamp: 1577351699655,
  mode: 'active',
  beacon: { hwid: '013752551b', dm: '', type: 'enter' } }
```

รูปที่ 4.2 ตัวอย่างของ beacon event ที่ส่งมาจาก LINE Bot Server

เมื่อ Service server ได้รับข้อมูลที่ถูกส่งมาจะนำค่า source.userId และ beacon.hwid ไปตรวจสอบว่าผู้ใช้นั้นเป็นพนักงานในบริษัทหรือไม่ และสถานที่ที่เข้ามาใช้งานนั้นตรงกับข้อมูลที่ถูกบันทึกไว้ในฐานข้อมูลหรือไม่



รูปที่ 4.3 ตรวจสอบข้อมูลผู้ใช้ตามเงื่อนไข

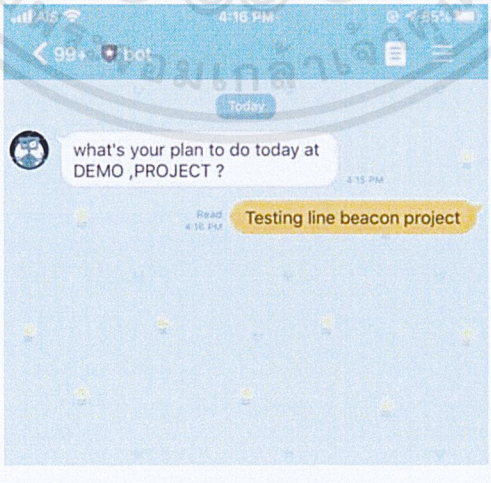
หากตรงตามเงื่อนไขที่กล่าวมาระบบจะนำ source.userId และ timestamp ไปเปรียบเทียบกับข้อมูลของพนักงานในวันนั้นซึ่งจะถูกแบ่งออกเป็น 2 กรณี

กรณีที่ 1 การลงเวลาเข้างาน

เมื่อระบบทำการตรวจสอบแล้วไม่พบข้อมูลการลงเวลา ระบบจะเก็บ timestamp ที่ได้เป็นการลงเวลาเข้างาน โดยจะทำการสร้าง activity ตามโครงสร้างของ activity model และทำการกำหนดค่าพื้นฐานให้กับ activity แล้วบันทึกลงในฐานข้อมูล ดังรูปที่ 4.4

รูปที่ 4.4 ระบบทำการสร้าง activity และบันทึกลงฐานข้อมูลลงฐานข้อมูล

จากนั้นระบบจะทำการส่ง response ไปยังพนักงานในรูปแบบของข้อความ เพื่อถามความคืบหน้าของโครงงานที่ได้รับมอบหมาย ดังรูปที่ 4.5



