เว็บแอพพลิเคชั่นสำหรับสถานีประจุไฟฟ้าโดยใช้โปรโตคอล OCPP

นายปฐมภัทร์ สูงปานเขา

นายณัฐพงษ์ วิญญูนุรักษ์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของกรศึกษาตามหลักสูตร

ปริญญาวิศวกรรมศาสตร์บัณฑิต

สาขาวิศวกรรมคอมพิวเตอร์ ภาควิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ

ปีการศึกษา 2561

Web Application for Electric Charger Station using OCPP protocol

Mr. Pathompat Sungpankhao

Mr. Nattapong Winyunuluk

A PROJECT REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF BACHELOR OF COMPUTER ENGINEERING DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

FACULTY OF ENGINEERING

KING MONGKUT’S UNIVERSITY OF TECHNOLOGY NORTH BANGKOK

ACADEMIC YEAR 2018

ปริญญานิพนธ์เรื่อง : เว็บแอพพลิเคชั่นสำหรับสถานีประจุไฟฟ้าโดยใช้โปรโตคอล OCPP

โดย : นายปฐมภัทร์ สูงปานเขา

นายณัฐพงษ์ วิญญูนุรักษ์

สาขาวิชา : วิศวกรรมคอมพิวเตอร์

ภาควิชา : วิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะ : วิศวกรรมศาสตร์

อาจารย์ที่ปรึกษา : ผู้ช่วยศาสตราจารย์ ดร.พงษ์ศักดิ์ กีรติวินทกร

ปีการศึกษา : 2561

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ อนุมัติให้  
ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต  
สาขาวิชาวิศวกรรมคอมพิวเตอร์

…………………………………………………..……… หัวหน้าภาควิชาวิศวกรรมไฟฟ้า

(ผู้ช่วยศาสตราจารย์ ดร.นภดล วิวัชรโกเศศ) และคอมพิวเตอร์

…………………………………………………..……… ประธานกรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.พงษ์ศักดิ์ กีรติวินทกร)

…………………………………………………..……… กรรมการ

(ดร.ดนุชา ประเสริฐสม)

…………………………………………………..……… กรรมการ

(อาจารย์โสภณ อภิรมย์วรการ)

ลิขสิทธิ์ของภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

มหาวิทยาลัยเทคโนโลยีพระจอมเกล้า พระนครเหนือ

Project Report Title : Web Application for Electric Charger Station using OCPP protocol

Name : Mr. Pathompat Sungpankhao

Mr. Nattapong Winyunuluk

Major Field : Computer Engineer

Department : Electrical and Computer Engineering

Faculty : Engineering

Project Advisor : Assoc. Prof. Dr. Phongsak Keeratiwintakorn

Academic Year : 2561

Accepted by the Faculty of Engineering, King Mongkut’s University of Technology North Bangkok in Partial Fulfillment of the Requirement for the Degree of Bachelor   
of Computer Engineering

…………………………………………………..……..… Chairperson of Department of Electrical

(Asst. Prof. Dr. Nophadon Wiwatcharagoses) and Computer Engineering

…………………………………………………..……..… Chairperson

(Assoc. Prof. Dr. Phongsak Keeratiwintakorn)

…………………………………………………..……..… Member

(Dr. Danucha Prasertsom)

…………………………………………………..……..… Member

(Mr. Sopon Apiromvorakarn)

Copyright of the Department of Electrical and Computer Engineering, Faculty of Engineering

King Mongkut’s University of Technology North Bangkok

**บทคัดย่อ**

โครงงานนี้มีวัตถุประสงค์เพื่อศึกษาและพัฒนารูปแบบการให้บริการสถานีประจุไฟฟ้า (Electric Charger Station) สำหรับยานพาหนะไฟฟ้า (Electric Vehicle : EV) โดยแบ่งการพัฒนาออกเป็น 2 ส่วน Gateway - Server ทำหน้าที่รับส่งข้อมูลของสถานีประจุไฟฟ้าและเว็บแอปพลิเคชั่น (Web Application) สำหรับให้บริการเจ้าของยานพาหนะไฟฟ้าที่ต้องการใช้บริการสถานีประจุไฟฟ้าผ่านช่องทางออนไลน์ ในส่วนของ Gateway - Server ทางคณะผู้จัดทำได้ทำการศึกษาและพัฒนาโดยใช้โปรโตคอล OCPP (Open Charge Point Protocol) เวอร์ชั่น 1.6 สำหรับสื่อสารกับสถานีประจุไฟฟ้าผ่านแพลทฟอร์ม NodeJs และ Firebase ในภาษา JavaScript ในส่วนของสถานีประจุไฟฟ้าใช้ตัวไมโครคอนโทรลเลอร์บีเกิ้ลโบนส์ (Beagle Bone Black : BBB) สำหรับสื่อสารกับ Gateway - Server และส่วนของเว็บแอปพลิเคชั่นยังมีการให้บริการ ระบบการจอง (Reservation System) ระบบร้องขอการติดตั้งสถานีอัดประจุไฟฟ้าใหม่ (Request Station System) ระบบเติมเงิน (Topup System) ผ่านผู้ให้บริการตัวกลางทางการชำระเงิน (Third Party Payment Gateway)

**Abstract**

This project has an objective to study and develop the model of electric charger stations for electric vehicles. We divide a project development into 2 parts. First, Gateway - Server is responsible for sending and receiving data of electric charger stations Secondly, Web applications is for providing electric vehicles owners, who want to use electric charge stations, via online channels. In the Gateway-Server part, we have studied and developed using the OCPP protocol (Open Charge Point Protocol) version 1.6 for communicating with software electric charge stations using NodeJS and Firebase platforms in JavaScript language. In the electric charge stations part, we Beagle Bone Black (BBB) microcontroller for communication with the Gateway-Server and Web applications, have services which the following, Reservation System, New Request Station System, Top up System and Third-Party Payment Gateway Services.

**กิตติกรรมประกาศ**

โครงงานพิเศษฉบับนี้สามารถสำเร็จลุล่วงได้ด้วยความอนุเคราะห์ ความช่วยเหลือ ให้คำปรึกษา และคำแนะนำในทุกด้านตลอดจนการตรวจสอบ แก้ไขข้อบกพร่องต่าง ๆ ที่มีเป็นอย่างดีมาโดยตลอดจาก ผู้ช่วยศาสตราจารย์ ดร.พงษ์ศักดิ์ กีรติวินทกร จนกระทั่งโครงงานพิเศษฉบับนี้สำเร็จลุล่วงไปได้ด้วยดี ซึ่งทางคณะผู้จัดทำขอกราบขอบพระคุณเป็นอย่างสูงมา ณ โอกาสนี้

ขอกราบขอบพระคุณท่านอาจารย์ทุกท่านในภาควิชาวิศวกรรมไฟฟ้ารวมทั้งอาจารย์ทุกท่าน  
ที่ประสิทธิ์ประสาทวิชาความรู้ให้แก่ทางคณะผู้จัดทำจนกระทั่งสำเร็จการศึกษา รวมทั้งครอบครัวของทางคณะผู้จัดทำที่ให้การสนับสนุนส่งเสริมและให้กำลังใจมาโดยตลอด รวมถึงมิตรภาพ ความรู้สึกดี ๆ   
ที่ได้รับมาโดยตลอดทั้งจากพี่ ๆ เพื่อน ๆ และน้อง ๆ ในภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

สุดท้ายนี้ ทางคณะผู้จัดทำหวังว่าโครงงานพิเศษฉบับนี้จะเป็นประโยชน์กับผู้สนใจไม่มากก็น้อย หากมีข้อผิดพลาดประการใดทางคณะผู้จัดทำขออภัยมา ณ ที่นี้ด้วย

ปฐมภัทร์ สูงปานเขา

ณัฐพงษ์ วิญญูนุรักษ์

**สารบัญ**

หน้า

บทคัดย่อภาษาไทย จ

บทคัดย่อภาษาอังกฤษ ฉ

กิตติกรรมประกาศ ช

สารบัญตาราง ญ

สารบัญภาพ ฎ

บทที่ 1 บทนำ 1

1.1 ความเป็นมาและความสำคัญ 1

1.2 วัตถุประสงค์โครงงาน 2

1.3 ขอบเขตของโครงการ 2

1.4 นิยามเฉพาะ 2

1.5 ประโยชน์ที่คาดว่าจะได้รับ 3

บทที่ 2 ทฤษฏีและงานวิจัยที่เกี่ยวข้อง 4

2.1 OCPP (Open Charge Point Protocol) 4

2.2 WebSocket Protocol (RFC-6455) 7

2.3 RPC (Remote Procedure Call) 7

2.4 JSON (Javascript Object Notation) 10

2.5 เก็บข้อมูลของเว็บ 14

2.6 หลักการส่งข้อมูลแบบพิกกี้แบ็ค 17

2.7 ตัวกลางการรับชำระเงิน 19

2.8 เทคโนโลยีของเว็บไซต์ (Web Technology) 20

2.9 RFID (Radio-Frequency Identification) 21

บทที่ 3 ขั้นตอนการดำเนินงาน 27

3.1 ขั้นตอนการดำเนินงาน 28

3.2 กระบวนการทำงานระบบ Account Information 29

3.3 กระบวนการ Reservation 30

**สารบัญ (ต่อ)**

หน้า

3.4 กระบวนการทำงานของระบบ Billing/Topup ผ่าน API ของ Omise 32

3.5 กระบวนการทำงานของสถานีประจุไฟฟ้า 34

บทที่ 4 ผลการดำเนินการ 37

4.1 ผลการทดลองเข้าใช้เว็บแอพพลิเคชั่น 37

4.2 ผลการทดลองเติมเงินผ่านเว็บแอพพลิเคชั่น 39

4.3 ผลการทดลองการใช้งานสถานีประจุไฟฟ้า 41

4.4 ผลการทดลองยกเลิกใช้งานสถานีประจุไฟฟ้า 46

4.5 ผลการทอดลองระบบการจอง 47

4.6 ผลการทดลองยกเลิกการจอง 49

บทที่ 5 สรุปและวิเคราะห์การดำเนินงาน 51

5.1 สรุปและวิเคราะห์การดำเนินงาน 51

5.2 ข้อเสนอแนะและการดำเนินงานต่อไป 52

เอกสารอ้างอิง 53

ประวัติผู้แต่ง 55

**สารบัญภาพ**

ภาพที่ หน้า

2-1 รูปแบบของโปรโตคอล RPC 8

2-2 โครงสร้างข้อมูลแบบออบเจ็ค 11

2-3 โครงสร้างข้อมูลแบบอาเรย์ 12

2-4 ค่า (Value) ที่สามารถเก็บได้ 12

2-5 ค่าของสายอักขระ (String) 13

2-6 ค่าของตัวเลข (Number) 13

2-7 ไดอะแกรมลำดับการส่งข้อมูลแบบพิกกี้แบ็ค 18

2-8 ขั้นตอนในการจ่ายเงินของบริษัท Omise 19

2-9 ภาพรวมของระบบ RFID 21

2-10 Passive Tag (คีย์การ์ดหอพัก) 22

2-11 Active Tag ที่มีแบตเตอรี่ 23

2-12 ตัวอย่างเครื่องอ่านแบบต่าง ๆ 24

2-13 ความถี่ย่านที่ระบบ RFID ถูกใช้งาน 25

2-14 โครงสร้างการเก็บข้อมูลแบบ JSON 26

3-1 Use case diagram 28

3-2 Process Finite State Machine Diagram ของระบบ Account Information 30

3-3 Reservation Sequence Diagram 31

3-4 Omise Flowchart 33

3-5 ส่วนประกอบของสถานีประจุไฟฟ้า 34

3-6 Finite State Machine Diagram ของ Charger 35

4-1 ข้อมูลผู้ใช้ในฐานข้อมูล Firebase 37

4-2 หน้าเว็บไซต์ Login สำหรับผู้ใช้ 38

4-3 หน้าเว็บไซต์เมื่อ Login เข้าใช้เว็บแอพพลิเคชั่น 39

4-4 เลขบัตรเคดิตที่ใช้ทดสอบระบบเติมเงินจากเว็บไซต์ Omise 40

4-5 หน้าเว็บไซต์แสดงข้อมูลบัตรเคดิตที่ต้องกรอก 40

4-6 หน้า Dashboard จากเว็บไซต์ Omise 41

**สารบัญภาพ (ต่อ)**

ภาพที่ หน้า

4-7 หน้าจอสถานีประจุไฟฟ้าเสดงให้แตะบัตร 41

4-8 หน้าจอสถานีประจุไฟฟ้าเสดงข้อมูลผู้ใช้เมื่อแตะบัตร 42

4-9 หน้าจอสถานีประจุไฟฟ้าแสดงให้กดปุ่มเริ่มชาร์จ 43

4-10 หน้าจอสถานีประจุไฟฟ้าแสดงให้เสียบหัวชาร์จ 44

4-11 หน้าจอสถานีประจุไฟฟ้าแสดงข้อมูลสถานะกำลังชาร์จ 45

4-12 หน้าจอสถานีประจุไฟฟ้าแสดงให้กดปุ่มยกเลิกการชาร์จ 46

4-13 หน้าเว็บไซต์แสดงสถานะ Connector ที่สามารถจองได้ 48

4-14 หน้าเว็บไซต์แสดงสถานะการจองทางเว็บแอพพลิเคชั่น 48

4-15 หน้าจอสถานีประจุไฟฟ้ามีการจองแล้ว 49

4-16 หน้าเว็บไซต์แสดงสถานะยกเลิกการจอง 49

**สารบัญตาราง**

ตารางที่ หน้า

2-1 ความแตกต่างระหว่าง Cookie, SessionStorage และ LocalStorage 16

2-2 Method/Property ทั้งหมดของ WebStorage 17

**บทที่ 1**

**บทนำ**

**1.1** **ความเป็นมาและความสำคัญ**

ในประเทศไทยมีเป้าหมายที่จะอนุรักษ์การใช้พลังงานทั้งหมดโดยถือตามแผนการอนุรักษ์พลังงาน โดยเฉพาะภาคขนส่ง ซึ่งถูกวางเป้าหมายในการอนุรักษ์พลังงานให้ได้ 15,100 พันตัน เทียบเท่ากับน้ำมันดิบ และลดปริมาณการปล่อยก๊าซคาร์บอนไดออกไซด์ให้ได้ 53 ล้านตันต่อปี ภายในปี พ.ศ. 2573   
ทั้งนี้ทางภาครัฐได้มีการส่งเสริมการใช้ยานพาหนะที่ใช้พลังงานไฟฟ้าเป็นมาตรการหนึ่ง  
ในแผนการอนุรักษ์พลังงาน

ปัจจุบันเทคโนโลยีการใช้ยานพาหนะไฟฟ้า (Electric Vehicle) ในการภาคขนส่งเริ่มได้รับ  
ความสนใจมากขึ้น เนื่องจากเทคโนโลยีดังกล่าวอาศัยพลังงานไฟฟ้าซึ่งจะเก็บสะสมไว้ในแบตเตอรี่  
เป็นแหล่งพลังงานหลักในการขับเคลื่อนยานพาหนะดังกล่าว เมื่อเปรียบเทียบกับยานพาหนะในปัจจุบัน  
ที่ใช้เครื่องยนต์สันดาปภายใน ซึ่งใช้น้ำมันหรือก๊าซธรรมชาติเป็นเชื้อเพลิงหลัก ซึ่งจะพบว่ายานพาหนะไฟฟ้าจะไม่ปล่อยมลพิษในรูปของไอเสียสู่สภาพแวดล้อมเลย เมื่อพิจารณาถึงประสิทธิภาพการใช้พลังงาน การใช้เชื้อเพลิงความร้อนมาผลิตไฟฟ้าเพื่อนำมาจ่ายให้กับยานพาหนะไฟฟ้าก็ยังถือว่าคุ้มค่า  
หรือมีประสิทธิภาพของการใช้พลังงานสูงกว่าการนำเชื้อเพลิงความร้อนมาใช้ในยานพาหนะ  
ที่เป็นเครื่องยนต์สันดาปภายในโดยตรงเป็นอย่างมาก

ปัญหาที่ทำให้การใช้ยานพาหนะไฟฟ้ายังไม่สามารถมาแทนที่ยานพาหนะในปัจจุบัน  
ที่ใช้เครื่องยนต์สันดาปภายในได้ในทันที ถึงแม้ว่ารถยนต์ไฟฟ้าที่มีจำหน่ายอยู่ในตลาดและมีสมรรถนะ  
เพียงพอที่จะสามารถนำมาใช้งานได้จริงในชีวิตประจำวันแทนรถยนต์ที่ใช้น้ำมันในปัจจุบันได้แล้ว  
คือ ราคาที่ยังคงสูง และสถานีบริการประจุไฟฟ้าที่มีไม่แพร่หลายอย่างทั่วถึง แต่อย่างไรก็ตามราคารถยนต์ไฟฟ้าแม้จะมีราคาแนวโน้มที่ลดลงถ้าหากในอนาคตมีการใช้กันอย่างแพร่หลายมากยิ่งขึ้น

