ระบบรวบรวมและจัดการมาตรฐานสำหรับการจัดทำเอกสาร Web API

A Standard Collection and Management System for Web API Documentation

โดย

ขันติชัย รุจิตระการโชติกุล

Kantichai Rujitrakarnchotikul

อาจารย์ที่ปรึกษา

ผศ.ดร. สุภกิจ นุตยะสกุล

รายงานนี้เป็นส่วนหนึ่งของวิชาการศึกษาอิสระ 2
หลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ
คณะเทคโนโลยีสารสนเทศ
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ภาคเรียนที่ 2 ปีการศึกษา 2564

A Standard Collection and Management System for Web API Documentation

Kantichai Rujitrakarnchotikul

A REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE

REQUIREMENTS OF THE COURSE INDEPENDENT STUDY 2

MASTER OF SCIENCE PROGRAM IN INFORMATION TECHNOLOGY

FACULTY OF INFORMATION TECHNOLOGY

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2/2021

ใบรับรองโครงการ

การศึกษาอิสระ 2 (INDEPENDENT STUDY 2)

เรื่อง

ระบบบรวบรวมและจัดการมาตรฐานสำหรับการจัดทำเอกสาร Web API

A Standard Collection and Management System for Web API Documentation

ขันติชัย รุจิตระการโชติกุล

รหัสประจำตัว 63606041

ขอรับรองว่ารายงานฉบับนี้ ข้าพเจ้าไม่ได้คัดลอกมาจากที่ใด
รายงานฉบับนี้ได้รับการตรวจสอบและอนุมัติให้เป็นส่วนหนึ่งของ
การศึกษาวิชาการศึกษาอิสระ 2 หลักสูตรวิทยาศาสตรมหาบัณฑิต (เทคโนโลยีสารสนเทศ)
ภาคเรียนที่ 2 ปีการศึกษา 2564

.....**อาจารย์ที่ปรึกษา**

(ผศ.ดร. สุภกิจ นุตยะสกุล)

.....**กรรมการสอบ**

(ผศ.ดร. โอฬาร วงศ์วิรัตน์)

.....**กรรมการสอบ**

(ผศ.ดร. อนันตพัฒน์ อนันตชัย)

หัวข้อ	ระบบรวบรวมและจัดการมาตรฐานสำหรับการจัดทำเอกสาร Web API
รหัสนักศึกษา	63606041
นักศึกษา	นายขันติชัย รุจิตระการโชติกุล
ปริญญา	วิทยาศาสตรมหาบัณฑิต
สาขาวิชา	เทคโนโลยีสารสนเทศ
แขนงวิชา	เทคโนโลยีสารสนเทศและการจัดการ
ปีการศึกษา	2564
อาจารย์ที่ปรึกษา	ผศ.ดร. สุภกิจ นุตยะสกุล

บทคัดย่อ

การค้นคว้าอิสระนี้นำเสนอเอกสารเกี่ยวกับระบบการจัดการ Web API ที่ช่วยให้หน่วยงานด้านเทคโนโลยีสารสนเทศ จัดการระบบการเชื่อมต่อข้อมูลระหว่างหน่วยงาน หรือระบบอื่นๆ ก่อนมีการพัฒนาระบบนี้ ผู้พัฒนาพบว่าประเภทของการเชื่อมต่อข้อมูลแบบดั้งเดิม ขาดวิธีการพัฒนาที่มีประสิทธิภาพ และการจัดทำข้อมูลที่มีซ้ำซ้อนโดยมนุษย์ ระบบที่ได้พัฒนานี้สามารถจัดระเบียบมาตรฐานของการจัดทำเอกสาร Web API และระบุจุดเชื่อมต่อของ Web API สิ่งนี้ช่วยให้ผู้พัฒนาระบบสร้างการเชื่อมต่อใหม่ไปยังจุดเชื่อมต่อ Web API ได้อย่างง่ายดาย ที่สามารถช่วยลดความซับซ้อนและความซ้ำซ้อนได้

คำสำคัญ : การรวบรวมจุดเชื่อมต่อ Web API, การแนะนำมาตรฐานในการจัดการ Web API, การจัดทำเอกสาร Web API

Title	A Standard Collection and Management System for Web API Documentation
Student ID	63606041
Student	Mr. Kantichai Rujitrakarnchotikul
Degree	Master of Science
Program	Information Technology
Major	Information Technology and Management
Academic Year	2021
Advisor	Asst. Prof. Dr.Supakit Nootyaskool

ABSTRACT

This independent study presents a document on the Web API management system. The proposed system helps the department of information technology to organize the data transferring between the department or the different systems. Before developing the proposed system, the developer found that type of legacy data connection, which causes of lack of effective development and occurring the redundancy data creations by human. The proposed system can organize the standard of Web API documents and indicate the Web API endpoints. This helps the developer set a new connection to the Web API endpoints easily, which can reduce complexity and redundancy.

Keywords : Web API Endpoints Collection, Web API Standard Recommendation, Web API Documentation

กิตติกรรมประกาศ

โครงการพัฒนาระบบในหัวข้อเรื่อง “ระบบรวบรวมและจัดการมาตรฐานสำหรับการจัดทำเอกสาร Web API” ที่ได้ศึกษาดังกล่าวสำเร็จลุล่วงไปได้ด้วยดี โดยได้รับการให้คำแนะนำ สั่งสอน สนันสนุน และความช่วยเหลือจากอาจารย์หลายๆท่าน โดยเฉพาะอย่างยิ่งอาจารย์ที่ปรึกษา อาจารย์ ผศ.ดร.สุภกิจ นุตยะสกุล ที่กรุณาชี้แนะและให้คำแนะนำในการทำโครงการมาโดยตลอด ผู้พัฒนามีความซาบซึ้งในความกรุณาของท่านอาจารย์เป็นอย่างยิ่ง

ผู้พัฒนาขอขอบพระคุณ พี่ๆ งานระบบสารสนเทศ สำนักทะเบียนและประมวลผล สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ทั้งครอบครัวและเพื่อนๆของผู้พัฒนาทุกคนที่ค่อยเป็นกำลังใจ ช่วยเหลือ และอดทนมาด้วยกัน

ในท้ายที่สุด ผู้พัฒนาขอขอบคุณตนเองที่อดทน สู้ ฝ่าฟัน ปัญหา อุปสรรต่างๆมาได้จนสำเร็จ รวมถึงขอบขอบพระคุณทุกท่านที่ได้กล่าวมาและที่มิได้กล่าวมา ณ ที่นี้ด้วย ที่มีส่วนสนับสนุนและเป็นกำลังใจให้จนทำให้การศึกษาวิจัยสำเร็จลุล่วงไปได้ด้วยดี

ขันติชัย รุจิตระการโชติกุล

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

ตารางที่ **หน้า**

สารบัญรูป

รูปที่ **หน้า**

สารบัญรูป (ต่อ)

รูปที่ **หน้า**

สารบัญรูป (ต่อ)

รูปที่ **หน้า**

บทที่ 1
บทนำ

ปัจจุบันระบบสารสนเทศมีความยุ่งยากซับซ้อนสำหรับการเชื่อมต่อข้อมูลระหว่างหน่วยงาน ผู้พัฒนาจึงมีแนวคิดพัฒนาระบบรวบรวมและจัดการมาตรฐานสำหรับการจัดทำเอกสาร Web API เพื่อสนับสนุนการทำงานของฝ่ายระบบสารสนเทศ ซึ่งเป็นเอกสารที่เข้ามาช่วยให้กลุ่มผู้พัฒนา/ดูแลระบบสารสนเทศภายในและภายนอก สามารถทำงานได้อย่างมีประสิทธิภาพ ลดเวลาทำงานซ้ำซ้อน และลดความสับสน สามารถแบ่งขั้นตอนการศึกษาเพื่อพัฒนาได้ดังต่อไปนี้

1.1 ประวัติความเป็นมา

API (Application Programming Interface) คือชุดของงาน จุดเชื่อมต่อสื่อสาร และเครื่องมือสำหรับการพัฒนาแอปพลิเคชัน (Application) โดยในระบบงานนี้กล่าวถึง Web API เป็น API ที่เชื่อมต่ออยู่บนพื้นฐานของ HTTP ซึ่งใช้งานบนเว็บไซต์ เป็นประเภทของ API ที่นิยมมากที่สุด เนื่องจากเข้าถึงง่าย และเว็บแอปพลิเคชัน (Web Application) เป็นที่นิยมใช้งานในปัจจุบัน

สำนักทะเบียนและประมวลผล สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง เป็นสำนักงานที่เกี่ยวข้องกับข้อมูลของนักศึกษาโดยตรง โดยในปีงบประมาณ 2565 นี้ได้แบ่งหน่วยงานใหม่ในสำนักเป็น 8 หน่วยงานคือ งานรับเข้าศึกษา, งานทะเบียนระดับปริญญาตรี, งานทะเบียนระดับบัณฑิตศึกษา, งานบริการเคาน์เตอร์ (Counter Service), งานสารบรรณ, งานบริหารจัดการกลาง, งานระบบสารสนเทศ และงานวิเคราะห์ข้อมูล (Data Analytics) โดยมีพันธกิจและนโยบายดังนี้

พันธกิจหลักคือ ยกระดับ S ทั้ง 4 ประการอย่างต่อเนื่อง หมายความถึง การปรับปรุง/เปลี่ยนแปลง งานบริการ (Services) และบุคลากร (Staffs) เพื่อมุ่งสู่เป้าหมายความพึงพอใจ (Satisfaction) ของผู้มีส่วนได้ส่วนเสีย และความยั่งยืน (Sustainability) ในด้านต่างๆ ที่เพิ่มขึ้นอย่างต่อเนื่อง

นโยบายคือ การกำหนดนโยบายการบริหารงานสำนักทะเบียนและประมวลผล ต้องมีการกำหนดเป้าหมายที่ชัดเจนก่อน โดยเป็นเป้าหมายที่สนับสนุนนโยบายของสถาบันฯ เป้าหมายในการดำเนินงานของสำนักฯ มุ่งเน้นผลที่เกิดกับผู้รับบริการและผู้อื่น ในส่วนของผู้รับบริการนั้น ความพึงพอใจ เป็นเป้าหมายสูงสุดที่ควรคำนึงถึง ส่วนผลที่เกิดกับผู้อื่นนั้น เป็นผลทางอ้อมที่เกิดกับสิ่งแวดล้อม จึงกำหนดให้ ความยั่งยืน เป็นเป้าหมายสูงสุดอีกประการ

มีนโยบายด้านบริการคือ พัฒนาปรับปรุงและเปลี่ยนแปลงงานบริการ โดยเน้นที่รูปแบบ วิธีการ เครื่องมือ ระบบสารสนเทศ นวัตกรรม ฯลฯ ที่ไม่เกี่ยวกับคน (กรณีของคน อยู่ในนโยบายด้านบุคลากร) แบ่งเป็น รูปแบบการให้บริการ ตอบสนองกลุ่มเป้าหมายที่หลากหลาย บริการที่ดีขึ้นและเพิ่มขึ้น และลดปัญหา/อุปสรรคในการบริการ

มีนโยบายด้านบุคลากรคือ บุคลากรของสำนักฯ เป็นฟันเฟืองที่สำคัญในการขับเคลื่อนและผลักดันในงานบริการบรรลุเป้าหมายได้ การพัฒนาบุคลากรให้มีความเป็นมืออาชีพมากขึ้น จึงต้องพัฒนาทั้งทักษะด้านความรู้/ความสามารถ (Hard Skill) และทักษะทางสังคม/อารมณ์ (Soft Skill) ผ่านกระบวนการพัฒนาทักษะเดิมที่มีอยู่และเสริมสร้างทักษะใหม่ๆ โดยต้องดำเนินการอย่างเป็นขั้นตอน เพื่อให้บุคลากรมีระดับของทักษะเพิ่มขึ้น

1.2 วัตถุประสงค์ของโครงการ

1. เพื่อศึกษา วิธีสร้างระบบรวบรวมและจัดการมาตรฐานสำหรับการจัดทำเอกสาร Web API ให้มีความสอดคล้องกับ กระบวนงานของสำนักทะเบียนและประมวลผลด้านระบบสารสนเทศ
2. เพื่อวิเคราะห์และออกแบบ ระบบรวบรวมและจัดการมาตรฐานสำหรับการจัดทำเอกสาร Web API เพื่อจัดทำเอกสารและคู่มือ API

1.3 ขอบเขตของโครงการ

1. รวบรวมข้อมูลที่มีความเกี่ยวข้องกับระบบรวบรวมและจัดการมาตรฐานสำหรับการจัดทำเอกสาร Web API
2. อธิบายการทำงานของ Web API รวมถึงตัวอย่างที่ใช้ภายในองค์กรอื่นๆ
3. การวิเคราะห์และออกแบบ ระบบรวบรวมและจัดการมาตรฐานสำหรับการจัดทำเอกสาร Web API
4. การจัดทำเอกสาร Web API

1.4 ปัญหาและการแก้ปัญหา

1.4.1 ปัญหาของระบบปัจจุบัน

สมัยเริ่มแรก การรับ-ส่งข้อมูลทั้งภายในและระหว่างหน่วยงานเป็นแบบ กระดาษและไฟล์ Excel โดยส่งผ่านซองเอกสาร อีเมล เว็บไซต์ และอุปกรณ์เก็บข้อมูลแบบพกพา ปัญหาที่เกิดขึ้นคือ ข้อมูลไม่สามารถอัปเดตแบบเรียลไทม์ (Real-time) ได้ แล้วต้องมีการจัดทำข้อมูลใหม่ทุก

ครั้งในการส่งข้อมูลในแต่ละครั้ง ทำให้ข้อมูลเกิดความล่าช้า และใช้เวลาในการจัดทำข้อมูลแบบซ้ำๆ

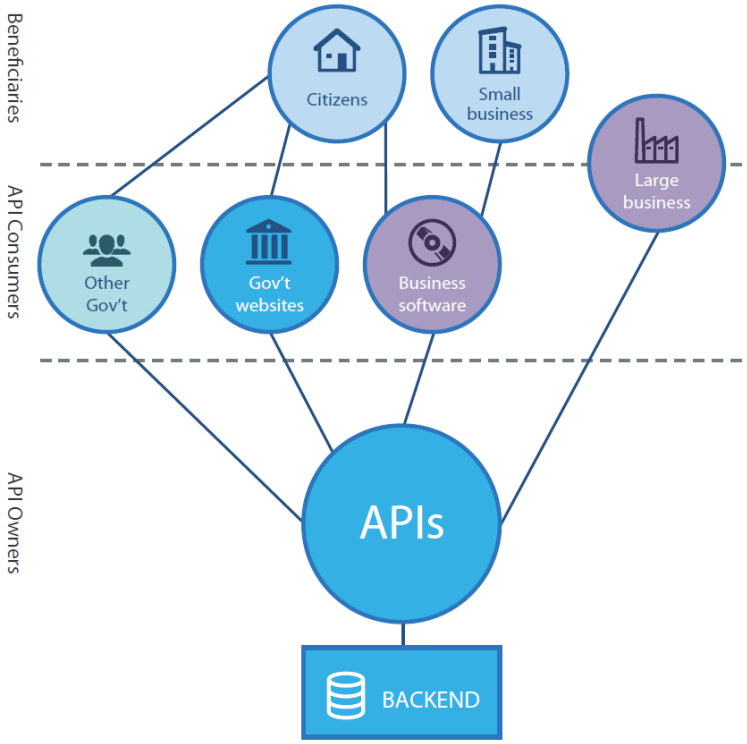
การทำงานในปัจจุบันที่เกิดขึ้นคือมีช่องทางให้ข้อมูลแบบเรียลไทม์ 2 วิธีหลักๆ คือ API และ View Table ทำให้เจ้าหน้าที่สามารถดึงข้อมูลต่างๆ ได้แบบออนดีมานด์ (On-demand) แต่ปัญหาที่เกิดขึ้นและยังไม่ได้รับการแก้ไขคือ มีหลายหน่วยงานทั้งภายนอกและภายใน ขอเชื่อมต่อข้อมูลเข้ามาเป็นจำนวนมาก พร้อมกับมีการแก้ไขรูปแบบการให้ข้อมูลหลายครั้ง ทำให้เกิดการกระจัดกระจายของจุดเชื่อมต่อข้อมูล ผู้จัดทำข้อมูลไม่มีเอกสารที่รวบรวมจุดเชื่อมต่อข้อมูลอย่างเป็นระบบ ทำให้เกิดการทำงานซ้ำซ้อน เพราะจำไม่ได้ว่าพัฒนาจุดเชื่อมต่อไว้ที่ใดบ้าง รวมถึงการแก้ปัญหาการใช้งานของจุดเชื่อมต่อข้อมูลเดิม เพราะต้องเริ่มไล่เส้นทางจุดเชื่อมข้อมูลตั้งแต่แรกใหม่ ว่าจุดที่ต้องการแก้ไขอยู่ที่ใด และมีอัตราการเพิ่มขึ้นของการนำ Web API มาใช้งานดังรูปที่ 1.1

รูปที่ 1.1 อัตราการเพิ่มขึ้นของการนำ Web API มาใช้งานของ ProgrammableWeb

(ที่มา : ProgrammableWeb.com (2019))

1.4.2 การแก้ปัญหา

ระบบรวบรวมและจัดการมาตรฐานสำหรับการจัดทำเอกสาร Web API นี้เพื่อใช้ในการแก้ไขปัญหาดังกล่าว ตามข้อ 1.4.1 โดยมีเป้าหมาย ให้ผู้พัฒนาจุดเชื่อมต่อ และผู้ใช้งานจากทั้งหน่วยงานภายในและภายนอก สามารถใช้เอกสารนี้เป็นที่อ้างอิงการใช้งานจุดเชื่อมต่อ Web API โดยแนะนำและรวบรวม วิธีการใช้งาน, วิธีสร้าง/เชื่อมจุดเชื่อมต่อ Web API ตามมาตรฐาน และรวบรวมจุดเชื่อมต่อ Web API ทั้งหมดที่มีอยู่ในระบบสารสนเทศของสำนักทะเบียนและประมวลผล โดยแสดงแผนผังตัวอย่างของจุดเชื่อมต่อดังรูปที่ 1.2



รูปที่ 1.2 แผนผังจุดเชื่อมต่อ APIs ของ Tax Office

(ที่มา : Australia – Australian Tax Office (2018))

1.5 แผนการดำเนินงาน

ตารางที่ 1.1 แผนการดำเนินงาน

หัวข้อกิจกรรม	เดือน									
	ส.ค.	ก.ย.	ต.ค.	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค.	เม.ย.	พ.ค.
1. ศึกษาระบบและความเป็นมาของ API										
2. อธิบายและวิเคราะห์ให้สอดคล้องกับงาน										
4. ออกแบบระบบงาน										
6. ดำเนินการจัดทำเอกสาร API										
7. วิเคราะห์ผลและปรับปรุงข้อผิดพลาดต่างๆ										
8. จัดทำเอกสารเพื่อตีพิมพ์, รวบรวมองค์ความรู้ทั้งหมดและสรุปผล										

1.6 ประโยชน์ที่คาดว่าจะได้รับ

1. เข้าใจหลักการทำงานของ API ที่ใช้ในระดับองค์กร
2. มีเอกสารรวบรวม API ทั้งหมดที่ใช้ในระดับองค์กร
3. ลดภาระการทำงานซ้ำซ้อน ที่เกิดจากพัฒนาของเดิมซ้ำ ทั้งที่มีอยู่ก่อนแล้ว
4. ลดระยะเวลาในกระบวนการทำงาน

1.7 สรุป

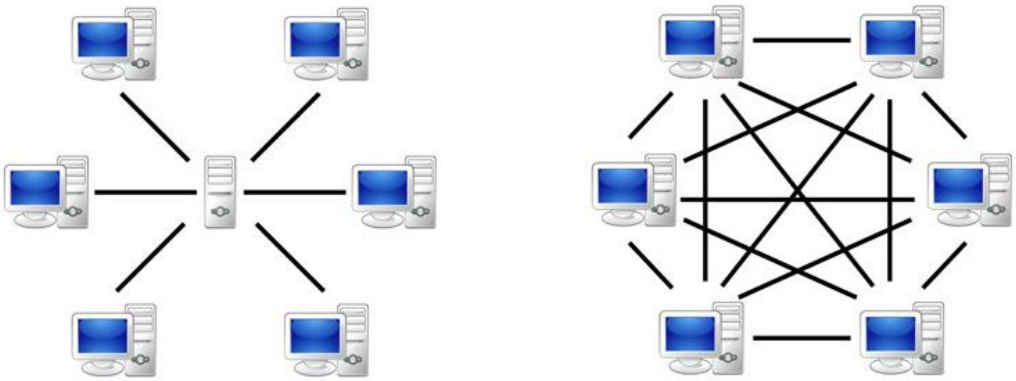
จากเนื้อหาทั้งหมดในบทที่ 1 ทำให้เห็นภาพรวมของระบบงาน ที่ช่วยสนับสนุนให้ข้อมูลภายในและระหว่างหน่วยงาน ภายใต้การรับ-ส่งข้อมูลผ่าน API ซึ่งช่วยรวบรวมจุดเชื่อมต่อ API รวมถึงรายละเอียดของแต่ละจุดเชื่อมต่ออีกด้วย ทำให้เกิดประสิทธิภาพทำงาน ด้านช่วยลดระยะเวลาในการทำงาน ลดภาระการทำงานซ้ำซ้อน ทำให้สามารถนำเวลาไปใช้ในการพัฒนาระบบอื่นต่อไปได้ โดยแสดงรายละเอียดของระบบในบทถัดไป

บทที่ 2
ทฤษฎีและหลักการที่เกี่ยวข้อง

เนื้อหาในบทที่ 2 นี้กล่าวถึงทฤษฎีต่างๆ ที่ได้นำมาใช้ในการทำระบบงาน โดยรวบรวมจากกระบวนงานที่สำนักทะเบียนและประมวลผลได้ทำในปัจจุบัน ผนวกรวมกับสิ่งที่ได้ศึกษาค้นคว้ามา เพื่อนำไปวิเคราะห์และออกแบบต่อในบทถัดไป สามารถแบ่งหัวข้อได้ดังนี้

2.1 ประวัติความเป็นมาของการเชื่อมต่อข้อมูล

อินเทอร์เน็ต (Internet) เกิดขึ้นในปี ค.ศ. 1969 (พ.ศ. 2512) เกิดขึ้นมาจากเครือข่าย ARPANET (Advanced Research Projects Agency NETwork) ซึ่งเป็นเครือข่ายคอมพิวเตอร์จากโครงการวิจัยขั้นสูงของกระทรวงกลาโหม ประเทศสหรัฐอเมริกา โดยมีวัตถุประสงค์หลักคือเพื่อให้คอมพิวเตอร์สามารถเชื่อมต่อกัน และมีปฏิสัมพันธ์กันได้ ต่อมาได้พัฒนาในสาธารณะใช้กลายเป็น เครือข่ายอินเทอร์เน็ตในปัจจุบัน โดยแบ่งประเภทเป็นแบบ Server-based และ P2P-network ดังรูปที่ 2.1



รูปที่ 2.1 แผนภาพการเชื่อมต่อแบบ Server-based (ซ้าย) และ P2P-network (ขวา)

อินเทอร์เน็ตในปัจจุบันพัฒนาอย่างมากแบบก้าวกระโดด ซึ่งใช้อยู่ในชีวิตประจำวันของเรา เช่น ส่งไปรษณีย์อิเล็กทรอนิกส์ หรือ อีเมล (E-Mail), สืบค้นข้อมูล, สื่อสารระหว่างบุคคลแบบตัวอักษร (Chat), สื่อสารระหว่างบุคคลแบบเห็นภาพและเสียง (Video Conference), ซื้อสินค้าออนไลน์, เล่นเกมออนไลน์ เป็นต้น

เมื่อมีอินเทอร์เน็ตแล้ว ทำให้เกิดการเชื่อมต่อข้อมูลทั้งภายในองค์กร และระหว่างองค์กรผ่านโพรโทคอล (Protocol) มากมาย อาทิเช่น

- HTTP (Hypertext Transfer Protocol) : ใช้ในเว็บไซต์ (Website) ที่แสดงอยู่บนเบราว์เซอร์ (Browser) เป็นที่นิยมใช้งานมากที่สุดที่ปัจจุบัน เนื่องจากผู้ใช้งานทั่วไปใช้งานเข้าถึงง่าย
- FTP (File Transfer Protocol) : ใช้ในการรับ-ส่งไฟล์ ระหว่างเซิร์ฟเวอร์ (Server) และไคลเอ็นต์ (Client)
- SMTP (Simple Mail Transfer Protocol) : ใช้สำหรับการส่งอีเมลไปยัง Mail Server ส่วนสำหรับการเปิดอ่านอีเมลของฝั่งผู้ใช้งานใช้โพรโทคอล IMAP หรือ POP3

2.2 ประเภทของ Web API

จากการรวบรวมข้อมูลจาก Stoplight (2565) ได้จำแนกประเภทวัตถุประสงค์การใช้งานของ Web API แบ่งได้เป็นดังนี้