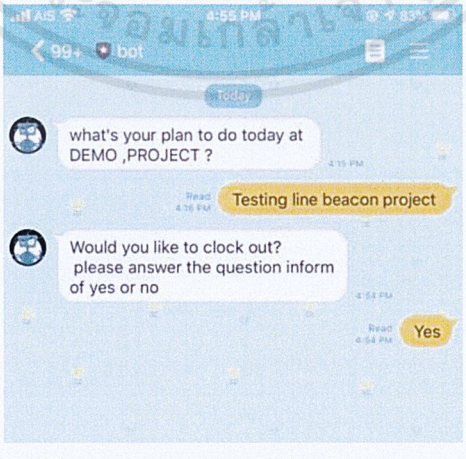
รูปที่ 4.5 การตอบสนองของลงเวลาเข้างาน

เมื่อได้รับการตอบกลับจากพนักงานระบบจะทำการอัปเดตข้อมูลที่อยู่ในฐานข้อมูล และส่งรายละเอียดการลงเวลาเข้างานไปยังแชทกลุ่มในรูปแบบของFlex message ดังรูปที่ 4.6

รูปที่ 4.6 รายละเอียดการลงเวลาเข้างาน

กรณีที่ 2 การลงเวลาออกงาน

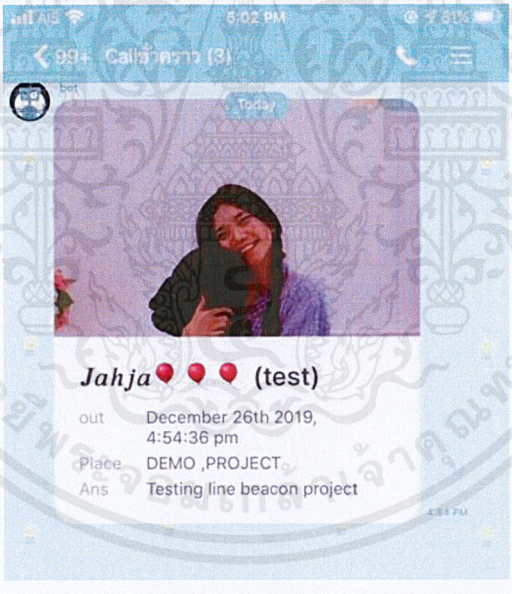
หากระบบนำ source.userId และ timestamp ไปตรวจสอบและพบข้อมูลการลงเวลาเข้างาน แต่ยังไม่พบการลงเวลาออกงาน ระบบจะทำการส่งข้อความไปยังแชทของพนักงาน เพื่อเป็นการยืนยันว่าพนักงานต้องการที่จะลงเวลาออกงาน ดังรูปที่ 4.7 หากผู้ใช้ต้องการลงเวลาออกจากงาน ระบบจะทำการอัปเดตค่าในฟิลด์ clock out ทั้งในฐานข้อมูล ดังรูปที่ 4.8 หลังจากนั้นทำการส่งรายละเอียดการลงเวลาออกงานไปยังแชทกลุ่ม ดังรูปที่ 4.9



รูปที่ 4.7 การตอบสนองของลงเวลาออกงาน

```
_id: ObjectId("5e047a14adde7a1bf48e2afc")
userId: "U5924eb56f756b1cbc1a565a5467be412"
displayName: "Jahja🎈🎈🎈"
type: "out"
clockin: 2019-12-26T09:14:59.655+00:00
clockout: 2019-12-26T09:54:36.199+00:00