**1.2 วัตถุประสงค์โครงงาน**

โครงงานชิ้นนี้มีวัตถุประสงค์เพื่อค้นคว้าและพัฒนาแนวทางการให้บริการสถานีประจุไฟฟ้าสำหรับพาหนะไฟ้ฟ้าในส่วนของการรับ-ส่งข้อมูลระหว่างสถานีประจุไฟฟ้ากับระบบศูนย์กลาง  
หรือฐานข้อมูล ตลอดจนเว็บแอพพลิเคชั่นสำหรับให้บริการผู้ใช้งานยานพาหนะไฟฟ้า   
ช่วยในด้านการใช้งานในระบบออนไลน์อย่างเต็มรูปแบบทั้งการเติมเงิน คำนวณค่าไฟฟ้า   
และการจองสถานีอัดประจุไฟฟ้า

**1.3 ขอบเขตของโครงงาน**

1.3.1 ออกแบบและพัฒนาระบบศูนย์กลางและระบบฐานข้อมูลเพื่อติดต่อรับ – ส่งข้อมูลกับ  
สถานีประจุไฟฟ้า โดยพัฒนาตามมาตรฐาน OCPP-J เวอร์ชั่น 1.6

1.3.2 ออกแบบและพัฒนาเว็บแอพพลิเคชั่นสำหรับให้บริการผู้ใช้ยานพาหนะไฟฟ้า   
โดยครอบคลุมผู้ใช้คอมพิวเตอร์ แท็บเล็ต และสมาร์ทโฟน

1.3.3 พัฒนาฟังก์ชันเพิ่มเติมของสถานีประจุไฟฟ้า คือ ระบบการจอง และระบบสั่งการจาก  
ระบบศูนย์กลางมายังสถานีประจุไฟฟ้า

1.3.4 ทดลองกระบวนการทำงานของสถานีประจุไฟฟ้า ตั้งแต่เริ่มต้นเสียบหัวชาร์จ   
จนกระทั่งถอดหัวชาร์จออก ว่ามีการรับส่งข้อมูลระหว่างอุปกรณ์อย่างไร และมีลำดับการทำงาน  
ของแต่ละข้อความอย่างไรบ้าง

**1.4 นิยามคำเฉพาะ**

1.4.1 ยานพาหนะไฟฟ้า (Electric Vehicle) คือ ยานพาหนะที่ขับเคลื่อนโดยมอเตอร์ไฟฟ้า  
แทนการใช้เครื่องยนต์ที่มีการเผาไหม้สันดาปภายใน

1.4.2 สถานีบริการประจุไฟฟ้า (Electric Charger Station) คือ สถานีให้บริการชาร์จ  
ประจุไฟฟ้าสำหรับยานพาหนะไฟฟ้า (Electric Vehicle) และอุปกรณ์ไฟฟ้าอื่น ๆ

1.4.3 IOT (Internet of things) คือ อุปกรณ์อิเล็กทรอนิกส์หรือทุกๆสิ่งที่เชื่อมต่อกับอินเตอร์เน็ตหรือควบคุมอุปกรณ์ผ่านทางอินเตอร์เน็ต

1.4.4 BBB (Beagle Bone Black) คือ Gateway Controller ที่เชื่อมต่ออยู่กับ Controller   
ของเครื่องชาร์จไฟฟ้า

**1.5 ประโยชน์ที่คาดว่าจะได้รับ**

สามารถพัฒนาสถานีประจุไฟฟ้า และเว็บแอพพลิเคชั่นให้ใช้ได้จริงในเชิงพาณิชย์   
เพื่อให้ผู้ใช้บริการยานพาหนะไฟฟ้ามีความสะดวกสบายมากยิ่งขึ้นในการใช้บริการผ่านช่องทางออนไลน์ เช่น การจองการใช้บริการสถานีประจุไฟฟ้าล่วงหน้า และการชำระเงินผ่านช่องทางออนไลน์

**บทที่ 2**

**ทฤษฎีและงานวิจัยที่เกี่ยวข้อง**

โครงงานพิเศษนี้มีวัตถุประสงค์เพื่อศึกษาและพัฒนาการให้บริการสถานีประจุไฟฟ้า  
ของยานพาหานะไฟฟ้า ซึ่งได้สืบค้นเกี่ยวกับรูปแบบการส่งข้อมูล (Protocol) ที่ใช้การรับ - ส่งข้อมูลระหว่างสถานีประจุไฟฟ้า, ระบบศูนย์กลาง, ระบบฐานข้อมูล ตลอดจนเครื่องมือ (Tools)   
และแพลทฟอร์ม (Platform) ที่ใช้ในการพัฒนาเว็บแอพพลิชั่นสำหรับให้บริการผู้ใช้ โดยมีทฤษฎีและงานวิจัยที่เกี่ยวข้อง ซึ่งมีรายละเอียดดังต่อไปนี้

1. OCPP (Open Charge Point Protocol)
2. เว็บซอกเก็ต (WebSocket Protocol : RFC-6455)
3. RPC (Remote Procedure Call)
4. JSON (Java Script Object Notation)
5. ที่เก็บข้อมูลของเว็บ (Web Storage)
6. หลักการส่งข้อมูลแบบพิกกี้แบ็ค (Piggybacking Principle)
7. ตัวกลางการรับชำระเงิน (Payment Gateway)
8. เว็บไซต์แบบหน้าเดียว (Single Page Application)
9. RFID (Radio-Frequency Identification)
10. ระบบฐานข้อมูลที่ไม่ใช้ภาษา SQL (NoSQL Databases)

**2.1 OCPP (Open Charge Point Protocol)**

OCPP (Open Charge Point Protocol) เป็นมาตรฐานที่ใช้ในการติดต่อสื่อสารระหว่าง สถานีประจุไฟฟ้า (Charge Point) และระบบศูนย์กลาง (Central System) ถูกออกแบบมาเพื่อใช้สำหรับสถานีประจุไฟฟ้าทุกชนิด OCPP นั้นจะมีแนวทางการพัฒนา 2 ทาง คือ OCPP-S ที่ใช้โปรโตคอล SOAP และ XML   
ในการทำงาน OCPP-J คือโปรโตคอลที่ใช้ JSON และ Web Socket ในการทำงาน

       OCPP จะประกอบด้วยมาตรฐานที่ระบุว่าแต่ละข้อความนั้นควรมีรูปแบบการส่งอย่างไร มีการกำหนดชนิดของตัวแปร ชื่อของตัวแปร ความยาวของข้อมูลในตัวแปร ข้อความทั้งหมดจะถูกแบ่งเป็น

ประเภทย่อย ๆ คือ Cores, Firmware Management, Local Auth List Management, Reservation, Smart Charging, Remote Trigger

2.1.1 ฟังก์ชันหลัก

ฟังก์ชันหลัก (Cores) เป็นข้อความสำหรับการรับ-ส่งข้อมูลพื้นฐาน ข้อความที่ถูกจัดประเภทเป็นฟังก์ชันหลัก (Cores) มีทั้งหมด 15 ข้อความ ดังนี้

* Authorize สำหรับการยืนยันตัวตน
* BootNotification สำหรับว่ามีการลงทะเบียนสถานีประจุไฟฟ้าภายในระบบศูนย์กลางหรือไม่
* ChangeAvailability สำหรับการเปลี่ยนแปลงความพร้อมใช้งานที่สถานีประจุไฟฟ้า
* ChangeConfiguration สำหรับการกำหนดค่าให้กับสถานีประจุไฟฟ้า
* ClearCache สำหรับการล้างข้อมูลโดยที่ระบบศูนย์กลางจะร้องขอมาที่สถานีประจุไฟฟ้าว่าจะยอมรับไหม
* DataTransfer สำหรับการถ่ายโอนข้อมูล
* GetConfiguration สำหรับการดูค่าที่ถูกกำหนด
* Heartbeat สำหรับการเช็คว่าสถานีประจุไฟฟ้ายังทำงานอยู่
* RemoteStartTransaction สำหรับการร้องขอการเริ่มต้นชาร์จ
* RemoteStopTransaction สำหรับการร้องขอการสิ้นสุดชาร์จ
* Reset สำหรับการกลับสู่ค่าเริ่มของสถานีประจุไฟฟ้า
* StartTransaction สำหรับการต้นเริ่มชาร์จ
* StatusNotification สำหรับแสดงสถานะของสถานีประจุไฟฟ้า
* StopTransaction สำหรับการสิ้นสุดการชาร์จ
* UnlockConnector สำหรับการแสดงสถานะของสถานีประจุไฟฟ้าว่าปลดล็อกตัวเชื่อมต่อหรือไม่
* MeterValues สำหรับแสดงค่าพลังงานที่ถูกชาร์จไป

2.1.2 ฟังก์ชันสำหรับการอัพเดทเฟิร์มแวร์

ฟังก์ชันสำหรับการอัพเดทเฟิร์มแวร์ (Firmware Management) เป็นข้อความสำหรับ  
การสั่งอัพเดทเฟิร์มแวร์ หรือการเรียกดู Log ของสถานีประจุไฟฟ้า ประกอบด้วยข้อความทั้งหมด   
4 ข้อความดังนี้

* GetDiagnostics สำหรับการวินิจฉัย
* DiagnosticsStatusNotification สำหรับแสดงสถานะการวินิจฉัย
* FirmwareStatusNotification สำหรับแสดงสถานะการติดตั้ง Firmware
* UpdateFirmware สำหรับอัปโหลด Firmware

2.1.3 ฟังก์ชันสำหรับการยืนยันตัวตนแบบออฟไลน์

ฟังก์ชันสำหรับการยืนยันตัวตนแบบออฟไลน์ (Local Auth List Management) เป็นข้อความสำหรับการใช้งานสถานีอัดประจุไฟฟ้าขณะไม่ได้เชื่อมต่ออินเตอร์เน็ต เช่น การแคชข้อมูล Authorize เป็นต้น ประกอบด้วยข้อความทั้งหมด 2 ข้อความ ดังนี้

* GetLocalListVersion สำหรับแสดงรายชื่อสถานีประจุไฟฟ้าที่ออฟไลน์
* SendLocalList สำหรับแสดงรายชื่อสถานีประจุไฟฟ้าที่กลับมาออนไลน์

2.1.4 ฟังก์ชันสำหรับการจอง

ฟังก์ชันสำหรับการจอง (Reservation) เป็นข้อความสำหรับการจองสถานีประจุไฟฟ้า ประกอบด้วยข้อความทั้งหมด 2 ข้อความ ดังนี้

* ReserveNow สำหรับการจองสถานีประจุไฟฟ้า
* CancelReservation สำหรับการยกเลิกจองสถานีประจุไฟฟ้า

2.1.5 ฟังก์ชันสำหรับการชาร์จอัจฉริยะ

ฟังก์ชันสำหรับการชาร์จอัจฉริยะ (Smart Charging) เป็นข้อความสำหรับการจัดการ Charging Profile ประกอบด้วย 3 ข้อความดังนี้

* ClearChargingProfile สำหรับการล้างโปรไฟล์ของสถานีประจุไฟฟ้า
* GetCompositeSchedule สำหรับคำนวณการจ่ายไฟที่คาดว่าจะได้สำหรับตัวเสียบของสถานีประจุไฟฟ้า
* SetChargingProfile สำหรับเซ็ตข้อมูลการชาร์จ

2.1.6 ฟังก์ชันสำหรับการรีโมต

ฟังก์ชันสำหรับการชาร์จการรีโมต (Remote Trigger) เป็นข้อความสำหรับการเข้าใช้งานสถานีประจุไฟฟ้าระยะไกล ประกอบด้วยข้อความ 1 ข้อความดังนี้

* TriggerMessage สำหรับการเข้าใช้งานสถานีประจุไฟฟ้าระยะไกล

**2.2 WebSocket Protocol (RFC-6455)**

WebSocket เป็นวิธีการติดต่อสื่อสาร (Protocol) ในคอมพิวเตอร์ ซึ่งสามารถรับส่งข้อมูลได้แบบสองทาง คือในเวลาเดียวกันนั้นสามารถเป็นได้ทั้งผู้รับและผู้ส่ง (Full Duplex) ผ่านช่องทาง TCP WebSocket และถูกกำหนดมาตรฐานโดย IETF ที่ได้รับมาตรฐาน RFC 6455 ในปี ค.ศ. 2011

WebSockets เป็น Services ที่ทำหน้าที่เป็น Protocol ของฝั่ง Server เมื่อมี Client   
ทำการเชื่อมต่อเข้ามา Services ดังกล่าวจะติดตาม Client เพื่อตรวจสอบว่า Client ยังทำการเชื่อมต่อและพร้อมที่จะรับ-ส่งข้อมูลให้ตลอดเวลา โดยที่ไม่จำเป็นว่าฝั่ง Client จะมีการ Request   
มาหรือไม่ ฉะนั้นเมื่อฝั่ง Server มีข้อมูลมาใหม่ ๆ ตัว Services จะทำหน้าที่ Push ข้อมูลเพื่อส่ง  
ให้กับทุก ๆ Client ที่เชื่อมต่ออยู่ในขณะนั้น ซึ่งสามารถตรวจสอบได้ว่า มี Client ใดที่กำลังทำการเชื่อมต่ออยู่ ซึ่งแตกต่างกับการ Request จาก Client โดยทั่วไปที่เราจะต้องใช้ Client ทำการ Request   
ส่งข้อความ Post หรือ Get ไปยัง Server จากนั้นรอให้ Server ทำการ Response ค่ากลับมา   
ซึ่ง Server ก็จะไม่มีทางรู้ว่า Client ได้ทำการปิดการเชื่อมต่อไปแล้วหรือไม่

การทำงานของ Websocket เริ่มแรกเมื่อหน้าเว็บถูกโหลด Client และ Server จะมีการเชื่อมต่อไว้ จากนั้นก็จะใช้ช่องทางนี้ในการรับ-ส่งข้อมูลกัน ซึ่งจะไม่เกี่ยวกับ HTTP request อื่น ๆ โดยระหว่างเชื่อมต่อก็ส่งข้อความชนิด Heartbeat ไว้เพื่อดูแล Session ทำให้ลด Request ที่ต้องส่งได้   
ส่งผลให้ลด Overhead ค่อนข้างมากเมื่อเทียบกับ Http Request ทั่วไป

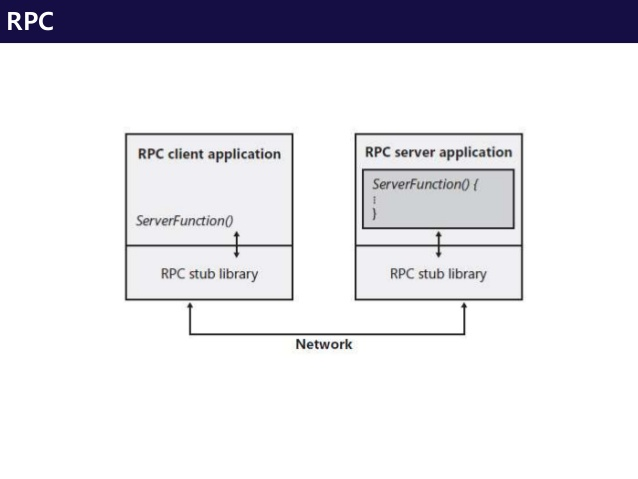
WebSockets ยังคงรองรับเฉพาะเบราว์เซอร์บางตัวเท่านั้น เช่น Google Chrome, Firefox และ Safari เป็นต้น ส่วน Internet Explorer, Opera หรือเบราว์เซอร์อื่น ๆ ต้องตรวจสอบเวอร์ชั่นปัจจุบันว่ารองรับโปรโตคอล WebSocket หรือไม่ โดยปัจจุบันยังมี Socket.io ซึ่งเป็นโมดูลที่เพิ่มเติมเข้ามาทำให้เบราว์เซอร์ที่ล้าสมัย เช่น IE6, IE7 สามารถรัน WebSocket ได้

**2.3 RPC (Remote Procedure Call)**

RPC (Remote Procedure Call) เป็นโปรโตคอลหนึ่งที่ใช้รูปแบบการจำลองเป็นแบบ Client/Server โดยเป็นปฏิบัติการแบบพร้อมกันกล่าวคือ โปรแกรมที่ขอเป็น Client จะทำหน้าที่ขอเรียกใช้บริการและโปรแกรมที่ให้บริการเป็น Server จะทำหน้าที่ตอบรับคำขอใช้บริการที่ถูกเรียกใช้กลับไป

RPC มีแนวความคิดอยู่ว่าจะต้องทำให้การเรียกใช้ Procedure ของผู้ให้บริการ นั้นมีความใกล้เคียงกับการเรียกใช้ฟังก์ชันทั่วไปให้มากที่สุด

จากภาพที่ 2-1 การเรียกใช้โพรซีเดอร์ของแม่ข่ายที่อยู่ไกล โปรแกรมของผู้ใช้บริการจะต้องถูกนำมาเชื่อมต่อเข้ากับโพรซีเดอร์ที่เป็นไลบราลี่ (Library Procedure) เรียกว่า Client Stub ทางฝั่งผู้ให้บริการ จะมีโพรซีเดอร์ที่เป็นไลบราลี่ เรียกว่า “Server Stub”