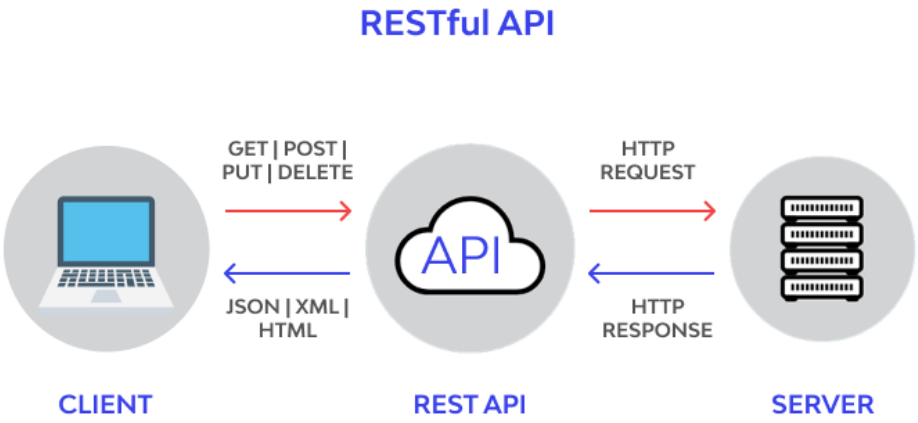
- Open API : รู้จักกันในชื่อ Public API ซึ่งสามารถให้ผู้พัฒนาและผู้ใช้งานภายนอก เข้าถึงได้โดยมีข้อจำกัดสิทธิ์ในการเข้าถึงต่ำ เช่น ข้อมูลสถิติของประชากรในโลก เป็นต้น
- Internal API : ออกแบบมาเพื่อให้ใช้งานระหว่างภายในองค์กร โดยจำกัดสิทธิ์การเข้าถึงของบุคคลภายนอก โดยมีมาตรฐานการเข้าถึงเป็นของตัวเองขึ้นอยู่กับการออกแบบของแต่ละองค์กร
- Partner API : ในทางปฏิบัติเหมือน Open API แต่มีลักษณะเฉพาะในการจำกัดสิทธิ์การเข้าถึง โดยส่วนใหญ่ใช้ API Gateway ในการจัดการ เช่น ค่าธรรมเนียมในการเข้าถึงข้อมูลตามที่ตกลงกัน เป็นต้น
- Composite API : เป็นการแบ่งย่อย/กระจายหลายจุดการเชื่อมต่อ เพื่อตอบสนองการออกแบบสถาปัตยกรรมของระบบแบบแบ่งระบบออกเป็นส่วนย่อยๆ (Microservice) เพื่อลดโหลดของ และง่ายต่อการแบ่งการทดสอบออกเป็นส่วนๆ (Module Testing)

ประเภทสถาปัตยกรรมของ API แบ่งได้เป็นดังนี้

- REST : ย่อมาจาก REpresentational State Transfer เป็นที่นิยมใช้งานมากที่สุดในสถาปัตยกรรมแบบ Web API โดยต้องออกแบบตามหลักการดังต่อไปนี้
    1. เป็นสถาปัตยกรรมแบบไคลเอ็นต์ - เซิร์ฟเวอร์ (Client - Server)
    2. Statelessness คือส่งคำขอและรับคำตอบกลับจากเซิร์ฟเวอร์แล้วปิดการเชื่อมต่อ
    3. Cacheability คือสามารถแคช (Cache) ได้ และต้องสามารถกำหนดได้ว่าจะแคชหรือไม่ เพื่อป้องกันไม่ให้ไคลเอ็นต์ได้รับข้อมูลเก่า
    4. Layered System คือเชื่อมต่อโดยตรงกับเซิร์ฟเวอร์ปลายทาง หรือไปยังตัวกลางอื่นๆระหว่างทาง ตัวกลางควรมีความสามารถในการขยายระบบได้

- JSON-RPC และ XML-RPC : RPC (Remote Procedure Call) คือขั้นตอนของโพรโทคอลในการเรียกใช้งาน โดยการเข้ารหัสตัวอักษร (Encode) แบบ JSON และ XML เพื่อใช้ในการสื่อสาร ซึ่งมีคุณลักษณะหลายรูปแบบ จึงต้องการสถาปัตยกรรมแบบ REST เพื่อให้มีมาตรฐานในการออกแบบ
- SOAP : ย่อมาจาก Simple Object Access Protocol เป็นการสื่อสารในรูปแบบ XML ที่เป็นโครงสร้างแบบ Object เป็นนิยมในช่วงเวลาหนึ่ง แต่ด้วยการออกแบบที่มีข้อจำกัดสูง REST จึงมีบทบาทเข้ามามากกว่า

Web Service ที่ใช้สถาปัตยกรรม REST เป็นที่รู้จักในชื่อ RESTful Web service โดยผู้ใช้งานส่งคำขอ (Request) ไปยัง URI ที่กำหนด จากนั้นเซิร์ฟเวอร์ส่งผลตอบรับ (Response) กลับเป็นมาเป็น Payload ในรูปแบบ HTML, XML, JSON หรือรูปแบบอื่นๆ ดังรูปที่ 2.2



รูปที่ 2.2 แผนภาพการเชื่อมต่อรับ-ส่งข้อมูลของ RESTful API

ระบบงานปัจจุบันใช้ RESTful API ในการพัฒนาระบบ ซึ่งเอกสาร API ที่จัดทำอยู่ในรูปแบบที่สนับสนุน RESTful API

2.3 มาตรฐานของ HTTP ที่ใช้ใน Web API

HTTP/1.1 ใช้มาตรฐาน RFC 2616

โดยแบ่งย่อยมาตรฐานที่เกี่ยวข้องกับ Web API ได้ดังต่อไปนี้

2.3.1 HTTP Authentication with JWT

HTTP/1.1 Authentication ใช้มาตรฐาน RFC 7235

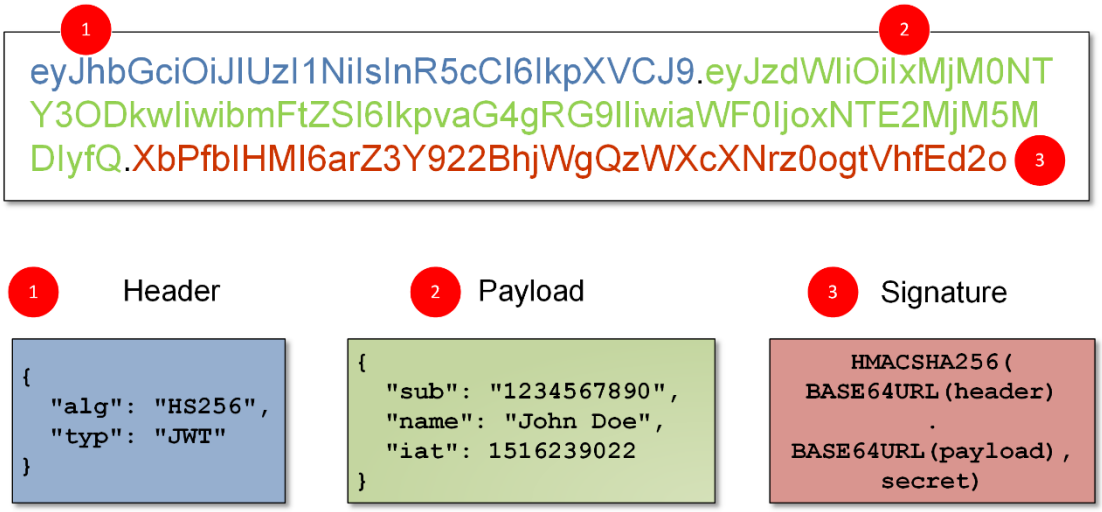
HTTP มีรูปแบบการตรวจสอบสิทธิ์หลายรูปแบบ สำหรับ Web API การตรวจสอบสิทธิ์ที่เป็นที่นิยมคือ JWT (JSON Web Tokens) ใช้มาตรฐาน RFC 7519

โทเคน (Token) ที่เซิร์ฟเวอร์ส่งให้ไคลเอ็นต์ใช้งาน เป็นภาษาที่ถูกเข้ารหัสโดยประกอบไปด้วย Header, Payload และ Signature โดยอัลกอริทึมที่นิยมใช้งานเข้ารหัสคือ HS256

จากนั้นเมื่อไคลเอ็นต์ต้องการเข้าถึงข้อมูลต้องส่งโทเค็นนี้มาใน HTTP Header

```
Authentication: Bearer {Token}
```

เพื่อยืนยันการตรวจสอบสิทธิ์ของไคลเอ็นต์ ตามแผนภาพโครงสร้างดังรูปที่ 2.3



รูปที่ 2.3 แผนภาพโครงสร้างการเข้ารหัสของ JWT

2.3.2 HTTP/1.1 Request Messages

จากการรวบรวมข้อมูลจาก Lokesh Gupta (2564) และ Iterable (2565) ข้อความคำขอเป็นข้อความที่ส่งจากไคลเอ็นต์ไปยังเซิร์ฟเวอร์เป้าหมาย

Syntax คำขอประกอบไปด้วย Request Method, เว้นวรรค, URL เป้าหมาย, เว้นวรรค และปิดท้ายด้วยเวอร์ชันของโพรโทคอล ตัวอย่างเช่น

```
GET /images/logo.png HTTP/1.1
```

เมธอดของคำขอ (Request Methods) หรือเรียกว่า กริยา (Verbs) เป็นการระบุความต้องการใช้งานข้อมูล โดยเมธอดที่นิยมใช้ใน Web API มีดังต่อไปนี้

- GET : ใช้สำหรับการขอข้อมูลในระบบเป้าหมาย โดยไม่มีการเปลี่ยนแปลงข้อมูล ตัวอย่างเช่น

```
GET api/users?size=20&page=5
```

- POST : ใช้สำหรับการสร้างข้อมูลใหม่ในระบบเป้าหมาย ตัวอย่างเช่น

```
POST api/users/123/accounts
```

- PUT : ใช้สำหรับอัปเดต/แก้ไขข้อมูลที่มีอยู่แล้วในระบบเป้าหมาย ตัวอย่างเช่น

```
PUT api/users/123/accounts/456
```

- DELETE : ใช้สำหรับลบ/เลิกใช้ข้อมูลที่มีอยู่ในระบบเป้าหมาย ตัวอย่างเช่น

```
DELETE api/users/123/accounts/456
```

- OPTION : ใช้สำหรับสถานการณ์ที่ต้องการข้อกำหนดร่วมกันในการสื่อสาร ตัวอย่างเช่น CORS (Cross-Origin Resource Sharing) คือการใช้ข้อมูลข้ามโดเมนกัน โดยทำการ pre-filght เมธอด OPTION ไปยังเซิร์ฟเวอร์เป้าหมาย ว่ายอมรับการใช้ข้อมูลข้ามโดเมนนี้ได้หรือไม่ เป็นต้น

ข้อมูลของคำขอ (Request Body) ใช้เป็นที่บรรจุข้อมูล (Payload) ที่ต้องการส่งไปยังเครื่องข่ายเป้าหมาย มีหลายรูปแบบ ตามแผนภาพดังรูปที่ 2.4 แต่ยกตัวอย่างที่นิยมใช้ใน Web API มีดังต่อไปนี้

- URL-path : เป็นข้อมูลที่บรรจุอยู่ใน URL ตัวอย่างเช่น

```
GET /api/resource/v1/v2
```

- Query string : หรือเรียกว่า Query params เป็นข้อมูลที่บรรจุอยู่ข้างหลังต่อจาก URL ตัวอย่างเช่น

```
GET /api/resource?p1=v1&p2=v2
```

- Form data : เป็นข้อมูลที่บรรจุอยู่ในฟอร์ม ซึ่งเป็นที่นิยมใช้งานในการออกแบบเว็บไซต์ด้วยภาษา PHP สามารถส่งไฟล์ได้

- JSON payload : เป็นข้อมูลที่บรรจุอยู่ใน Payload เป็นที่นิยมในข้อมูลที่มีลักษณะซับซ้อน ตัวอย่างเช่น

```
{
    "p1":"v1",
    "p2":"v2"
}
```

ทั้งนี้เมธอด GET และ DELETE บรรจุข้อมูลประเภท URL-path หรือ Query string ส่วนเมธอด POST และ PUT บรรจุข้อมูลประเภท Form data หรือ JSON payload

2.3.3 HTTP/1.1 Response Messages

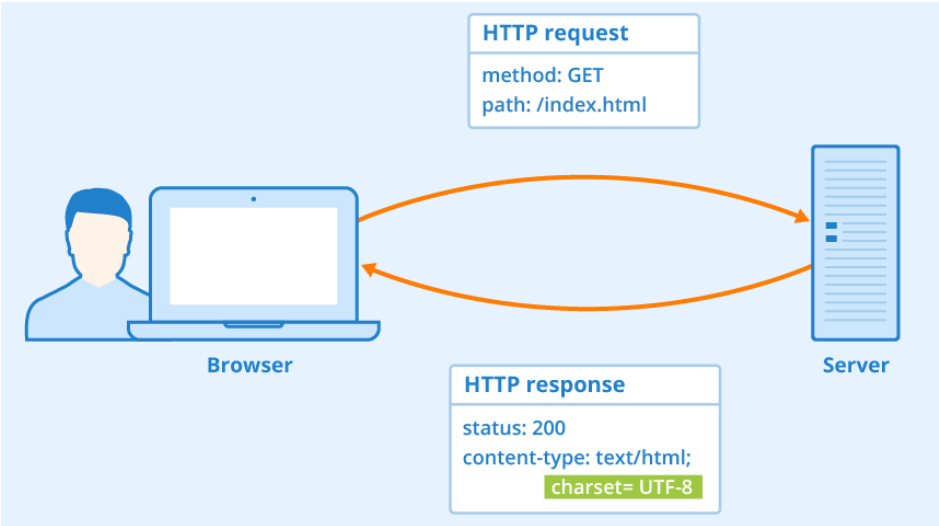
จากการรวบรวมข้อมูลจาก restfulapi.net (2564) และ Iterable (2565) ข้อความคำตอบกลับเป็นข้อความที่ได้รับจากเซิร์ฟเวอร์ไปยังไคลเอ็นต์ที่ส่งคำขอมา

Syntax เริ่มต้นของคำตอบกลับประกอบไปด้วย เวอร์ชันของโพรโทคอล, เว้นวรรค, รหัสสถานะ, เว้นวรรค และปิดท้ายด้วยข้อความของสถานะ (อาจเป็นค่าว่าง) ตัวอย่างเช่น

```
HTTP/1.1 200 OK
```

ฟิลด์ส่วนหัวของคำตอบกลับ (Response Header Fields) เป็นข้อมูลเพิ่มเติมที่เซิร์ฟเวอร์ส่งผ่านมายังไคลเอ็นต์ได้ (อาจเป็นค่าว่างหรือมากกว่าหนึ่ง) ประกอบไปด้วย ชื่อฟิลด์, โคลอน, เว้นวรรค และค่าของฟิลด์นั้น ตามแผนภาพดังรูปที่ 2.4 ตัวอย่างเช่น

```
Content-Type: application/json
```



รูปที่ 2.4 แผนภาพตัวอย่าง Header ของการเชื่อมต่อของ HTTP

รหัสสถานะของคำตอบกลับ (Response Status Codes) สำหรับแจ้งให้ไคลเอ็นต์ทราบว่าส่งผลให้ไคลเอ็นต์ทำกิจกรรมอย่างใดต่อไป โดยรหัสสถานะที่นิยมใช้ใน Web API มีดังต่อไปนี้

- 200 (OK) : สถานะปกติ
- 400 (Bad Request) : คำขอไม่ถูกต้องตามข้อกำหนด
- 401 (Unauthorized) : ตรวจสอบสิทธิ์ไม่ผ่าน
- 403 (Forbidden) : ไม่มีสิทธิ์เข้าถึงข้อมูล
- 404 (Not Found) : ไม่พบข้อมูล
- 405 (Method Not Allowed) : ไม่มีสิทธิ์ใช้งานเมธอดนี้
- 409 (Conflict) : ข้อมูลมีความขัดแย้ง เช่น มีอีเมลผู้ใช้งานนี้อยู่แล้ว เป็นต้น
- 410 (Gone) : ตัวอย่างการใช้งานเช่น JWT ที่ใช้งานหมดอายุ เป็นต้น
- 429 (Too Many Requests) : มีการส่งคำขอมากเกินที่กำหนดไว้
- 503 (Service Unavailable) : จุดเชื่อมต่อนี้ไม่สามารถใช้งานได้

ตารางที่ 2.1 สรุปการใช้งานเมธอดของ HTTP

HTTP Method	**CRUD**	**Collection Resource (e.g. /users)**	**Single Resouce (e.g. /users/123)**
GET	Read	200 (OK), list of users. Use pagination, sorting, and filtering to navigate big lists	200 (OK), single user. 404 (Not Found), if ID not found or invalid
POST	Create	201 (Created), ‘Location’ header with link to /users/{id} containing new ID	Avoid using POST on a single resource
PUT	Update/Replace	405 (Method not allowed), unless you want to update every resource in the entire collection of resource	200 (OK) or 204 (No Content). Use 404 (Not Found), if ID is not found or invalid
DELETE	Delete	405 (Method not allowed), unless you want to delete the whole collection — use with caution	200 (OK). 404 (Not Found), if ID not found or invalid

2.4 การออกแบบ RESTful Web API

จากการรวบรวมข้อมูลการออกแบบ RESTful Web API ของ Microsoft (2565), European Commission (2562) และ OECD (2562) เว็บแอปพลิเคชันสมัยใหม่ส่วนใหญ่ใช้ Web API สำหรับให้ผู้ใช้งานมีปฏิสัมพันธ์กับแอปพลิเคชัน การออกแบบ Web API ที่ดีควรมุ่งเป้าเพื่อสนับสนุนดังต่อไปนี้

- ความเป็นอิสระของแพลตฟอร์ม (Platform Independence) คือผู้ใช้งานสามารถเรียกใช้งาน API ได้ ไม่ว่า API ถูกนำไปใช้งานอย่างไร โดยใช้โพรโทคอลมาตรฐาน ที่มีกลไกที่ผู้ใช้งานและ Web Service ใช้รูปแบบของข้อมูลที่ใช้แลกเปลี่ยนกัน
- วิวัฒนาการของบริการ (Service Evolution) คือ Web API ควรสามารถพัฒนาและเพิ่มฟังก์ชันการทำงาน ได้เป็นอิสระจากแอปพลิเคชัน เมื่อ Web API พัฒนาเพิ่มเติม ฟังก์ชันทั้งหมดควรทำงานต่อไปได้โดยไม่มีการแก้ไข

2.4.1 การจัดรูปแบบการออกแบบ API ให้สอดคล้องกับข้อมูล (Organizing the API design around resources)

URI ควรยึดตามคำนาม ไม่ใช่กริยา ตัวอย่างเช่น

```
GET api/orders # Good

GET api/create-order # Avoid
```

แนวทางปฏิบัติในการตั้งชื่อ URI โดยทั่วไปให้ใช้คำนามพหูพจน์ พร้อมลำดับชั้นการเรียกใช้ ตัวอย่างเช่น /customers คือเส้นทางไปยังข้อมูลลูกค้า; /customers/5 คืออ้างอิงถึงข้อมูลลูกค้าหมายเลข 5; customers/5/orders คือเส้นทางไปยังข้อมูลออเดอร์ของลูกค้าหมายเลข 5 เป็นต้น

หลีกเลี่ยงการใช้คำที่อ้างอิงถึงแหล่งข้อมูลพื้นฐาน เช่น ชื่อฟิลด์ในฐานข้อมูล เป็นต้น หากจำเป็นแนะนำให้มีเลเยอร์ขั้นกลางระหว่างฐานข้อมูลและ Web API ด้วย

2.4.2 การกำหนดการทำงานของ API ในรูปแบบเมธอดของ HTTP (Defining API operations in terms of HTTP methods)

เมธอด HTTP ที่ใช้ใน Web API เป็นไปตามที่กล่าวไว้ในหัวข้อ 2.3.2 ส่วนการนำเมธอดมาประยุกต์ใช้งานขึ้นอยู่กับลักษณะและสถานการณ์ของงานนั้นๆ

2.4.3 การออกแบบให้สอดคล้องกับข้อกำหนดของ HTTP (Conforming to HTTP semantics)

ส่วนนี้อธิบายถึงข้อควรพิจารณาบางประการสำหรับการออกแบบ API ที่สอดคล้องกับข้อกำหนดของ HTTP โดยไม่ได้ครอบคลุมทุกรายละเอียด แต่สามารถศึกษากำหนดของ HTTP เพิ่มเติมได้

ในโพรโทคอล HTTP รูปแบบข้อมูลถูกกำหนดตามการใช้งานประเภทของสื่อ หรือเรียกว่า MIME type สำหรับข้อมูลที่ไม่ใช่ประเภทไบนารี (Non-binary) Web API ส่วนใหญ่ใช้ JSON (MIME type = application/json) หรืออาจเป็น XML (MIME type = application/xml)

การดำเนินการแบบอะซิงโครนัส (Asynchronous operations) บางครั้งการดำเนินการตามคำขออาจใช้เวลาประมวลผล หากรอจนประมวลผลเสร็จสิ้นก่อนที่ส่งตอบกลับไปยังไคลเอ็นต์อาจทำให้เวลาตอบสนองต่อผู้ใช้งานไม่สามารถยอมรับได้ ให้พิจารณา ส่งคำตอบกลับรหัสสถานะเป็น 202 (Accepted) เพื่อระบุว่าคำขอได้รับแล้วแต่ยังไม่เสร็จสมบูรณ์ จากนั้นเปิดจุดเชื่อมต่อเพื่อให้สามารถตรวจสอบสถานะปัจจุบันของคำขอ หรืออาจรวมประมาณเวลาที่เสร็จสิ้น หรืออาจเพิ่มลิงก์เพื่อยกเลิกการดำเนินการ

2.4.4 การกรองและแบ่งหน้าของข้อมูล (Filtering and pagination data)

การเปิดจุดเชื่อมต่อข้อมูลผ่าน URI เดียว อาจทำให้แอปพลิเคชันดึงข้อมูลจำนวนมากเกินความจำเป็น หากต้องการแค่ข้อมูลบางส่วน ทำให้กระบวนการประมวลผลไม่มีประสิทธิภาพ จากการใช้แบนด์วิดธ์ของเครือข่าย และกำลังประมวลผลของเซิร์ฟเวอร์

โดย Web API อนุญาตให้สามารถส่งตัวกรอง ที่เป็นการจำกัดจำนวน (Limit) และออฟเซ็ต (Offset) ของข้อมูลได้ภายในคำขอ ตัวอย่างเช่น

```
GET api/orders?limit=25&offset=50
```

จากตัวอย่างนี้ เป็นการขอข้อมูลจำนวน 25 ออเดอร์ โดยเริ่มต้นออเดอร์ที่ 50

2.4.5 การรองรับการแบ่งการตอบสนองออกเป็นส่วนๆสำหรับข้อมูลไบนารีขนาดใหญ่ (Supporting partial responses for large binary resources)

ข้อมูลอาจมีไบนารีขนาดใหญ่ เช่น ไฟล์, รูปภาพ เป็นต้น เพื่อแก้ปัญหาที่เกิดจากการเชื่อมต่อที่ไม่เสถียรและไม่ต่อเนื่อง ให้พิจารณาเปิดใช้งานดึงข้อมูลดังกล่าวเป็นส่วนๆ เช่นนี้ Web API ควรสนับสนุนฟิลด์ส่วนหัวชื่อ Accept-Range สำหรับคำขอ

2.4.6 การใช้ HATEOAS เพื่อใช้นำทางไปยังข้อมูลที่เกี่ยวข้อง (Using HATEOAS to enable navigation to related resources)

HATEOAS (Hypermedia As The Engine Of Application State) เป็นการนำข้อมูลมาสร้างไฮเปอร์ลิงก์ภายใน Web API เพื่อให้รูปแบบลิงก์บนเว็บไซต์มีความเป็นอิสระ และไม่ต้องกำหนดตายตัวบนเว็บไซต์ที่ยากต่อการแก้ไข

2.4.7 การกำหนดเวอร์ชันของ RESTful Web API (Versioning a RESTful Web API)

แทบเป็นไปไม่ได้ที่ Web API ไม่ได้รับการแก้ไข เนื่องด้วยความเปลี่ยนแปลงจากความต้องการทางนโยบายขององค์กร การกำหนดเวอร์ชันช่วยให้ผู้ใช้งานสามารถเลือกเวอร์ชันที่ใช้งาน

รวมถึงยังคงใช้เวอร์ชันเดิมอยู่ได้ ในขณะที่มีการพัฒนาเวอร์ชัน Web API ใหม่แล้ว โดยแต่ละแบบมีข้อดี-ข้อเสียแตกต่างกันออกไป ดังนี้

- ไม่มีการทำเวอร์ชัน : เป็นวิธีการที่เรียบง่ายที่สุด คือการเพิ่มข้อมูลเพิ่มเติม โดยไม่แก้ไขข้อมูลที่มีอยู่เดิม ทำให้ระบบเดิมยังสามารถทำงานอยู่ได้ แต่เมื่อเพิ่มข้อมูลมากขึ้นเรื่อยๆ ทำให้เกิดการส่งข้อมูลมากเกินความจำเป็น อาจส่งผลต่อประสิทธิภาพของระบบได้
- การทำเวอร์ชันบน URI : เป็นวิธีการทำเวอร์ชันในระดับลิงก์ ซึ่ง Web API เป็นตัวจัดการเส้นทาง ตัวอย่างเช่น

  เวอร์ชันแรก

```
GET api/v1/customers/3
```

  ส่วนเวอร์ชันที่สอง

```
GET api/v2/customers/3
```

- การทำเวอร์ชันบน Query String : วิธีการนี้คือกำหนดเวอร์ชันบน Query String ตัวอย่างเช่น