› location: Object
askstate: true
dialogs: true
plan: "Testing line beacon project "
url: "https://profile.line-scdn.net/0hZ9Wd11JVBRZqHS7rdOl6QVZYC3sdMwNeEnodcU..."
__v: 0
```

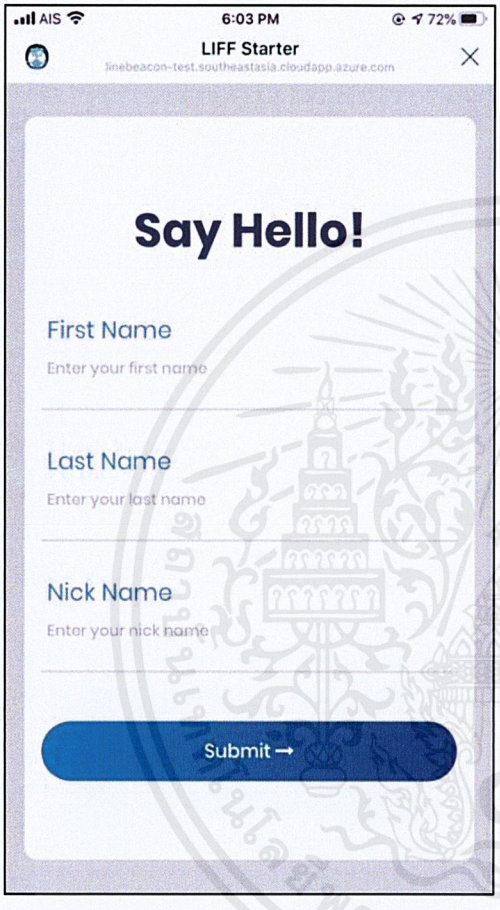

รูปที่ 4.8 ระบบทำการอัปเดตค่าภายในฐานข้อมูล



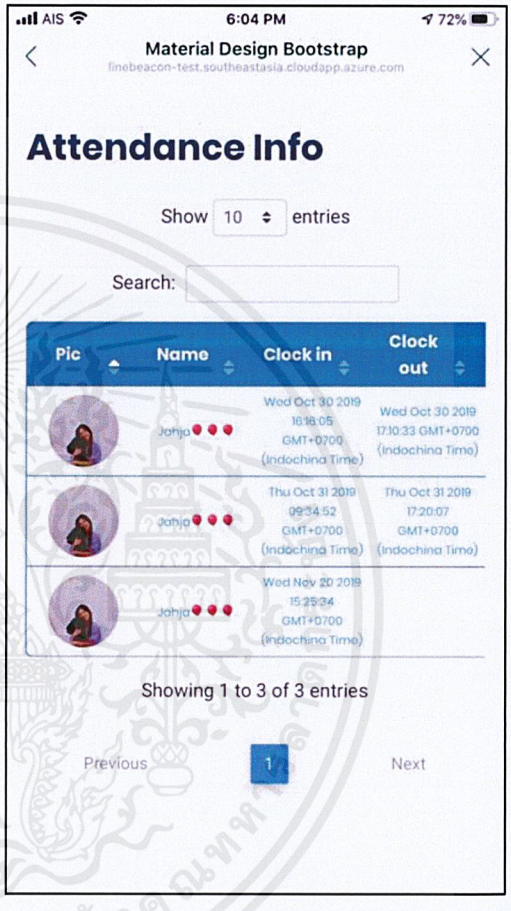
รูปที่ 4.9 รายละเอียดการลงเวลาออกงาน

4.2 การแสดงผล

พนักงานสามารถกรอกประวัติส่วนและดูประวัติการทำงานผ่าน LINE Front-end Framework (LIFF) ซึ่งทำหน้าที่เป็น Web view ทำให้สามารถเปิดเว็บต่าง ๆ ผ่านแอปพลิเคชันไลน์ได้ทันที

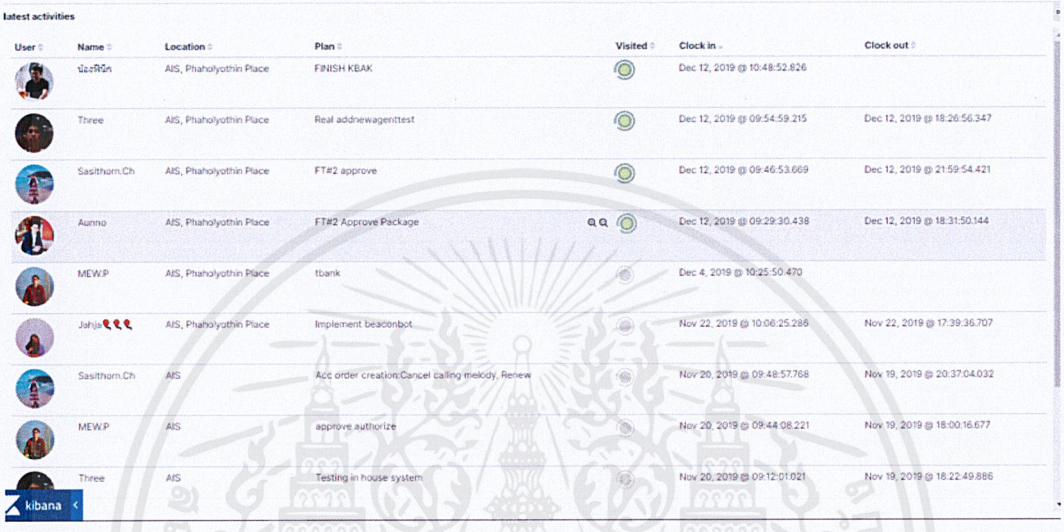


รูปที่ 4.10 หน้าจอ User Profile

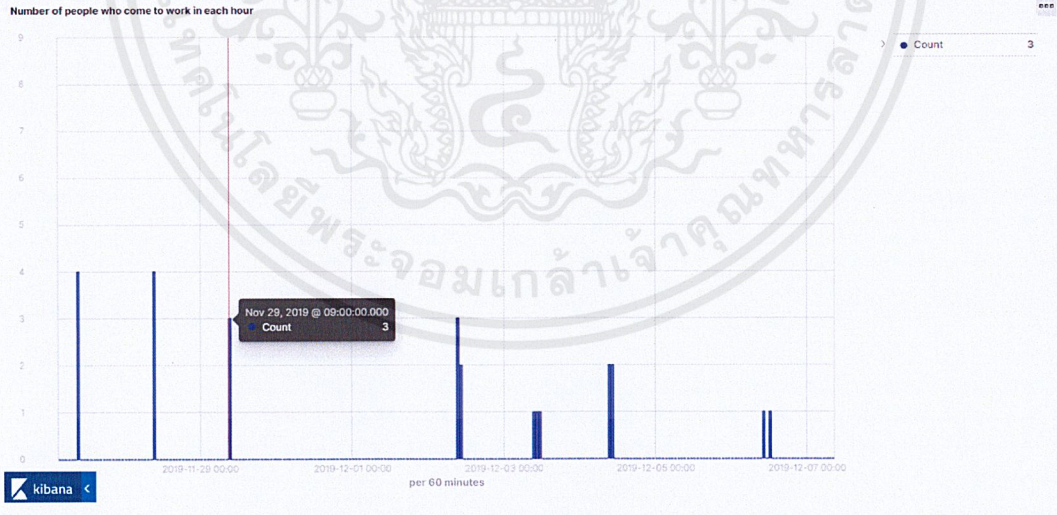


รูปที่ 4.11 หน้าจอ History Profile

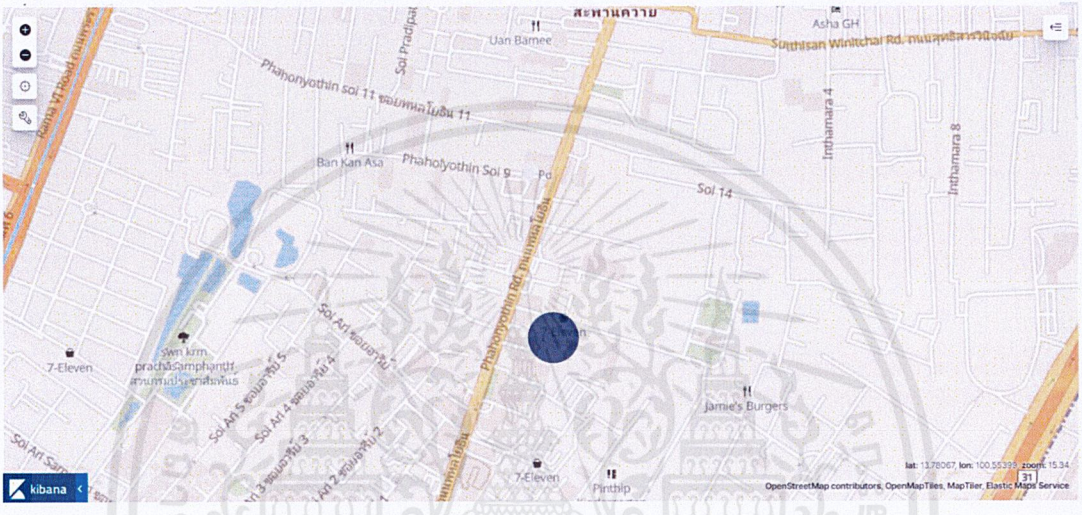
ในส่วนของผู้จัดการนั้นสามารถดูการเข้าทำงานของพนักงานได้ผ่านแชทกลุ่ม และดูการสรุปรวบรวมข้อมูลในรูปแบบของ Dashboard โดยแสดงผ่าน Kibana โดยประกอบไปด้วย



รูปที่ 4.12 หน้าจอแสดงรายละเอียดการลงเวลาเข้า-ออก



รูปที่ 4.13 หน้าจอแสดงจำนวนพนักงานที่เข้ามาทำงานในแต่ละชั่วโมง