**ภาพที่ 2-1** รูปแบบของโปรโตคอล RPC

2.3.1 RPC Framework ใน OCPP-J

RPC Framework เป็นการนำเอาแนวคิดของ RPC มาทำการสร้างเป็นรูปแบบของข้อมูลที่ใช้ระหว่างสถานีอัดประจุไฟฟ้ากับระบบศูนย์กลาง โดยมีโปรโตคอล OCPP และ WebSocket   
เป็นโปรโตคอลที่ใช้รับ - ส่งข้อมูล ใน RPC Framework จะแบ่งเป็น 2 ชนิด คือ ผู้เรียกใช้โพรซีเยอร์ (Caller) และผู้ถูกเรียกใช้ (Callee) โดยทั้งสถานีอัดประจุไฟฟ้าและระบบศูนย์กลางสามารถเป็นได้  
ทั้งผู้เรียกใช้และผู้ถูกเรียกใช้

ในแต่ละส่วนของข้อความ RPC Message จะประกอบด้วย 2 ส่วนที่เหมือนกัน   
คือ ชนิดของข้อความ (Message Type) , ไอดีเฉพาะของข้อความ (Message ID) ส่วนประกอบ  
ที่เหลือจะไม่เหมือนกันตามแต่ชนิดของข้อความโดยแบ่งข้อความออกเป็น 3 ชนิด ดังนี้

**1) CALL**

ชนิดของข้อความ Call จะใช้ในการส่งจะประกอบด้วย 4 ส่วน คือ ชนิดของข้อความ   
(Message Type ID), ไอดีเฉพาะของข้อความ (Uniquid), ชื่อของฟังก์ชัน (Action) และข้อมูลที่บรรจุ (Payload) ตามลำดับ ทำให้มีหน้าตาของข้อมูลในรูปแบบของ JSON ดังนี้

[<MessageTypeId>,<UniqueId>,<Action>,<Payload>]

จากตัวอย่างด้านล่าง ข้อมูล CALL ในข้อความ BootNotification ส่วนแรก ชนิดของข้อความเป็น 2 บ่งบอกว่าคือข้อความที่ส่งไป ส่วนที่สอง ไอดีเฉพาะของข้อความจะถูก Generate ขึ้นมาเอง ส่วนที่สาม ชื่อของฟังก์ชันเป็น BootNotification ส่วนที่สี่ ข้อมูลที่ถูกบรรจุไว้ด้วย Key:Value   
รูปแบบของ JSON โดยข้อมูลที่จะบรรจุ จะกำหนดไว้ตามโปรโตคอล OCPP ว่าใส่อะไรได้บ้าง  
ใน Payload

[2,

"19223201",

"BootNotification",

{"chargePointVendor": "VendorX", "chargePointModel": "SingleSocketCharger"}

]

**2) CALLRESULT**

ชนิดของข้อความ CallResult จะใช้ในการตอบกลับข้อความ CALL จะประกอบด้วย 3 ส่วน คือ ชนิดของข้อความ (Message Type ID), ไอดีเฉพาะของข้อความ (UniqueId) และข้อมูลที่บรรจุตอบกลับ (Payload) ตามลำดับ ทำให้มีหน้าตาของข้อมูลในรูปแบบของ JSON ดังนี้

[<MessageTypeId>,<UniqueId>,<Payload>]

จากตัวอย่างด้านล่าง ข้อมูล CALLRESULT ในข้อความ BootNotification ส่วนแรก   
ชนิดของข้อความเป็น 3 บ่งบอกว่าคือข้อความตอบกลับ ส่วนที่สอง ไอดีเฉพาะของข้อความ  
จะถูก Generate ขึ้นมาเอง ส่วนที่สาม ข้อมูลที่ถูกบรรจุไว้ด้วย Key:Value รูปแบบของ JSON   
โดยข้อมูลที่จะบรรจุ จะกำหนดไว้ตามโปรโตคอล OCPP ว่าใส่อะไรได้บ้างใน Payload   
จากตัวอย่างใน Payload มีสถานะ เวลาปัจจุบันของการตอบกลับ ช่วง Heart Beat

[3,

"19223201",

{"status":"Accepted", "currentTime":"2013-02-01T20:53:32.486Z", "heartbeatInterval":300}

]

**3) CALLERROR**

ข้อความชนิด CALLERROR จะเกิดได้จาก 2 สถานการณ์ คือ เกิดความผิดพลาดในการส่งข้อมูล และข้อความไม่ได้อยู่ในมาตรฐานที่กำหนดไว้ จะประกอบด้วย 5 ส่วน คือ ชนิดของข้อความ   
(Message Type ID) , ไอดีเฉพาะของข้อความ (UniqueId), หมายเลขข้อผิดพลาด (Error Code), คำอธิบายข้อผิดพลาด (Error Description) และรายละเอียดข้อผิดพลาด (Error Detail) ตามลำดับ   
ทำให้มีหน้าตาของข้อมูลในรูปแบบของ JSON ดังนี้

[<MessageTypeId>, "<UniqueId>", "<errorCode>", "<errorDescription>", {<errorDetails>}]

**2.4 JSON (Javascript Object Notation)**

JSON (JavaScript Object Notation) คือ รูปแบบของข้อมูลที่ใช้สำหรับแลกเปลี่ยนข้อมูลที่มีขนาดเล็ก รูปแบบข้อมูลสามารถทำความเข้าใจได้ง่าย รูปแบบนี้จะถูกสร้าง และอ่านโดยเครื่องคอมพิวเตอร์ได้ไม่ยากจนเกินไป โดยมาตรฐานของ JSON ถูกกำหนดภายใต้ภาษา JavaScript (JavaScript Programming Language, Standard ECMA-262 3rd Edition – December 1999.) เป็นรูปแบบข้อมูลตัวอักษรที่มีความเป็นอิสระอย่างสมบูรณ์ แต่รูปแบบนี้จะมีหลักของการเขียนที่นักเขียนโปรแกรมภาษาต่างๆ คุ้นเคยไม่ว่าจะเป็นภาษา C, C++, C#, Java, Javascript, Perl, Python และอื่น ๆ คุณสมบัติเหล่านี้ทำให้ JSON ถือเป็นภาษาแลกเปลี่ยนข้อมูลที่มีสมบูรณ์แบบ ในการทำงานหลายอย่างกับ Javascript จะพบว่า JSON เข้ามาเกี่ยวข้องด้วย ยกตัวอย่างการทำงานกับ Script หลาย ๆ ตัวที่มีการเรียกข้อมูลแบบ AJAX ก็มักจะส่งข้อความตอบกลับมาในรูปแบบ JSON

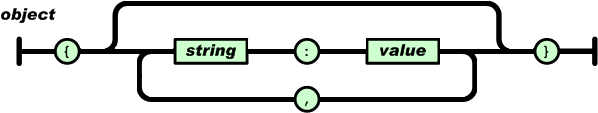
มาตรฐานของฟอร์แมต JSON คือ RFC 4627 มี Internet Media Type เป็น Application/JSON และมีนามสกุลของไฟล์เป็น .json

มาตรฐานของฟอร์แมต JSON คือ RFC 4627 มี Internet Media Type เป็น Application/JSON และมีนามสกุลของไฟล์เป็น .json ปัจจุบัน JSON นิยมใช้ในเว็บแอปพลิเคชัน โดยเฉพาะ AJAX   
โดย JSON เป็นฟอร์แมตทางเลือกในการส่งข้อมูล นอกเหนือไปจาก XML ซึ่งนิยมใช้ในอดีต สาเหตุที่ JSON เริ่มได้รับความนิยมเป็นเพราะกระชับและเข้าใจง่ายกว่า XML

JSON สามารถสร้างได้ 2 รูปแบบ คือ 1. การจัดเก็บในชุดข้อมูลที่มีชื่อข้อมูลและข้อมูลคู่กัน   
ในภาษาต่าง ๆ ข้อมูลจะจัดอยู่ในรูปแบบของ Object , Record, Struct, Dictionary, Hash Table, Keyed List หรือ Assosiative Array 2. ลำดับของค่าข้อมูล ในภาษาโปรแกรมส่วนใหญ่ จะจัดอยู่ในรูปแบบของ Array, Vector, List หรือ Sequence ใน JSON มีรูปแบบการเก็บข้อมูล ดังนี้

2.4.1 Object

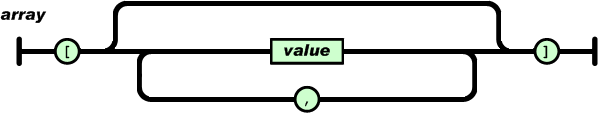
เป็นชุดของข้อมูลที่มีชื่อข้อมูลและค่าของข้อมูลนั้นคู่กัน ซึ่งจะถูกเริ่มต้นด้วยเครื่องหมาย {   
และจะปิดท้ายข้อมูลด้วยเครื่องหมาย } ข้อมูลแต่ละค่าจะมีเครื่องหมาย : (Colon) ระหว่างชื่อข้อมูล (String) กับค่าของข้อมูล (Value) และแต่ละข้อมูลจะมีเครื่องหมาย , คั่น ดังแสดงในภาพที่ 2-2



**ภาพที่ 2-2** โครงสร้างข้อมูลแบบออบเจ็ค

2.4.2 Array

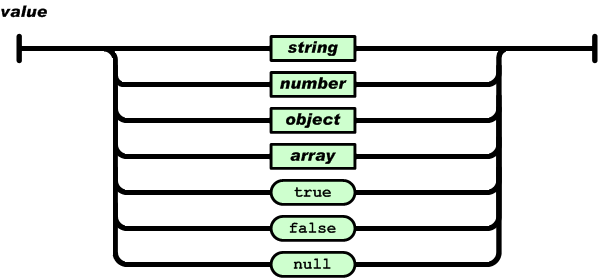
อาเรย์ (Array) เป็นลำดับของข้อมูล ซึ่งจะถูกเริ่มต้นด้วยเครื่องหมาย [ และจะจบด้วยเครื่องหมาย (Value) ] แต่ละค่าของข้อมูลจะถูกคั่นด้วยเครื่องหมาย , ดังแสดงในภาพที่ 2-3



**ภาพที่ 2-3** โครงสร้างข้อมูลแบบอาเรย์

2.4.3 Value

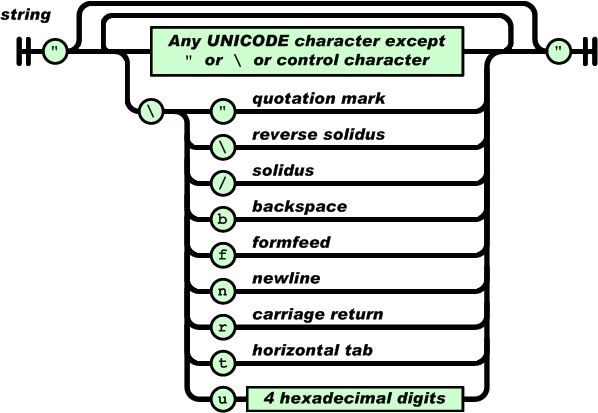
ค่า (Value) เป็นสายอักขระ (String) ที่อยู่ในเครื่องหมาย “”, ตัวเลข, ค่าทางตรรกศาสตร์, NULL , ออบเจ็ค (Object) หรืออาเรย์ (Array) ซึ่งโครงสร้างสามารถวางซ้อนกันได้ ดังแสดงในภาพที่ 2-4



**ภาพที่ 2-4** ค่า (Value) ที่สามารถเก็บได้

2.4.4 String

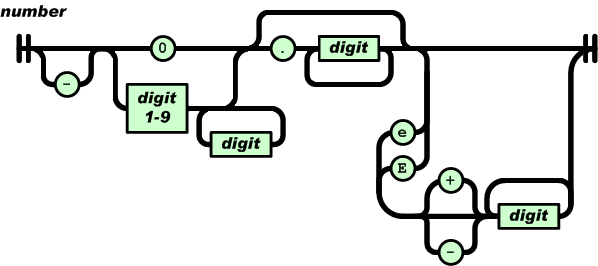
สายอักขระ (String) เป็นลำดับของตัวอักษรตั้งแต่ 0 ตัวอักษรหรือมากกว่า ซึ่งอยู่ภายใต้เครื่องหมายคำพูด (“”) และจะใช้เครื่องหมาย ในการใส่เครื่องหมายกำกับต่าง ๆ ซึ่งจะมีลักษณะคล้ายกับสายอักขระ (String) ในภาษา C หรือภาษา Java ดังแสดงในภาพที่ 2-5



**ภาพที่ 2-5** ค่าของสายอักขระ (String)

2.4.5 Number

ตัวเลข (Number) นั้นมีความคล้ายคลึงกับ Number ในภาษา C หรือภาษา Java แต่จะไม่สามารถใช้เลขฐาน 8 กับเลขฐาน 16 ได้ ตัวอย่างเช่น 4.5E76, 4.5E+76, 4.5e+76 และ 4.5e76 เป็นต้น ดังแสดงในภาพที่ 2-6



**ภาพที่ 2-6** ค่าของตัวเลข (Number)

2.4.6 โครงสร้างของ JSON

JSON จัดอยู่ในหมวดภาษาโปรแกรม แต่มักจะถูกมองว่าเป็นภาษาที่ใช้ในการแลกเปลี่ยนข้อมูลมากกว่า มีโครงสร้างคล้ายคลึงกับ Object ในภาษา Javascript ประกอบด้วย Key และ Value ซึ่ง Value สามารถเป็นข้อมูลได้หลายชนิดทั้ง String, Number, Array หรือ Object ในการใช้งาน JSON นั้นมีประโยชน์ที่สามารถส่งข้อมูลข้ามภาษาได้ โดยส่งข้อมูลผ่านรูปแบบ JSON ตัวอย่างของโครงสร้าง JSON ดังตัวอย่างด้านล่าง โดยจะมีข้อมูลเป็น firstname, lastname และ address โดยในส่วนของ address จะมีค่ามากกว่าหนึ่ง จึงมีข้อมูลภายในอีก ประกอบด้วย address1, province และ country

[

      {"firstname" : "name",

       "lastname": "name",

       "address" : [

                          {

                            "address1" : "address",

                            "province" : "bangkok",

                            "country" : "Thailand"

                          }

                       ]

        }

 ]

**2.5 ที่เก็บข้อมูลของเว็บ**

ที่เก็บข้อมูลของเว็บ (Web Storage) ใช้สำหรับจัดเก็บข้อมูลที่ใช้งานของเบราว์เซอร์เอาไว้  
ที่ฝั่ง Client ลักษณะคล้ายกับ Cookies แต่ Web Strorage ถูกออกแบบมาเพื่อให้ใช้กับภาษาโปรแกรมมิ่งฝั่ง Client โดยเฉพาะ ทำให้มีข้อได้เปรียบกว่า Cookie ตรงที่ HTML5 ใช้ JavaScript เก็บค่าตัวแปรที่ Client ด้วย LocalStorage และ SessionStorage ในการเขียนโปรแกรมที่ทำงานบนเว็บในปัจจุบันจะเห็นได้ว่ามีการพัฒนาส่วนติดต่อผู้ใช้ (User Interface : UI) ที่สามารถทำงานได้อย่างรวดเร็ว สามารถตอบสนองข้อมูลได้ทันที ส่วนหนึ่งเป็นเพราะมีการจัดการกับค่าตัวแปรที่ต้องการนำไปกับการจัดการกับ UI ต่าง ๆ ได้ในทันที ซึ่งในอดีตการส่งค่าตัวแปรไปยัง Web Page ต่าง ๆ อาจจะต้องส่งผ่าน Cookies ของ JavaScript หรืออาจจะส่งไปยัง Web Server แล้วภาษาฝั่ง Server (เช่น PHP) เก็บค่าแล้วจึงส่งกลับมายัง Client ซึ่งจะเห็นว่าในหลาย ๆ ครั้งการเรียกค่าตัวแปรมาใช้งานค่อนข้างจะยุ่งยาก

ใน HTML5 จึงได้ออกแบบ Web Storage เข้ามาจัดการกับปัญหาเหล่านี้ โดยในรูปแบบของ Storage จะมีอยู่ 2 ประเภทคือ LocalStorage และ SessionStorage การทำงานของทั้ง 2 ตัวนี้จะไม่แตกต่างกันมาก โดย LocalStorage จะมีหลักการทำงานเหมือน Cookies คือจะอยู่ตลอดไป แม้ว่าจะปิด Web Browser หรือ Restart เครื่องก็ตาม แต่ SessionStorage จะมีหลักการเหมือนกับ Session คือจะหายไป เมื่อทำการปิด Web Browser หรือ Restart

จากเหตุผลที่กล่าวไปทำให้ผู้ใช้งานสามารถเลือกได้ว่าจะใช้งาน Web Storage ชนิดใด  
และมีข้อสังเกตคือตัว Web Storage จะถูกจัดเก็บไว้ที่ Client ดังนั้นข้อมูลหรือตัวแปรที่จัดเก็บ  
จะมีความปลอดภัยที่ต่ำมาก ดังนั้นในการใช้งานควรจะเลือกจัดเก็บค่าที่คิดว่าไม่สำคัญ สามารถเปิดเผยได้โดยไม่มีผลในด้านความปลอดภัย เมื่อถูกคุกคามจากผู้ที่ไม่ประสงค์ดี และอีกความสามารถหนึ่งของ Web Storage คือสามารถเก็บข้อมูลขนาดใหญ่ได้ถึง 5 MB โดยสามารถเก็บค่าได้เฉพาะ String หรือข้อความเท่านั้น