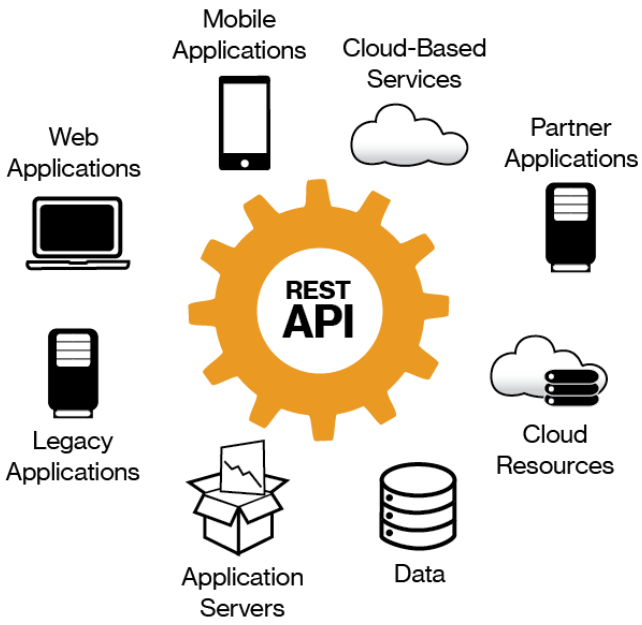
```
GET api/customers/3?version=2
```

- การทำเวอร์ชันบนส่วนหัวขอคำขอ : วิธีการนี้คือกำหนดเวอร์ชันบนส่วนหัวของคำขอ ตัวอย่างเช่น

```
GET api/customers/3 HTTP/1.1
Custom-Header: api-version=2
```

2.4.8 การริเริ่มในการทำ Open API (Open API Initiative)

การทำ Open API เป็นการสร้างมาตรฐานให้กับองค์กร ที่มีข้อดีสำหรับการทำงานร่วมกันระหว่างหน่วยงาน แต่ต้องใช้ความระมัดระวังในการออกแบบ โดยให้สอดคล้องกับข้อกำหนดขององค์กร มีวัตถุประสงค์ของการใช้งานดังรูปแผนภาพตัวอย่างที่ 2.5



รูปที่ 2.5 แผนภาพตัวอย่างภาพรวมวัตถุประสงค์ของการใช้งาน RESTful API

2.5 การพัฒนา RESTful Web API

จากการรวบรวมข้อมูลการพัฒนา RESTful Web API ของ Microsoft (2565), European Commission (2562) และ OECD (2562) กล่าวว่า Web API ที่ได้รับการออกแบบมาอย่างดี สามารถกำหนดทรัพยากร ความสัมพันธ์ และรูปแบบการดำเนินการของแอปพลิเคชันที่ผู้ใช้งานสามารถเข้าถึงได้ เมื่อมีการปรับใช้ Web API ควรพิจารณาข้อกำหนดทางกายภาพของเซิร์ฟเวอร์ และวิธีการสร้าง Web API มากกว่าโครงสร้างเชิงตรรกะของข้อมูล คำแนะนำนี้เน้นไปที่แนวปฏิบัติที่ดีที่สุดสำหรับการนำ Web API ไปใช้งานและเผยแพร่

2.5.1 การประมวลผลคำขอ (Processing requests)

การประมวลผลคำขอที่ได้รับมา ให้พิจารณาประเด็นดังต่อไปนี้

- ใช้เมธอดใดๆ ซ้ำๆ ในกิจกรรมเดิม ควรได้ผลลัพธ์แบบเดิม
- สร้างข้อมูลใหม่ หากสร้างข้อมูลเดิมซ้ำ ควรไม่มีการเปลี่ยนแปลงข้อมูลที่มีอยู่เดิม เช่น การสมัครสมาชิกของผู้ใช้งาน เป็นต้น
- หลีกเลี่ยงการพัฒนากระบวนการที่ใช้เมธอด POST, PUT และ DELETE มากเกินความจำเป็น
- ปฏิบัติตามข้อกำหนดของ HTTP ให้เป็นเป็นตามมาตรฐาน เมื่อมีการส่งคำตอบกลับ
- สนับสนุนการนำทางและค้นหาข้อมูลในรูปแบบ HATEOAS

2.5.2 การจัดการข้อยกเว้น (Handling exceptions)

การดำเนินการที่ส่งผลถึงข้อยกเว้นที่ไม่ตรวจจับไม่ได้ ให้พิจารณาประเด็นดังต่อไปนี้

- ตรวจจับข้อยกเว้น จากนั้นส่งรหัสสถานะประเภทข้อยกเว้น กลับไปยังผู้ส่งคำขอ

- จัดการข้อยกเว้นอย่างสม่ำเสมอ และบันทึกประวัติข้อมูลเกี่ยวกับข้อผิดพลาด
- แยกแยะระหว่างข้อผิดพลาดฝั่งไคลเอ็นต์และข้อผิดพลาดฝั่งเซิร์ฟเวอร์

2.5.3 การเพิ่มประสิทธิภาพการเข้าถึงข้อมูลของฝั่งไคลเอ็นต์ (Optimizing client-side data access)

สภาพแวดล้อมแบบแยกส่วนการทำงานระหว่างไคลเอ็นต์และเซิร์ฟเวอร์ หนึ่งในข้อกังวลคือระบบมีโอกาสเกิดคอขวดได้มาก โดยเฉพาะแอปพลิเคชันที่ไคลเอ็นต์ส่งคำขอและรับข้อมูลบ่อยครั้ง ดังนั้นควรมีเป้าหมายที่ลดปริมาณการรับส่งข้อมูลที่ไหลผ่านเครือข่ายให้น้อยที่สุด ให้พิจารณาประเด็นดังต่อไปนี้

- สนับสนุนการทำแคชที่ฝั่งไคลเอ็นต์ (Client-Side Caching)
- ใช้ ETag (Entity Tag) เพื่อเพิ่มประสิทธิภาพการประมวลผล เมื่อไคลเอ็นต์ส่ง ETag เดิม เซิร์ฟเวอร์ควรใช้ข้อมูลที่อยู่ในแคชตาม ETag นั้นนำมาประมวลผล

2.5.4 การจัดการคำขอและคำตอบกลับขนาดใหญ่ (Handling large requests and responses)

อาจมีบางครั้งที่แอปพลิเคชันจำเป็นต้องส่งคำขอหรือรับข้อมูลที่มีขนาดใหญ่หลายเมกะไบต์ หรือมากกว่า การรอในขณะที่ส่งข้อมูลอาจทำให้แอปพลิเคชันไม่ตอบสนอง ให้พิจารณาประเด็นดังต่อไปนี้

- ใช้การกรองและแบ่งหน้าของข้อมูล และการรองรับการแบ่งการตอบสนองออกเป็นส่วนๆ สำหรับข้อมูลไบนารีขนาดใหญ่ ตามบทการออกแบบ Web API
- มีการบีบอัดข้อมูลก่อนส่งคำตอบกลับ

2.5.5 การรักษาการตอบสนอง ความสามารถในการปรับขนาด และความพร้อมใช้งาน (Maintaining responsiveness, scalability, and availability)

Web API อาจถูกใช้โดยแอปพลิเคชันจำนวนมากที่ทำงานอยู่ สิ่งสำคัญคือต้องตรวจสอบให้แน่ใจว่า Web API ยังสามารถตอบสนองภายใต้ภาระงานจำนวนมาก เพื่อให้สามารถปรับขนาดกำลังประมวลผลของเซิร์ฟเวอร์ เพื่อรับประกันความพร้อมใช้งานต่อองค์กรที่สำคัญ ให้พิจารณาประเด็นดังต่อไปนี้

- ให้มั่นใจว่าแต่ละคำขอเป็นกระบวนการทำงานแบบ Stateless
- รองรับการถูกโจมตีรูปแบบ DDOS (Distributed Denial-Of-Service) Attack เมื่อมีแนวโน้มที่ถูกโจมตีให้พักหรือปิดกั้นการเข้าถึงชั่วคราวเพื่อนำไปตรวจสอบ แล้วส่งรหัสสถานะ 429 (Too Many Requests) ตอบกลับไป

- เมื่อเซิร์ฟเวอร์ถึงจุดจำกัดของการประมวลผล ให้ส่งรหัสสถานะ 503 (Service Unavailable) ตอบกลับไป จนกว่าพร้อมให้บริการอีกครั้ง เพื่อป้องกันไม่ให้กระบวนการที่กำลังประมวลผลอยู่ได้รับผลกระทบ

2.5.6 การเผยแพร่และจัดการ Web API (Publishing and managing a Web API)

จัดทำเอกสาร Web API เพื่อรวมจุดเชื่อมต่อทั้งหมด โดยกำหนดรายละเอียดของแต่ละจุดเชื่อมต่อ รวมถึงการยืนยันสิทธิ์เพื่อเข้าถึงข้อมูล

2.5.7 การทดสอบ Web API (Testing a Web API)

หลังจากได้รับความต้องการและพัฒนา Web API เพิ่มเติมแล้ว ต้องตรวจสอบว่าสามารถทำงานได้อย่างปกติและถูกต้อง ให้พิจารณาประเด็นดังต่อไปนี้

- ทดสอบทุกเส้นทาง ของทุกเมธอด
- ตรวจสอบเส้นทางที่ต้องการการยืนยันสิทธิ์
- ทดสอบข้อยกเว้น ให้เป็นไปตามข้อกำหนดของ HTTP
- ตรวจสอบคำขอและคำตอบกลับให้อยู่ในรูปแบบที่ดี ตามที่ได้ออกแบบไว้
- ตรวจสอบ HATEOAS ลิงก์ว่าสามารถใช้งานได้
- ตรวจสอบว่าคำตอบกลับส่งรหัสสถานะถูกต้องตามประเภทของสถานะ

2.5.8 การเฝ้าติดตาม Web API (Monitoring a Web API)

สามารถพัฒนาหรือใช้ซอร์ฟแวร์เสริมเพื่อรวบรวมข้อมูลการใช้งาน และวิเคราะห์การรับ-ส่งข้อมูลผ่านบริการการจัดการ Web API โดยรวบรวมข้อมูล ให้พิจารณาประเด็นดังต่อไปนี้

- ระยะเวลาตอบสนองของเซิร์ฟเวอร์
- จำนวนคำขอ และรายละเอียดของแต่ละคำขอ
- คำขอที่ตอบสนองช้าที่สุด โดยนับจากระยะเวลาตอบสนองเฉลี่ย
- รายละเอียดของคำขอที่ล้มเหลว
- จำนวนไคลเอ็นต์ที่แตกต่างกันของเบราว์เซอร์และ User Agents
- การเข้าถึงของผู้ใช้งานที่มีบทบาทแตกต่างกัน

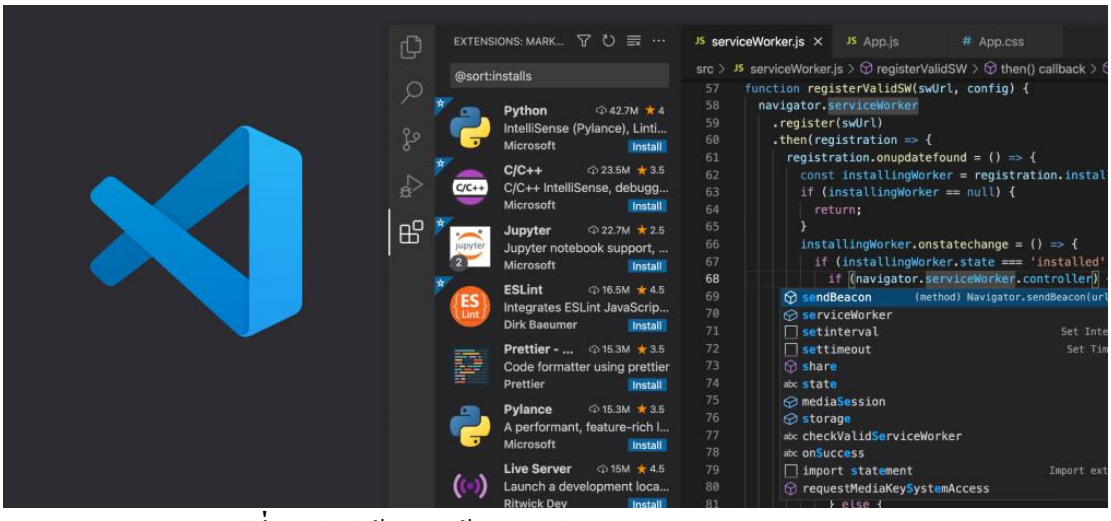
2.6 เครื่องมือที่ใช้ในการพัฒนา

การจัดทำเอกสาร Web API เริ่มต้นด้วยการออกแบบระบบที่ใช้งาน พัฒนาระบบ จากนั้นมีจุดเชื่อมต่อ Web API หลายรายการ โดยเครื่องมือที่เกี่ยวข้องมีดังต่อไปนี้

2.6.1 โปรแกรมสำหรับพัฒนาโค้ด (Code Editor)

สำหรับพัฒนาซอร์สโค้ด (Source Code) จำเป็นต้องใช้โปรแกรมในการพัฒนา ซึ่งมีหลายโปรแกรมด้วยกัน Code Editor ที่นิยมใช้ในปัจจุบันคือ Visual Studio code (VS Code) พัฒนา

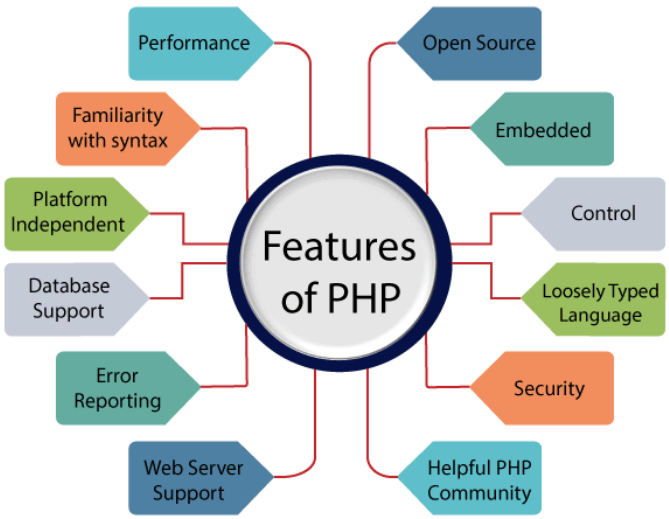
โดยบริษัทไมโครซอฟท์ ตัวอย่างคุณลักษณะเช่น แบ่งสีในแต่ละคำสั่งของแต่ละภาษา, มี Extension มากมายเพื่อรองรับการพัฒนา, มี Debugging เพื่อตรวจสอบความผิดพลาดของซอร์สโค้ด, มีจัดหน้าและรูปแบบของซอร์สโค้ดเพื่อให้อ่านง่ายและเป็นไปตามมาตรฐานเดียวกัน เป็นต้น โลโก้และหน้าตาของโปรแกรมเป็นดังรูปที่ 2.6



รูปที่ 2.6 โลโก้และหน้าตาของโปรแกรม Visual Studio code

2.6.2 ภาษาโปรแกรมมิ่ง (Programming Language)

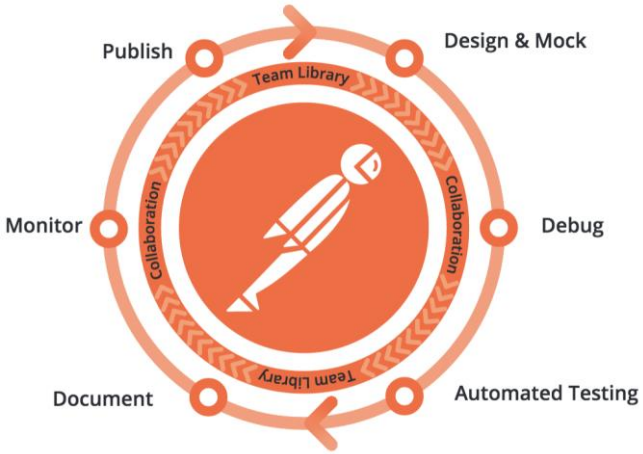
ภาษาที่ใช้ในการพัฒนาโปรแกรม ซึ่งมีหลายภาษาด้วยกัน ภาษาที่ยกมาพิจารณาและมีหลายเว็บแอปพลิเคชันใช้งานอยู่คือ PHP เป็นคำย่อแบบกล่าวซ้ำ จากคำว่า PHP Hypertext Preprocessor สามารถใช้นำการพัฒนา Web API ในฝั่งเซิร์ฟเวอร์ได้ โดยมี Framework หลายตัวที่สามารถใช้พัฒนา Web API ได้ ตัวอย่างคุณสมบัติดังรูปที่ 2.7



รูปที่ 2.7 ตัวอย่างคุณสมบัติของภาษา PHP

2.6.3 เครื่องมือพัฒนาทดสอบ และจัดทำเอกสาร Web API

Postman เป็นโปรแกรมที่เป็นที่นิยมและคุณลักษณะออกแบบมาเพื่อพัฒนา Web API โดยเฉพาะ ตัวอย่างคุณลักษณะเช่น แบ่ง Workspaces สำหรับทำงานได้, จัดหมวดหมู่ของคำขอได้, ที่เซิร์ฟเวอร์จำลองได้, ตรวจสอบ (Monitoring) ได้, เก็บประวัติการเรียกใช้คำขอได้, จัดทำเอกสารจากหมวดหมู่ได้ เป็นต้น โลโก้และตัวอย่างคุณสมบัติเป็นดังรูปที่ 2.8



รูปที่ 2.8 โลโก้และตัวอย่างคุณสมบัติที่เป็น Lifecycle ของ Postman

2.7 รูปแบบของเอกสาร Web API ในปัจจุบัน

ในโลกอินเทอร์เน็ตมีเอกสาร Web API มากมาย โดยยกตัวอย่างเว็บแอปพลิเคชันที่ชื่อว่า Bitfinex เป็นแพลตฟอร์มสำหรับแลกเปลี่ยนสกุลเงินดิจิทัล คริปโตเคอร์เรนซี ที่ทำเอกสาร ไว้อย่างมีประสิทธิภาพ อ่านง่าย และนำไปประยุกต์ใช้งานได้ง่าย คุณสมบัติดังรูปที่ 2.9 และ 2.10

รูปที่ 2.9 คุณสมบัติของเว็บแอปพลิเคชัน Bitfinex

รูปที่ 2.10 คุณสมบัติของเว็บแอปพลิเคชัน Bitfinex (ต่อ)

คู่มือการใช้งาน Web API ของ Bitfinex สามารถเข้าถึงได้ที่ลิงก์ https://docs.bitfinex.com/ โดยเอกสาร Web API ส่วน REST แบ่งเป็น 4 ส่วนหลักคือ

2.7.1 ข้อมูลทั่วไป

ส่วนประกอบมีดังนี้

- URL ของจุดเชื่อมต่อ
- รูปแบบของพารามิเตอร์ของการใช้งานจุดเชื่อมต่อ
- การจำกัดอัตราการส่งคำขอ โดยทาง Bitfinex จำกัดไว้ที่ 10 ถึง 90 คำขอต่อนาที

2.7.2 จุดเชื่อมต่อข้อมูลสาธารณะ

ส่วนประกอบมีดังนี้

- ตัวอย่างการใช้งานด้วยภาษา JavaScript, Ruby และ Go
- รายการจุดเชื่อมต่อข้อมูลสาธารณะทั้งหมด

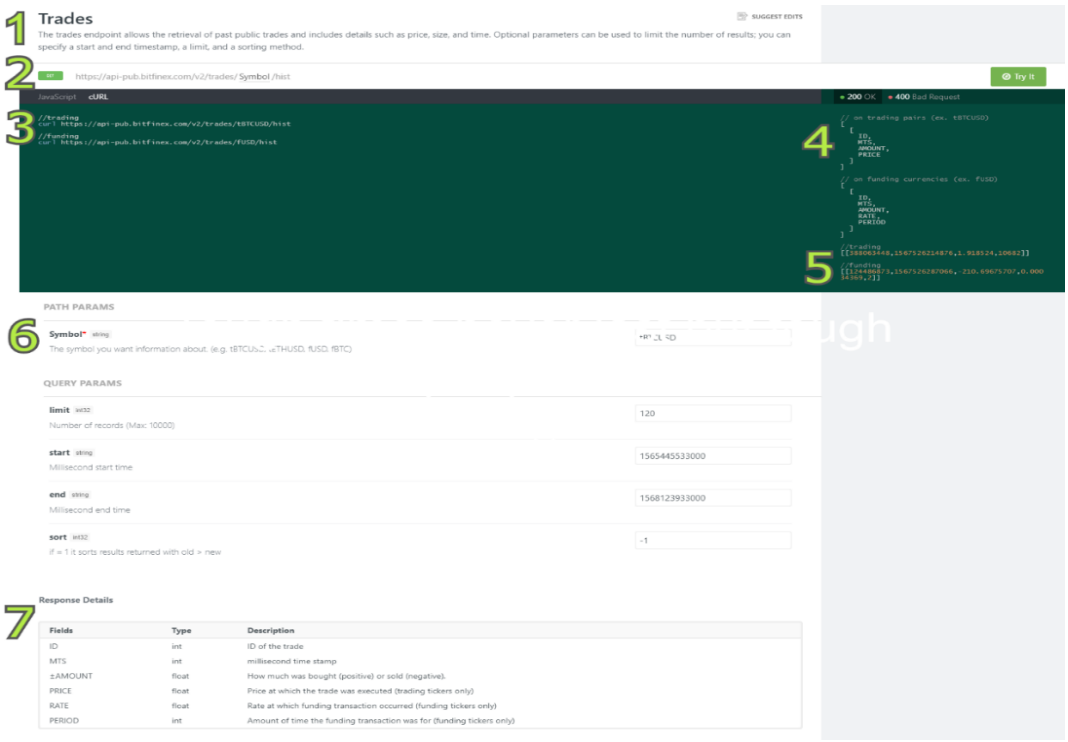
2.7.3 จุดเชื่อมต่อข้อมูลแบบต้องยืนยันสิทธิ์

ส่วนประกอบมีดังนี้

- ตัวอย่างการใช้งานพร้อมการยืนยันสิทธิ์ด้วยภาษา JavaScript
- รายการจุดเชื่อมต่อข้อมูลแบบต้องยืนยันสิทธิ์ทั้งหมด

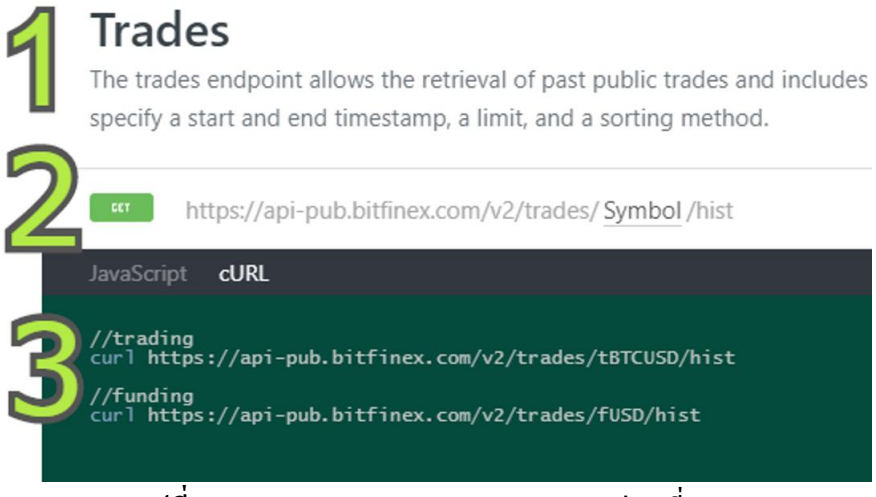
2.7.4 วิธีการอ่านเอกสาร

เอกสารโดยภาพรวม ดังรูปที่ 2.11



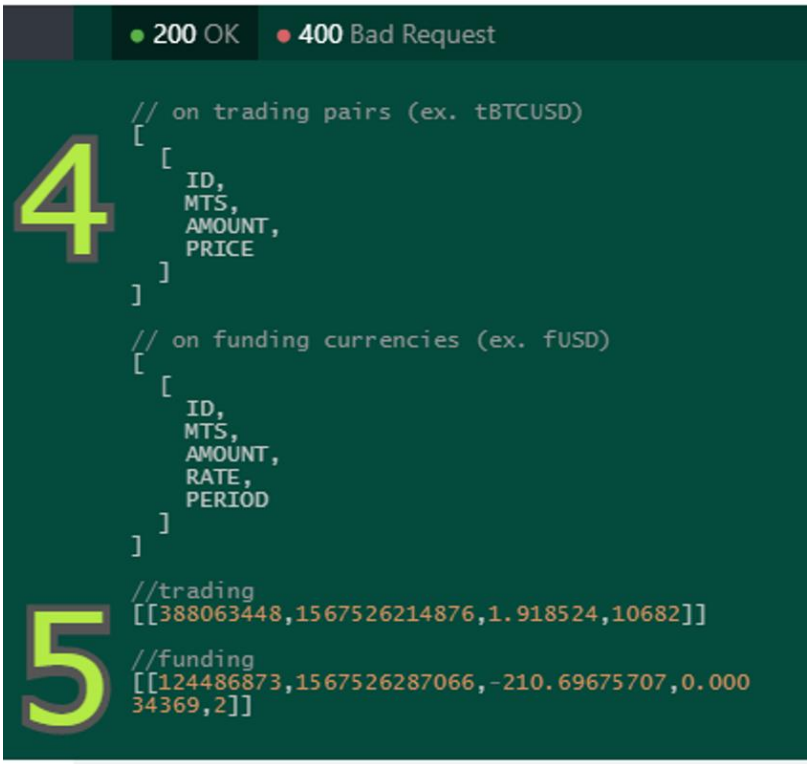
รูปที่ 2.11 เอกสาร Web API ของ Bitfinex โดยรวม

การอธิบายรายละเอียดในแต่ละจุดเชื่อมต่อมีโครงสร้างที่เป็นทำนองเดียวกัน แบ่งเป็น 7 ส่วน ดังต่อไปนี้