รูปที่ 4.14 หน้าจอแสดงสถานที่ที่พนักงานเข้าทำงาน

บทที่ 5
สรุปผลการดำเนินงาน ปัญหา และข้อเสนอแนะ

5.1 สรุปผลการดำเนินงาน

ระบบลงเวลาของพนักงานด้วยไลน์บีคอนเพิ่มความสะดวกในการลงเวลาเข้า-ออกให้แก่พนักงานได้มากขึ้น อีกทั้งยังสามารถใช้ในการตรวจสอบการเข้าทำงานของพนักงาน พร้อมแจ้งเตือนข้อความสถานะการเข้า-ออกงาน สถานที่ที่เข้าใช้งาน และสามารถดูประวัติผ่านทางแอปพลิเคชันไลน์

นอกจากนั้นหัวหน้าทีมยังสามารถตรวจสอบประวัติการทำงานของพนักงาน สถานะการทำงานในปัจจุบัน ติดตามความคืบหน้าในการทำงานของพนักงานแต่ละคน หรือข้อมูลต่าง ๆ ผ่านทางKibana ที่ได้ทำการสรุปและรวบรวมผลลัพธ์นำมาแสดงในรูปแบบของกราฟ ตาราง เพื่อให้ง่ายและสะดวกต่อการทำความใจ

5.2 ปัญหาและอุปสรรคที่พบ

1) Message API ของไลน์มีข้อจำกัดในการทำงานต่าง ๆ หากใช้บริการในรูปแบบไม่เสียค่าใช้จ่าย เช่น การบรอดแคสข้อความจำกัดอยู่ที่ 1000 request/month ฟีเจอร์ที่ใช้ในการรับข้อมูล userID ของสมาชิกทั้งหมดที่อยู่ภายในกลุ่ม หรือฟีเจอร์ที่ใช้ในการรับข้อมูล userID ของสมาชิกทั้งหมดที่อยู่ภายในห้องแชท เป็นต้น