2.5.1 ความแตกต่างระหว่าง Web Storage และ Cookies

นักพัฒนาเว็บแอพพลิเคชั่นส่วนใหญ่มักใช้ Cookies ในการจัดเก็บข้อมูลของผู้ใช้งาน   
แต่เนื่องจาก Cookies ถูกสร้างขึ้นมาเพื่อประโยชน์สำหรับภาษา Script ทางฝั่ง Server เป็นหลัก ทำให้มีข้อจำกัดหลายอย่าง แต่สำหรับ HTML เวอร์ชั่น 5 ยังมีอีกวิธีในการเก็บข้อมูลชนิดหนึ่งที่น่าสนใจ   
คือการใช้ Web Storage โดยการเก็บข้อมูลชนิดดังกล่าวไม่ได้ถูกสร้างมาเพื่อทดแทน Cookies   
เลยทีเดียว อาจมีบางแอพพลิเคชั่นที่ต้องการใช้ประโยชน์จาก Cookies จึงไม่สามารถบอกได้ว่าการเก็บข้อมูลแบบ Cookies หรือ Web Storage ดีกว่ากัน แต่ขึ้นอยู่กับว่าจะเอาไปใช้ในแอพพลิเคชั่นประเภทใดมากกว่า ซึ่งทางผู้จัดทำได้ทำตารางอธิบายความแตกต่าง ดังนี้

**ตารางที่ 2-1** ความแตกต่างระหว่าง Cookie , SessionStorage และ LocalStorage

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Cookie** | **SessionStorage** | **LocalStorage** |
| **การส่งข้อมูล** | ข้อมูลถูกส่งทุกครั้งที่ทำการส่ง HTTP Request | ข้อมูลไม่ได้ถูกส่งไปกับ HTTP Request จะถูกใช้เมื่อเรียกใช้เท่านั้น | ข้อมูลไม่ได้ถูกส่งไปกับ HTTP Request จะถูกใช้เมื่อเรียกใช้เท่านั้น |
| **ขนาดของข้อมูลที่เก็บได้** | 4 KB | 5 MB | 5 MB |
| **ความเร็ว** | ช้า เนื่องจาก Overhead ของ HTTP Request | เร็ว | เร็ว |
| **ระยะเวลา** | แล้วแต่การกำหนด timeout | จะหายไปเมื่อปิด Tab หรือ Window | คงอยู่ตลอด จนกว่าจะทำลาย |

2.5.2 การใช้งาน Web Storage

เนื่องจาก Web Storage เป็น Features ของ Web Browser ที่รองรับ HTML5 ดังนั้นในการพัฒนาเว็บไซต์ในยุคปัจจุบันจะสามารถใช้งาน Web Storage ได้ในทันที ซึ่งจะอยู่ที่ปัจจัยของ Version บน Web Browser โดยการใช้งาน LocalStorage และ SessionStorage นั้นก็ต้องทำความเข้าใจว่าทั้งสองต่างก็เป็น Object สำหรับจัดเก็บข้อมูล และทั้งคู่ยังมี Methods และ Property ที่เหมือนกันทุกประการ ดังนี้

**ตารางที่ 2-2** Method/Property ทั้งหมดของ WebStorage

|  |  |
| --- | --- |
| **ชื่อ Method / Property** | การใช้งาน |
| **setItem(key, value)** | เก็บข้อมูล |
| **getItem(key)** | ดึงข้อมูลที่เก็บไว้ออกมาใช้ตาม key ที่ระบุ |
| **removeItem(key)** | ลบข้อมูลที่เคยเก็บไว้ตาม key ที่ระบุ |
| **key(n)** | แสดงชื่อของ key ตาม index ที่ระบุ (เริ่มต้นที่ 0) |
| **clear()** | ลบข้อมูลที่จัดเก็บไว้ทั้งหมด (เฉพาะโดเมนนั้น ๆ) |
| **Length** | แสดงจำนวนข้อมูลที่จัดเก็บไว้ (เฉพาะโดเมนนั้น ๆ) |

2.5.3 การนำไปประยุกต์ใช้งาน

นักพัฒนาสามารถนำความสามารถของ Web Storage ไปประยุกต์ใช้กับ Web Application   
ได้หลายรูปแบบด้วยกันดังนี้

- เก็บค่า Preferences ต่าง ๆ ที่ users สามารปรับแต่งได้เอง เช่น ขนาดตัวอักษร หรือมุมมอง เป็นต้น

- เก็บสถานะการใช้งานล่าสุด เพื่อเก็บรักษาสถานะของ Web Application หรือค่า Input   
ต่าง ๆ ในฟอร์มให้คงอยู่เหมือนเดิมแม้ว่าจะปิด Web Application ไปแล้ว

- แคช (Cache) ข้อมูลส่วนที่ไม่ต้องการการอัพเดทแบบ Real-Time เพื่อลดภาระของ server ให้น้อยลง

**2.6 หลักการส่งข้อมูลแบบพิกกี้แบ็ค**

หลักการส่งข้อมูลแบบพิกกี้แบ็ค (Piggybacking Principle) เป็นหลักการที่ใช้ในรูปแบบ  
การส่งข้อมูลแบบสองทาง (Two-way Communication) เมื่อผู้รับได้รับ Data Frame จะยังไม่ส่ง ACK   
กลับทันที ผู้รับจะรอ Network ส่ง Data Packet มาก่อน แล้วจึงแนบ Packet + ACK ส่งกลับไปยังผู้ส่ง

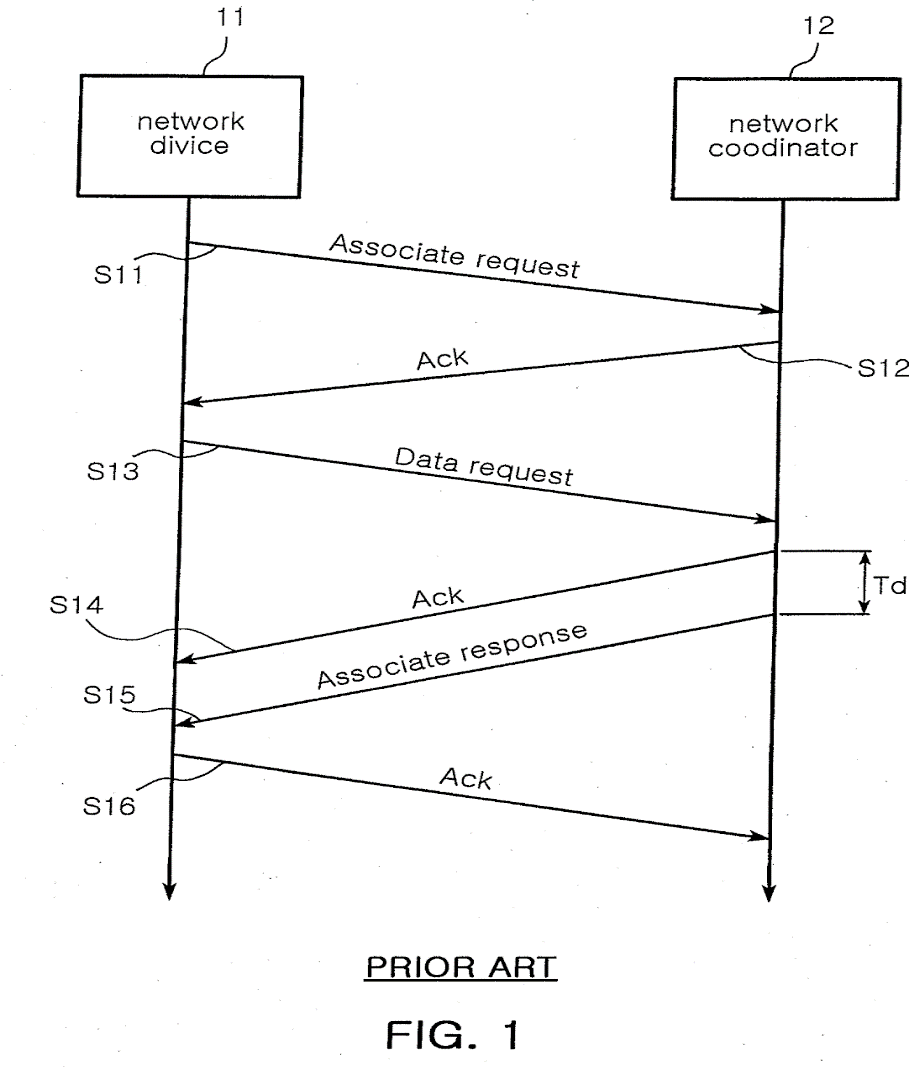
หลักการส่งข้อมูลแบบพิกกี้แบ็คจะมีความแตกต่างกับ Sliding Window Protocol   
ที่ใช้ใน OSI Model เล็กน้อย โดยใน Data Frame จะนำข้อมูลที่ต้องการส่งมารวมกับ Acknowledgment เมื่อ A ต้องการส่งข้อมูลไปให้ B  จะส่ง Data ตามด้วย ACK ข้อกำหนด  
ของหลักการส่งข้อมูลแบบอาศัยไปด้วยจะมีอยู่ 3 ประการ คือ

1. ถ้า Station A ต้องการส่งทั้งข้อมูลและ ACK จะสามารถส่งได้ทั้งสองอย่าง

2. ถ้า Station A ต้องการส่งแค่ ACK ก็จะแบ่ง ACK frame ส่งไปได้

3. ถ้า Station A ต้องการส่งแค่ข้อมูล ให้นำ ACK Frame สุดท้ายส่งไปกับข้อมูล Station B จะไม่สนใจ ACK Frame จาก Station A เนื่องจากเป็น ACK Frame ซ้ำ

จากภาพที่ 2-7 network divice ส่งคำขอการเชื่อมต่อไปยัง network coodinator เมื่อเชื่อมต่อสำเร็จ network coodinator จะส่ง Ack กลับมา และ network divice ส่ง Data request เพื่อขอข้อมูล network coodinator จะตอบกลับมาพร้อมกับข้อมูล และ Ack เมื่อ network divice ได้รับ จะส่ง Ack กลับไปเพื่อบอก network coodinator ว่าได้รับข้อมูลแล้ว



**ภาพที่ 2-7** ไดอะแกรมลำดับการส่งข้อมูลแบบพิกกี้แบ็ค

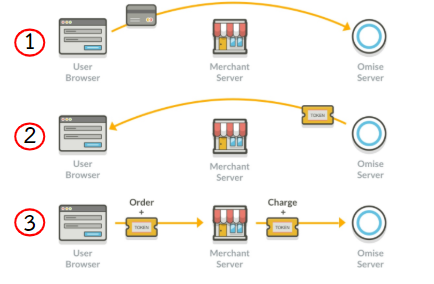
2.6.1 ข้อดีและข้อเสียของการใช้หลักการส่งข้อมูลแบบพิกกี้แบ็ค

ข้อดีของการส่งแบบดังกล่าวคือ เพิ่มประสิทธิภาพในการรับ - ส่งข้อมูล สามารถใช้ช่องทางแบนด์วิธได้อย่างมีประสิทธิภาพมากขึ้น ส่วนข้อเสียที่หลีกเลี่ยงไม่ได้คือ ผู้รับสามารถรบกวนในเวลาที่ไม่มีการส่งข้อมูลได้ ปัญหานี้สามารถแก้ไขโดยการเพิ่มตัวนับ (Counter) ที่ Data frame

**2.7 ตัวกลางการรับชำระเงิน**

ตัวกลางการรับชำระเงิน (Payment Gateway) จะใช้เทคนิคการทำ Token เพื่อให้ความเสี่ยงในการเก็บข้อมูลบัตรมาอยู่ที่ผู้ให้บริการแทน ทางเว็บไซต์จะมีไม่มีข้อมูลบัตรเครดิตวิ่งผ่าน ถ้าเกิดกรณีที่เว็บไซต์ถูกผู้ไม่พึ่งประสงค์โจมตี (Hack) ก็จะไม่มีข้อมูลบัตรของผู้ใช้หลุดออกไป

จากภาพที่ 2-8 ขั้นตอนที่ 1 ข้อมูลบัตรเครดิตจะถูกเข้ารหัสและส่งมาให้ Omise Server โดยใช้ Source Code Omise.js ที่ Omise มีให้ ขั้นตอนที่ 2 หลังจากข้อมูลบัตรผ่านการตรวจสอบแล้ว Omise จะส่ง Token (รหัสตัวเลข) มาให้เว็บไซต์ ขั้นตอนที่ 3 ผู้ใช้จะสั่งสินค้าพร้อมกับส่ง Token มาให้กับ Server ของเว็บไซต์ แล้วเว็บไซต์จึงใช้ Token ในการสั่งตัดบัตรเครดิตกับทาง Omise



**ภาพที่ 2-8** ขั้นตอนในการจ่ายเงินของบริษัท Omise

**2.8 เทคโนโลยีของเว็บไซต์ (Web Technology)**

เว็บไซต์ในปัจจุบันมีการพัฒนาไปอย่างมากทั้งในด้านเทคโนโลยีการรับ - ส่งข้อมูล การติดต่อ  
กับระบบฐานข้อมูลหรือการจัดการข้อมูลฝั่งไคลเอนต์ ทำให้มีความรวดเร็วในการใช้งานสูง ตอบสนองผู้ใช้ได้อย่างทันท่วงที หากเปรียบเทียบกับเว็บไซต์ในอดีตก็คงต่างกันโดยสิ้นเชิง   
โดยหากเราแบ่งเว็บไซต์ทั้งหมดตามรูปแบบของเทคโนโลยีที่ใช้ก็จะแบ่งได้ 3 รูปแบบดังนี้

2.8.1 เว็บไซต์แบบ Static

เว็บไซต์แบบ Static คือ เว็บไซต์ที่มีข้อมูลนิ่งและมักไม่ตอบสนองต่อผู้ใช้ ซึ่งโดยส่วนมากจะเขียนด้วยภาษา HTML มีลักษณะเด่นคือ ข้อมูลจะเป็นแบบ Text File หรือเป็นแบบ HTML ทั้งหมด ข้อมูลของเว็บจะไม่เปลี่ยนจนกว่านักพัฒนาเว็บจะแก้ไขข้อมูลและทำการอัปโหลดขึ้นไปใหม่ แต่จะสามารถแสดงผลผ่าน Web Browser ได้เร็ว

2.8.2 เว็บไซต์แบบ Dynamic

เว็บไซต์แบบ Dynamic คือเว็บไซต์ที่แตกต่างกับเว็บไซต์แบบ Static คือจะมีการตอบสนองกับผู้ใช้งานผ่านการกรอกข้อมูลหรือแสดงผลต่าง ๆ มีลักษณะเด่น คือ บน Server ต้องมีโปรแกรมเพื่อประมวลผลภาษาที่ใช้เขียน โดยการเรียกเว็บทุกครั้ง Web Server จะต้องประมวลผลก่อนที่จะส่งข้อมูลมาแสดงผลที่ Web Browser ผลที่ได้อาจไม่เหมือนกันซึ่งจะขึ้นอยู่กับข้อมูลของผู้ใช้งาน โดยข้อมูลจะเรียกจากฐานข้อมูลหรือข้อมูลของระบบ เป็นต้น การพัฒนาเว็บไซต์แบบนี้จะมีความยากมากกว่าแบบแรก เพราะส่วนใหญ่จะเน้นเพื่อติดต่อข้อมูลภายนอก เช่น ฐานข้อมูลและไฟล์ เป็นต้น

2.8.3 เว็บไซต์แบบ Active

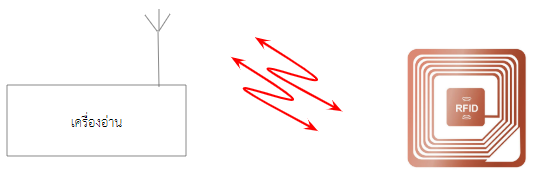
เว็บไซต์แบบ Active เป็นเว็บไซต์ที่ตอบสนองผู้ใช้แบบเรียลไทม์ มีลักษณะเด่นคือ การโหลดเว็บจะเป็นการโหลดซอร์สโปรแกรมมาประมวลที่เครื่องไคลเอนต์ (Web Browser) ด้วย ซึ่งหมายความว่า Web Browser จะต้องมีตัวประมวลผลซอร์สโปรแกรม เช่น Java Runtime หรืออื่น ๆ นอกเหนือจาก HTML นอกจากนี้ผู้พัฒนายังสามารถเขียนซอร์สโปรแกรมให้รีเฟรชเฉพาะข้อมูลได้ โดยไม่จำเป็นต้องโหลดหน้าเว็บมาใหม่เหมือน Dynamic Web แต่จากเหตุผลข้างต้นก็ทำให้เว็บไซต์แบบ Active นั้นพัฒนายากกว่า Static และ Dynamic

**2.9 RFID (Radio-Frequency Identification)**

RFID เป็นระบบที่ใช้ระบุลักษณะของวัตถุใดวัตุหนึ่ง ด้วยคลื่นความถี่วิทยุ ซึ่งจัดเป็นเทคโนโลยีใหม่ถ้าเทียบกับยุคสมัยปัจจุบัน ซึ่งปัจจุบันนี้มีผู้คนให้ความสนใจและเข้าใจถึงความสำคัญของเทคโนโลยี RFID นี้ค่อนข้างมาก เพราะวิธีการนี้มีความสะดวก และช่วยลดระยะเวลาค่อนข้างมาก แถมผลลัพธ์นั้นยังมีประสิทธิภาพที่ดีอีกด้วย