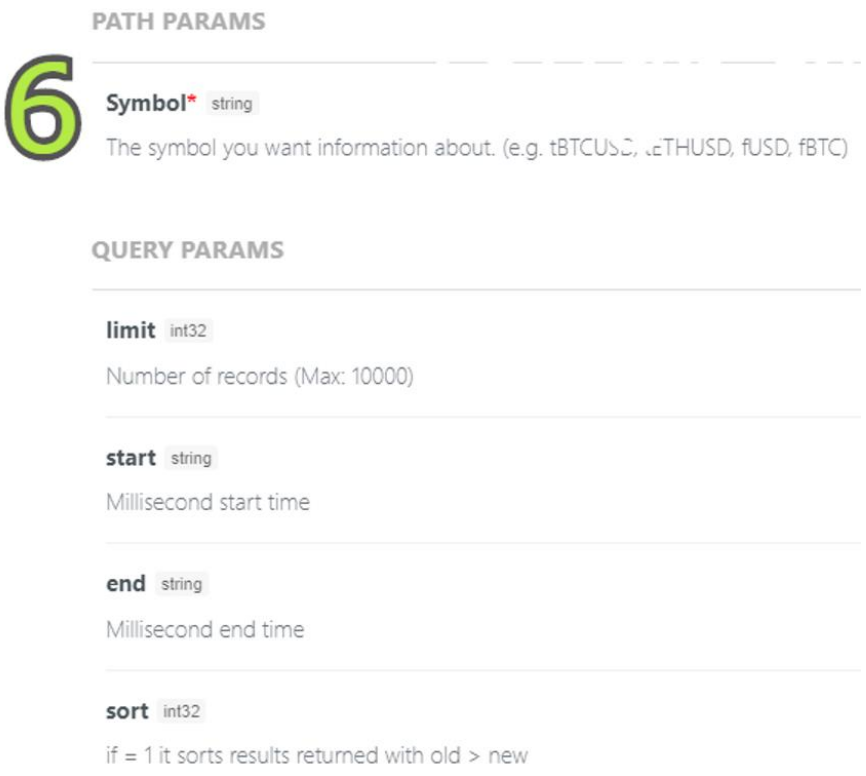
รูปที่ 2.12 เอกสาร Web API ของ Bitfinex ส่วนที่ 1-3

1. หัวข้อและคำอธิบายรายละเอียดของจุดเชื่อมต่อ ดังรูปที่ 2.12
2. ตัวอย่าง URL ที่มีพาธและพารามิเตอร์ประกอบ ดังรูปที่ 2.12
3. ตัวอย่างโค้ดที่ใช้เขียนในภาษา JavaScript และ cURL ดังรูปที่ 2.12



รูปที่ 2.13 เอกสาร Web API ของ Bitfinex ส่วนที่ 4-5

4. ตัวอย่างผลของคำตอบกลับที่มีการจัดรูปแบบตัวอักษร ดังรูปที่ 2.13
5. ตัวอย่างผลของคำตอบกลับ ดังรูปที่ 2.13



รูปที่ 2.14 เอกสาร Web API ของ Bitfinex ส่วนที่ 6

6. พาธ และ Query params ที่ใช้ในคำขอ ดังรูปที่ 2.14

Response Details

Fields	Type	Description
ID	int	ID of the trade
MTS	int	millisecond time stamp
±AMOUNT	float	How much was bought (positive) or sold (negative).
PRICE	float	Price at which the trade was executed (trading tickers only)
RATE	float	Rate at which funding transaction occurred (funding tickers only)
PERIOD	int	Amount of time the funding transaction was for (funding tickers only)

รูปที่ 2.15 เอกสาร Web API ของ Bitfinex ส่วนที่ 7

7. คำอธิบายและประเภทข้อมูลของแต่ละฟิลด์ในคำตอบกลับ ดังรูปที่ 2.15

2.8 รูปแบบวิธีการตรวจสอบสิทธิ์ (Authentication Methods) ที่ใช้ใน Web API

ระบบสำหรับใช้เข้าถึงข้อมูลเพื่อทำกิจกรรมต่างๆเกี่ยวกับข้อมูล จำเป็นต้องมีการตรวจสอบสิทธิ์ เพื่อให้ระบบสามารถจำแนกได้ว่า ผู้ส่งข้อความคำขอนี้ ระบบสามารถจัดหาข้อมูลให้ได้หรือไม่ และสามารถจัดหาข้อมูลให้ได้ระดับใด

Maggie Summers (2563) และ RestCase (2562) ได้จำแนกวิธีการตรวจสอบสิทธิ์ที่ใช้ใน Web API มีรูปแบบทิศทางเดียวกันของทั้ง 2 แหล่งข้อมูลอยู่ 3 รูปแบบคือ

2.8.1 HTTP Authentication Schemes

แบบแผนการตรวจสอบสิทธิ์ของ HTTP (HTTP Authentication Schemes) เช่น Basic, Bearer, Digest, HOBA เป็นต้น โดยที่นิยมใช้สำหรับ Web API คือ

- Basic Authentication ใช้มาตรฐาน RFC 7617 โดยส่ง username:password ที่แปลงเป็น base64 แนบไปยังส่วนหัวของคำขอ ปัจจุบันไม่นิยมใช้ เนื่องจากปัญหาด้านความปลอดภัย

```
Authorization: Basic bG9sOnNlY3VyZQ==
```

- Bearer Authentication ใช้มาตรฐาน RFC 6750 รู้จักกันในชื่อโทเคนเป็นส่วนหนึ่งของการใช้งานเช่น JWT, OAuth เป็นต้น เหมือนกับ Basic Authentication แต่ต่างกันตรงที่ส่งข้อความที่ถูกเข้ารหัสตัวอักษรโดยเซิร์ฟเวอร์ แทน base64 ของ username:password

```
Authorization: Bearer DqQW5ry60eXxp74Ydz2Ob1Ph0LRXdXk7
```

2.8.2 API Keys

API Keys มีรูปแบบวิธีการใช้งานที่หลากหลาย ขึ้นอยู่กับการออกแบบของผู้พัฒนาระบบ โดยสามารถกำหนดอยู่บนส่วนต่างๆ เช่น Header, Query String, Body เป็นต้น

โดยยกตัวอย่างการใช้งานที่นิยมใช้ใน Web API ของ Cryptocurrency Exchange ซึ่งมีความปลอดภัยสูงโดยเข้ารหัสข้อความร่วมกับ Key ที่ออกโดย Platform

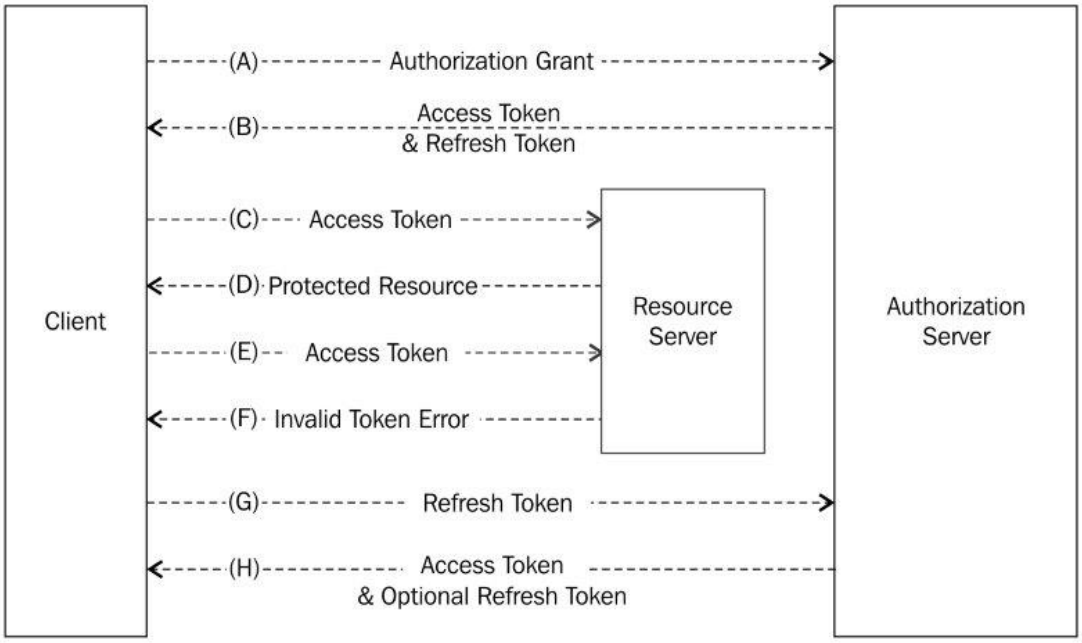
- API Key : เป็นคีย์ที่ถูกออกโดยแพลตฟอร์ม โดยใช้ระบุผู้ใช้งาน
- API Secret Key : เหมือนกับ API Key แต่ต่างกันตรงถูกแสดงเพียงครั้งเดียวในขั้นตอนการออก API Key ถ้าลืมหรือสูญหายต้องออก API Key ใหม่
- Nonce : ส่วนใหญ่ใช้เป็น Timestamp โดย Nonce ที่ส่งไปพร้อมกับคำขอ ต้องมีค่ามากกว่าคำขอก่อนหน้าเสมอ
- Signature : เป็นส่วนสำคัญสำหรับการสร้างสัญญาที่ไม่สามารถโต้แย้งหรือปฏิเสธได้ (Non-repudiation) โดยวิธีการสร้างที่นิยมคือนำ URL Path, คำขอ, Nonce มาเข้ารหัสตัวอักษรร่วมกับ API Secret Key ด้วยฟังก์ชัน HMAC อัลกอริทึม SHA-384 หรือตามรูปแบบที่แพลตฟอร์มกำหนด

จากนั้นนำชุดข้อมูลดังกล่าวส่งคำขอไปยังแพลตฟอร์มเมื่อรูปแบบคำขอเป็นไปตามที่แพลตฟอร์มกำหนด ส่งสถานะคำขอ 200 OK และข้อมูลที่ร้องขอตอบกลับมา

2.8.3 OAuth

OAuth ใช้มาตรฐาน RFC 6749 เป็นมาตรฐานหรือ Framework สำหรับมอบสิทธิ์การเข้าถึง เป็นที่นิยมใช้ในเว็บแอปพลิเคชันมากมาย ดังซิสเต็มซีเควนไดอะแกรมรูปที่ 2.16 โดยโคเทนที่ใช้งานมี 2 รูปแบบคือ

- Access Token : สำหรับใช้งานร่วมกับการส่งคำขอ โดยมีเวลาหมดอายุ
- Refresh Token : สำหรับใช้ขอ Access Token ใหม่ เพื่อขยายเวลาหมดอายุ, ปรับเปลี่ยนขอบเขตการเข้าถึง และเปลี่ยนโครงสร้างของโทเคน



รูปที่ 2.16 ซิสเต็มซีเควนไดอะแกรมการขอยืนยันสิทธิ์ของ OAuth 2.0 Framework

บทที่ 3
การวิเคราะห์และการออกแบบ

เนื้อหาในบทที่ 3 นี้กล่าวถึงการวิเคราะห์และการออกแบบ ที่ได้นำมาใช้ทำระบบงาน ที่ได้จากการศึกษาทฤษฎีและหลักการที่เกี่ยวข้อง สามารถแบ่งหัวข้อได้ดังนี้

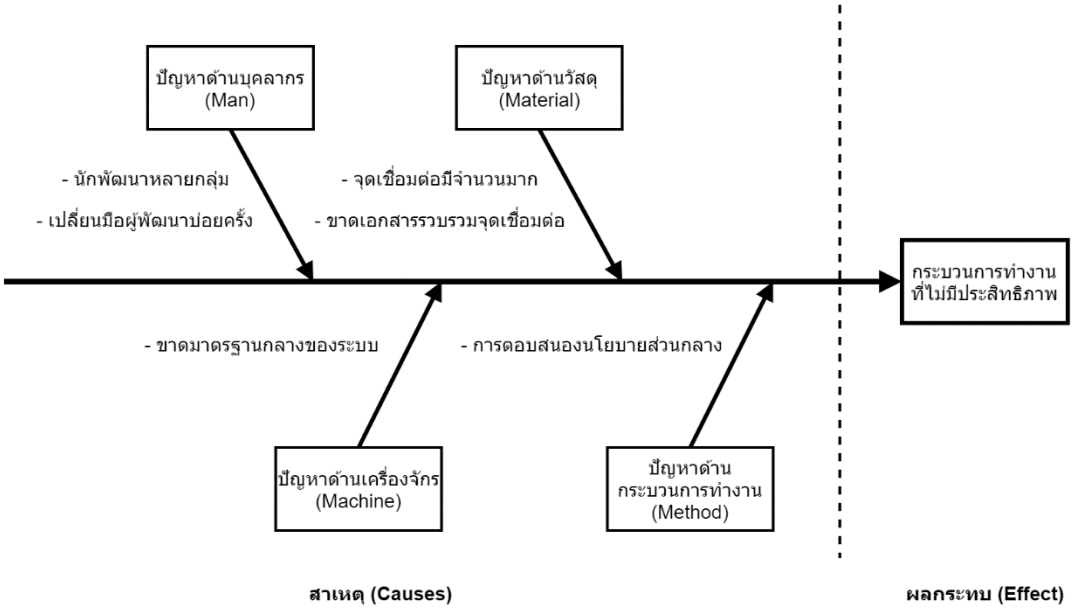
3.1 การวิเคราะห์ Web API ที่ใช้งานในปัจจุบัน

ในปัจจุบันสำนักทะเบียนและประมวลผล ได้แบ่งกลุ่มของจุดเชื่อมต่อ Web API ออกเป็น 2 กลุ่มดังต่อไปนี้

1. กลุ่มจุดเชื่อมต่อทั่วไป : คือกลุ่มจุดเชื่อมต่อที่เป็นพื้นฐานของระบบต่างๆ ที่สามารถใช้ร่วมกันได้ เช่น ข้อมูลพื้นฐานของ คณะ, สาขา, วิชาเรียน, โรงเรียน, มหาวิทยาลัย, ที่อยู่ (เขต, จังหวัด, รหัสไปรษณีย์, ประเทศ ฯลฯ), คำนำหน้านาม, กรุ๊ปเลือด รวมถึงการยืนยันสิทธิ์ด้วย JWT เป็นต้น
2. กลุ่มจุดเชื่อมต่อเฉพาะในแต่ละระบบ : คือกลุ่มจุดเชื่อมต่อที่ใช้งานระบบงานนั้นๆ เช่น ระบบรับสมัคร, ระบบตารางเรียน-ตารางสอน, ระบบลงทะเบียน เป็นต้น

3.2 การวิเคราะห์ปัญหาของการใช้งาน Web API ในปัจจุบัน

ผู้พัฒนาวิเคราะห์โดยใช้ ผังก้างปลา 4M 1H เป็นเครื่องมือทางการบริหารรูปแบบหนึ่ง ช่วยวิเคราะห์หาสาเหตุของปัญหาที่ก่อให้เกิดผลกระทบ โดยแยกแยะสาเหตุต่างๆ จากคน (Man), เครื่องจักร (Machine), วัสดุอุปกรณ์ (Material), กระบวนการทำงาน (Method) และสภาพแวดล้อม (Environment) โดยปกติใช้เป็นเครื่องมือนี้ในการประชุมระดมความคิดจากระดับผู้บริหารและบุคลากร ดังรูปที่ 3.1



รูปที่ 3.1 แผนผังก้างปลาวิเคราะห์ปัญหาโดยใช้ 4M 1H ของการใช้งาน Web API ในปัจจุบัน

การวิเคราะห์ปัญหาของสำนักทะเบียนและประมวลผล โดยใช้ 4M 1H มีรายละเอียดดังนี้

1. ปัญหาด้านบุคลากร (Man)

- นักพัฒนาหลายกลุ่ม : นอกจากกลุ่มผู้พัฒนาระบบภายในสำนักแล้ว ยังมีกลุ่มผู้พัฒนาระบบภายในสถาบันฯ และกลุ่มผู้พัฒนาระบบภายนอกสถาบันฯ อีกด้วย
- เปลี่ยนมือผู้พัฒนาบ่อยครั้ง : ในระบบใดๆ หลังจากพัฒนาแล้ว มีโอกาสที่ผู้พัฒนาเดิมอาจยุติความรับผิดชอบต่อระบบนั้น แล้วนำผู้พัฒนาคนใหม่มาแทน ซึ่งอาจทำให้ขาดองค์ความรู้เดิมของจุดเชื่อมต่อ ที่ผู้พัฒนาเดิมได้พัฒนาไว้

2. ปัญหาด้านวัสดุ (Material)

- จุดเชื่อมต่อมีจำนวนมาก : ด้วยอัตราการเติบโตของจุดเชื่อมต่อที่เพิ่มขึ้นแบบทวีคูณ ผลจากการเติบโตขึ้นของระบบใหม่ต่างๆ
- ขาดเอกสารรวบรวมจุดเชื่อมต่อ : ปัจจุบันยังไม่มีเอกสารที่รวบรวมจุดเชื่อมต่อ API

3. ปัญหาด้านเครื่องจักร (Machine)

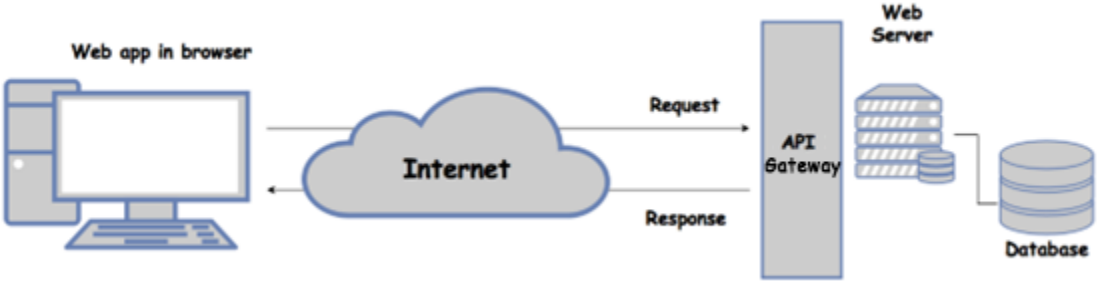
- ขาดมาตรฐานกลางของระบบ : การพัฒนาระบบใดๆ อาจทำให้ใช้งานได้ในระบบนั้นๆ ไม่ได้ออกแบบให้ใช้มาตรฐานกลางร่วมกันแต่แรก

4. ปัญหาด้านกระบวนการทำงาน (Method)

- การตอบสนองนโยบายส่วนกลาง : เป็นเหตุการณ์ที่เกิดขึ้นปกติในองค์กรที่ต้องการพัฒนาอย่างก้าวกระโดด ทำให้ระบบที่มีต้องตอบสนองนโยบายขององค์กรด้วย

จากปัญหาที่รวบรวมมาเป็นปัญหาหลักๆ ที่เกี่ยวข้องกับ Web API ผลกระทบโดยตรงคือทำให้เกิดกระบวนการทำงานที่ไม่มีประสิทธิภาพเท่าที่ควร

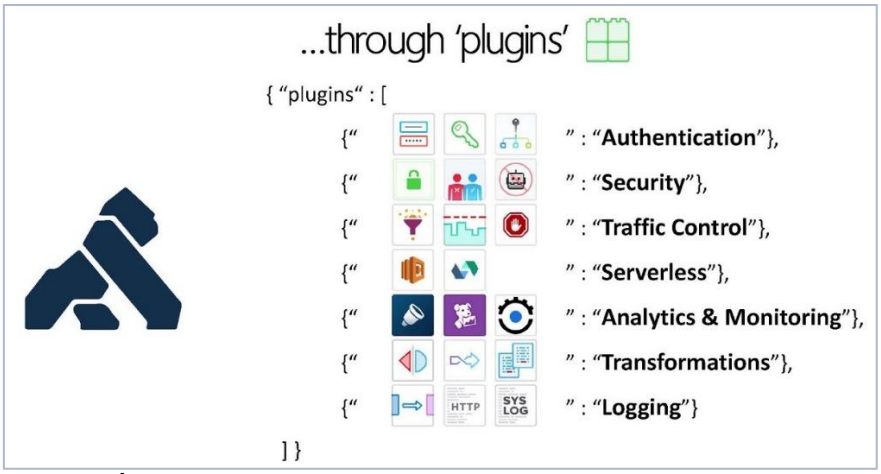
3.3 การออกแบบซิสเต็มไดอะแกรม (System Diagram)



รูปที่ 3.2 ซิสเต็มไดอะแกรมโดยภาพรวมของ Web API

ซิสเต็มไดอะแกรมนี้แสดงภาพรวมของส่วนที่เกี่ยวข้องกับ Web API ดังรูปที่ 3.2 โดยมีส่วนประกอบดังนี้

1. Web Application : ส่วนหลักที่ใช้ขับเคลื่อนกิจกรรมของระบบนั้นๆ เป็นแอปพลิเคชันที่ถูกเขียนขึ้นมาให้สามารถเปิดใช้ในเว็บเบราว์เซอร์ได้โดยตรง ไม่จำเป็นต้องโหลดแอปพลิเคชันแบบเต็มๆ ลงบนเครื่องของไคลเอนต์ ทำให้โดยรวมแล้วใช้ทรัพยากรค่อนข้างต่ำ
2. API Gateway : ส่วนที่ค่อยจัดหาข้อมูลเพื่อตอบสนองการทำงานของเว็บแอปพลิเคชัน โดยเป็นส่วนที่เกี่ยวข้องกับ Web API โดยตรง มีโปรแกรมที่ออกแบบมาเพื่อเป็น API Gateway เช่น Kong เป็นต้น โลโก้และตัวอย่างความสามารถดังรูปที่ 3.3



รูปที่ 3.3 โลโก้และตัวอย่างความสามารถของ Kong API Gateway

3. Database : ส่วนที่ใช้เก็บข้อมูลเพื่อให้ Web API อ่าน-เขียนข้อมูล นำไปประมวลผลและใช้งานในระบบนั้นๆ ต่อไป

จากการวิเคราะห์ในหัวข้อที่ 3.1 และ 3.2 นำไปสู่รูปซิสเต็มไดอะแกรมที่ 3.2 ซึ่งเป็นรูปแบบพื้นฐานของการใช้งาน Web API โดยทั่วไป แต่ยังไม่สามารถจัดหมวดหมู่, อ้างอิงจุดเชื่อมต่อ Web API และระบุรูปแบบวิธีการตรวจสอบสิทธิ์ได้

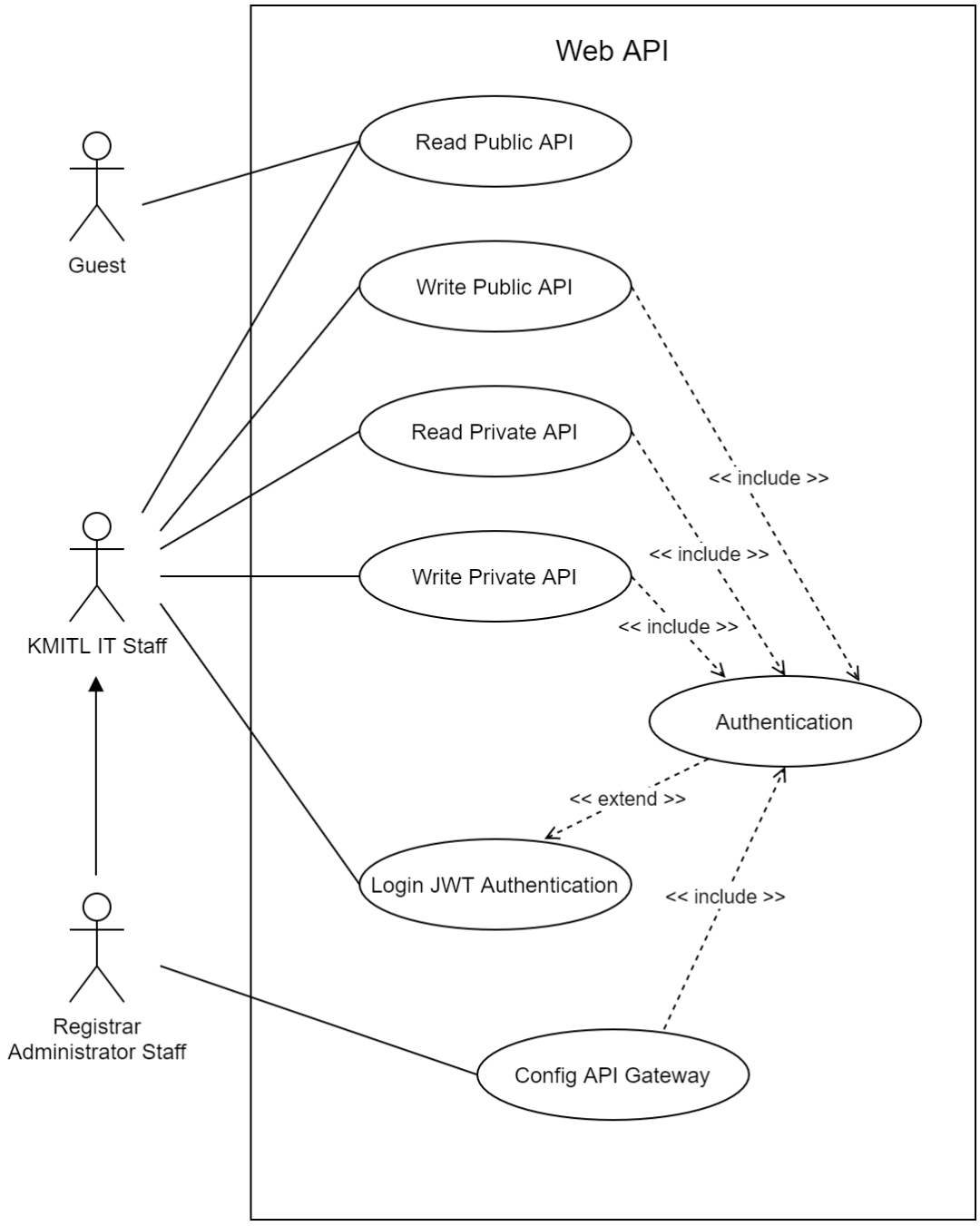
จึงได้ศึกษาหัวข้อที่ 2.6, 2.7 และ 2.8 ซึ่งเกี่ยวข้องกับการจัดหมวดหมู่, อ้างอิงจุดเชื่อมต่อ Web API และระบุรูปแบบวิธีการตรวจสอบสิทธิ์ เพื่อนำมาบูรณาการออกแบบยูสเคสไดอะแกรม (Use Case Diagram), ซิสเต็มซิสเต็มซีเควนไดอะแกรม (System Sequence Diagram) และเอกสาร Web API โดยใช้โปรแกรม Postman ในหัวข้อถัดไป