2) เนื่องจากในการทำระบบลงเวลางานต้องอาศัยความรู้ในหลาย ๆ ด้านและทำความเข้าใจกระบวนการการทำงานทั้งหมด ดังนั้นจึงต้องใช้เวลาในการแสวงหาข้อมูลและทำความเข้าใจ

3) เนื่องจาก event ต่างๆ ไม่ว่าจะเป็น Webhook event, Message event, Beacon event นั้น ทำการรับและส่งด้วยไลน์แพลตฟอร์ม ดังนั้นหากไลน์แพลตฟอร์มมีปัญหาหรือระบบล่ม พนักงานจะไม่สามารถลงเวลาได้

5.3 ข้อเสนอแนะ

1) ควรมีการเตรียมความพร้อมศึกษาหาความรู้ก่อนเข้าร่วมสหกิจศึกษา เนื่องจากในการทำงานนั้นต้องใช้ความรู้ในหลายๆด้าน หรือความรู้ที่ไม่ได้ศึกษามาก่อน ดังนั้นจึงควรศึกษาเทคโนโลยีหรือความรู้ที่เกี่ยวข้องในการทำงาน

2) ระบบลงเวลาของพนักงานด้วยไลน์บีคอนยังเป็นเพียงต้นแบบของระบบ ถึงแม้ว่าสามารถใช้งานได้แล้ว แต่ยังขาดฟังก์ชันอีกหลายฟังก์ชันที่จะทำให้ระบบสมบูรณ์ ซึ่งในอนาคตสามารถนำระบบนี้ไปพัฒนาต่อยอดและแก้ไขข้อบกพร่องต่าง ๆ เพื่อเพิ่มประสิทธิภาพในการทำงานมากยิ่งขึ้น

เอกสารอ้างอิง

[1] รู้จักกับ Visual Studio Code (วิชวล สตูดิโอ โค้ด) โปรแกรมฟรีจากค่ายไมโครซอฟท์

ที่มา :

https://www.mindphp.com/%E0%B8%9A%E0%B8%97%E0%B8%84%E0%B8%A7%E0%B8%B2%E0%B8%A1/microsoft/4829-visual-studio-code.html

[2] Express คืออะไร ?