2.9.1 ส่วนประกอบของระบบ RFID

RFID จะมีส่วนประกอบหลักๆอยู่ 2 ส่วนด้วยกัน อันดับแรก คือ ทรานสปอนเดอร์ หรือแท็ก (Transponder/Tag) ซึ่งจะใช้ติดกับวัตถุต่าง ๆ ที่เราต้องการ โดยแท็กที่นำไปติดนั้นจะบันทึกข้อมูลเกี่ยวกับวัตถุชิ้นนั้น ๆ เอาไว้ และส่วนประกอบที่สอง คือ เครื่องสำหรับอ่าน/เขียนข้อมูลภายในแท็ก (Interrogator/Reader) โดยใช้คลื่นความถี่วิทยุ ดังแสดงในภาพที่ 2-9



**ภาพที่ 2-9** ภาพรวมของระบบ RFID

**1) แท็ก (Tag)**

พื้นฐานโครงสร้างภายในของแท็กจะประกอบด้วย 2 ส่วน ซึ่งได้แก่ ขดลวดขนาดเล็ก ซึ่งทำหน้าที่เป็นสายอากาศ (Antenna) เพื่อสำหรับรับ-ส่งสัญญาณคลื่นความถี่วิทยุและสร้างพลังงานส่งให้ส่วนที่สอง คือไมโครชิป (Microchip) ที่ต้องทำหน้าที่เก็บข้อมูลของวัตถุ และแท็กมีการแบ่งโครงสร้างการใช้งานกันอยู่ 2 ชนิดใหญ่ ๆ คือ Passive RFID Tags และ Active RFID Tags

Passive RFID Tags แท็กชนิดนี้ไม่ต้องอาศัยแหล่งจ่ายไฟภายนอกใด ๆ เพราะภายในแท็กจะมีวงจรกำเนิดไฟฟ้าเหนี่ยวนำขนาดเล็กเป็นแหล่งจ่ายไฟในตัวอยู่ทำให้การอ่านข้อมูลทำได้ไม่ไกลมากนัก ระยะอ่านสูงสุดประมาณ 1 เมตร ขึ้นอยู่กับความแรงของเครื่องส่งและคลื่นความถี่วิทยุที่ใช้ ปกติแท็กชนิดนี้ส่วนมากจะมีหน่วยความจำขนาดเล็กโดยทั่วไปประมาณ 16 ถึง 1,024 ไบต์ มีขนาดเล็กและน้ำหนักเบา ราคาถูก ไอซีของแท็กชนิด Passive ที่มีการผลิตออกมาจะมีทั้งขนาดและรูปร่างเป็นแท่ง หรือแผ่นขนาดเล็กจนแทบไม่สามารถมองเห็นได้ไปจนถึงขนาดใหญ่ ซึ่งก็ต่างมีความเหมาะสมกับชนิดงานที่แตกต่างกัน ส่วนโครงสร้างภายในที่เป็นไอซีของแท็กนั้น ก็จะประกอบด้วย 3 ส่วนหลัก ๆ ได้แก่ ส่วนควบคุมการทำงานของภาครับ-ส่งสัญญาณวิทยุ (Analog Front-End) ส่วนควบคุมภาคลอจิก (Digital Control Unit) ส่วนของหน่วยความจำ (Memory) ซึ่งอาจจะเป็นแบบ ROM หรือ EEPROM ดังตัวอย่างในภาพที่ 2-10



**ภาพที่ 2-10** Passive Tag (คีย์การ์ดหอพัก)

Active RFID Tags แท็กชนิดนี้จะต้องอาศัยแหล่งจ่ายไฟจากแบตเตอรี่ภายนอกเพื่อจ่ายพลังงานให้กับวงจรภายในทำงานแท็กชนิดนี้ มีหน่วยความจำภายในขนาดใหญ่ได้ถึง 1 เมกะไบต์ และสามารถอ่านได้ในระยะไกลสูงสุดประมาณ 10 เมตร ประเภทจากรูปแบบในการใช้งานมีอยู่ 3 แบบ คือ แบบที่สามารถถูกอ่านและเขียนข้อมูลได้อย่างอิสระ (Read-Write), แบบเขียนได้เพียงครั้งเดียวเท่านั้นแต่อ่านได้อย่างอิสระ (WriteOnce Read-Many หรือ WORM) และแบบอ่านได้ เพียงอย่างเดียว (Read-Only) ดังตัวอย่างในภาพที่ 2-11



**ภาพที่ 2-11** Active Tag ที่มีแบตเตอรี่

**2) เครื่องอ่าน (Reader)**

หน้าที่ของเครื่องอ่านคือ การเชื่อมต่อเพื่อเขียนหรืออ่านข้อมูลลงในแท็กด้วยสัญญาณความถี่วิทยุ โดยภายในจะประกอบด้วยเสาอากาศที่ทำจากขดลวดทองแดง เอาไว้เพื่อใช้รับส่งสัญญาณภาครับและภาคส่งสัญญาณวิทยุ วงจรควบคุมการอ่าน-เขียนข้อมูลจำพวกไมโครคอนโทรลเลอร์ และในส่วนของการติดต่อสื่อสารกับคอมพิวเตอร์

โดยทั่วไปเครื่องอ่านจะประกอบด้วยส่วนประกอบหลักดังนี้

* ภาครับและส่งสัญญาณวิทยุ
* ภาคสร้างสัญญาณพาหะ
* ขดลวดที่ทำหน้าที่เป็นสายอากาศ
* วงจรจูนสัญญาณ
* หน่วยประมวลผลข้อมูลและภาคติดต่อกับคอมพิวเตอร์

หน่วยประมวลข้อมูลที่อยู่ภายในเครื่องอ่านมักจะใช้เป็นไมโครคอนโทรลเลอร์ ซึ่งอัลกอริทึมที่อยู่ภายในโปรแกรมจะทำหน้าที่ถอดรหัสข้อมูล (Decoding) ที่ได้รับ และทำหน้าที่ติดต่อกับคอมพิวเตอร์ ลักษณะ ขนาด และรูปร่างของเครื่องอ่านจะแตกต่างกันไปตามประเภทของการใช้งาน เช่น แบบมือถือขนาดเล็กหรือติดผนัง จนไปถึงขนาดใหญ่เท่าประตู (Gate size) ดังตัวอย่างในภาพที่ 2-12

****

**ภาพที่ 2-12** ตัวอย่างเครื่องอ่านแบบต่าง ๆ

2.9.2 หลักการและเทคนิคที่ใช้รับและส่งข้อมูลระหว่างแท็กและเครื่องอ่าน

เทคนิคในการรับส่งข้อมูลระหว่างเครื่องอ่านและแท็กจะใช้หลักการมอดูเลตทางแอมปลิจูด (Amplitude Modulation: AM) หรือใช้การมอดูเลตทางแอมปิลจูดบวกกับการเข้ารหัสแมนเชสเตอร์ (Manchester encoded AM) แต่ในปัจจุบันก็มีแท็กที่ใช้การมอดูเลตแบบอื่น ๆ ด้วย เช่น การมอดูเลชั่นแบบเฟสชิฟคีย์อิง (Phase Shift Keying: PSK), ฟรีเควนซี่ชิฟคีย์อิง (Frequency Shift Keying: FSK) หรือการใช้การมอดูเลตทางความถี่ (Frequency Modulation: FM)

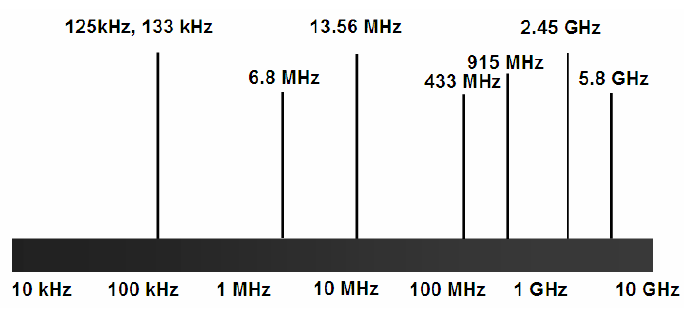
ในการรับส่งข้อมูลหรือสัญญาณวิทยุระหว่างแท็กกับเครื่องอ่านจะทำได้อย่างมีประสิทธิภาพก็ต่อเมื่อสายอากาศมีความยาวที่เหมาะสมกับความถี่พาหะที่ใช้งาน เช่น เมื่อความถี่ใช้งานเป็น   
13.56 เมกะเฮิรตซ์ ความยาวของเสาอากาศ (เป็นเส้นตรง) ที่เหมาะสมก็คือ 22.12 เมตร สายอากาศที่ดูเหมาะจะใช้ร่วมกับแท็กมากที่สุด ก็คือสายอากาศที่เป็นขดลวดขนาดเล็กหรือสายอากาศแบบแมกเนติกไดโพล (magnetic dipole antenna) รูปแบบของสายอากาศแบบนี้ก็จะมีอยู่หลากหลาย ทั้งแบบที่เป็นขดลวดพันบนแกนอากาศหรือแกนเฟอร์ไรต์แบบที่เป็นวงลูปที่ทำขึ้นจากลายทองแดงบนแผนวงจรพิมพ์ทั้งที่เป็นลูปแบบวงกลมและสี่เหลี่ยม

2.9.3 คลื่นพาหะในระบบ RFID

ในปัจจุบันคลื่นพาหะที่ใช้งานกันในระบบ RFID จะอยู่ในย่านความถี่ ISM (Industrial-Scientific-Medical) ซึ่งเป็นย่านความถี่ที่กำหนดการใช้งานในเชิงอุตสาหกรรม วิทยาศาสตร์และการแพทย์ สามารถใช้งานได้โดยไม่ตรงกับย่านความถี่ทที่ใช้งานในการสื่อสารทั่วไปสำหรับคลื่นพาหะที่ใช้กันในระบบ RFID อาจแบงออกได้เป็น 3 ย่านความถี่ใช้งานหลัก ได้แก่

* ย่านความถี่ต่ำ (Low Frequency: LH) ต่ำกว่า 150 kHz
* ย่านความถี่สูง (High Frequency: HF) 13.56 MHz
* ย่านความถี่สูงยิ่ง (Ultra High Frequency: UHF) 433/868/915 MHz

การใช้งาน 2 ย่านความถี่แรกจะเหมาะสำหรับใช้กับงานที่มีระยะการสื่อสารข้อมูลในระยะใกล้ (LF ระยะอ่านประมาณ 10-20 เซนติเมตร และ HF ระยะอ่านประมาณ 1 เมตร) เช่น การตรวจสอบการผ่านเข้าออกพื้นที่และเก็บประวัติต่าง ๆ ส่วนย่านความถี่สูงยิ่ง จะถูกใช้กับงานที่มีระยะการสื่อสารข้อมูลในระยะไกล (UHF ระยะอ่านประมาณ 1-10 เมตร) เช่น ระบบเก็บค่าบริการทางด่วนและในปัจจุบันระบบ RFID กำลังถูกวิจัยและพั ฒนาในย่านความถี่ไมโครเวฟที่ความถี่ 2.4 GHz และความถี่ 5.8 GHz เพื่อใช้งานที่มีระยะอ่านที่ไกลกว่า 10 เมตร ดังตัวอย่างในภาพที่ 2-13 ซึ่งในโปรเจคนี้จะใช้ย่านความถี่ 13.56 MHz หรือ HF จะมีระยะอ่านประมาณ 1 เมตร ใช้สำหรับยืนยันตัวตนใช้งานสถานีประจุไฟฟ้า

****

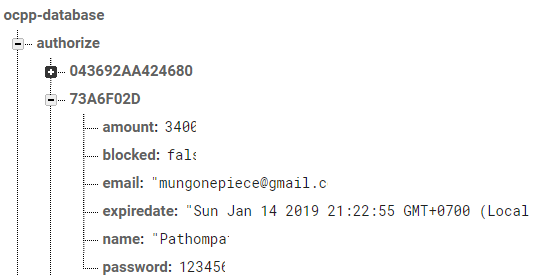
**ภาพที่ 2-13** ความถี่ย่านที่ระบบ RFID ถูกใช้งาน

**2.10 ระบบฐานข้อมูลที่ไม่ใช้ภาษา SQL (NoSQL Databases)**

NoSQL Databases เป็นอีกหนึ่งในเทคโนโลยีที่ใช้ในการจัดเก็บข้อมูลที่เป็นที่นิยมในปัจจุบัน เนื่องจากมีความยืดหยุ่นในการเก็บข้อมูลและความเร็วในการอ่านเขียนข้อมูลสูง จึงมักจะใช้ออกแบบฐานข้อมูลสำหรับข้อมูลที่มีขนาดใหญ่และใช้สำหรับระบบที่ต้องการการทำงานแบบ Realtime

2.10.1 Document Database

เป็นฐานข้อมูล NoSQL ที่ได้ความนิยมที่สุด ใช้รูปแบบการบันทึกข้อมูลเป็น JSON Structure ซึ่งเป็นชุดของข้อความที่มีความยาว ทำให้มีอิสระในการจัดเก็บข้อมูลประเภทใด ๆ ก็ได้ไม่ว่าจะเป็น Integer, String หรือข้อความใด ๆ ก็ตาม และไม่ต้องกำหนดประเภทหรือรูปแบบของข้อมูลล่วงหน้า ดังภาพที่ 2-14 เป็นตัวอย่างโครงสร้างการเก็บข้อมูลของ User ที่ถูกลงทะเบียนไว้ในระบบ มี TagID เป็น 73A6F02D ในการยืนยันตัวตนเข้าใช้เว็บแอพพลิเคชั่นและใช่งานสถานีประจุไฟฟ้า เนื่องจากโครงงานนี้เป็นการรับส่งข้อมูลแบบ JSON จึงเลือกใช้ฐานข้อมูลแบบ NoSQL ในการเก็บข้อมูล



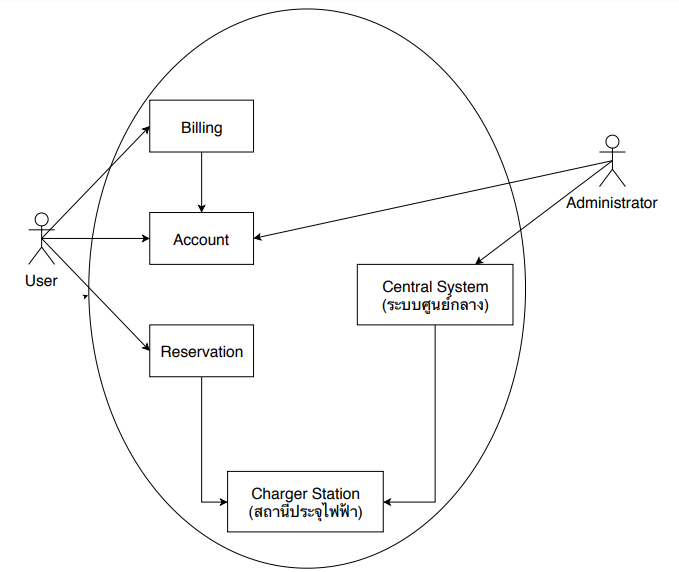
**ภาพที่ 2-14** โครงสร้างการเก็บข้อมูลแบบ JSON

**บทที่ 3**

**ขั้นตอนการดำเนินงาน**

โครงงานนี้มีวัตถุประสงค์เพื่อศึกษาและพัฒนาการให้บริการสถานีประจุไฟฟ้า  
ของยานพาหานะไฟฟ้า ซึ่งได้สืบค้นเกี่ยวกับรูปแบบการส่งข้อมูล (Protocol) ที่ใช้การรับ - ส่งข้อมูลระหว่างสถานีประจุไฟฟ้า (Charger Station), ระบบศูนย์กลาง (Central System), ระบบฐานข้อมูล (Database) ตลอดจนเครื่องมือ (Tools) และแพลทฟอร์ม (Platform) ที่ใช้ในการพัฒนาเว็บแอพพลิชั่นสำหรับให้บริการผู้ใช้

จากภาพที่ 3-1 เป็น Use case diagram เมื่อผู้ใช้เข้าใช้เว็บแอพพลิชั่นจะสามารถเติมเงินได้   
เห็นข้อมูลของผู้ใช้และสามารถจองสถานีประจุไฟฟ้าได้ ระบบการจองจะติดต่อกับสถานีประจุไฟฟ้า  
เมื่อมีการจองในส่วนของระบบศูนย์กลางจะติดต่อกับสถานีประจุไฟฟ้าอยู่ตลอดเวลาเมื่อสถานีประจุไฟฟ้ายังออนไลน์อยู่ ในส่วนของ Admin ที่สามารถเข้าแก้ไขได้คือ ข้อมูลของผู้ใช้และระบบศูนย์กลาง



**ภาพที่ 3-1** Use case diagram

**3.1 ขั้นตอนการดำเนินงาน**

1. ศึกษา Protocol OCPP ที่เป็นมาตรฐานกลางในการติดต่อสื่อสารระหว่างสถานีอัดประจุไฟฟ้ากับ ระบบศูนย์กลาง

2. กำหนดช่องทางในการ รับ - ส่งข้อมูล โดยการ รับ - ส่งข้อมูลผ่าน Websocket Protocol