3.4 การออกแบบยูสเคสไดอะแกรม (Use Case Diagram)

ยูสเคสไดอะแกรมที่แสดงกิจกรรมในมุมมองผู้ใช้งานดังรูปที่ 3.4 ประกอบด้วย

แอกเตอร์ (Actor) จำนวน 4 แอกเตอร์ คือ เจ้าหน้าที่ดูแลระบบสำนักทะเบียนและประมวลผล, เจ้าหน้าที่สารสนเทศของสถาบันฯ และบุคคลภายนอก

ยูสเคสจำนวน 4 ยูสเคส โดยอธิบายรายละเอียดแต่ละยูสเคสในหัวข้อย่อยถัดไป



รูปที่ 3.4 ยูสเคสไดอะแกรมของ Web API

3.4.1 คำอธิบายยูสเคส (Use Case Specification)

ตารางที่ 3.1 แสดงคำอธิบายยูสเคส Read Public API

Use Case Name :	Read Public API	
Brief Description :	ใช้สำหรับอ่านข้อมูลสาธารณะ	
Actors :	Guest, KMITL IT Staff และ Registrar Administrator Staff	
Related Use Cases :	-	
Preconditions :	ผู้ใช้งานสร้างคำขอเพื่อส่งไปยังระบบ	
Postconditions :	ผู้ใช้งานได้รับข้อมูลจากระบบเพื่อนำไปใช้งาน	
Flow of Events :	**Actor**	**System**
	1. ผู้ใช้งานส่งคำขอข้อมูล 5. ผู้ใช้งานได้รับคำตอบกลับของข้อมูล	2. ระบบประมวลผลคำขอ 3. ระบบประมวลข้อมูลตามคำขอ 4. ระบบส่งคำตอบกลับพร้อมข้อมูล
Exception Conditions :	จาก Flow ที่ 2 : รูปแบบของคำขอไม่ถูกต้อง จาก Flow ที่ 3 : ไม่มีข้อมูลที่ระบบสามารถจัดหาให้ได้	

ตารางที่ 3.2 แสดงคำอธิบายยูสเคส Write Public API

Use Case Name :	Write Public API	
Brief Description :	ใช้สำหรับเขียนข้อมูลสาธารณะ	
Actors :	KMITL IT Staff และ Registrar Administrator Staff	
Related Use Cases :	Authentication	
Preconditions :	ผู้ใช้งานสร้างคำขอพร้อมข้อมูลและโทเค็นเพื่อส่งไปยังระบบ	
Postconditions :	ผู้ใช้งานได้รับผลการเขียนข้อมูลจากระบบ	
Flow of Events :	**Actor**	**System**
	1. ผู้ใช้งานส่งคำขอเขียนข้อมูล 5. ผู้ใช้งานได้รับคำตอบกลับของผลการเขียนข้อมูล	2. ระบบประมวลผลคำขอ 3. ระบบประมวลข้อมูลตามคำขอ 4. ระบบส่งคำตอบกลับพร้อมผลการเขียนข้อมูล
Exception Conditions :	จาก Flow ที่ 2 : ไม่มีสิทธิ์ในการเขียนข้อมูล จาก Flow ที่ 2 : รูปแบบของคำขอไม่ถูกต้อง	

ตารางที่ 3.3 แสดงคำอธิบายยูสเคส Read Private API

Use Case Name :	Read Private API	
Brief Description :	ใช้สำหรับอ่านข้อมูลส่วนตัว	
Actors :	KMITL IT Staff และ Registrar Administrator Staff	
Related Use Cases :	Authentication	
Preconditions :	ผู้ใช้งานสร้างคำขอพร้อมโทเค็นเพื่อส่งไปยังระบบ	
Postconditions :	ผู้ใช้งานได้รับข้อมูลจากระบบเพื่อนำไปใช้งาน	
Flow of Events :	**Actor**	**System**
	1. ผู้ใช้งานส่งคำขอข้อมูล 5. ผู้ใช้งานได้รับคำตอบกลับของข้อมูล	2. ระบบประมวลผลคำขอ 3. ระบบประมวลข้อมูลตามคำขอ 4. ระบบส่งคำตอบกลับพร้อมข้อมูล
Exception Conditions :	จาก Flow ที่ 2 : ไม่มีสิทธิ์ในการเขียนข้อมูล จาก Flow ที่ 2 : รูปแบบของคำขอไม่ถูกต้อง จาก Flow ที่ 3 : ไม่มีข้อมูลที่ระบบสามารถจัดหาให้ได้	

ตารางที่ 3.4 แสดงคำอธิบายยูสเคส Write Private API

Use Case Name :	Write Private API	
Brief Description :	ใช้สำหรับเขียนข้อมูลส่วนตัว	
Actors :	KMITL IT Staff และ Registrar Administrator Staff	
Related Use Cases :	Authentication	
Preconditions :	ผู้ใช้งานสร้างคำขอพร้อมข้อมูลและโทเค็นเพื่อส่งไปยังระบบ	
Postconditions :	ผู้ใช้งานได้รับผลการเขียนข้อมูลจากระบบ	
Flow of Events :	**Actor**	**System**
	1. ผู้ใช้งานส่งคำขอเขียนข้อมูล 5. ผู้ใช้งานได้รับคำตอบกลับของผลการเขียนข้อมูล	2. ระบบประมวลผลคำขอ 3. ระบบประมวลข้อมูลตามคำขอ 4. ระบบส่งคำตอบกลับพร้อมผลการเขียนข้อมูล
Exception Conditions :	จาก Flow ที่ 2 : ไม่มีสิทธิ์ในการเขียนข้อมูล จาก Flow ที่ 2 : รูปแบบของคำขอไม่ถูกต้อง	

ตารางที่ 3.5 แสดงคำอธิบายยูสเคส Authentication

Use Case Name :	Authentication
Brief Description :	ใช้สำหรับยืนยันสิทธิ์การใช้งาน
Actors :	-
Related Use Cases :	Read Public API, Write Public API, Read Private API, Write Private API, Login JWT Authentication และ Config API Gateway
Preconditions :	ระบบต้องการยืนยันสิทธิ์การใช้งาน
Postconditions :	ระบบได้รับผลการยืนยันสิทธิ์
Flow of Events :	**System**
	1. ระบบถอดรหัสโทเค็น 2. ระบบประมวลข้อมูลของโทเค็นที่ได้ถอดรหัสมา 3. ระบบส่งผลการยืนยันสิทธิ์
Exception Conditions :	จาก Flow ที่ 2 : รูปแบบของโทเค็นไม่ถูกต้อง

ตารางที่ 3.6 แสดงคำอธิบายยูสเคส Login JWT Authentication

Use Case Name :	Login JWT Authentication	
Brief Description :	ใช้สำหรับขอโทเค็นเพื่อใช้ในการยืนยันสิทธิ์	
Actors :	KMITL IT Staff และ Registrar Administrator Staff	
Related Use Cases :	Authentication	
Preconditions :	ผู้ใช้งานสร้างคำขอพร้อมข้อมูลที่ใช้ในการยืนยันสิทธิ์เพื่อส่งไปยังระบบ	
Postconditions :	ผู้ใช้งานได้รับโทเค็นจากระบบ	
Flow of Events :	**Actor**	**System**
	1. ผู้ใช้งานส่งคำขอพร้อมข้อมูลที่ใช้ในการยืนยันสิทธิ์ 5. ผู้ใช้งานได้รับคำตอบกลับเป็นโทเค็นจากระบบ	2. ระบบประมวลผลคำขอ 3. ระบบประมวลข้อมูลตามคำขอ 4. ระบบส่งคำตอบกลับเป็นโทเค็น
Exception Conditions :	จาก Flow ที่ 2 : รูปแบบของคำขอไม่ถูกต้อง จาก Flow ที่ 2 : ไม่มีสิทธิ์ในการขอโทเค็น	

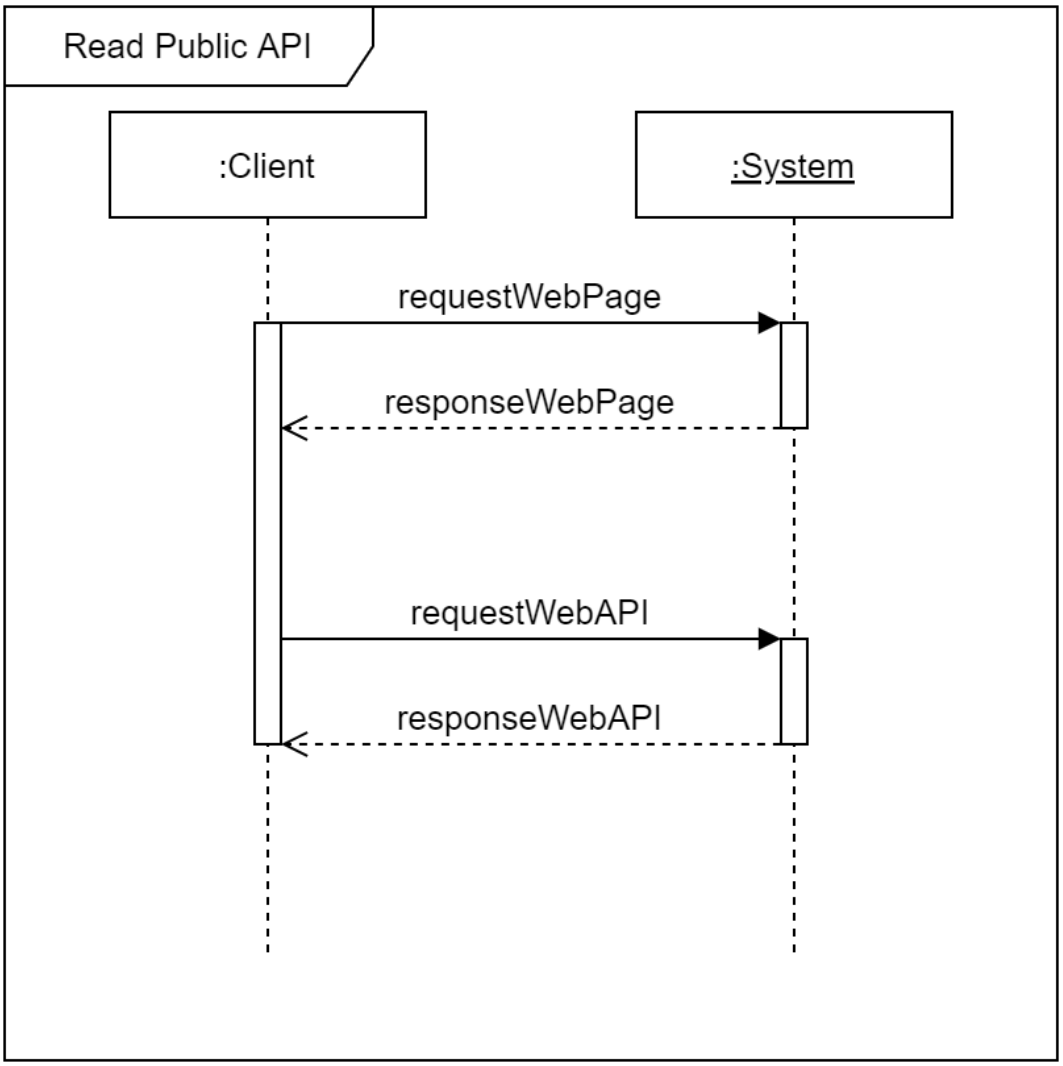
ตารางที่ 3.7 แสดงคำอธิบายยูสเคส Config API Gateway

Use Case Name :	Config API Gateway	
Brief Description :	ใช้สำหรับตั้งค่าการใช้งานของ API Gateway	
Actors :	Registrar Administrator Staff	
Related Use Cases :	Authentication	
Preconditions :	ผู้ใช้งานสร้างคำขอพร้อมข้อมูลการตั้งค่าและโทเค็นเพื่อส่งไปยังระบบ	
Postconditions :	ผู้ใช้งานได้รับผลการตั้งค่า	
Flow of Events :	**Actor**	**System**
	1. ผู้ใช้งานส่งคำขอข้อมูลการตั้งค่า 5. ผู้ใช้งานได้รับคำตอบกลับของผลการตั้งค่า	2. ระบบประมวลผลคำขอ 3. ระบบประมวลข้อมูลตามคำขอ 4. ระบบส่งคำตอบกลับเป็นผลการตั้งค่า
Exception Conditions :	จาก Flow ที่ 2 : ไม่มีสิทธิ์ในการตั้งค่า จาก Flow ที่ 2 : รูปแบบของคำขอไม่ถูกต้อง	

3.5 การออกแบบซิสเต็มซิสเต็มซีเควนไดอะแกรม (System Sequence Diagram)

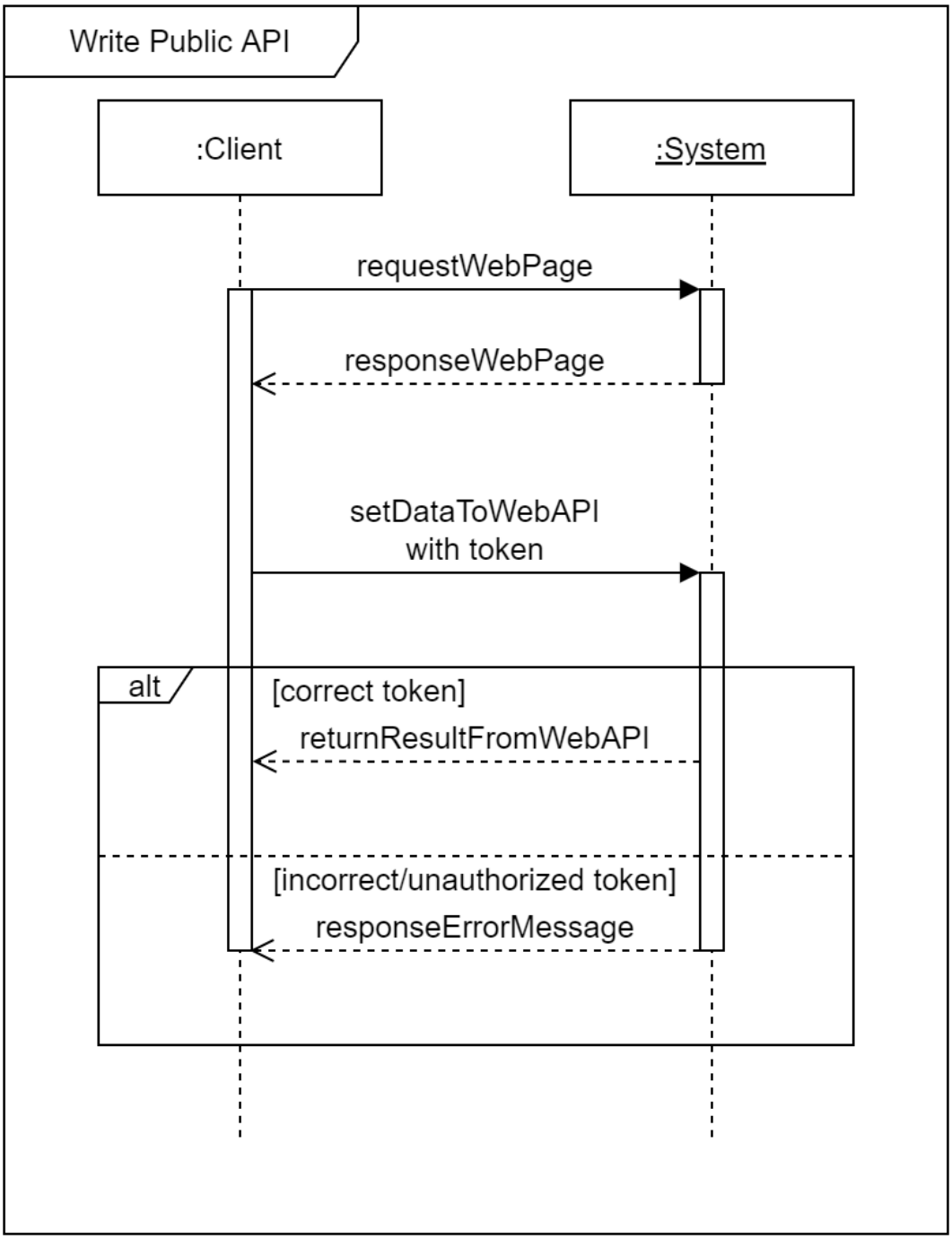
ซิสเต็มซีเควนไดอะแกรมแสดงกิจกรรมของลำดับกระบวนการทำงานของระบบ ที่สอดคล้องกับยูสเคสไดอะแกรมทั้ง 7 ยูสเคส ดังต่อไปนี้

3.5.1 ซิสเต็มซีเควนไดอะแกรมการอ่านข้อมูลสาธารณะ



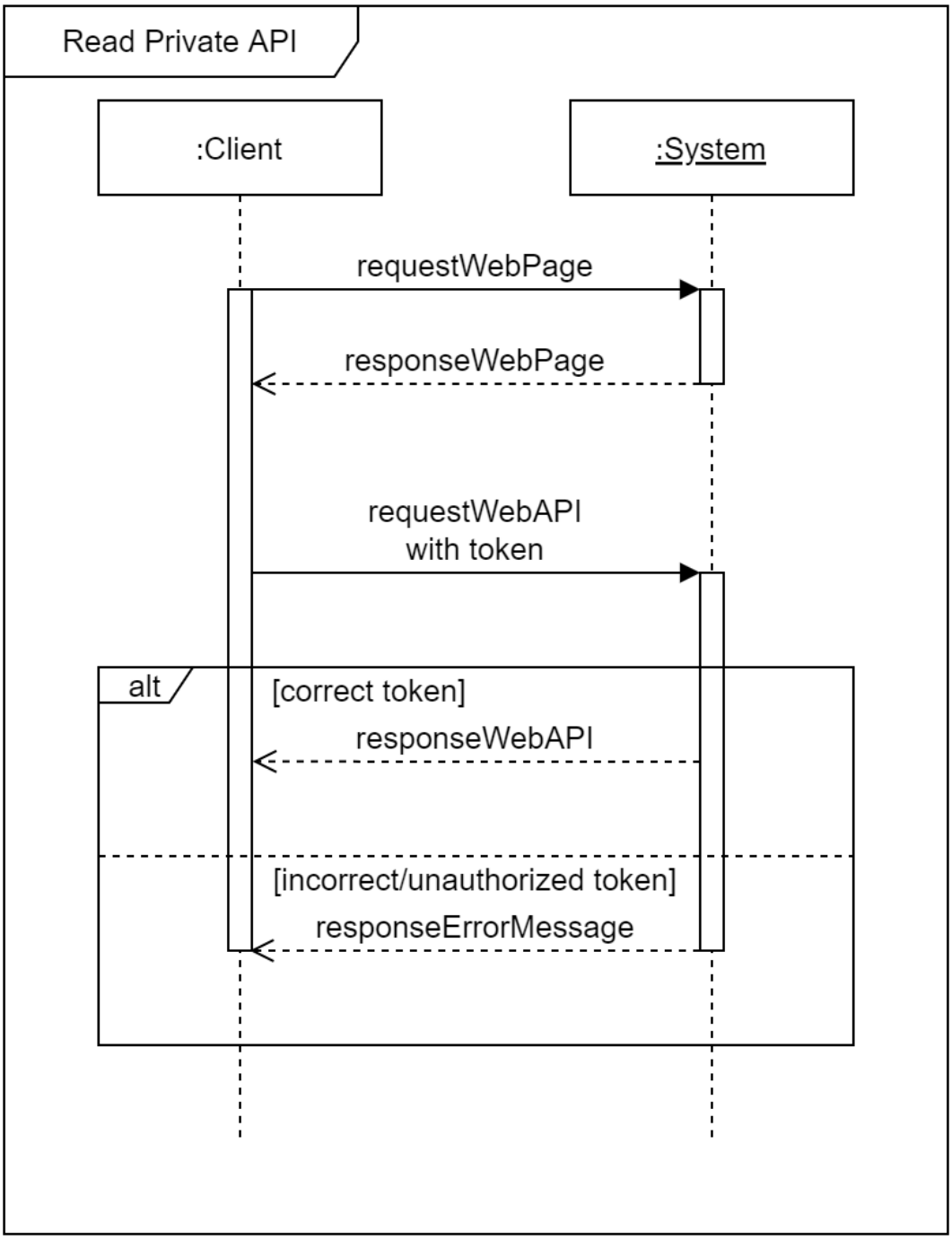
รูปที่ 3.5 ซิสเต็มซีเควนไดอะแกรมการอ่านข้อมูลสาธารณะ

3.5.2 ซิสเต็มซีเควนไดอะแกรมการเขียนข้อมูลสาธารณะ



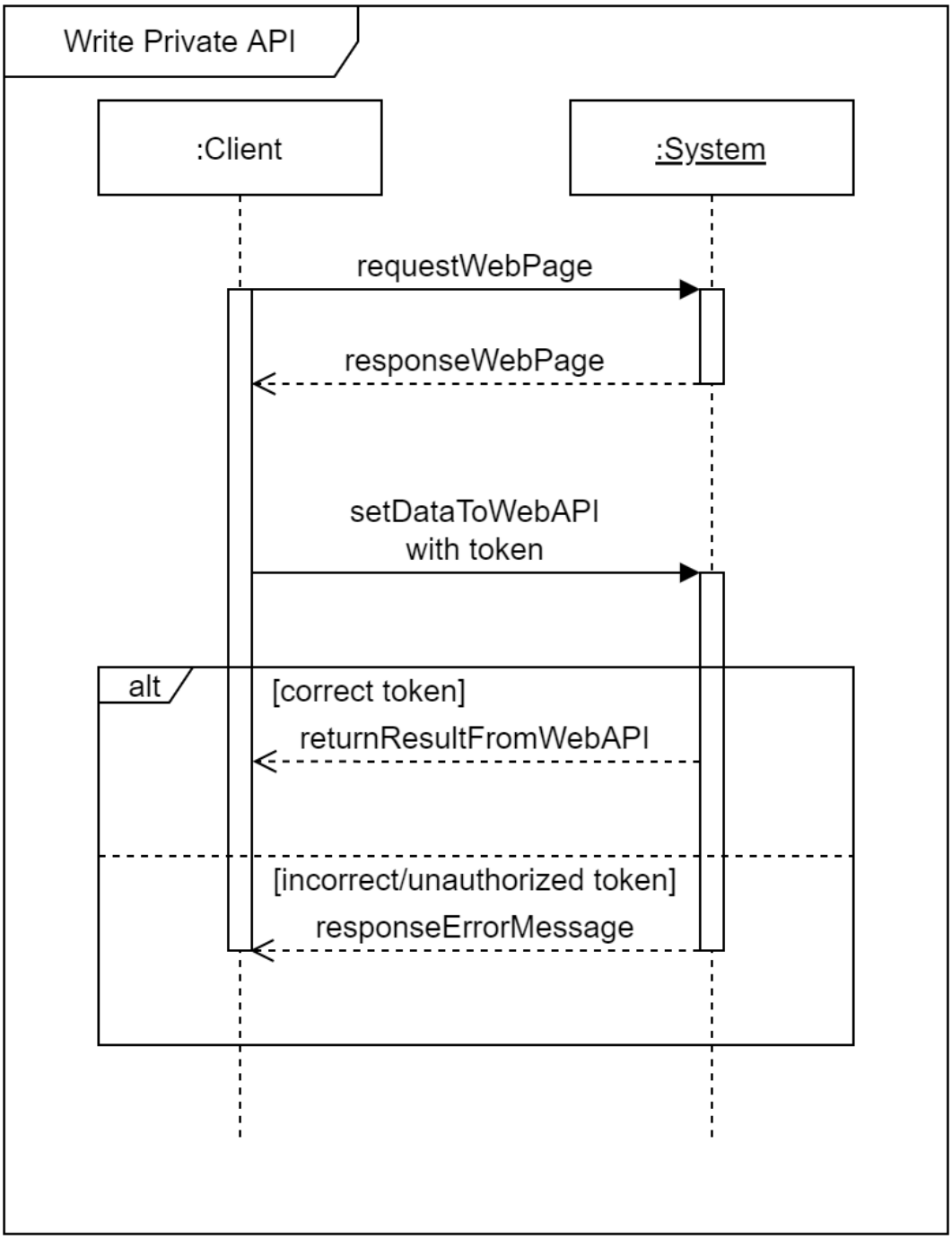
รูปที่ 3.6 ซิสเต็มซีเควนไดอะแกรมการเขียนข้อมูลสาธารณะ