ที่มา : https://zakoschool.herokuapp.com/lesson/nodejs/express/home

[3] Express.js เอ็กเพรส ดอทเจเอส คืออะไร Express.js คือ เป็น Web Application Framework ชื่อดังที่ได้รับความนิยมมาก สำหรับทำงานบน platform ของ Node.js

ที่มา :

https://www.mindphp.com/%E0%B8%84%E0%B8%B9%E0%B9%88%E0%B8%A1%E0%B8%B7%E0%B8%AD/73-%E0%B8%84%E0%B8%B7%E0%B8%AD%E0%B8%AD%E0%B8%B0%E0%B9%84%E0%B8%A3/3874-what-is-express-js.html

[4] การกำหนด และใช้งาน Routing ใน Express เบื้องต้น

ที่มา :

https://www.ninenik.com/%E0%B8%81%E0%B8%B2%E0%B8%A3%E0%B8%81%E0%B8%B3%E0%B8%AB%E0%B8%99%E0%B8%94_%E0%B9%81%E0%B8%A5%E0%B8%B0%E0%B9%83%E0%B8%8A%E0%B9%89%E0%B8%87%E0%B8%B2%E0%B8%99_Routing_%E0%B9%83%E0%B8%99_Express_%E0%B9%80%E0%B8%9A%E0%B8%B7%E0%B9%89%E0%B8%AD%E0%B8%87%E0%B8%95%E0%B9%89%E0%B8%99-908.html

[5] รู้จักกับ Middleware ฟังก์ชัน ใน Express และการใช้งานเบื้องต้น

ที่มา :

https://www.ninenik.com/%E0%B8%A3%E0%B8%B9%E0%B9%89%E0%B8%88%E0%B8%B1%E0%B8%81%E0%B8%B1%E0%B8%9A_Middleware_%E0%B8%9F%E0%B8%B1%E0%B8%87%E0%B8%81%E0%B9%8C%E0%B8%8A%E0%B8%B1%E0%B9%88%E0%B8%99_%E0%B9%83%E0%B8%99_Express_%E0%B9%81%E0%B8%A5%E0%B8%B0%E0%B8%81%E0%B8%B2%E0%B8%A3%E0%B9%83%E0%B8%8A%E0%B9%89%E0%B8%87%E0%B8%B2%E0%B8%99%E0%B9%80%E0%B8%9A%E0%B8%B7%E0%B9%89%E0%B8%AD%E0%B8%87%E0%B8%95%E0%B9%89%E0%B8%99-910.html

[6] MongoDB คืออะไร? + สอนวิธีใช้งานเบื้องต้น

ที่มา : https://devahoy.com/blog/2015/08/getting-started-with-mongodb/

[7] Elasticsearch คืออะไร?

ที่มา : https://www.kanouivirach.com/2014/04/elasticsearch-%E0%B8%84%E0%B8%B7%E0%B8%AD%E0%B8%AD%E0%B8%B0%E0%B9%84%E0%B8%A3/

[8] Elasticsearch คืออะไร?

ที่มา : https://www.kanouivirach.com/2014/04/elasticsearch-%E0%B8%84%E0%B8%B7%E0%B8%AD%E0%B8%AD%E0%B8%B0%E0%B9%84%E0%B8%A3/

[9] Merkel Dirk, "Docker: Lightweight linux containers for consistent development and deployment", Linux J., vol. 2014, no. 239