3. กำหนดรูปแบบของการเก็บข้อมูลเป็น JSON สำหรับการส่งข้อมูลใน Websocket

4. ใช้มาตรฐานของชุดข้อมูล ที่บ่งบอกสถานะของผู้ส่ง - ผู้รับ โดยหลักการมาจาก RPC

5. ทำการ Register Message ของ Protocal OCPP ไว้ที่ Central System สำหรับรับข้อมูลจาก Charger Station โดยใช้ภาษา Javascript บน NodeJS Framework สร้าง Central System

6. กำหนดฟังก์ชันรับ Massage ตอบกลับจาก Central System ใน Charger Station   
โดยใช้ภาษา Python ในเขียนฟังก์ต่าง ๆ ในการรับ - ส่งข้อมูลจาก Central System สำหรับนำข้อมูลที่ได้จาก Central System มาแสดงบนจอของ Charger Station เพื่อสื่อสารกับ User ได้เข้าใจในการใช้งาน Charger Station

7. ออกแบบ Database ในการเก็บข้อมูลซึ่งรูปแบบข้อมูลเป็น JSON จึงเลือกใช้ Firebase   
ที่เป็น NoSQL และเป็น Database Realtime โดยจะมี Source Code ในการเชื่อมต่อกับ Firebase   
จะเป็นภาษา Javascript ดังนั้น Central System จะรับข้อมูลจาก Charger Station มาเก็บไว้ใน Firebase และเช็ค User ว่ามีอยู่ใน Database หรือไม่เพื่อเป็นการยืนยันตัวตนในการใช้งาน   
Charger Station

8. ออกแบบ Web Application โดยมีการ Login ในการเข้าใช้ Web Application ซึ่งจะใช้ Polymer 2.0 เป็น Web Framewort เพราะเป็น JavaScript Based, Realtime และเป็น Web Component ที่เราสามารถสร้าง Custom Element หรือไว้สร้าง HTML Element ของตัวเองได้ไว้ใช้ได้

ฟังก์ชันหลัก ๆ ของ Web Application จะมีดั้งนี้

1. แนะนำวิธีการใช้ Charger Station

2. ระบบ Account Information

3. แสดงตำแหน่งและสถานะของ Station ในแผนที่

4. สามารถจอง Charger Station ในแผนที่ได้

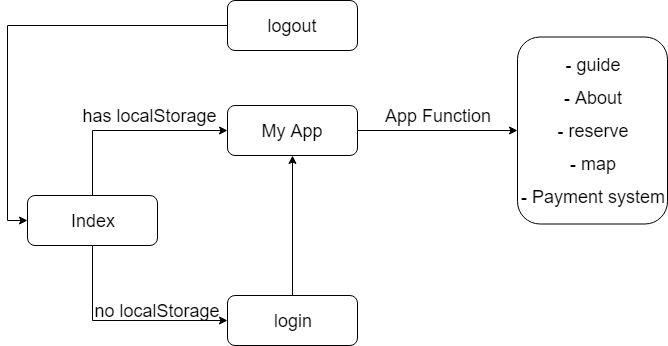
5. สามารถติดตามสถานะการจองและยกเลิกการจองได้

6. ระบบ Billing/Topup ผ่าน API ของ Omise

**3.2 กระบวนการทำงานของระบบ Account Information**

หน้า Web Application จะขึ้นหน้า Login เป็นหน้าแรกให้ User ทำการ Login เพื่อเข้าใช้งาน Web Application เมื่อผู้ใช้ได้ทำการใส่ User-Email และ Password ลงไปในช่องกรอกระบบจะทำการเช็คกับ Database เพื่อยืนยันความถูกต้อง ถ้ามีข้อมูลใน Database จะเข้าใช้ Web Application   
ได้ตามปกติและจะทำการเก็บค่า User-Email ไว้ที่ Local Storege เมื่อผู้ใช้ปิด Browser ไป   
เมื่อเข้ามาใหม่ผู้ใช้ที่ Login ค้างไว้ก็ยังอยู่ในระบบเปรียบเสมือน Facebook ที่ผู้ใช้ได้ Login ค้างไว้   
ถ้าไม่มีข้อมูลใน Database ก็จะไม่สามารถเข้าใช้ Web Application ได้

ดังแสดงในภาพที่ 3-2 ถ้ายังไม่มี User-Email เก็บไว้ที่ Local Storege หน้า Index   
ก็จะแสดงให้ User ทำการ Login ก่อนเข้าใช้ Web Application เมื่อทำการ Login เรียบร้อยก็จะเข้าใช้งาน Web Application ได้ตามปกติ และจะทำการเก็บค่า User-Email ไว้ที่ Local Storege   
เมื่อผู้ใช้ปิด Browser ไป เมื่อเข้ามาใหม่ผู้ใช้ที่ Login ค้างไว้ ในหน้า Index จะไม่แสดงให้ Login   
แต่จะเข้าใช้ Web Application ได้เลย เมื่อ Logout ก็จะกลับไปหน้า Index และลบค่าที่เก็บไว้ที่ Local Storege ด้วย ดังนั้น จะไม่มีค่าใน Local Storege ในหน้า Index จะแสดงให้ User ทำการ Login ก่อน

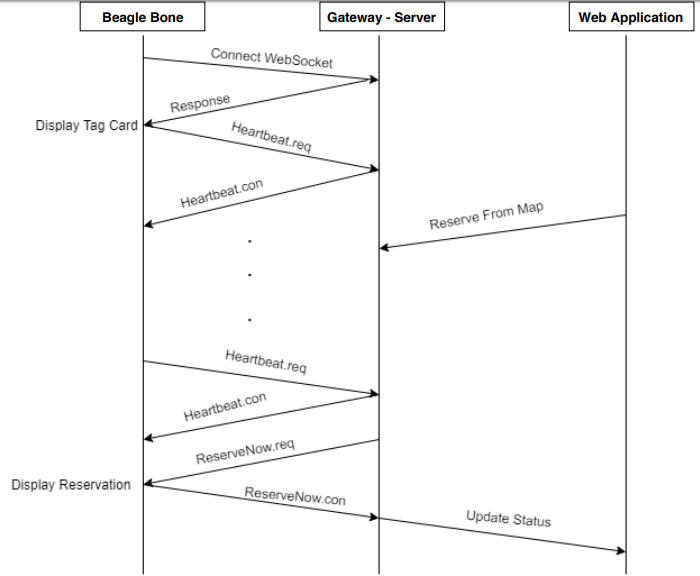


**ภาพที่ 3-2** Process Finite State Machine Diagram ของระบบ Account Information

**3.3 กระบวนการทำงานของระบบ Reservation**

จากภาพที่ 3-3 เว็บแอพพลิเคชั่นส่งคำขอจองไปยัง Gateway - Server ซึ่งจะเพิ่มค่าใน Database ส่วน Reservation เพิ่มมา 1 ข้อมูล สถานะ “Pending” Gateway - Server รอรับข้อความ Heartbeat จาก Charger Station (Beagle Bone) Gateway - Server จะเช็คว่าใน Database มี Reservation ที่มีสถานะ Pending และมี cpid ตรงกับที่ต้องการจองหรือไม่ ถ้าตรงจะส่งข้อความ ReserveNow ไปพร้อมข้อความ Heartbeat ไปยัง Charger Station เพื่อแสดงบนจอ Charger Station (Beagle Bone) ว่ามีผู้ใช้ได้ทำการจองไว้แล้ว Gateway - Server จะรอข้อความตอบกลับจาก Charger Station (Beagle Bone) ถ้า Status ตอบกลับเป็น Accept จะเปลี่ยนค่าใน Database เป็น Accepted และปิดปุ่มจองใน Map สำหรับจองสำเร็จ ในระหว่างการรอจองและรอชาร์จจะสามารถยกเลิกได้ตลอดเวลา โดยจะส่งข้อความ CancelReservation ไปยัง Gateway - Server และ Gateway - Server จะเปลี่ยนสถานะใน Database เป็น Canceled และรอส่งข้อความ CancelReservation ไปพร้อมกับ Heartbeat ไปยัง Charger Station (Beagle Bone) เพื่อเปลี่ยนจากจอที่แสดงผลว่ามีผู้ใช้ได้ทำการจองไว้แล้วเปลี่ยนมาใช้ได้ปกติโดยที่ยังไม่มีผู้ใช้อื่นจองไว้ก่อน เมื่อผู้ใช้ชาร์จเสร็จแล้ว Charger Station (Beagle Bone) จะส่งข้อความ StopTransaction ไปยัง Gateway - Server เพื่อจะเปลี่ยนสถานะใน Database เป็น Finished และบน Map จะสามารถจองได้เหมือนเดิม

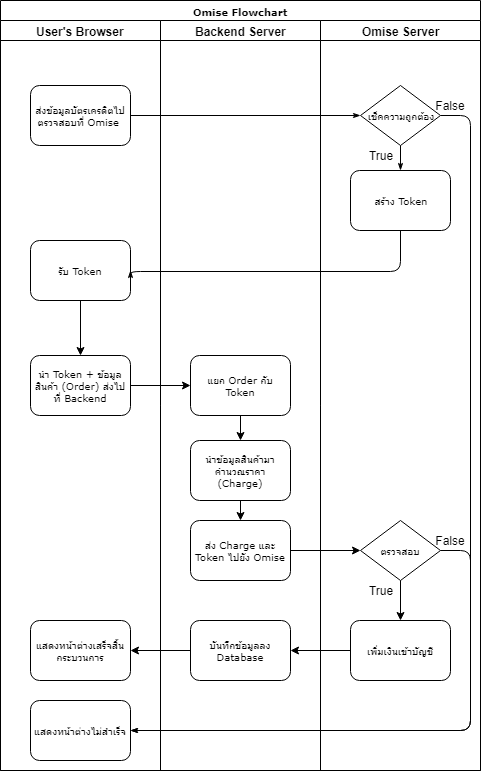
Server (ระบบศูนย์กลาง) จะรอข้อความตอบกลับจาก Charger Station (Beagle Bone) ถ้า Status ตอบกลับเป็น Accept จะเปลี่ยนค่าใน Database เป็น Accepted และปิดปุ่มจองใน Map สำหรับจองสำเร็จ



**ภาพที่ 3-3** Reservation Sequence Diagram

**3.4 กระบวนการทำงานของระบบ Billing/Topup ผ่าน API ของ Omise**

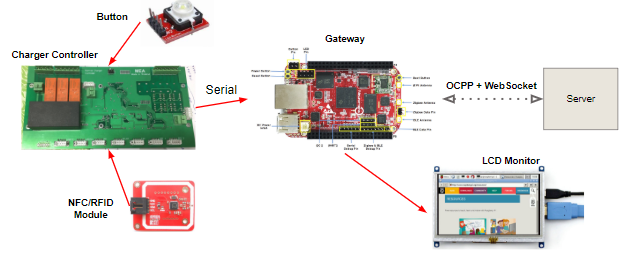
จากภาพที่ 3-4 เมื่อกรอกข้อมูลบัตรเครดิตที่หน้าเว็บไซต์ (User’s Browser) ข้อมูลบัตรเครดิตจะถูกเข้ารหัสและส่งมาให้ Omise Server โดยใช้ API ของ Omise หลังจากข้อมูลบัตรผ่านการตรวจสอบแล้ว Omise จะส่ง Token (รหัสตัวเลข) มาให้เว็บไซต์ (User’s Browser) ถ้าตรวจสอบไม่ผ่านหน้าเว็บไซต์ (User’s Browser) จะแสดงหน้าต่างไม่สำเร็จ ผู้ใช้จะสั่งสินค้าพร้อมกับส่ง Token มาให้กับ Backend Server ของเว็บไซต์ (User’s Browser) Backend Server จะทำการแยก Order (ข้อมูล)   
กับ Token ออกก่อนและทำการนำเอาข้อมูลสินค้ามาคำนวณราคา จากนั้นส่งราคา (charge) กับ Token ไปยัง Omise Server ตรวจสอบ เมื่อตรวจสอบผ่านแล้วจะทำการเพิ่มเงินเข้าในบัญชีของ Omise Server แล้วส่งจำนวนเงินนั้นมาให้ Backend Server บันทึกลงใน Database แล้วหน้าเว็บไซต์ (User’s Browser) จะแสดงหน้าต่างสำเร็จกระบวนการ เมื่อตรวจสอบไม่ผ่านหน้าเว็บไซต์ (User’s Browser)   
จะแสดงหน้าต่างไม่สำเร็จ



**ภาพที่ 3-4** Omise Flowchart

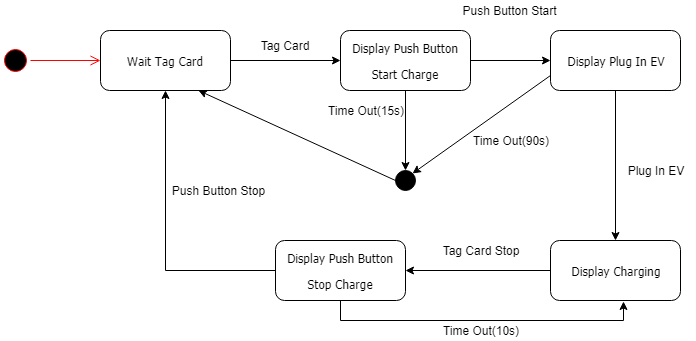
**3.5 กระบวนการทำงานของสถานีประจุไฟฟ้า**

จากภาพที่ 3-5 ตัวควบคุม (Charger Controller) จะรับอินพุตจากผู้ใช้ เช่น แตะบัตร RFID   
เพื่อยืนยันตัวตน และกดปุ่มเพื่อเริ่มชาร์จหรือหยุดชาร์จ และส่งข้อมูลผ่าน Serial ให้กับ Beagle Bone   
ที่เป็น Gateway แพ็คข้อมูลส่งไปยังระบบศูนย์กลาง (Server) และแสดงผลบนจอ LCD Monitor



**ภาพที่ 3-5** ส่วนประกอบของสถานีประจุไฟฟ้า

จากภาพที่ 3-6 แสดงถึง State Machine Diagram Charger (ขั้นตอนการทำงานของ  
สถานีประจุไฟฟ้า) เริ่มแรกสถานีประจุไฟฟ้าจะให้ผู้ใช้มาทำการแตะบัตร RFID เพื่อยืนยันตัวตน  
ก่อนจากนั้นสถานีประจุไฟฟ้าจะแสดงผลที่หน้าจอ Monitor ให้ผู้ใช้กดปุ่มเพื่อเริ่มจะทำการชาร์จ   
ถ้าไม่กดปุ่มจะมี Time Out อยู่ที่ 15 วินาที จะกลับไปให้แตะบัตรใหม่ เมื่อกดปุ่มแล้วสถานีจะแสดงผล  
ที่หน้าจอ Monitor ให้ผู้ใช้เสียบสายชาร์จเข้าที่ยานพาหนะไฟฟ้า ถ้าไม่เสียบสายชาร์จจะมี Time Out   
อยู่ที่ 90 วินาที จะกลับไปให้แตะบัตรใหม่ เมื่อเสียบสายชาร์จแล้วสถานีจะแสดงผลที่หน้าจอ Monitor   
ว่ากำลังชาร์จ เมื่อชาร์จเสร็จหรือว่าจะทำการหยุดชาร์จ จะต้องแตะบัตรที่ผู้ใช้แตะตอนเริ่มชาร์จ  
ถึงจะหยุดชาร์จได้ เมื่อแตะบัตรสถานีประจุไฟฟ้าจะแสดงผลที่หน้าจอให้กดปุ่มเพื่อหยุดชาร์จ   
ถ้าไม่กดปุ่มจะมี Time Out อยู่ที่ 10 วินาที จะกลับไปแสดงผลว่ากำลังชาร์จ เมื่อกดปุ่มจะกลับไปเริ่มให้แตะบัตรใหม่ เป็นการสิ้นสุดกระบวนการชาร์จ



**ภาพที่ 3-6** Finite State Machine Diagram ของ Charger

จากภาพที่ 3-7 แสดงถึงลำดับการส่งข้อมูลระหว่าง Controller ไปยัง Beagle Bone   
ที่เป็น Gateway ทำหน้าที่แพ็คข้อมูลส่งไปยังระบบศูนย์กลาง Controller และ Beagle Bone   
เป็นอุปกรณ์ภายในสถานีประจุไฟฟ้า มีลำดับการส่งข้อมูลดังต่อไปนี้

1. Beagle Bone จะส่ง Heartbeat ทุก ๆ 1 วินาที เพื่อเป็นการบอกระบบศูนย์กลางว่าสถานีประจุไฟฟ้ายังใช้ได้

2. Controller รอรับอินพุตจากผู้ใช้ เมื่อผู้ใช้แตะบัตร Controller จะส่ง TagCard มาให้ Beagle Bone แพ็คข้อมูลและส่งไปยังระบบศูนย์กลางตรวจสอบ

3. เมื่อตรวจสอบเสร็จระบบศูนย์กลางตอบกลับ Confirm ไปยัง Beagle Bone

4. Beagle Bone ส่ง Confirm ไปยัง Controller

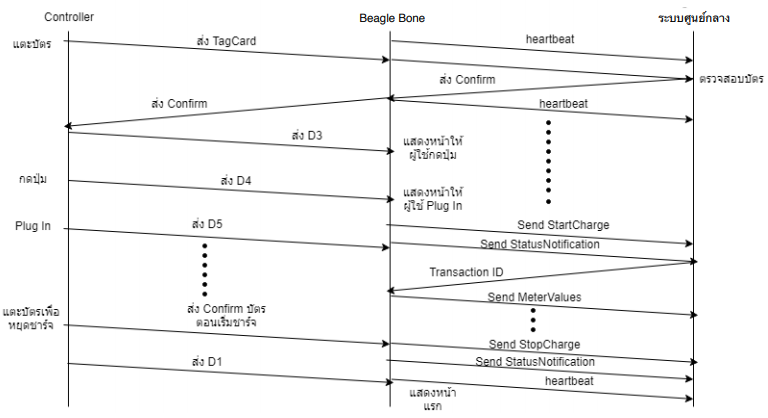
5. Controller จะส่ง D3 มายัง Beagle Bone ให้แสดงหน้าให้ผู้ใช้กดปุ่ม (D3 เป็น State ระหว่าง Controller กับ Beagle Bone ว่าจะสั่งให้ Beagle Bone แสดงหรือทำอะไร)

6. เมื่อกดปุ่ม Controller จะส่ง D4 มายัง Beagle Bone ให้แสดงหน้าให้ผู้ใช้เสียบสายชาร์จกับยานพาหนะไฟฟ้า

7. เมื่อผู้ใช้เสียบสายชาร์จกับยานพาหนะไฟฟ้า Controller จะส่ง D5 มายัง Beagle Bone ให้แสดงหน้ากำลังชาร์จ และ Beagle Bone จะส่งแพ็คเกจ StartCharge, StatusNotification และ MeterValues ไปยังระบบศูนย์กลางเพื่อบอกว่าสถานีประจุไฟฟ้ากำลังชาร์จอยู่

8. เมื่อชาร์จเสร็จหรือจะหยุดชาร์จให้แตะบัตรที่ถูก Confirm ตอนเริ่มชาร์จ Controller   
จะส่ง TagCard ไปยัง Beagle Bone ให้แสดงหน้ากดปุ่มเพื่อหยุดชาร์จ และ Beagle Bone จะส่งแพ็คเกจ StopCharge และ StatusNotification ไปยังระบบศูนย์กลางเพื่อบอกว่าสถานีประจุไฟฟ้าหยุดชาร์จแล้ว

9. เมื่อผู้ใช้กดปุ่ม Controller จะส่ง D1 มายัง Beagle Bone ให้กลับมาแสดงหน้าแรกให้ผู้ใช้แตะบัตรใช้งานเหมือนเดิม



**ภาพที่ 3-7** ลำดับการส่งข้อมูลระหว่าง Controller, Beagle Bone และ Gateway - Server

**บทที่ 4**

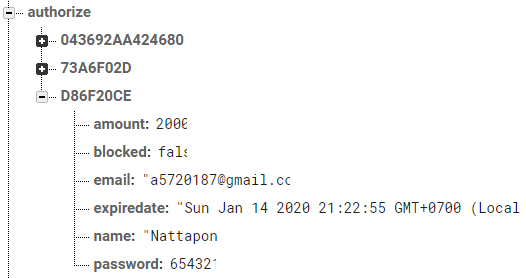
**ผลการดำเนินการ**

ในบทความนี้จะทดลองและแสดงผลของการใช้งานเว็บแอพพลิเคชั่นตั้งแต่เริ่มต้นใช้งานและฟังก์ชันหลัก ๆ ในการใช้งานเว็บแอพพลิเคชั่น และการติอต่อสื่อสารระหว่างสถานีประจุไฟฟ้ากับ Gateway - Server หรือระบบศูนย์กลางที่ใช้ OCPP Potocol ในการติดต่อสื่อสาร ซึ่งมีรายละเอียดการทดลองดังต่อไปนี้ ทดลองเข้าใช้เว็บแอพพลิเคชั่น ทดลองเติมเงินผ่านเว็บแอพพลิเคชั่น ทดลองการใช้งานสถานีประจุไฟฟ้า ทดลองยกเลิกการใช้งานสถานีประจุไฟฟ้า ทดลองระบบการจองและทดลองยกเลิกระบบการจอง

**4.1 ผลการทดลองเข้าใช้เว็บแอพพลิเคชั่น**

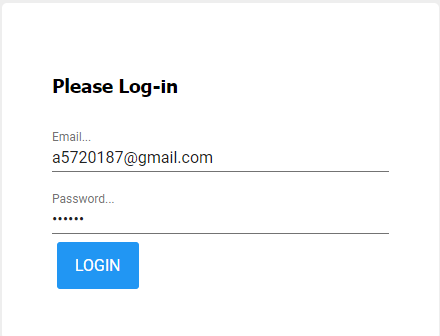
การเข้าใช้งานเว็บแอพพลิเคชั่นจะใช้ Email และ Password ในการเข้าใช้งานเว็บแอพพลิเคชั่นที่ผู้ใช้ได้ทำการลงทะเบียนไว้ ผู้ใช้ที่ลงทะเบียนจะมีการ์ด RFID เพื่อสำหรับยืนยันตัวตนในการใช้งานสถานีประจุไฟฟ้า ในส่วนของ Email และ Password ของผู้ใช้จะถูกเก็บไว้ในฐานข้อมูล

จากภาพที่ 4-1 แสดงถึงข้อมูลในฐานข้อมูลที่ผู้ใช้ได้ทำการลงทะเบียนไว้ จะมี Email และ Password ที่ไว้เข้าใช้งานเว็บแอพพลิเคชั่น



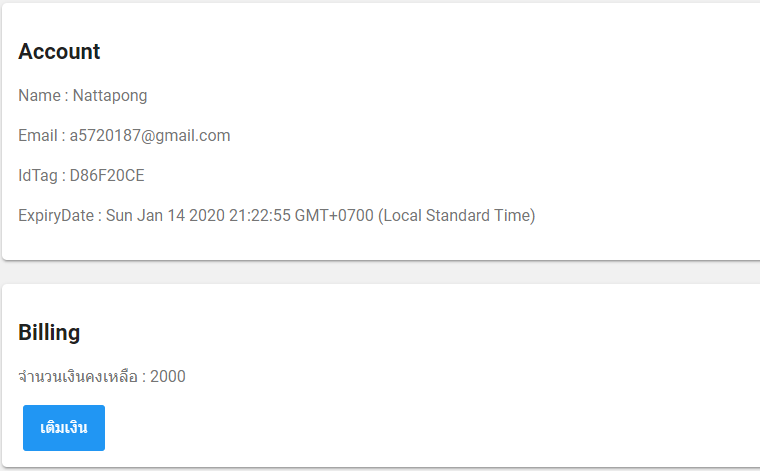
**ภาพที่ 4-1** ข้อมูลผู้ใช้ในฐานข้อมูล Firebase

จากภาพที่ 4-2 นำ Email และ Password ที่ผู้ใช้ได้ลงทะเบียนไว้มาใช้เพื่อ Login เข้าใช้งานเว็บแอพพลิเคชั่น



**ภาพที่ 4-2** หน้าเว็บไซต์ Login สำหรับผู้ใช้

จากภาพที่ 4-3 เมื่อเข้าใช้งานเว็บแอพพลิเคชั่นแล้ว หน้าเว็บจะแสดงข้อมูลของผู้ใช้ที่ Login   
เข้ามา ข้อมูลที่แสดงจะมี ชื่อ, Email, ID Tag, วันหมดอายุ และจำนวนเงินของผู้ใช้ ผู้ใช้สามารถเติมเงินผ่านเว็บแอพพลิเคชั่นเข้าบัตร RFID ที่ผู้ใช้ได้ลงทะเบียนไว้ได้



**ภาพที่ 4-3** หน้าเว็บไซต์เมื่อ Login เข้าใช้เว็บแอพพลิเคชั่น

**4.2 ผลการทดลองเติมเงินผ่านเว็บแอพพลิเคชั่น**

จากภาพที่ 4-4 โดยระบบเติมเงินนี้จะเป็นระบบทดสอบที่ทางบริษัท Omise มีให้เพื่อให้นักพัฒนาได้ทำการทดสอบการเติมเงินจริง ๆ โดยจะมีเลขบัตรต่าง ๆ ให้



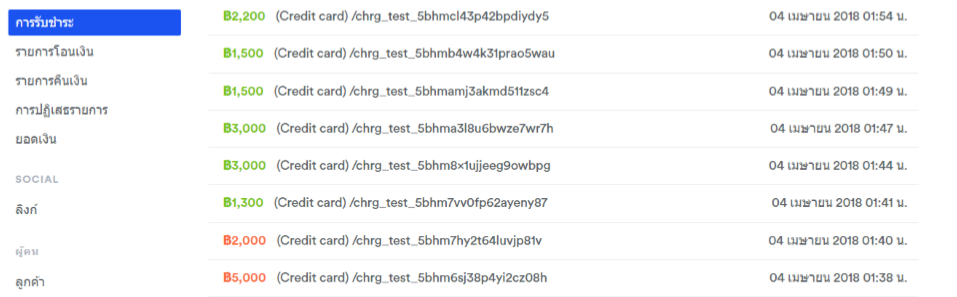
**ภาพที่ 4-4** เลขบัตรเครดิตที่ใช้ทดสอบระบบเติมเงินจากเว็บไซต์ Omise

จากภาพที่ 4-3 เมื่อกดเติมเงิน จะแสดงหน้าดังภาพที่ 4-5 ให้ใส่ข้อมูลสมมติลงไปยกเว้นเลขบัตร เลขบัตรจะใช้ของ Omise ที่มีให้ทดสอบจากภาพที่ 4-4 เลือกมาใส่ในช่อง Card ID ในภาพที่ 4-5



**ภาพที่ 4-5** หน้าเว็บไซต์แสดงข้อมูลบัตรเครดิตที่ต้องกรอก

จากภาพที่ 4-6 เป็น Dashboard ในโหมดทดสอบของบริษัท Omise แสดงจำนวนเงินผู้ใช้ที่ทำการเติมเข้ามา ถ้าขึ้นเป็นตัวเลขสีเขียวคือเติมเงินสำเร็จ ถ้าเป็นสีแดงคือไม่สำเร็จ



**ภาพที่ 4-6** หน้า Dashboard จากเว็บไซต์ Omise

**4.3 ทดลองการใช้งานสถานีประจุไฟฟ้า**

จากภาพที่ 4-7 เมื่อสถานีประจุไฟฟ้าเชื่อมต่อกับ Gateway - Server สำเร็จจะแสดงหน้าแตะบัตร เพื่อบอกให้ผู้ใช้แตะบัตรที่ได้ลงทะเบียนไว้ก่อนใช้บริการสถานีประจุไฟฟ้า



**ภาพที่ 4-7** หน้าจอสถานีประจุไฟฟ้าแสดงให้แตะบัตร

จากตัวอย่างด้านล่างเป็นข้อความจาก Console เป็น Message Heartbeat ที่ Gateway - Server ได้รับ สถานีประจุไฟฟ้าส่งไปยัง Gateway - Server เป็นข้อความสำหรับบอกสถานีประจุไฟฟ้ายังทำงานอยู่ Message Heartbeat จะส่งทุก ๆ 1 วินาที และ Gateway - Server จะตอบกลับเป็นเวลาปัจจุบัน

|  |
| --- |
| received: [2, "de3983fe4074caa3d51256c2661138f18f76d0ce", "Heartbeat", {}] sent: [3, "de3983fe4074caa3d51256c2661138f18f76d0ce", {"currentTime":  "2018-08-20T03:25:26.420Z"}] |

จากภาพที่ 4-8 เมื่อผู้ใช้ได้ทำการแตะบัตรสถานีประจุไฟฟ้าจะมีข้อมูลของผู้ใช้ขึ้นมาแสดงประมาณ 3 วินาที



**ภาพที่ 4-8** หน้าจอสถานีประจุไฟฟ้าแสดงข้อมูลผู้ใช้เมื่อแตะบัตร

Message Authorize สถานีประจุไฟฟ้าส่งไปยัง Gateway - Server เป็นข้อความสำหรับยืนยันตันตนของผู้ใช้ Gateway - Server จะนำเอา ID Tag ไปตรวจสอบในฐานข้อมูลและตอบกลับ โดยจะมี Status บ่งบอกว่าถูกต้องหรือไม่

จากตัวอย่างด้านล่างเป็นข้อความจาก Console มี Status เป็น Accepted ที่ผู้ใช้สามารถใช้งานสถานีประจุไฟฟ้าได้

|  |
| --- |
| received: [2, "BQMYei0kseAoZ2aij7mbTs37UNGCFLhv", "Authorize", {"idTag": "D86F20CE"}] sent: [3, "BQMYei0kseAoZ2aij7mbTs37UNGCFLhv", {"idTagInfo": {"status": "Accepted", "name": "Nattpong", "amount": 2000}}] |

จากตัวอย่างด้านล่างเป็นข้อความจาก Console มี Status เป็น Blocked ที่ผู้ใช้ไม่สามารถใช้งานสถานีประจุไฟฟ้าได้ เพราะบัตรหมดอายุหรือบัตรถูกระงับไว้

|  |
| --- |
| received: [2, "BQMYei0kseAoZ2aij7mbTs37UNGCFLhv", "Authorize", {"idTag": "D86F20CE"}] sent: [3, "BQMYei0kseAoZ2aij7mbTs37UNGCFLhv", {"idTagInfo": {"status": "Blocked"}}] |

จากภาพที่ 4-9 เมื่อยืนยันตัวตนผ่านแล้วสถานีประจุไฟฟ้าจะแสดงหน้าให้ผู้ใช้กดปุ่มเพื่อเริ่มชาร์จ



**ภาพที่ 4-9** หน้าจอสถานีประจุไฟฟ้าแสดงให้กดปุ่มเริ่มชาร์จ

จากภาพที่ 4-10 เมื่อกดปุ่มสถานีประจุไฟฟ้าจะแสดงหน้าให้ผู้ใช้เสียบหัวชาร์จเข้ากับยานพาหนะไฟฟ้าของผู้ใช้



**ภาพที่ 4-10** หน้าจอสถานีประจุไฟฟ้าแสดงให้เสียบหัวชาร์จ

จากภาพที่ 4-11 เมื่อผู้ใช้ทำการเสียบหัวชาร์จเข้ากับยานพาหนะของผู้ใช้แล้ว สถานีประจุไฟฟ้าจะแสดงหน้ากำลังชาร์จไฟฟ้า โดยจะมีข้อมูลพลังงาน กระแสไฟ และราคาแสดงอยู่ด้วย เมื่อชาร์จเสร็จหรือจะหยุดชาร์จให้นำบัตรที่แตะตอนเริ่มชาร์จมาแตะเพื่อทำการหยุดชาร์จ



**ภาพที่ 4-11** หน้าจอสถานีประจุไฟฟ้าแสดงข้อมูลสถานะกำลังชาร์จ

จากตัวอย่างด้านล่างเป็นข้อความจาก Console เมื่อผู้ใช้ทำการเสียบหัวชาร์จเข้ากับยานพาหนะของผู้ใช้แล้ว Gateway - Server จะรับ Message StatusNotification , StartTransaction และ MeterValues สำหรับการเริ่มชาร์จ ส่วนแรก StatusNotification ที่ Gateway - Server ได้รับจะบ่งบอกสถานะของสถานีประจุไฟฟ้าว่ากำลังชาร์จอยู่ (status = Charging) และใช้หัวชาร์จอันไหน (connectorId = 1) ส่วนที่สอง StartTransaction ที่ Gateway - Server ได้รับจะบ่งบอกว่าใช้หัวชาร์จอันไหนเริ่มชาร์จ (connectorId = 1) ค่า Meter เริ่มชาร์จเท่าไหร่และ ID Tag ผู้เริ่มชาร์จและ Gateway - Server จะตอบกลับมาว่าเริ่มชาร์จด้วย Transaction ID เท่าไหร่ (transactionId = 10) ส่วนที่สาม MeterValues ที่ Gateway - Server ได้รับจะได้รับเรื่อยๆจนกว่าจะหยุดชาร์จและค่าพลังงาน (Value) ที่ชาร์จไปจะเพิ่มขึ้นเรื่อย ๆ สำหรับใช้คำนวณคิดเงินผู้ใช้

|  |
| --- |
| received: [2, "ZeERlwkKOEdm9qaCnTUpDXI95bxr1ot3", "StatusNotification", {"timestamp": "2018-08-20 05:3:01", "status": "Charging", "connectorId": 1, "errorCode": "NoError"}]  sent: [3, "ZeERlwkKOEdm9qaCnTUpDXI95bxr1ot3", {}]  received: [2, "XHgRau8AcLOUezhYFvZWKMi83DeP1bvC", "StartTransaction", {"connectorId": 1, "meterStart": 0, "idTag": "D86F20CE", "timestamp": "2018-08-20 05:3:01"}] sent: [3, "XHgRau8AcLOUezhYFvZWKMi83DeP1bvC", {"idTagInfo": {"status": "Accepted"}, "transactionId": 10}]  received: [2, "uIDHeJPq6QESJEKY2Tu1qzimdxgfBQNW", "MeterValues", {"connectorId": 1 "meterValue": {"sampledValue": {"Value": 0, "unit": "Wh"}}}]  sent: [3, "uIDHeJPq6QESJEKY2Tu1qzimdxgfBQNW", {}] |

**4.4 ทดลองยกเลิกการใช้งานสถานีประจุไฟฟ้า**

จากภาพที่ 4-12 เมื่อแตะบัตร สถานีประจุไฟฟ้าจะแสดงหน้าให้กดปุ่มหยุดชาร์จ กดหนึ่งทีเพื่อทำการหยุดชาร์จ



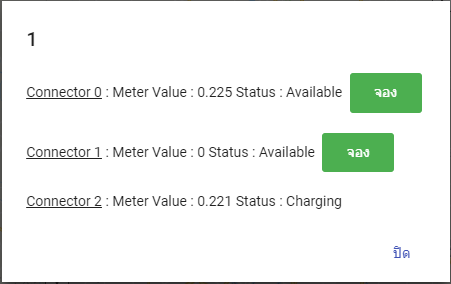
**ภาพที่ 4-12** หน้าจอสถานีประจุไฟฟ้าแสดงให้กดปุ่มยกเลิกการชาร์จ

จากตัวอย่างด้านล่างเป็นข้อความจาก Console เมื่อผู้ใช้ทำการแตะบัตรเพื่อหยุดชาร์จ Gateway - Server จะรับ Message StatusNotification และ StartTransaction สำหรับการหยุดชาร์จ ส่วนแรก StatusNotification ที่ Gateway - Server ได้รับจะบ่งบอกสถานะของสถานีประจุไฟฟ้าว่าว่างแล้วหรือทำการหยุดชาร์จไปแล้วสามารถให้ผู้ใช้อื่นมาใช้ต่อได้ (status = Available) และหัวชาร์จอันไหนที่ว่างแล้ว (connectorId = 1) ส่วนที่สอง StopTransaction ที่ Gateway - Server ได้รับจะบ่งบอกว่า Transaction ID ที่เท่าไหร่หยุดชาร์จ เวลาที่หยุดชาร์จ ID Tag ผู้ใช้ที่หยุดชาร์จและ Meter Stop เป็นเท่าไหร่ สุดท้าย Gateway - Server จะตอบกลับด้วยสถานะ Accepted คือทำการยกเลิกการชาร์จเรียบร้อยแล้ว

|  |
| --- |
| received: [2, "ZeERlwkKOEdm9qaCnTUpDXI95bxr1ot3", "StatusNotification", {"timestamp": "2018-08-20 05:3:01", "status": "Available", "connectorId": 1, "errorCode": "NoError"}]  sent: [3, "ZeERlwkKOEdm9qaCnTUpDXI95bxr1ot3", {}]  received: [2, "XHgRau8AcLOUezhYFvZWKMi83DeP1bvC", "StopTransaction", {"transactionId": 10,  "meterStop": 0, "idTag": "D86F20CE", "timestamp": "2018-08-20 05:3:01"}] sent: [3, "XHgRau8AcLOUezhYFvZWKMi83DeP1bvC", {"idTagInfo": {"status": "Accepted"}] |

**4.5 ผลการทดลองระบบการจอง**

ในเว็บแอพพลิเคชั่นจะมีบริการระบบจองสถานีประจุไฟฟ้า โดยใน Map จะมีตำแหน่งของสถานีประจุไฟฟ้าเมื่อผู้ใช้คลิกที่ตำแหน่งสถานีประจุไฟฟ้า จะแสดงถึงสถานะ Connector ที่สามารถจองได้   
ดังแสดงในภาพที่ 4-13

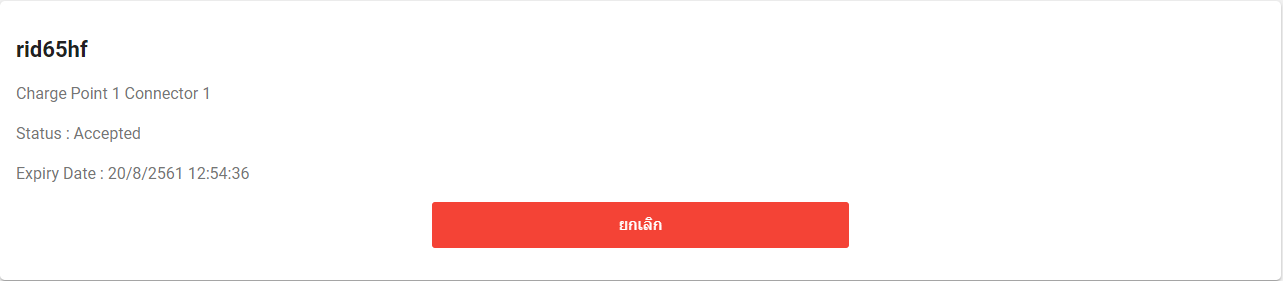


**ภาพที่ 4-13** หน้าเว็บไซต์แสดงสถานะ Connector ที่สามารถจองได้

จากตัวอย่างด้านล่างเป็นข้อความจาก Console เป็น Message ReserveNow ที่ส่งจากเว็บแอพพลิเคชั่นไปยัง Gateway - Server โดยให้ Gateway - Server ส่งไปยังสถานีประจุไฟฟ้าเพื่อทำการจอง ดังนั้น Connector ที่ถูกจองจะไม่สามารถจองได้อีก

|  |
| --- |
| received: [2, "ZeERlwkKOEdm9qaCnTUpDXI95bxr1ot3", "ReserveNow", {"connectorId": 1, "reservationId": 5, "idTag": "D86F20CE", "expiryDate": "20/8/2561 12:51:01"}] |

เมื่อจองสำเร็จในเว็บแอพพลิเคชั่นจะแสดงสถานะ Accepted และแสดงเวลาสิ้นสุดการจอง   
ผู้ใช้สามารถยกเลิกการจองได้ ดังแสดงในภาพที่ 4-14



**ภาพที่ 4-14** หน้าเว็บไซต์แสดงสถานะการจองทางเว็บแอพพลิเคชั่น

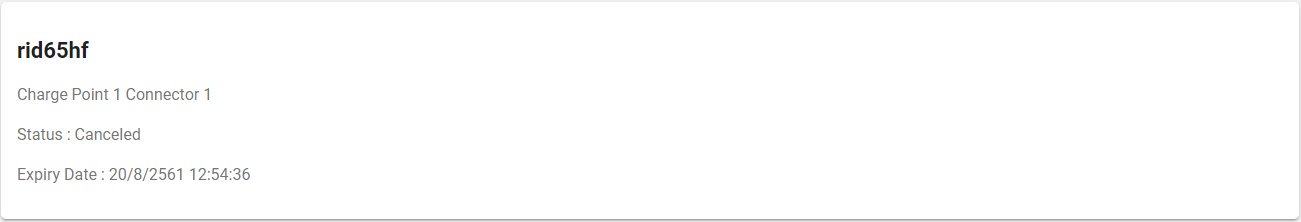
เมื่อมีการจองสถานีประจุไฟฟ้าจะแสดงหน้าว่ามีผู้ใช้ท่านอื่นได้ทำการจองไว้แล้ว   
ผู้ใช้ท่านอื่นจะไม่สามารถเข้าใช้ได้นอกจากผู้ใช้ที่ได้ทำการจองไว้มาแตะบัตรเพื่อใช้งาน ดังแสดงในภาพที่ 4-15



**ภาพที่ 4-15** หน้าจอแสดงผลสถานีประจุไฟฟ้ามีการจองแล้ว

**4.6 ผลการทดลองยกเลิกการจอง**

เมื่อผู้ใช้กดปุ่มยกเลิก ระบบจะทำการยกเลิกการจองให้ผู้ใช้ทันทีและจะแสดงสถานะ Canceled ดังแสดงในในภาพที่ 4-16



**ภาพที่ 4-16** หน้าเว็บไซต์แสดงสถานะยกเลิกการจอง

จากตัวอย่างด้านล่างเป็นข้อความจาก Console เป็น Message CancelReservation ที่ส่งจากเว็บแอพพลิเคชั่นไปยัง Gateway - Server โดยให้ Gateway - Server ส่งไปยังสถานีประจุไฟฟ้าเพื่อทำการยกเลิกการจอง จากนั้นผู้ใช้ท่านอื่นก็จะสามารถใช้งานสถานีประจุไฟฟ้านั้นได้และสถานะการจองจะเปิดให้จองได้อีกครั้ง

|  |
| --- |
| received: [2, "ZeERlwkKOEdm9qaCnTUpDXI95bxr1ot3", "CancelReservation", {"reservationId": 5}] |

**บทที่ 5**

**สรุปและวิเคราะห์การดำเนินงาน**

**5.1 สรุปผลการดำเนินงาน**

จากการออกแบบการดำเนินงาน ศึกษาโปรโตคอล OCPP (Open Charge Point Protocol)   
ที่เป็นมาตรฐานในการติดต่อสื่อสารระหว่างสถานีอัดประจุไฟฟ้า (Charge Point) และระบบศูนย์กลาง (Central System) ถูกออกแบบมาสำหรับสถานีอัดประจุไฟฟ้าทุกชนิด OCPP จะประกอบด้วยมาตรฐานที่ระบุว่าแต่ละข้อความนั้นควรมีรูปแบบการส่งอย่างไร เช่น การกำหนดชนิดของตัวแปร ชื่อของตัวแปร ความยาวของข้อมูลในตัวแปร เป็นต้น ซึ่งเป็นโปรโตคอลที่เข้าใจได้ไม่ยากเลย

ในแต่ละข้อความของ OCPP จะใช้รูปแบบในการรับ - ส่งข้อมูลแบบ JSON โดยแพ็คเข้ากับ RPC ที่เป็นมามาตรฐานของชุดข้อมูล ที่บ่งบอกสถานะของผู้รับ - ผู้ส่ง โดยใช่ช่องทาง Web Socket   
ในการติดต่อสื่อสาร เป็นช่องทางในการรับส่งข้อมูลในรูปแบบ Real Time และระบบศูนย์กลางจะทำหน้าที่รับข้อความมาตรวจสอบผู้ใช้ในฐานข้อมูลว่ามีหรือไม่และส่งกลับว่าถูกต้องหรือไม่ บันทึกสถานะต่าง ๆ ของสถานีประจุไฟฟ้าลงฐานข้อมูล จากนั้นเว็บแอพพลิเคชั่นจะนำสถานะของสถานีประจุไฟฟ้ามาแสดงในแผนที่ ฐานข้อมูลนั้นได้ใช้เป็น Firebase ซึ่งเป็นฐานข้อมูลแบบ Real Time และเป็นภาษา NoSQL ที่เก็บข้อมูลในรูปแบบ JSON จึงง่ายต่อการเก็บข้อมูลในรูปแบบการรับ - ส่งข้อมูลระหว่างสถานีประจุไฟฟ้าและระบบศูนย์กลาง ซึ่งเหมาะกับเว็บแอพพลิเคชั่นที่มีการอัพเดตค่าอยู่ตลอดเวลา เช่น สถานะของสถานีประจุไฟฟ้าที่นำมาแสดงในแผนที่และระบบเติมเงิน/จ่ายเงินที่จะต้องอัพเดตจำนวนเงิน เป็นต้น

ประโยชน์ที่ได้รับจากการทำปริญญานิพนธ์นี้

1. เว็บแอปพลิเคชันสามารถให้ผู้ใช้ที่ใช้ยานพาหนะไฟฟ้ามีความสะดวกสบายมากยิ่งขึ้นในการใช้บริการผ่านเว็บแอพพลิเคชั่น

2. สามารถจองสถานีอัดประจุไฟฟ้าล่วงหน้าได้ผ่านเว็บแอพพลิเคชั่น

3. สามารถชำระเงินผ่านแอพพลิเคชั่นได้

4. สามารถดูสถานะของสถานีอัดประจุไฟฟ้าและตำแหน่งของสถานีประจุไฟฟ้าผ่านทางเว็บแอพพลิเคชั่นได้

**5.2 ข้อเสนอแนะและการดำเนินงานต่อไป**

อย่างไรก็ตามเว็บแอพพลิเคชั่นสำหรับสถานีประจุไฟฟ้า ยังอยู่ในช่วงทดสอบจึงยังไม่สามารถใช้จริงได้ เนื่องจากเว็บแอพพลิเคชั่นยังใช้งานได้แค่บน Local Host และบางฟังก์ชันของเว็บแอพพลิเคชั่นยังต้องพัฒนาอีก เช่น หาตำแหน่งที่ใกล้สุดระหว่างสถานีประจุไฟฟ้ากับผู้ขับยานพาหนะไฟฟ้าและระบบเติมเงินเป็นโหมดทดสอบของ Omise เป็นต้น

**เอกสารอ้างอิง**

1. สํานักงานนโยบายและแผนพลังงาน. รายงานฉบับสมบูรณ์ (ฉบับปรับปรุง) : โครงการศึกษาการเตรียมความพร้อมรองรับการใช้ยานพาหนะไฟฟ้าในอนาคตสําหรับประเทศไทย. [Online]. Available:http://www.eppo.go.th/images/Infromation\_service/studyreport/  
ensol.pdf

2. HTML5 Web Storage.

[Online]. Available: <http://www.siamhtml.com/html5-web-storage/>

3. json.   
[Online]. Available: <http://www.boxsingle.com/?page=Blog.ShowBlogDetail&blogID=13>

4. HTML5 ใช้ JavaScript เก็บค่าตัวแปรที่ Client ด้วย localStorage และ sessionStorag.

[Online].Available:<https://www.thaicreate.com/community/html5javascriptlocalstorage-sessionstorage.html>

5. Websocket.   
[Online]. Available: <https://khasathan.in.th/archives/75/>

6. ทฤษฎีทำเว็บแบบ Realtime.

[Online]. Available: <https://engineering.thinknet.co.th>

7. วัชรากร หนูทอง, อนุกูล น้อยไม้ และปรินันท์ วรรณสว่าง. RFID เทคโนโลยีสารพัดประโยชน์.

[Online]. Available: <http://www.lampangtc.ac.th/mnfile/branch5/file/knowledge/RFID.pdf>

8. ฐานข้อมูล NoSQL.

[Online]. Available:<https://www.techtalkthai.com/4-basic-nosql-databases-you-should-know/>

9. Python (Version 2.7).

[Online]. Available: <https://www.python.org/downloads/>

10. OCPP 1.6 (Open Charge Point Protocol).

[Online]. Available: <https://www.openchargealliance.org/>

11. RPC (Remote Procedure Call).

[Online]. Available: <https://www.mindphp.com/>

**เอกสารอ้างอิง (ต่อ)**

12. หลักการ Piggybacking.

[Online]. Available: <https://en.wikipedia.org/wiki/Piggybacking_(data_transmission)>

13. Payment Gateway Omise.

[Online]. Available: <https://www.omise.co/th>

14. เทคโนโลยีของเว็บไซต์ (Web Technology).

[Online]. Available: <http://www.nextproject.net/contents/default.aspx?00052>

15. Beaglebone Black Getting Started.

[Online]. Available: <http://beagleboard.org/getting-started>

16. Serial Ports / UART Beaglebone Black.

[Online]. Available: <http://beaglebone.cameon.net/home/serial-ports-uart>

17. Webcomponents.

[Online]. Available: <https://www.webcomponents.org>

18. Polymer.

[Online]. Available: <https://www.polymer-project.org/>

19. Firebase ฐานข้อมูล NoSQL.

[Online]. Available: <https://firebase.google.com/>

**ประวัติผู้แต่ง**

ปริญญานิพนธ์เรื่อง : เว็บแอพพลิเคชั่นสำหรับสถานีประจุไฟฟ้าโดยใช้โปรโตคอล OCPP

สาขา : วิศวกรรมคอมพิวเตอร์

ภาควิชา : วิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะ : วิศวกรรมศาสตร์

ชื่อ : นายณัฐพงษ์ วิญญูนุรักษ์

ประวัติ

เกิดเมื่อวันที่ 22 พฤษภาคม พ.ศ. 2538 บ้านเลขที่ 142/135 หมู่ 7 หมู่บ้านพูนสุข ซอยวัดศรีวารีน้อย ถนนบางนาตราด ตำบลบางโฉลง อำเภอบางพลี จังหวัดสมุทรปราการ 10540 สำเร็จการศึกษาระดับมธัยมศึกษาตอนปลาย สาขาวิทยาศาสตร์-คณิตศาสตร์ จากโรงเรียนพูลเจริญวิทยาคม จังหวัดสมุทรปราการ ปีการศึกษา 2557 ผ่านการฝึกงานกับบริษัท เน็กซ์ คอร์ป จำกัด และสำเร็จการศึกษาระดับปริญญาตรี สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จากมหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ จังหวัดกรุงเทพมหานคร ปีการศึกษา 2562

ชื่อ : นายปฐมภัทร์ สูงปานเขา

ประวัติ

เกิดเมื่อวันที่ 22 มกราคม พ.ศ. 2539 บ้านเลขที่ 70/296 หมู่ 1 ถนนบางกรวย - ไทรน้อย ตำบลบางบัวทอง อำเภอบางบัวทอง จังหวัดนนทบุรี 11110 สำเร็จการศึกษาระดับมัธยมศึกษาตอนปลาย สาขาวิทยาศาสตร์-คณิตศาสตร์ จากโรงเรียนวัดเขมาภิรตาราม จังหวัดนนทบุรี ปีการศึกษา 2557 ผ่านการฝึกงานกับบริษัท เน็กซ์ คอร์ป จำกัด และสำเร็จการศึกษาระดับปริญญาตรี สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จากมหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ จังหวัดกรุงเทพมหานคร ปีการศึกษา 2561