3.5.3 ซิสเต็มซีเควนไดอะแกรมการอ่านข้อมูลส่วนตัว



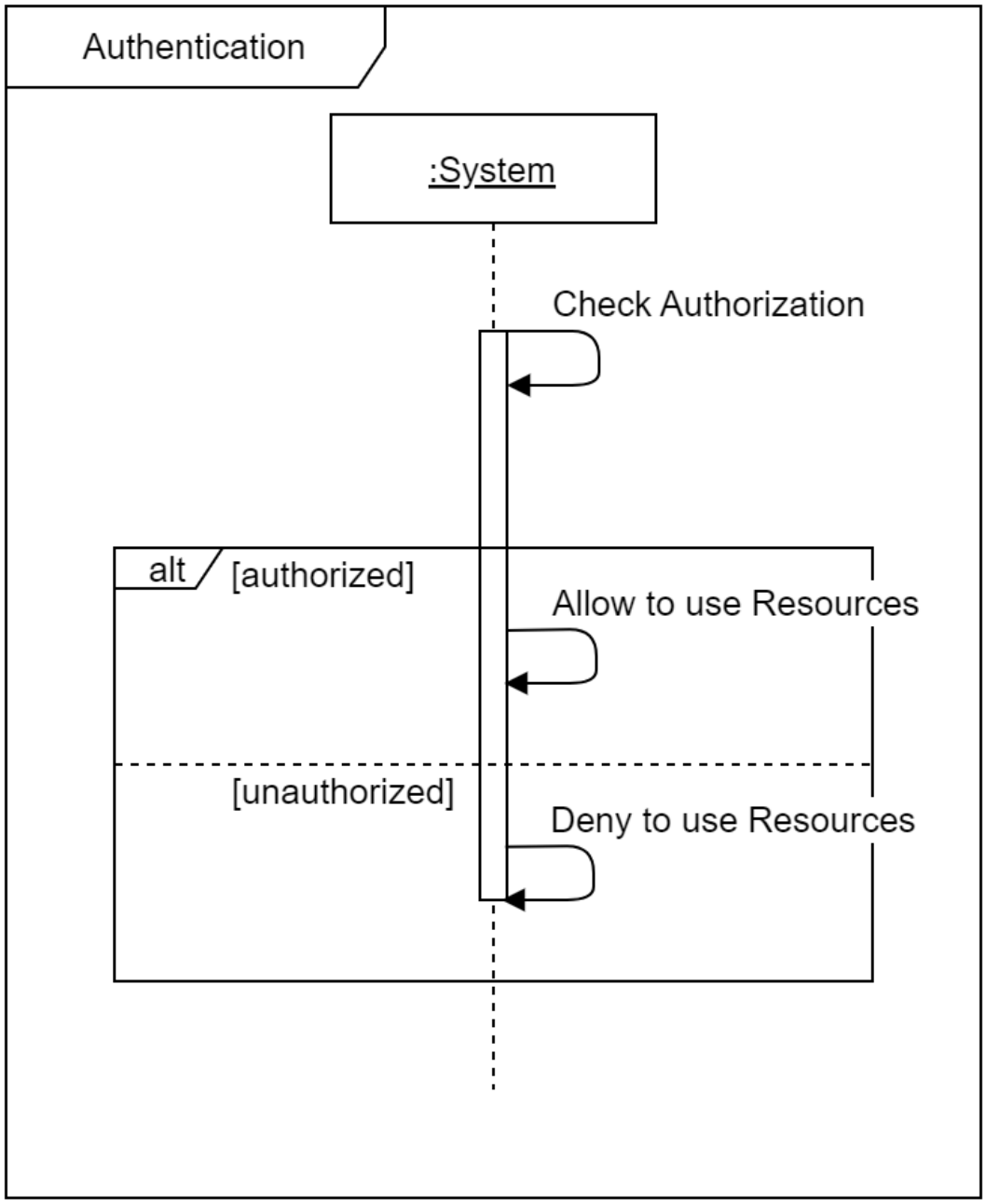
รูปที่ 3.7 ซิสเต็มซีเควนไดอะแกรมการอ่านข้อมูลส่วนตัว

3.5.4 ซิสเต็มซีเควนไดอะแกรมการเขียนข้อมูลส่วนตัว



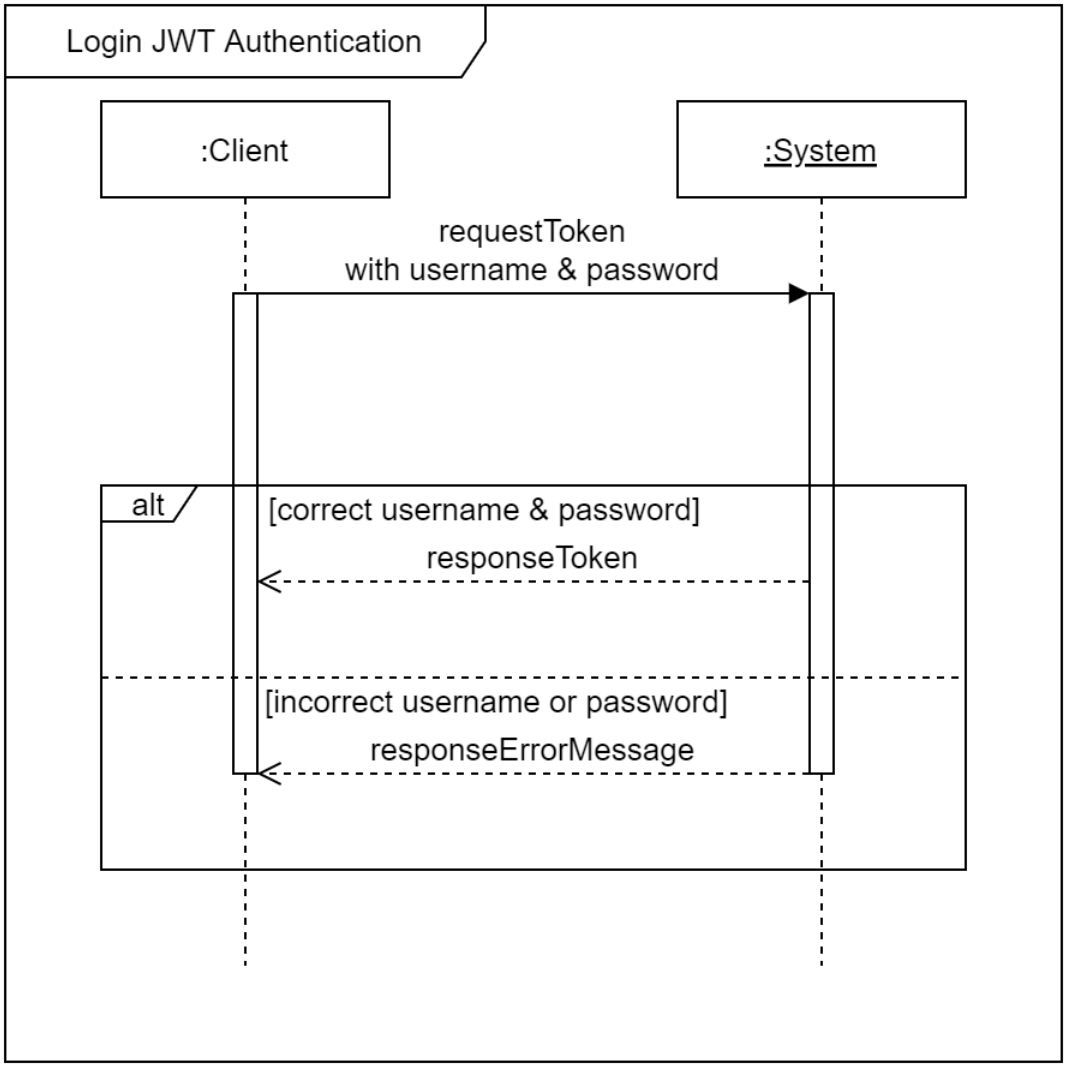
รูปที่ 3.8 ซิสเต็มซีเควนไดอะแกรมการเขียนข้อมูลส่วนตัว

3.5.5 ซิสเต็มซีเควนไดอะแกรมการยืนยันสิทธิ์การใช้งาน



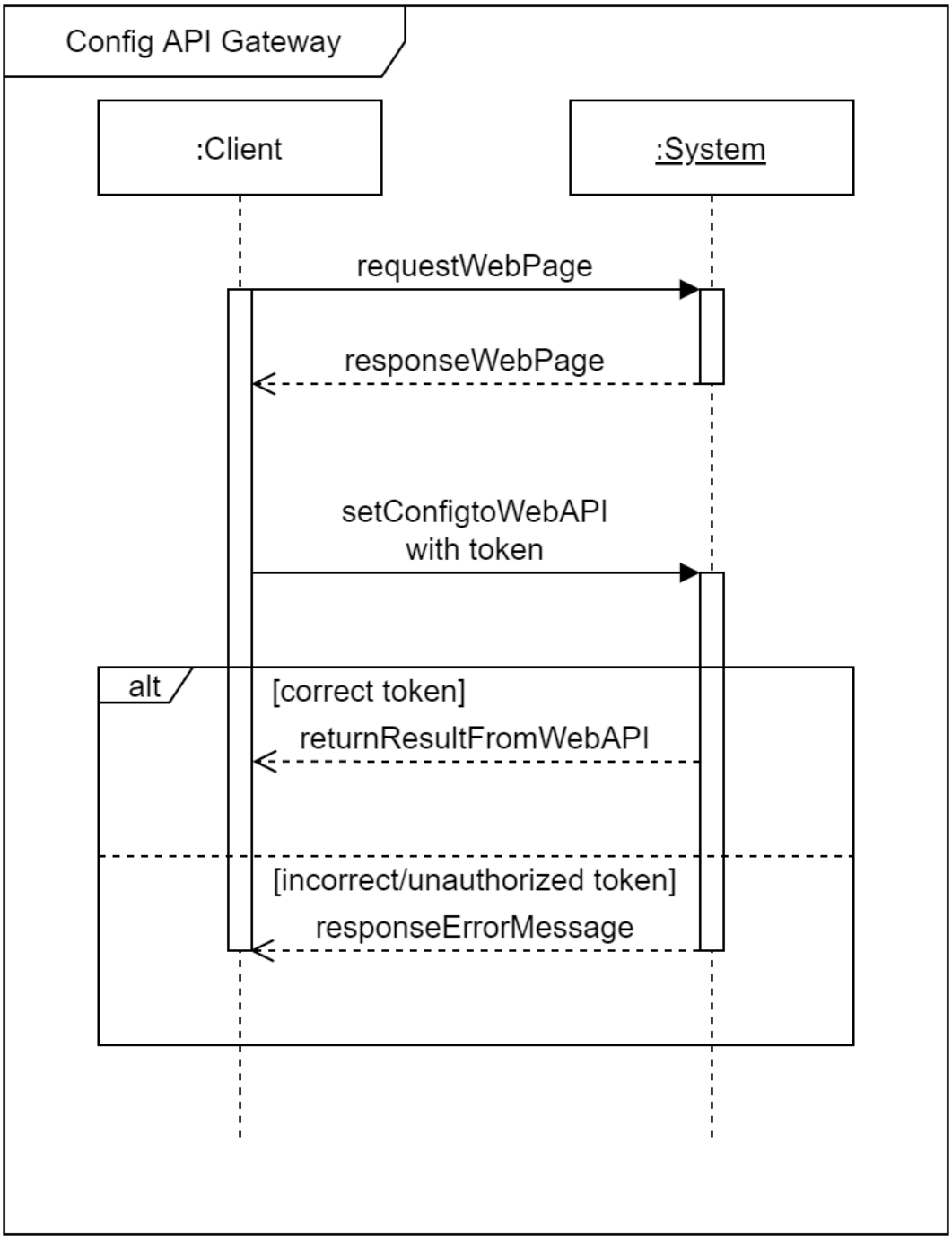
รูปที่ 3.9 ซิสเต็มซีเควนไดอะแกรมการยืนยันสิทธิ์การใช้งาน

3.5.6 ซิสเต็มซีเควนไดอะแกรมการขอโทเค็นเพื่อใช้ในการยืนยันสิทธิ์



รูปที่ 3.10 ซิสเต็มซีเควนไดอะแกรมการขอโทเค็นเพื่อใช้ในการยืนยันสิทธิ์

3.5.7 ซิสเต็มซีเควนไดอะแกรมการตั้งค่าการใช้งานของ API Gateway



รูปที่ 3.11 ซิสเต็มซีเควนไดอะแกรมการตั้งค่าการใช้งานของ API Gateway

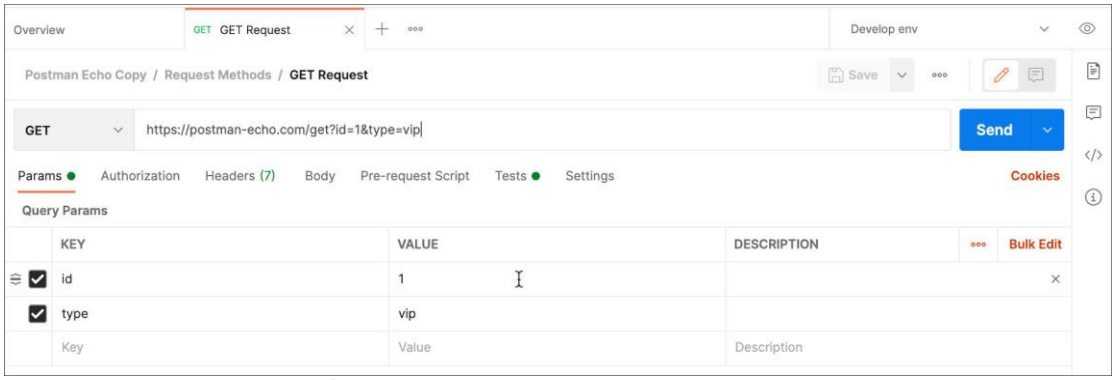
3.6 การออกแบบเอกสาร Web API โดยใช้โปรแกรม Postman

คู่มือการใช้งานโปรแกรม Postman สามารถเข้าถึงได้ที่ลิงก์ https://learning.postman.com/docs/

หลังจากพัฒนาจุดเชื่อมต่อ Web API สำเร็จแล้ว ขั้นตอนต่อไปคือการจัดหมวดหมู่ทดสอบจุดเชื่อมต่อ และจัดทำเอกสาร Web API ซึ่งโปรแกรม Postman สามารถตอบโจทย์การใช้งานได้อย่างครอบคลุม มี 4 ขั้นตอนหลักดังต่อไปนี้

3.6.1 การสร้างคำขอ

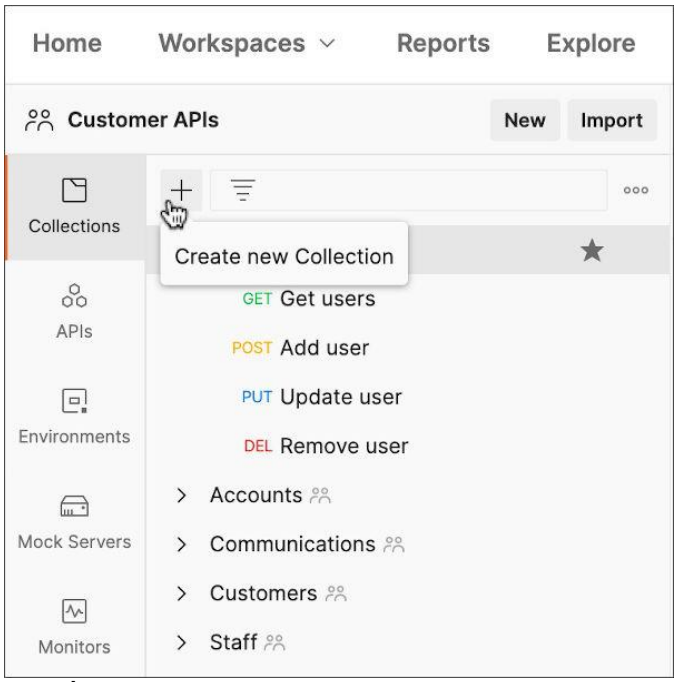
เริ่มต้นโดยเปิดแท็บใหม่ จากนั้นกรอกข้อมูลที่เป็นรายละเอียดหลักๆ คือ เมธอด, URL, Query Params (สำหรับเมธอด GET และ DELETE), เนื้อข้อมูล (ถ้ามี), การยืนยันสิทธิ์ (ถ้ามี) และอื่นๆ ตามวัตถุประสงค์ของงาน ดังรูปที่ 3.12



รูปที่ 3.12 การสร้างคำขอด้วยโปรแกรม Postman

3.6.2 การจัดหมวดหมู่คำขอ

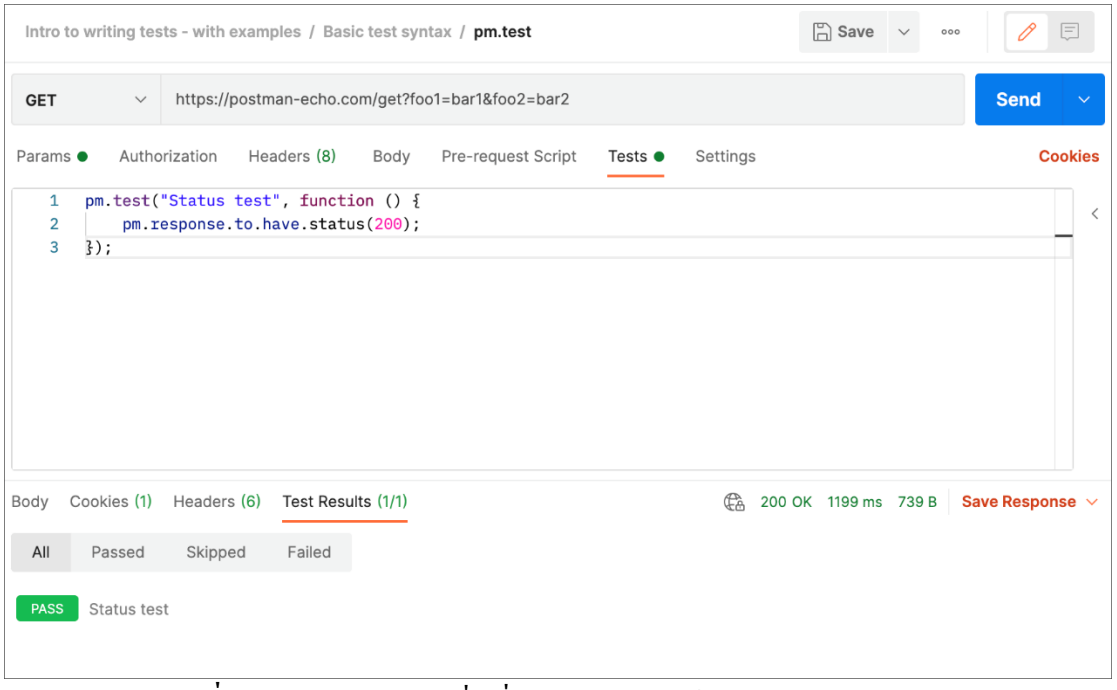
หลังจากตั้งค่าคำขอสมบูรณ์แล้ว ต่อไปเป็นการบันทึกคำขอเข้าไปอยู่ในหมวดหมู่ที่ได้ออกแบบไว้ ดังรูปที่ 3.13 โดยจากหัวข้อ 3.4 แต่ละหมวดหมู่สามารถแบ่งโฟลเดอร์ย่อยได้เป็น 4 รูปแบบคือ Read Public, Write Public, Read Private และ Write Private ตามความเหมาะสม



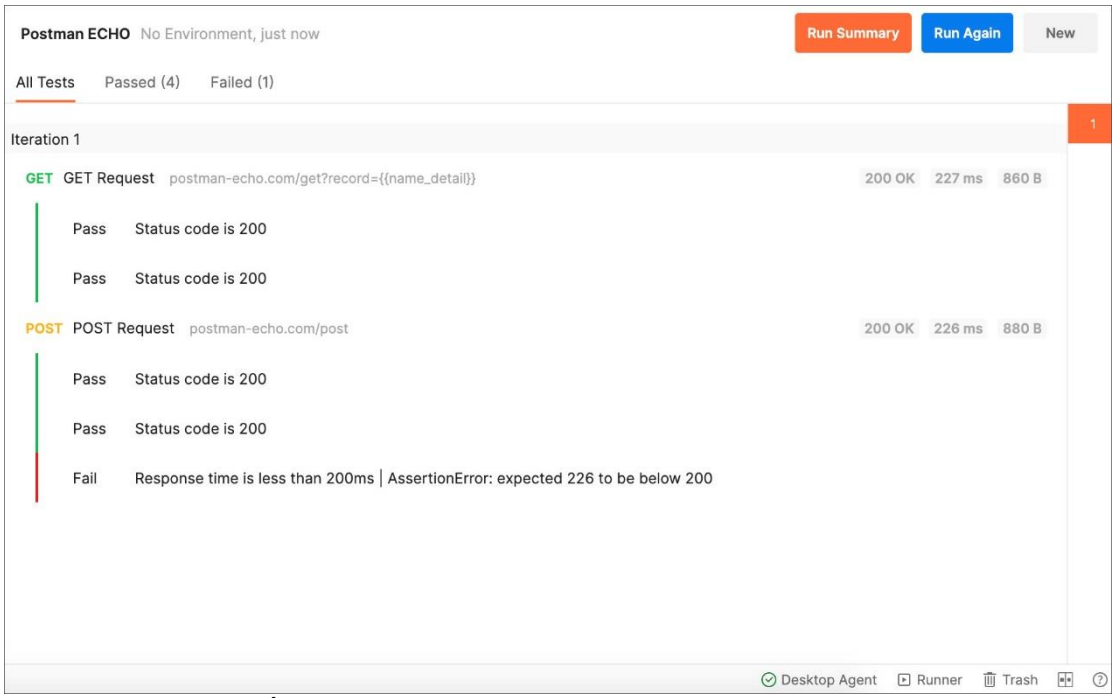
รูปที่ 3.13 การจัดหมวดหมู่คำขอด้วยโปรแกรม Postman

3.6.3 การทดสอบคำขอ

ในแต่ละคำขอสามารถเขียนเทสเคส (Test case) ให้ครอบคลุมการใช้งานเพื่อตรวจสอบว่าเมื่อได้ส่งข้อมูลในแต่ละรูปแบบไป เป็นตามผลที่ได้คาดหวังหรือไม่ ดังรูปที่ 3.14 และ 3.15



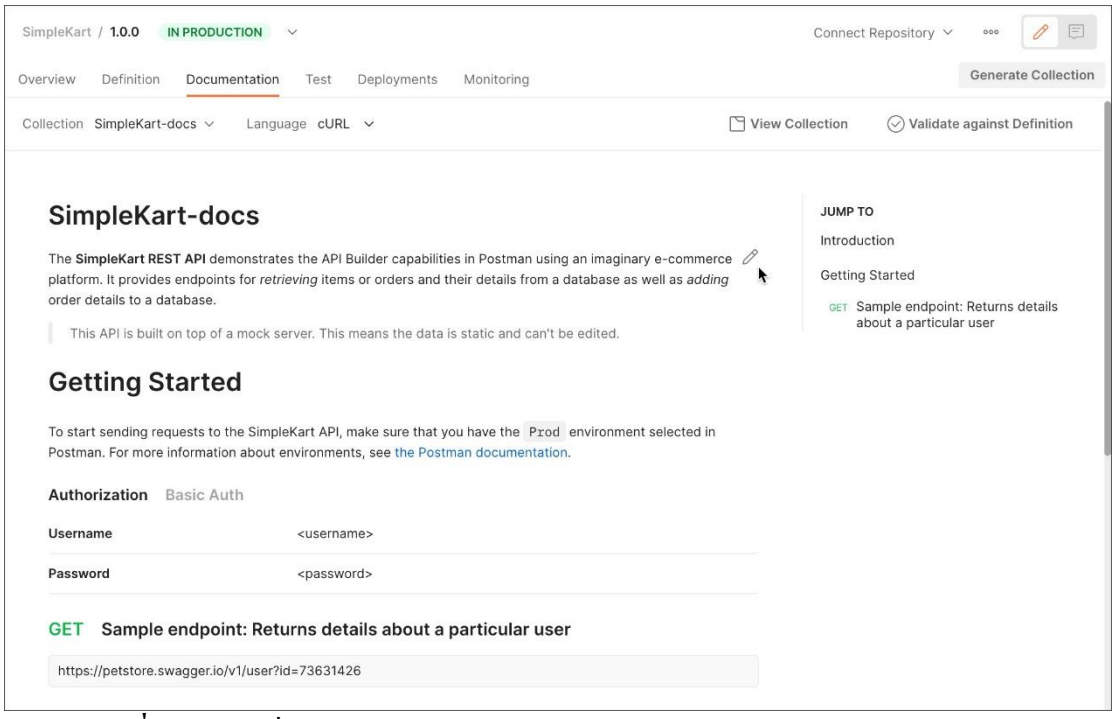
รูปที่ 3.14 การเขียนชุดคำสั่งเพื่อทดสอบคำขอด้วยโปรแกรม Postman



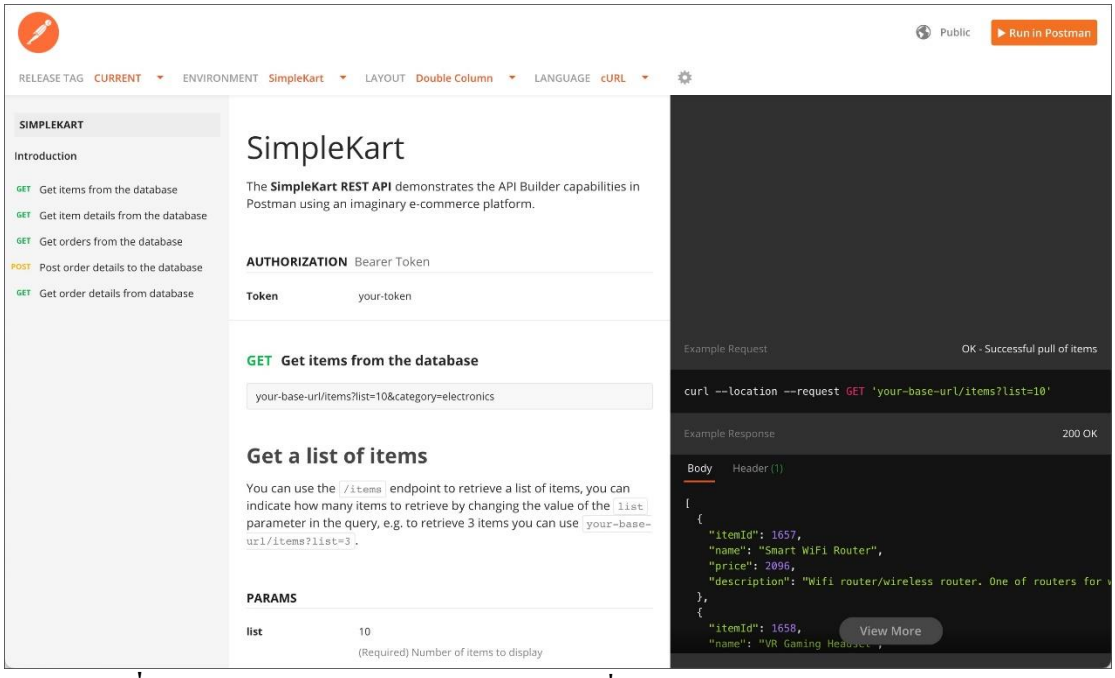
รูปที่ 3.15 ผลลัพธ์การทดสอบคำขอด้วยโปรแกรม Postman

3.6.4 การจัดทำเอกสาร Web API

เมื่อหมวดหมู่ของคำขอถูกพัฒนาเสร็จสิ้นตามความต้องการของระบบแล้ว สามารถสร้างเอกสาร Web API ได้จากหมวดหมู่ของคำขอ โดยเขียนอธิบายรายละเอียดให้ครอบคลุมและเป็นไปตามมาตรฐาน HTTP สำหรับ Web API ดังรูปที่ 3.16 และ 3.17



รูปที่ 3.16 การเพิ่มคำอธิบายในการจัดทำเอกสาร Web API ด้วยโปรแกรม Postman



รูปที่ 3.17 ตัวอย่างหน้าเว็บเอกสาร Web API ที่เผยแพร่แล้ว ด้วยโปรแกรม Postman

บทที่ 4
ผลการดำเนินงาน

จากการวิเคราะห์และการออกแบบระบบในบทที่ 3 สามารถนำไปจัดสร้างระบบและทดสอบผลการดำเนินงาน แบ่งออกเป็นหัวข้อ ดังนี้

4.1 เปรียบเทียบรูปแบบวิธีการตรวจสอบสิทธิ์ ที่ใช้ใน Web API

จากหัวข้อที่ 2.8 ได้ศึกษาถึงรูปแบบวิธีการตรวจสอบสิทธิ์ ที่ใช้ใน Web API โดยการตรวจสอบสิทธิ์นั้นจำเป็นอย่างยิ่ง เพื่อใช้จำกัดการเข้าถึงข้อมูล ควบคุมการใช้ข้อมูล และเก็บประวัติการใช้งาน ซึ่งรูปแบบวิธีการตรวจสอบสิทธิ์มีอยู่หลายรูปแบบ คุณสมบัติต่างกันไป จึงได้วิเคราะห์เปรียบเทียบแต่ละรูปแบบได้ ดังนี้

ตารางที่ 4.1 เปรียบเทียบรูปแบบวิธีการตรวจสอบสิทธิ์ ที่ใช้ใน Web API

รูปแบบวิธีการตรวจสอบสิทธิ์	ความซับซ้อนในการพัฒนา	ความหลากหลายในการปรับแต่ง	ความปลอดภัย
HTTP Basic Authentication	น้อย	น้อย	น้อยที่สุด
API Key Authentication	ขึ้นอยู่กับการออกแบบ	มากที่สุด	ขึ้นอยู่กับการออกแบบ
OAuth Authentication	ปานกลาง	น้อย	มากที่สุด
No Authentication	น้อยที่สุด	-	ไม่มี

จากตารางที่ 4.1 สังเกตได้ว่าการยืนยันสิทธิ์แบบ HTTP Basic ไม่เหมาะสมที่นำมาใช้งานในปัจจุบัน ที่ไม่มีการยืนยันสิทธิ์เลยอาจจำเป็นต้องใช้ใน Public API ซึ่งควบคุมการใช้งานโดยอ้างอิงด้วย IP Address แทนเป็นต้น ส่วนการยืนยันสิทธิ์แบบ API Key และ OAuth สามารถเลือกใช้ได้ตามความเหมาะสม API Key มีความหลากหลายในการออกแบบ ซึ่งต้องใช้ความชำนาญของผู้พัฒนาร่วมด้วย ส่วน OAuth เป็นมาตรฐาน Framework ที่สามารถร่วมใช้งานได้กับ Endpoint ของผู้ให้บริการ Web API อื่นๆ ที่ใช้การยืนยันสิทธิ์ด้วย OAuth เหมือนกัน

4.2 การจัดทำเอกสาร Web API

หลังจากศึกษาองค์ความรู้ของการออกแบบ การพัฒนา วิธีการพัฒนา และตัวอย่างที่ดีเรียบร้อยแล้ว จึงนำมาพัฒนาการรวบรวมจุดเชื่อมต่อ Web API ทั้งหมดที่สามารถจัดทำได้ ของระบบสารสนเทศสำนักทะเบียนและประมวลผล สจล. อยู่ 4 ระบบ คือ

1. ระบบศูนย์กลางสำหรับเป็นพื้นฐานของระบบอื่นๆ (Central API)
2. ระบบสมัครนักศึกษา (Admission System)
3. ระบบรายงานตัวออนไลน์ (Matriculation System)
4. ระบบลงทะเบียนเรียน (Registration System)

เมื่อรวบรวมเสร็จสิ้นแล้วจึงนำมาจัดทำเอกสาร Web API โดยใช้โปรแกรม Postman โดยมีรายละเอียดขั้นตอน ดังนี้

4.2.1 รวบรวมจุดเชื่อมต่อจากเอกสารเดิมที่มีอยู่ และซอร์สโค้ด

เอกสารที่มีอยู่ที่ได้รวบรวมมาคือ เอกสารส่งมอบระบบ, เอกสารคู่มือการใช้งานระบบ, เอกสารคู่มือสำหรับผู้พัฒนาระบบ และเอกสารโครงสร้างฐานข้อมูล

ซอร์สโค้ดแบ่งเป็น 2 ส่วน คือ โปรแกรมส่วนหน้า (Frontend) ที่ติดต่อกับผู้ใช้งาน ดังรูปที่ 4.1 และโปรแกรมส่วนหลัง (Backend) ที่เป็นส่วนประมวลผลข้อมูลและติดต่อกับฐานข้อมูล ดังรูปที่ 4.2

```
3   class UserProvider extends HttpRequest {
4     constructor() {
5       super(process.env.VUE_APP_ENDPOINT + 'api/user/')
6     }
7
8     async getJWTUserInfo(student_id) {
9       const { data } = await this.get('/', {
10        function: 'get-jwt-user-info',
11        student_id: student_id
12      })
13      return data
14    }
15
16    async loginJWT(email, password) {
17      const { data } = await this.post('/', {
18        function: 'login-jwt',
19        email: email,
20        password: password
21      })
22      return data
23    }
24  }
```

รูปที่ 4.1 ตัวอย่างซอร์สโค้ดที่แสดงจุดเชื่อมต่อของโปรแกรมส่วนหน้า

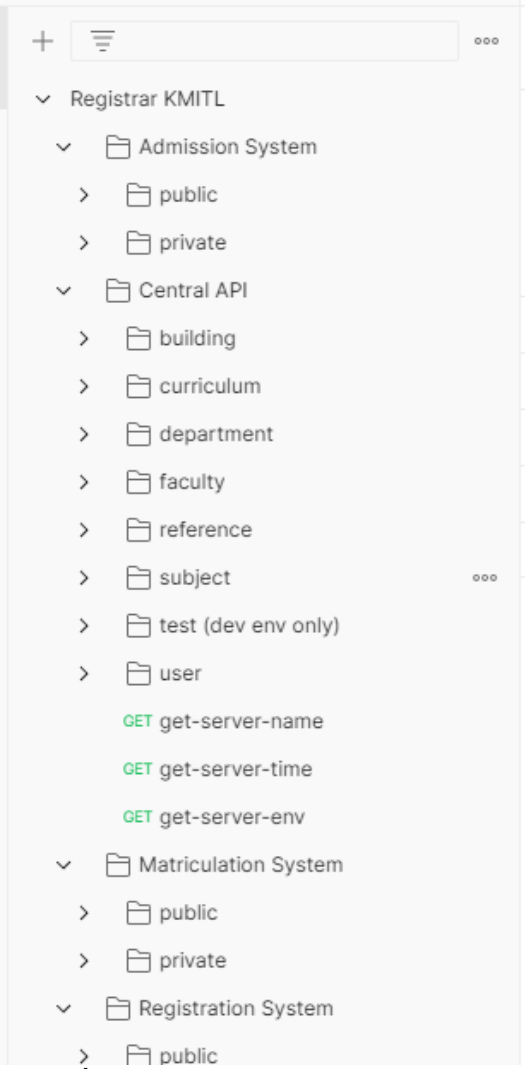
```php
# route
if (REQUEST_METHOD == 'GET') {
    switch ($_REQUEST['function']) {

        case 'get-registrar-user-account':
            $response = getRegistrarUserAccount($_REQUEST['user_id']);
            break;
        case 'logout':
            $response = logout();
            break;

        case 'get-jwt-user-info':
            $response = getJWTUserInfo($_REQUEST['student_id']);
            break;

        default:
            error('Bad Request', 400);
    }
} elseif (REQUEST_METHOD == 'POST') {
```

รูปที่ 4.2 ตัวอย่างซอร์สโค้ดที่แสดงจุดเชื่อมต่อของโปรแกรมส่วนหลัง

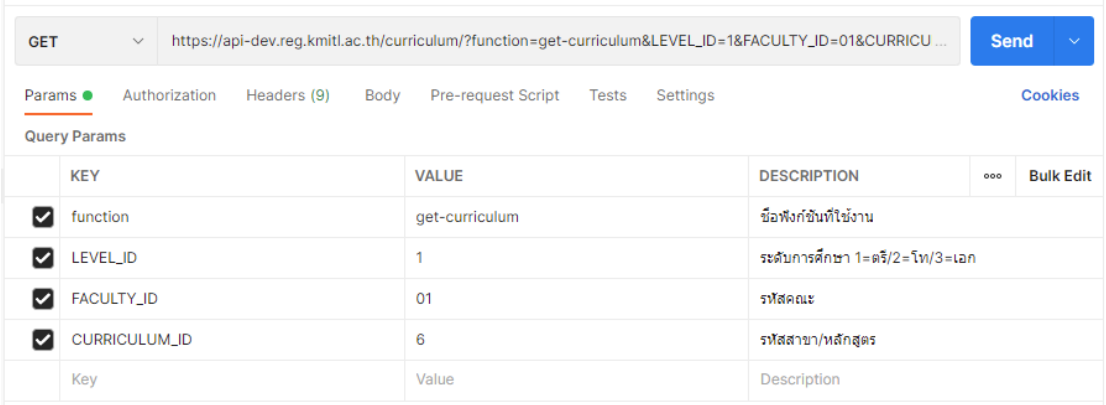
4.2.2 สร้างชุดของคำขอและจัดหมวดหมู่

สร้างหมวดหมู่ของแต่ละระบบ พร้อมโฟลเดอร์ย่อยในแต่ละส่วนของระบบ ดังรูปที่ 4.3 จากนั้นสร้างชุดของคำขอพร้อมพารามิเตอร์ต่างๆ ที่ใช้ร่วมกับคำขอนั้นๆ ดังรูปที่ 4.4 โดยจากหัวข้อที่ 3.4 แต่ละหมวดหมู่สามารถแบ่งโฟลเดอร์ย่อยได้เป็น 4 รูปแบบคือ Read Public, Write Public, Read Private และ Write Private ตามความเหมาะสม

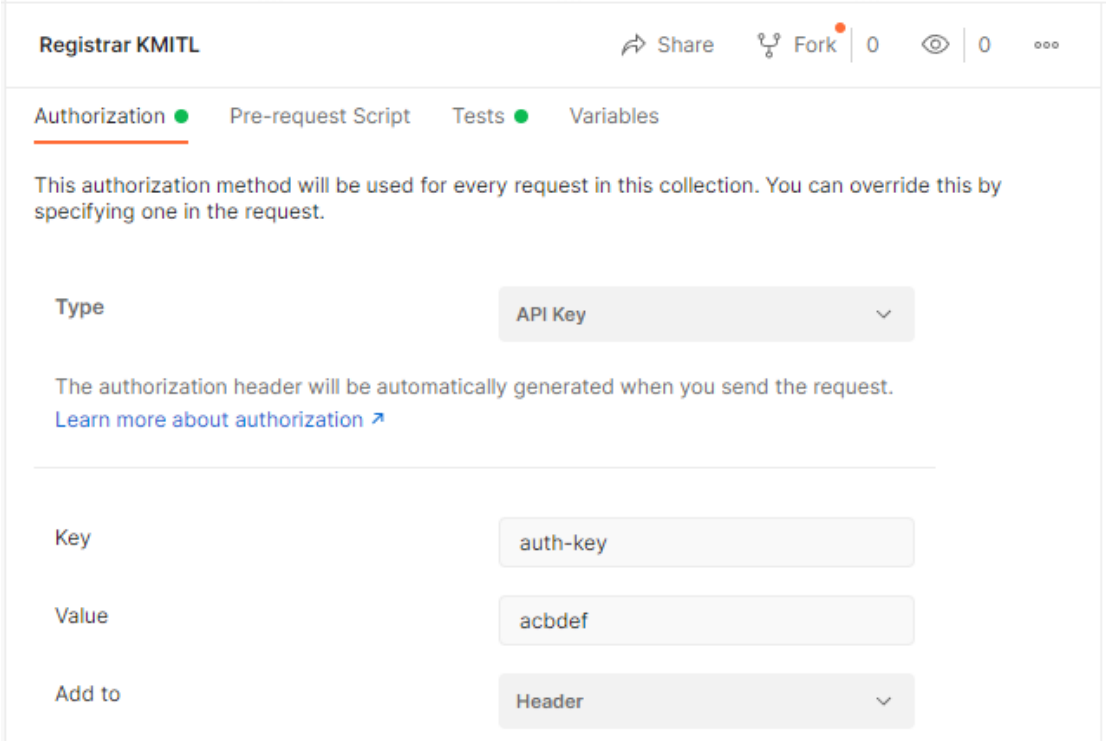
สามารถกำหนดรูปแบบวิธีการตรวจสอบสิทธิ์ได้ 3 ระดับ คือ ระดับคำขอ ระดับหมวดหมู่ และระดับครอบคลุมทุกจุดเชื่อมต่อ (Workspace) ตัวอย่างเช่น API Key, Bearer Token, OAuth 2.0 เป็นต้น ดังรูปที่ 4.5



รูปที่ 4.3 หมวดหมู่ของคำขอที่จัดทำแล้ว



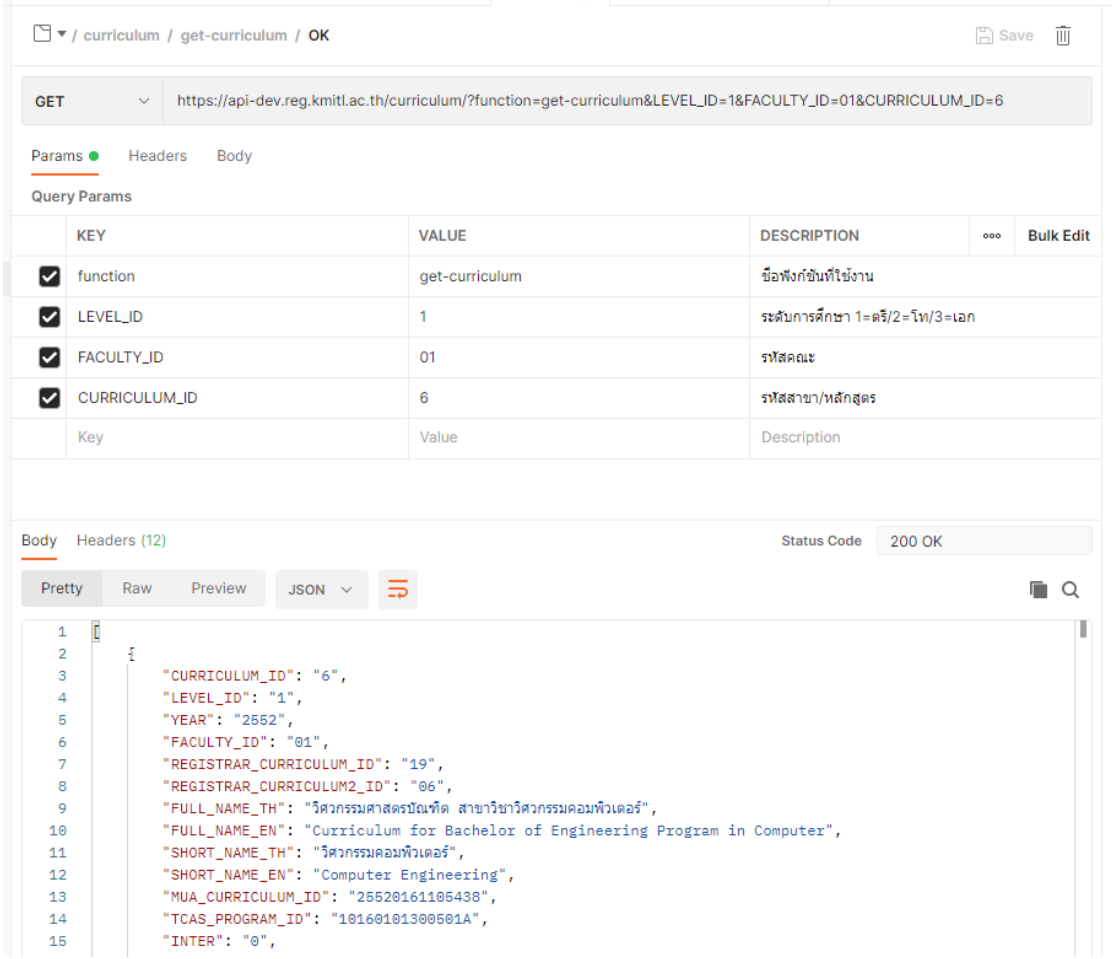
รูปที่ 4.4 ตัวอย่างคำขอข้อมูลหลักสูตร



รูปที่ 4.5 กำหนดรูปแบบวิธีการตรวจสอบสิทธิ์แบบ API Key ที่ระดับครอบคลุมทุกจุดเชื่อมต่อ

4.2.3 สร้างตัวอย่างการใช้งานของแต่ละคำขอ

เมื่อสร้างชุดของคำขอแล้ว ต่อไปเป็นการสร้างตัวอย่างของแต่ละคำขอ เพื่อแสดงสถานการณ์ต่างๆ ที่สามารถเกิดขึ้นได้ ซึ่งหากมีจำนวนตัวอย่างที่ครอบคลุมการใช้งาน ย่อมมีผลดีต่อผู้ที่ใช้เอกสารนี้ ดังรูปที่ 4.6



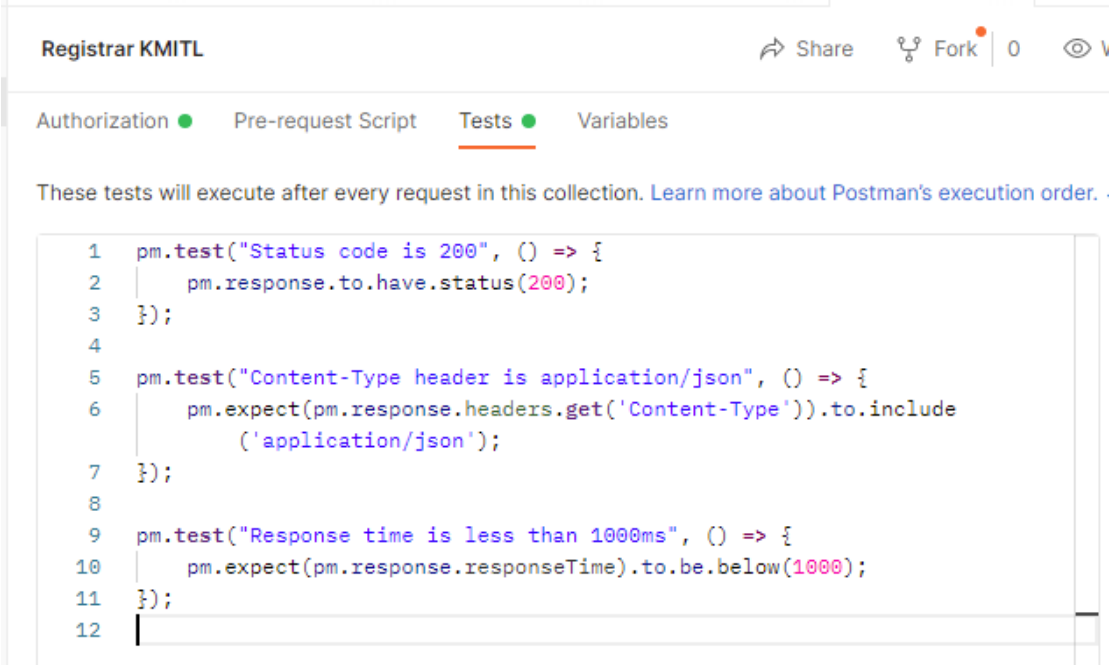
รูปที่ 4.6 สร้างตัวอย่างที่คำตอบกลับปกติของคำขอหลักสูตร

4.2.4 สร้างสคริปทดสอบจุดเชื่อมต่อ

สามารถสร้างสคริปทดสอบได้ 3 ระดับ คือ ระดับคำขอ ระดับหมวดหมู่ และระดับครอบคลุมทุกจุดเชื่อมต่อ ตัวอย่างสคริปท์สามารถศึกษาเพิ่มเติมได้ที่ภาคผนวก ก

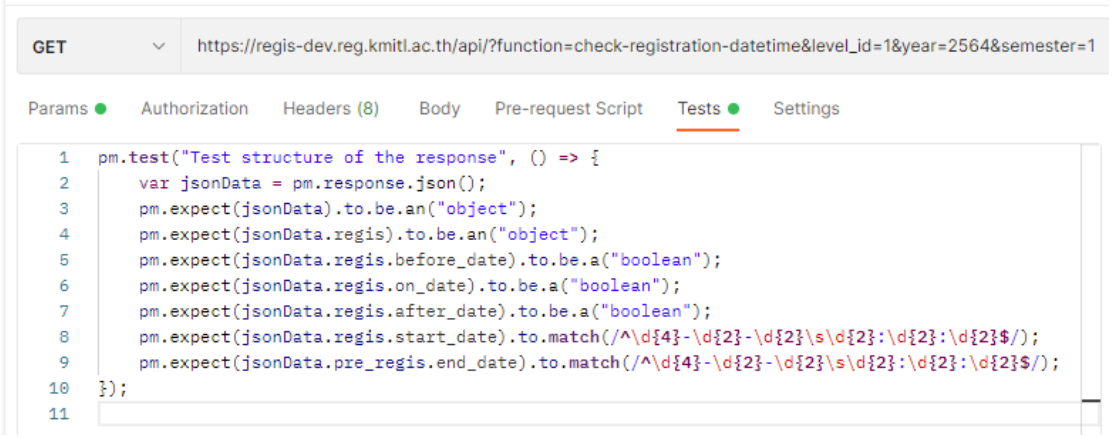
ในระดับครอบคลุมทุกจุดเชื่อมต่อได้สร้างสคริปทดสอบไว้ 3 คำสั่ง ดังรูปที่ 4.7 คือ

1. ทดสอบสถานะของคำตอบกลับ คาดหวังเป็น 200 OK
2. ทดสอบรูปแบบข้อมูลของคำตอบกลับ คาดหวังเป็น application/json
3. ทดสอบระยะเวลาของคำตอบกลับ คาดหวังไม่เกิน 1000 ms

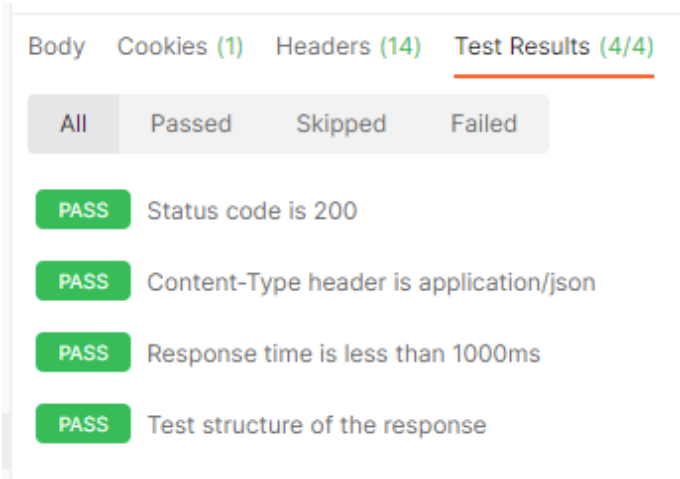


รูปที่ 4.7 สคริปทดสอบจุดเชื่อมต่อ ระดับครอบคลุมทุกจุดเชื่อมต่อ

ในระดับคำขอ การสร้างสคริปทดสอบขึ้นอยู่กับคุณสมบัติและการใช้งานของจุดเชื่อมต่อ มีคำสั่งพื้นฐานที่สามารถใช้ได้ในหลายคำขอคือ ทดสอบโครงสร้างข้อมูลของคำตอบกลับ ดังรูปที่ 4.8 และผลการทดสอบจุดเชื่อมต่อ ดังรูปที่ 4.9



รูปที่ 4.8 ตัวอย่างสคริปทดสอบจุดเชื่อมต่อ ระดับคำขอ

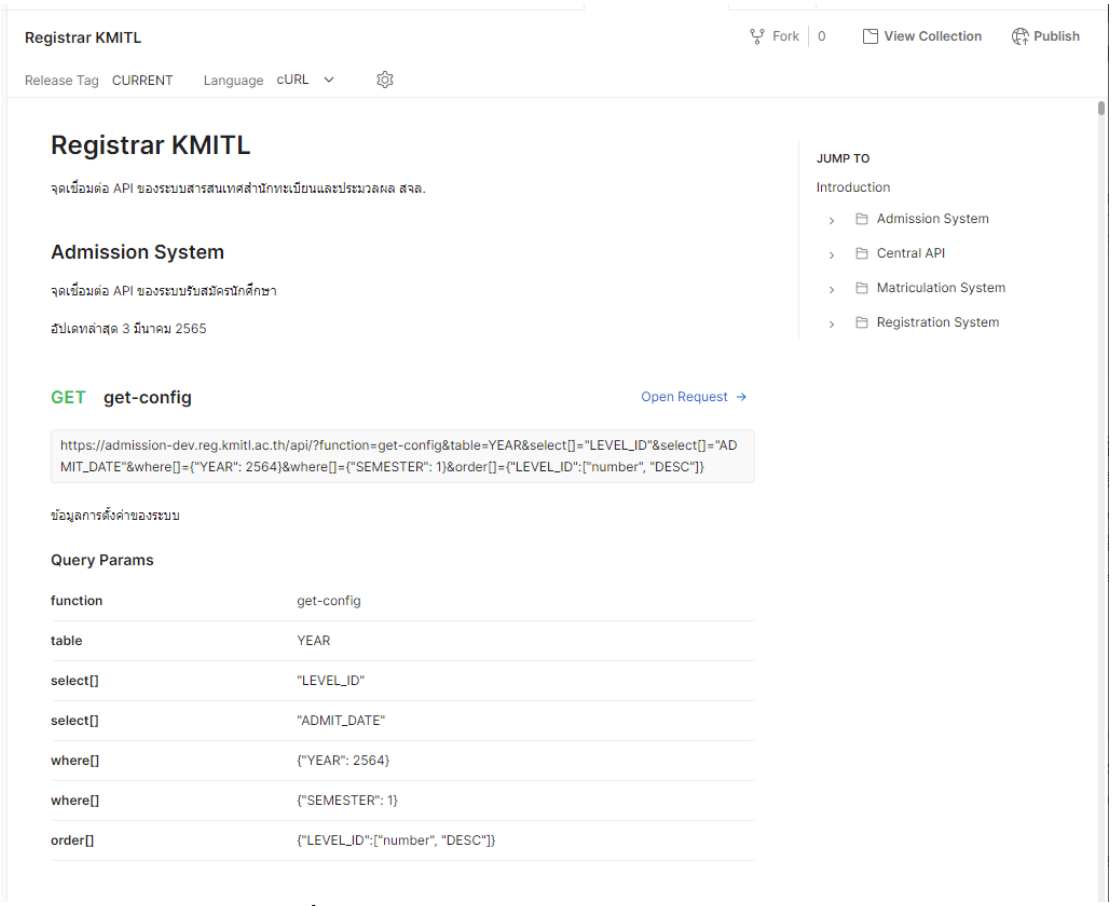


รูปที่ 4.9 ผลการทดสอบจุดเชื่อมต่อ

4.2.5 จัดทำเอกสาร Web API

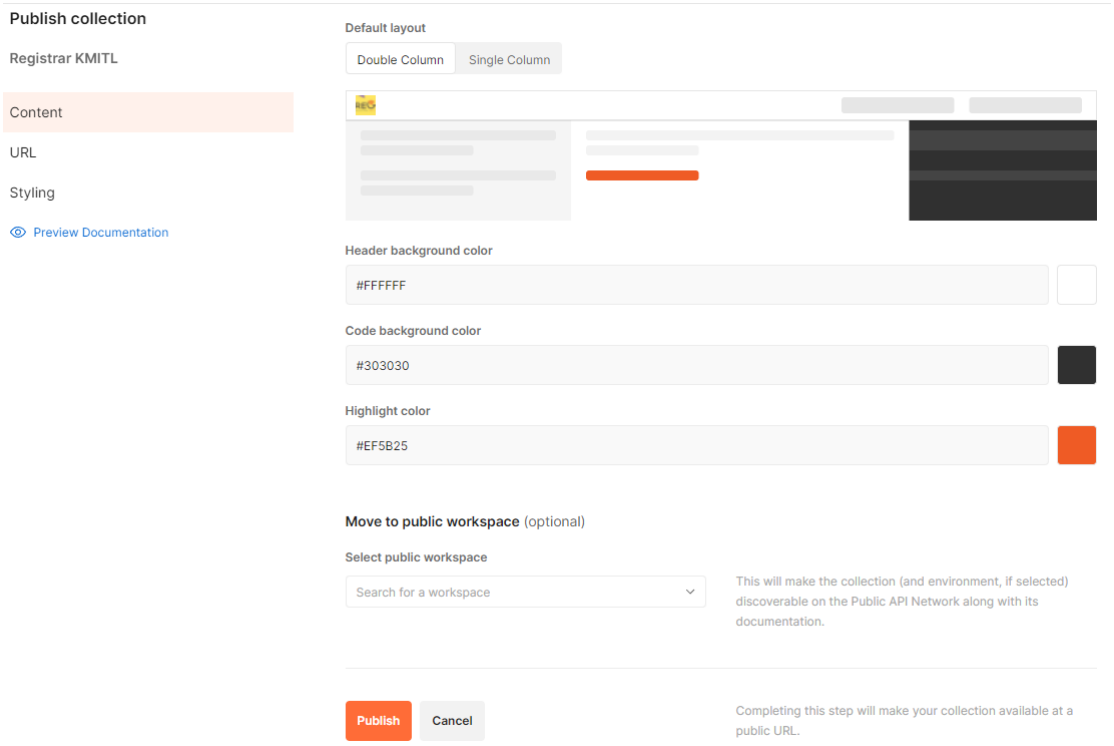
หลังจากสร้างจุดเชื่อมต่อทั้งหมดโดยสมบูรณ์ตามต้องการแล้ว กดที่เมนู Documentation จากนั้นใส่คำอธิบายรายละเอียดของส่วนต่างๆ ดังรูปที่ 4.10 ดังนี้

1. คำอธิบายรายละเอียดของเอกสาร Web API
2. คำอธิบายรายละเอียดของหมวดหมู่
3. คำอธิบายรายละเอียดของคำขอ
4. คำอธิบายรายละเอียดของพารามิเตอร์ต่างๆ

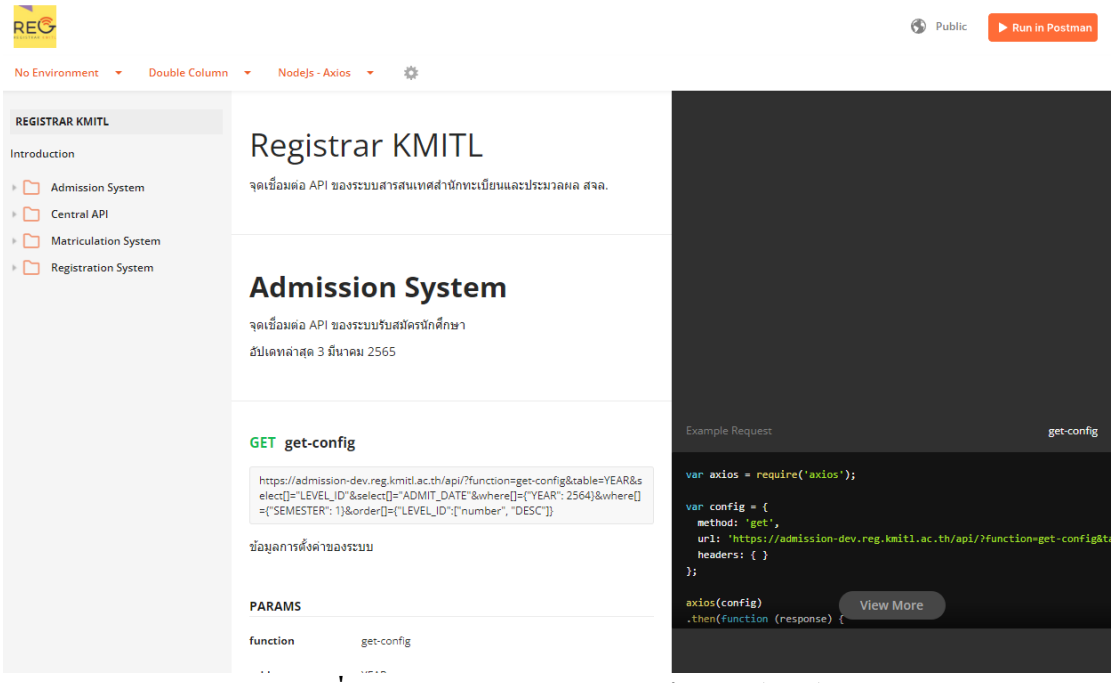


รูปที่ 4.10 เอกสาร Web API บนโปรแกรม Postman

เมื่อกรอกคำอธิบายรายละเอียดทั้งหมดโดยสมบูรณ์ตามต้องการแล้ว กดที่ปุ่ม Publish มุมบนขวาของโปรแกรม Postman ดังรูปที่ 4.11 เพื่อจัดทำเอกสารเป็นรูปแบบ Web Documentation ที่สามารถเข้าถึงได้ด้วยเว็บเบราว์เซอร์ต่อไป ดังรูปที่ 4.12



รูปที่ 4.11 ตั้งค่ารูปแบบเอกสาร Web API บนเว็บเบราว์เซอร์



รูปที่ 4.12 เอกสาร Web API บนเว็บเบราว์เซอร์

เอกสาร Web API ของระบบสารสนเทศสำนักทะเบียนและประมวลผล สจล. เข้าถึงได้ที่ลิงก์ https://documenter.getpostman.com/view/20533680/UyrHfYfr

4.3 การจัดทำคู่มือการออกแบบ พัฒนา และจัดทำเอกสาร Web API

หลังจากดำเนินการจัดทำเอกสาร Web API ที่สามารถใช้งานได้จริงของสำนักทะเบียนและประมวลผล สจล. แล้ว ต่อไปเป็นการจัดทำคู่มือการออกแบบ พัฒนา และจัดทำเอกสาร Web API เพื่อใช้เป็นมาตรฐานร่วมกัน ดังรูปที่ 4.13 สามารถให้หน่วยงานภายในสถาบัน ฯ ใช้คู่มือนี้เป็นมาตรฐานกลางร่วมกัน สามารถเข้าถึงคู่มือผ่านเว็บไซต์สำนักทะเบียนและประมวลผล สจล. ที่เมนูเอกสาร/ข้อมูล ได้ที่ลิงก์ https://reg.kmitl.ac.th/ โดยมีรายละเอียดของคู่มือ ดังนี้

1. บทนำ อธิบายถึง ประวัติความเป็นมา ปัญหาและการแก้ปัญหา
2. การออกแบบ Web API
3. การพัฒนา Web API
4. รูปแบบวิธีการตรวจสอบสิทธิ์ (Authentication Methods) ที่ใช้ใน Web API
5. การจัดทำเอกสาร Web API โดยแบ่งเป็น การสร้างคำขอ การจัดหมวดหมู่คำขอ การสร้างตัวอย่างของแต่ละคำขอ การทดสอบคำขอ และการจัดทำเอกสาร Web API

รูปที่ 4.13 หน้าปกคู่มือการออกแบบ พัฒนา และจัดทำเอกสาร Web API

4.4 ผลการทำแบบประเมิน

หลังจากจัดทำคู่มือการออกแบบ พัฒนา และจัดทำเอกสาร Web API เวอร์ชันแรกเสร็จสิ้นแล้ว จึงจัดทำแบบประเมินด้วย Google Forms สำหรับผู้บริหาร / อาจารย์ จำนวน 1 ท่านและผู้พัฒนาระบบจำนวน 4 ท่าน เพื่อทำการประเมินแบบประเมินวัดระดับ ข้อติชม และข้อเสนอแนะเพื่อนำมาปรับปรุงคู่มือนี้ต่อไป โดยมีผลลัพธ์การประเมิน ดังนี้

ตารางที่ 4.2 เกณฑ์การให้คะแนนของแบบประเมิน

เชิงคุณภาพ	เชิงปริมาณ	ความหมาย
มากที่สุด	4.51 – 5.00	คู่มือที่พัฒนามีระดับความพึงพอใจมากที่สุด
มาก	4.01 – 4.50	คู่มือที่พัฒนามีระดับความพึงพอใจมาก
ปานกลาง	3.51 – 4.00	คู่มือที่พัฒนามีระดับความพึงพอใจปานกลาง
น้อย	3.01 – 3.50	คู่มือที่พัฒนามีระดับความพึงพอใจน้อย
น้อยที่สุด	< 3.00	คู่มือที่พัฒนามีระดับความพึงพอใจน้อยที่สุด

หัวข้อที่ใช้ในการประเมินประสิทธิภาพของคู่มือ ตามรูปแบบในภาคผนวก ข มีดังนี้

1. ความเข้าใจของเนื้อหา
2. ความถูกต้องสมบูรณ์ของรูปแบบคู่มือ โดยภาพรวม
3. ความสามารถในการนำมาประยุกต์ใช้จัดทำเอกสาร Web API ได้จริง โดยภาพรวม
4. ข้อเสนอแนะ / ความคิดเห็น เกี่ยวกับคู่มือ
5. ข้อเสนอแนะ / ความคิดเห็น อื่นๆ

ตารางที่ 4.3 ผลการประเมินของคู่มือ

รายการประเมิน	ค่าเฉลี่ย	ระดับประเมิน
ความเข้าใจของเนื้อหา		
บท การออกแบบ Web API	5.00	มากที่สุด
บท การพัฒนา Web API	4.80	มากที่สุด
บท รูปแบบวิธีการตรวจสอบสิทธิ์ (Authentication Methods) ที่ใช้ใน Web API	5.00	มากที่สุด
บท การจัดทำเอกสาร Web API	4.80	มากที่สุด
ความถูกต้องสมบูรณ์ของรูปแบบคู่มือ โดยภาพรวม	4.80	มากที่สุด

ความสามารถในการนำมาประยุกต์ใช้จัดทำเอกสาร Web API ได้จริง โดยภาพรวม	5.00	มากที่สุด

ข้อเสนอแนะ / ความคิดเห็น เกี่ยวกับคู่มือ มีดังนี้

1. ทำออกมาค่อนข้างเข้าใจง่าย อ่านสบายตา ภาพประกอบเห็นแล้วเข้าใจตรงจุด
2. เป็นคู่มือที่มีประโยชน์และสามารถนำมาใช้ได้จริงสำหรับใช้ภายในหน่วยงานสำนักทะเบียนฯ และใช้เป็นข้อกำหนดรูปแบบการรับส่งข้อมูลที่เป็นมาตรฐานกับหน่วยงานภายนอก
3. ในส่วนของการจัดทำเอกสาร Web API หากผู้พัฒนาสามารถแบ่งเป็นลำดับขั้นตอนโดยละเอียด เป็นประโยชน์ต่อผู้ (เริ่มต้น) ในการนำคู่มือนี้ไปใช้งานต่อไป
4. เป็นประโยชน์ต่อการนำไปปรับใช้ในการพัฒนาระบบสารสนเทศของสำนักทะเบียนฯ

สร้างแบบประเมินด้วย Google Forms สามารถทำแบบประเมินผ่านลิงก์ https://forms.gle/tbUGXuV485WgrEKV6

บทที่ 5
สรุปผลการดำเนินงาน

จากการดำเนินงานทั้งหมดตามขอบเขตของโครงการตั้งแต่ ศึกษาประวัติความเป็นมา วิเคราะห์ปัญหา ค้นคว้าทฤษฎีและหลักการที่เกี่ยวข้อง จากนั้นทำการวิเคราะห์และออกแบบ เพื่อนำมาดำเนินงานตามที่ได้วางแผนไว้ สามารถสรุปผลการดำเนินงานทั้งหมดได้ดังนี้

5.1 ผลการดำเนินงาน

- ทราบถึงมาตรฐานการออกแบบและพัฒนาสำหรับ Web API ซึ่งเป็นความรู้พื้นฐานในการจัดทำเอกสาร Web API
- ได้จัดทำคู่มือการออกแบบ พัฒนา และจัดทำเอกสาร Web API ไว้เป็นคลังความรู้ของหน่วยงาน เพื่อให้ความรู้กับส่วนงานที่เกี่ยวข้องต่อไป
- ได้จัดทำเอกสาร Web API ของระบบสารสนเทศสำนักทะเบียนและประมวลผล สจล. ไว้เพื่อรวบรวมจุดเชื่อมต่อ และเป็นเอกสารประกอบสำหรับทำงานร่วมกับหน่วยงานอื่นๆ

5.2 ปัญหาและอุปสรรคของการดำเนินงาน

- เนื่องด้วยการจัดทำคู่มือและเอกสาร Web API เป็นช่วงเริ่มแรก ทำให้อาจยังไม่ทราบเงื่อนไขและข้อจำกัดได้อย่างครอบคลุม
- เนื่องด้วยระยะเวลาดำเนินงานค่อนข้างสั้น ทำให้กระบวนการทดสอบการใช้งานซอฟต์แวร์โดยผู้ใช้งานจริง (User Acceptance Test (UAT)) ทำได้ไม่เต็มที่

5.3 แนวทางในการแก้ไขปัญหา

- เก็บข้อมูล ปัญหา-ข้อเสนอแนะการใช้งานจากผู้ใช้งาน เพื่อนำไปวิเคราะห์และแก้ไขปัญหา
- ปรับปรุงคู่มือการออกแบบ พัฒนา และจัดทำเอกสาร Web API อย่างต่อเนื่อง เพื่อให้มีความสอดคล้องกับกระบวนการทำงานของหน่วยงาน

5.4 ข้อเสนอแนะและแนวทางในการพัฒนาเพิ่มเติม

- ค้นคว้าข้อมูลเกี่ยวกับ Web API เพิ่มเติม เพื่อนำมาเพิ่มประสิทธิภาพให้กับระบบและการจัดทำคู่มือ-เอกสาร Web API

- สนับสนุนให้หน่วยงานภายนอกจัดทำเอกสาร Web API เพื่อเพิ่มประสิทธิภาพการทำงานร่วมกันระหว่างหน่วยงาน

บรรณานุกรม

Bitfinex. (2564). **Bitfinex API documentation**. [Online] Available: https://docs.bitfinex.com/docs

European Commission. (2562). **Web Application Programming Interfaces (APIs): general-purpose standards, terms and European Commission initiatives**. Italy: European Commission. [Online] Available: https://publications.jrc.ec.europa.eu/repository/handle/JRC118082

Iterable. (2565). **API Overview and Sample Payloads**. [Online] Available: https://support.iterable.com/hc/en-us/articles/204780579-API-Overview-and-Sample-Payloads-

JWT. (2565). **Introduction to JSON Web Tokens**. [Online] Available: https://jwt.io/introduction

Lokesh Gupta. (2564). **HTTP Methods**. [Online] Available: https://restfulapi.net/http-methods

Maggie Summers. (2563). **API Keys: API Authentication Methods & Examples**. [Online] Available: https://blog.stoplight.io/api-keys-best-practices-to-authenticate-apis

Microsoft. (2565). **RESTful web API design**. [Online] Available: https://docs.microsoft.com/en-us/azure/architecture/best-practices/api-design

Microsoft. (2565). **Web API implementation.** [Online] Available: https://docs.microsoft.com/en-us/azure/architecture/best-practices/api-implementation

OECD. (2562). **Unlocking the Digital Economy - A guide to implementing application programme interfaces in Government**. Paris: OECD. [Online] Available: https://www.oecd.org/tax/forum-on-tax-administration/publications-and-products/unlocking-the-digital-economy-guide-to-implementing-application-programming-interfaces-in-government.htm

Postman. (2565). **Postman Documentation**. [Online] Available:
https://learning.postman.com/docs

RestCase. (2562). **4 Most Used REST API Authentication Methods**. [Online] Available:
https://blog.restcase.com/4-most-used-rest-api-authentication-methods/

Stoplight. (2565). **Types of APIs & Popular REST API Protocol**. [Online] Available:
https://stoplight.io/api-types

Wikipedia. (2565). **Hypertext Transfer Protocol**. [Online] Available:
https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol

ภาคผนวก ก
ตัวอย่าง Test Script ที่ใช้ในโปรแกรม Postman

ตัวอย่างสคริปท์สำหรับทดสอบคำตอบกลับ จากการส่งคำขอไปยังแพลตฟอร์มโดยคาดหวังว่าผลลัพธ์ของคำตอบกลับเป็นอย่างที่คาดหวัง

สคริปท์ทดสอบทั้งหมดสามารถศึกษาเพิ่มเติมได้ที่ลิงก์

https://learning.postman.com/docs/writing-scripts/script-references/test-examples/

ตัวอย่างสคริปท์ทดสอบสามารถศึกษาเพิ่มเติมได้ที่ลิงก์

https://www.postman.com/postman/workspace/test-examples-in-postman/

```
pm.test("Status code is 200", () => {
    pm.response.to.have.status(200);
});

pm.test("Content-Type header is application/json", () => {
    pm.expect(pm.response.headers.get('Content-
Type')).to.include('application/json');
});

pm.test("Test structure of the response", () => {
    var jsonData = pm.response.json();
    pm.expect(jsonData).to.be.an("object");
    pm.expect(jsonData.regis).to.be.an("object");
    pm.expect(jsonData.regis.before_date).to.be.a("boolean");
    pm.expect(jsonData.regis.on_date).to.be.a("boolean");
    pm.expect(jsonData.regis.after_date).to.be.a("boolean");
    pm.expect(jsonData.regis.start_date).to.match(/^\d{4}-\d{2}-
\d{2}\s\d{2}:\d{2}:\d{2}$/);
    pm.expect(jsonData.pre_regis.end_date).to.match(/^\d{4}-\d{2}-
\d{2}\s\d{2}:\d{2}:\d{2}$/);
});

pm.test("Response time is less than 200ms", () => {
    pm.expect(pm.response.responseTime).to.be.below(200);
});
```

ภาคผนวก ข

แบบประเมิน คู่มือการออกแบบ พัฒนา และจัดทำเอกสาร Web API

แบบประเมินสร้างด้วย Google Forms สามารถทำแบบประเมินผ่านลิงก์
https://forms.gle/tbUGXuV485WgrEKV6

แบบประเมินนี้ ใช้เพื่อเป็นส่วนหนึ่งของรายวิชา 06017772 Independent Study 2 ปีการศึกษา 2/2564 ขอความอนุเคราะห์ผู้ประเมินทำแบบประเมินนี้ เพื่อเป็นประโยชน์สำหรับการปรับปรุง-แก้ไขคู่มือต่อไป
ขันติชัย รุจิตระการโชติกุล รหัสนักศึกษา 63606041

ส่วนที่ 1 กลุ่มผู้ประเมิน *

- ◯ ผู้บริหาร / อาจารย์
- ◯ ผู้พัฒนาระบบ

ส่วนที่ 2 คำนำหน้าชื่อ ชื่อ นามสกุล ผู้ประเมิน (ภาษาไทย) *

_____

ส่วนที่ 3 ความเข้าใจของเนื้อหา *

1 = น้อยที่สุด 5 = มากที่สุด

	1	2	3	4	5	ไม่ประเมิน
บท การออกแบบ Web API	◯	◯	◯	◯	◯	◯
บท การพัฒนา Web API	◯	◯	◯	◯	◯	◯
บท รูปแบบวิธีการตรวจสอบสิทธิ์ (Authentication Methods) ที่ใช้ใน Web API	◯	◯	◯	◯	◯	◯
บท การจัดทำเอกสาร Web API	◯	◯	◯	◯	◯	◯

ส่วนที่ 4 ความถูกต้องสมบูรณ์ของรูปแบบคู่มือ โดยภาพรวม *

	1	2	3	4	5	
น้อยที่สุด	◯	◯	◯	◯	◯	มากที่สุด

ส่วนที่ 5 ความสามารถในการนำมาประยุกต์ใช้จัดทำเอกสาร Web API ได้จริง โดยภาพรวม *

	1	2	3	4	5	
น้อยที่สุด	◯	◯	◯	◯	◯	มากที่สุด

ส่วนที่ 6 ข้อเสนอแนะ / ความคิดเห็น เกี่ยวกับคู่มือ *

_____

_____

_____

ส่วนที่ 7 ข้อเสนอแนะ / ความคิดเห็น อื่นๆ

_____

_____

_____

ประวัติผู้เขียน

ชื่อ - นามสกุล	ขันติชัย รุจิตระการโชติกุล
วัน เดือน ปี เกิด	21 กันยายน 2538
ที่อยู่	488 ซอยพหลโยธิน 30 แขวงจันทรเกษม เขตจตุจักร กรุงเทพฯ 10900
ประวัติการศึกษา	2560 คณะวิศวกรรมศาสตร์ สาขาวิศวกรรมคอมพิวเตอร์
	สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ประสบการณ์การทำงาน	
พ.ศ. 2561 – ปัจจุบัน	สำนักทะเบียนและประมวลผล
	สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง