



การออกแบบและพัฒนาแพลตฟอร์มข้อมูลที่มีความหลากหลาย
Design and Development of Multitudinous Data Platform

สุนิสา จุฬรัตน์
Sunisa Junrat

วิทยานิพนธ์นี้สำหรับการศึกษาตามหลักสูตรปริญญา วิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์
มหาวิทยาลัยสงขลานครินทร์

A Thesis Submitted in Fulfillment of the Requirements for the Degree of Master
of Engineering in Computer Engineering
Prince of Songkla University

2567

ลิขสิทธิ์ของมหาวิทยาลัยสงขลานครินทร์



การออกแบบและพัฒนาแพลตฟอร์มข้อมูลที่มีความหลากหลาย
Design and Development of Multitudinous Data Platform

สุนิสา จุฬรัตน์
Sunisa Junrat

วิทยานิพนธ์นี้สำหรับการศึกษาตามหลักสูตรปริญญา วิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์
มหาวิทยาลัยสงขลานครินทร์

A Thesis Submitted in Fulfillment of the Requirements for the Degree of Master
of Engineering in Computer Engineering

Prince of Songkla University

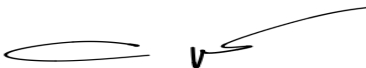
2567

ลิขสิทธิ์ของมหาวิทยาลัยสงขลานครินทร์

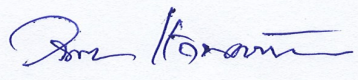
ชื่อวิทยานิพนธ์ การออกแบบและพัฒนาแพลตฟอร์มข้อมูลที่มีความหลากหลาย
ผู้เขียน นางสาวสุนิสา จุฬรัตน์
สาขาวิชา วิศวกรรมคอมพิวเตอร์

อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

คณะกรรมการสอบ

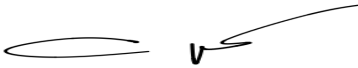

Digitally signed : 2024.10.29 21:38:11 +07:00

(ผู้ช่วยศาสตราจารย์ ดร.สุกญา เจริญปัญญาศักดิ์)


Digitally signed : 2024.10.30 16:01:45 +07:00

(ศาสตราจารย์ ดร.วนิดา แก่นอากาศ)

อาจารย์ที่ปรึกษาวิทยานิพนธ์ร่วม


Digitally signed : 2024.10.29 21:38:12 +07:00

(ผู้ช่วยศาสตราจารย์ ดร.สุกญา เจริญปัญญาศักดิ์)


Digitally signed : 2024.10.30 14:54:04 +07:00

(ผู้ช่วยศาสตราจารย์ ดร.วรรณรัช สันติอมรทัต)


Digitally signed : 2024.10.30 14:54:06 +07:00

(ผู้ช่วยศาสตราจารย์ ดร.วรรณรัช สันติอมรทัต)

บัณฑิตวิทยาลัย มหาวิทยาลัยสงขลานครินทร์ อนุมัติให้บัณฑิตวิทยานิพนธ์ฉบับนี้
สำหรับการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์


Digitally signed : 2024.10.31 14:52:32 +07:00

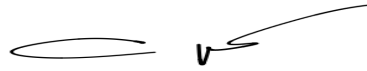
(ศาสตราจารย์ ดร.มิตรชัย จงเชื้อขำนาญ)

คณบดีบัณฑิตวิทยาลัย

(3)

ขอรับรองว่า ผลงานวิจัยนี้มาจากการศึกษาวิจัยของนักศึกษาเอง และได้แสดงความขอบคุณบุคคลที่มี
ส่วนช่วยเหลือแล้ว

ลงชื่อ



Digitally signed : 2024.10.29 21:38:14 +07:00

(ผู้ช่วยศาสตราจารย์ ดร.สุกษณา เจริญปัญญาศักดิ์)

อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

ลงชื่อ



Digitally signed : 2024.10.30 14:54:07 +07:00

(ผู้ช่วยศาสตราจารย์ ดร.วรรณรัช สันติอมรทัต)

อาจารย์ที่ปรึกษาวิทยานิพนธ์ร่วม

ลงชื่อ นางสาวสุนิสา จุลรัตน์

นักศึกษา

(4)

ข้าพเจ้าขอรับรองว่า ผลงานวิจัยนี้ไม่เคยเป็นส่วนหนึ่งในการอนุมัติปริญญาในระดับใดมาก่อน และ
ไม่ได้ถูกใช้ในการยื่นขออนุมัติปริญญาในขณะนี้

ลงชื่อ นางสาวสุนิสา จุรัตน์
นักศึกษา

ชื่อวิทยานิพนธ์	การออกแบบและพัฒนาแพลตฟอร์มข้อมูลที่มีความหลากหลาย
ผู้เขียน	นางสาวสุนิสา จุฬรัตน์
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
ปีการศึกษา	2567

บทคัดย่อ

ข้อมูลแต่ละชุมชนประกอบไปด้วยข้อมูลหลากหลายหมวดหมู่ หลากหลายชนิด ซึ่งแต่ละพื้นที่มีข้อมูลที่อาจเหมือนหรือแตกต่างกัน ขึ้นอยู่กับหลายปัจจัย เช่น สภาพพื้นที่ ภูมิอากาศ วิถีชีวิต วัฒนธรรม ประเพณี อาหารท้องถิ่น แหล่งพืชสมุนไพร องค์ความรู้ของคนในพื้นที่ ประเภทธุรกิจ ในชุมชน เป็นต้น ส่วนใหญ่ข้อมูลเหล่านี้ ยังไม่ได้ถูกจัดเก็บเป็นข้อมูลดิจิทัล ทำให้การวิเคราะห์ข้อมูลแต่ละชุมชน แต่ละเมือง ตำบล หรือจังหวัด จนไปถึงภาพรวมระดับประเทศ ยังทำได้ล่าช้าและไม่ครอบคลุม ข้อมูลบางส่วนซึ่งเคยถูกจัดเก็บลงระบบดิจิทัล ยังมีการแยกระบบจัดเก็บกันอย่างกระจัดกระจายและยังไม่เชื่อมโยงกัน ส่งผลให้ยังไม่สามารถบริหารจัดการข้อมูลร่วมกันได้ จึงควรจัดเก็บข้อมูลหลากหลายของแต่ละพื้นที่ให้อยู่ในรูปแบบดิจิทัลและสามารถนำมาวิเคราะห์ร่วมกันได้อย่างครอบคลุม ซึ่งสามารถเรียกดูข้อมูลและสืบค้นตามความต้องการได้ตลอดเวลา

วิทยานิพนธ์ฉบับนี้ จึงนำเสนอการออกแบบและพัฒนาระบบจัดการข้อมูลความหลากหลายของแต่ละชุมชน ซึ่งรองรับการจัดเก็บข้อมูลชนิดข้อความ ตัวเลข ตำแหน่งที่อยู่ วันเวลา รูปภาพ เอกสาร เป็นต้น โดยสามารถเชื่อมโยงข้อมูลจากระบบต่าง ๆ ที่มีอยู่แล้วผ่าน Application Programming Interface (API) ที่กำหนดได้ โดยระบบดำเนินการจัดเก็บข้อมูลดิบบนระบบ Google Cloud Firestore จัดเก็บไฟล์บนระบบ Google Cloud Storage และจัดเก็บข้อมูลสำหรับการสืบค้นอย่างรวดเร็วบนระบบ Elasticsearch นอกจากนั้นยังได้ออกแบบและพัฒนาระบบสำหรับบันทึกข้อมูลหลากหลายหมวดหมู่ของแต่ละชุมชน ซึ่งรองรับการสร้างโครงสร้างข้อมูลของแต่ละหมวดหมู่ได้โดยอัตโนมัติ เพื่ออำนวยความสะดวกให้ผู้ลงพื้นที่ของแต่ละชุมชนสามารถเก็บข้อมูลได้อย่างสะดวกมากยิ่งขึ้น การออกแบบและพัฒนาระบบจัดการข้อมูลหลากหลายนี้ จึงช่วยรวบรวมกลุ่มข้อมูลประเภทต่าง ๆ ของแต่ละพื้นที่ได้อย่างยืดหยุ่น ส่งผลให้นำไปประยุกต์ใช้ในการแก้ปัญหาและพัฒนาเมืองได้ตามต้องการ

คำสำคัญ : ข้อมูลหลากหลายชนิด, ระบบการจัดการข้อมูล, การเก็บข้อมูล, การวิเคราะห์ข้อมูล, การค้นหาข้อความ, ข้อมูลขนาดใหญ่

Thesis Title	Design and Development of Multitudinous Data Platform
Author	Miss Sunisa Junrat
Major Program	Computer Engineering
Academic Year	2024

ABSTRACT

Each community's data varies depending on factors such as condition, climate, way of life, culture, traditions, local knowledge, and types of businesses. Most of this data has not been stored digitally, causing delays and incomplete analyses. Even some digital data is scattered across separate, unconnected systems, making collective management difficult. Therefore, community data should be stored on a unified digital platform for seamless analysis, allowing on-demand browsing and searching.

This study proposes the design and development of a multi-data platform that supports various data types (text, numbers, locations, images, etc.) and integrates with existing systems via APIs. It stores raw data on Google Firebase Firestore, files on Google Cloud Storage, and retrieves data quickly via Elasticsearch. Additionally, the platform includes applications for data collection across communities, enabling automatic creation of data structures to simplify collection. This flexible system can be applied to support city development and problem-solving.

Keyword: Diversity Data, Data Platform, Data Collection, Data Analytics, Big Data

กิตติกรรมประกาศ

ขอขอบพระคุณ ผู้ช่วยศาสตราจารย์ ดร.สุกัญญา เจริญปัญญาศักดิ์ อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่ได้เสียสละเวลาในการให้คำปรึกษา พร้อมทั้งแนะนำแนวคิดในการทำวิทยานิพนธ์ รวมถึงการแนะนำวิธีการแก้ไขปัญหาลักษณะเกี่ยวกับการทำวิทยานิพนธ์ ตลอดจนตรวจสอบและแก้ไขวิทยานิพนธ์ให้ดำเนินไปอย่างลุล่วงและมีความสมบูรณ์

ขอขอบพระคุณ ผู้ช่วยศาสตราจารย์ ดร.วรรณรัช สันติอมรทัต อาจารย์ที่ปรึกษาวิทยานิพนธ์ร่วม ที่ได้กรุณาเสียสละเวลาในการให้คำปรึกษา ช่วยแนะนำแนวคิดและวิธีแก้ไขปัญหในการทำวิทยานิพนธ์ให้ดำเนินไปอย่างลุล่วง

ขอขอบพระคุณ ศาสตราจารย์ ดร.วนิดา แก่นอากาศ ที่ได้กรุณาเสียสละเวลาเป็นประธานกรรมการสอบวิทยานิพนธ์ ตลอดจนตรวจสอบและให้คำแนะนำอันเป็นประโยชน์ในการปรับปรุงวิทยานิพนธ์ฉบับนี้ให้มีความสมบูรณ์และดำเนินไปอย่างลุล่วง

ขอขอบพระคุณ คุณมงคล มโนพิรุฬห์พร คุณจิรัชศักดิ์ นพรัตน์ คณาจารย์ทุกท่าน คุณบงกช พฤกษ์พงษ์ ผู้คอยให้คำแนะนำเกี่ยวกับระเบียบและวิธีปฏิบัติเพื่อสำเร็จการศึกษา และเพื่อน ๆ ซึ่งได้สำเร็จการศึกษาระดับปริญญาโทและปริญญาเอกทุกท่าน ที่ได้กรุณาเสียสละเวลาให้คำปรึกษา และให้ความช่วยเหลือในด้านเทคนิคเฉพาะทาง เทคนิคการดำเนินงานวิจัย เป็นอย่างดีเสมอมา

และท้ายสุดนี้ ข้าพเจ้าขอโน้มรำลึกถึงพระคุณของบิดามารดาและครอบครัวที่ส่งเสริมและสนับสนุนข้าพเจ้าเสมอมาจนกระทั่งข้าพเจ้าสำเร็จการศึกษา

สุณิสา จุฬรัตน์

สารบัญ

หน้า

บทคัดย่อ	5
ABSTRACT	6
กิตติกรรมประกาศ	7
สารบัญ.....	8
บทที่ 1 บทนำ	1
1.1 ที่มาและความสำคัญ.....	1
1.2 วัตถุประสงค์	1
1.3 ขอบเขต	2
1.4 ขั้นตอนและวิธีการดำเนินงาน.....	2
1.5 ประโยชน์ที่คาดว่าจะได้รับ.....	2
บทที่ 2 เอกสารงานวิจัยที่เกี่ยวข้อง/วิธีการวิจัย.....	4
2.1 ตรวจสอบเอกสาร.....	4
2.2 ชนิดของข้อมูล	6
2.3 ชนิดของระบบจัดเก็บข้อมูล.....	8
2.4 วิธีการวิเคราะห์ข้อความ (Text Analyzer).....	13
บทที่ 3 วิธีดำเนินการวิจัย/วิธีการวิจัย	18
3.1 สถาปัตยกรรมของระบบจัดการข้อมูลความหลากหลาย.....	18
3.2 การออกแบบกระบวนการจัดการข้อมูล.....	19
3.3 รูปแบบโครงสร้างการเก็บข้อมูลแต่ละหมวดหมู่ (JSON Schema Definition)	29
3.4 เครื่องมือที่ใช้	32
3.5 ระบบอำนวยความสะดวกในการจัดการข้อมูลความหลากหลาย	34
3.6 การออกแบบโครงสร้างข้อมูลตามหมวดหมู่.....	39

3.7 วิธีการทดสอบประสิทธิภาพในการค้นหาด้วยคำหรือข้อความ.....	87
บทที่ 4 ผลการศึกษา/ผลการวิจัย	100
4.1 ผลการวิเคราะห์จากการทดลอง.....	100
4.2 ผลการวิเคราะห์สถิติการใช้งานระบบเก็บข้อมูลหลากหลาย.....	110
4.3 จำนวนผู้ใช้งานที่ระบบจัดการข้อมูลความหลากหลายรองรับ.....	114
บทที่ 5 สรุปผลการวิจัย/อภิปรายผล และข้อเสนอแนะ	116
5.1 สรุปผล.....	116
5.2 ข้อเสนอแนะ	116
บรรณานุกรม	117
ภาคผนวก	122
ภาคผนวก ก ผลงานเผยแพร่และตีพิมพ์.....	122
ประวัติผู้เขียน.....	133

บทที่ 1 บทนำ

1.1 ที่มาและความสำคัญ

ข้อมูลแต่ละชุมชนประกอบไปด้วยข้อมูลหลากหลายหมวดหมู่ ไม่ว่าจะเป็นข้อมูลพืช ท้องถิ่น อาหารประจำถิ่น ภูมิปัญญาท้องถิ่น แหล่งที่พัก ความเชี่ยวชาญของบุคคลในพื้นที่ สภาพ ภูมิอากาศและแหล่งน้ำ ข้อมูลเหล่านี้มีจำนวนมากและมีขนาดใหญ่มากขึ้นเรื่อย ๆ อีกทั้งข้อมูลส่วนใหญ่ยังไม่ได้รับการสำรวจและบันทึกลงบนระบบดิจิทัลที่มีการจัดเก็บข้อมูลหลากหลายชนิดที่มี ปริมาณเพิ่มขึ้นทุกขณะอย่างเหมาะสม เพื่อการสืบค้นตามช่วงเวลาที่น่าสนใจ นอกจากนี้ยังมีข้อมูล ชุมชนที่ถูกจัดเก็บกระจายอยู่บนระบบดิจิทัลอื่น ๆ ซึ่งยังไม่มีมีการเชื่อมโยงข้อมูลเหล่านั้นเข้า ด้วยกัน จึงยังคงเป็นอุปสรรคต่อการบริหารจัดการและวิเคราะห์ข้อมูลหลากหลายร่วมกัน

วิทยานิพนธ์ฉบับนี้ จึงนำเสนอการออกแบบและพัฒนาระบบจัดเก็บข้อมูลความ หลากหลายของแต่ละพื้นที่ ซึ่งรองรับชนิดข้อมูลรูปแบบข้อความ ตัวเลข ตำแหน่งที่อยู่ วันเวลา รูปภาพ เอกสาร เป็นต้น และสามารถเชื่อมโยงข้อมูลจากระบบต่าง ๆ ที่เคยบันทึกข้อมูลในรูปแบบ ดิจิทัลแล้วผ่าน APIs [1] ที่กำหนด โดยดำเนินการจัดเก็บข้อมูลดิบบนระบบ Google Cloud Firestore [2] จัดเก็บไฟล์บนระบบ Google Cloud Storage [3] และข้อมูลสำหรับการสืบค้นอย่าง รวดเร็วบนระบบ Elasticsearch [4] นอกจากนี้ยังได้ออกแบบและพัฒนาระบบสำหรับบันทึกข้อมูล หลากหลายหมวดหมู่ตามความต้องการแต่ละชุมชน ซึ่งรองรับการสร้างฐานข้อมูลหรือโครงสร้าง ข้อมูลของแต่ละหมวดหมู่โดยอัตโนมัติ เพื่ออำนวยความสะดวกให้กับผู้ลงสำรวจเก็บข้อมูล ซึ่งจะช่วย รวบรวมข้อมูลที่มีลักษณะเฉพาะถิ่น เพื่อนำมาวิเคราะห์และวางแผนบริหารจัดการเมืองต่อไป ข้อมูลที่ ผ่านการจัดเก็บลงสู่ระบบจัดเก็บข้อมูลความหลากหลายนี้ ผู้ใช้งานข้อมูล เช่น นักวิทยาศาสตร์ข้อมูล วิศวกรข้อมูล ผู้จัดการข้อมูลของแต่ละเมือง สามารถเลือกชุดข้อมูลที่สนใจ เพื่อนำมาวิเคราะห์ร่วมกับ ข้อมูลจากหลากหลายแหล่งได้ ซึ่งสามารถนำมาประยุกต์แก้ไขปัญหาและพัฒนาเมืองได้อย่าง เหมาะสม ดังเช่นตัวอย่าง การนำข้อมูลมาบริหารจัดการความเสี่ยงและภัยพิบัติ [5] การรวบรวมและ สรุปรความต้องการและความคาดหวังของประชาชนในเมืองลอนดอน [6] เป็นต้น

1.2 วัตถุประสงค์

1. การออกแบบและพัฒนาสถาปัตยกรรมของระบบจัดเก็บข้อมูลความหลากหลายของแต่ละ ชุมชน ซึ่งรองรับการค้นหาข้อมูลขนาดใหญ่ได้อย่างรวดเร็ว
2. ผู้ใช้งานข้อมูล สามารถสืบค้นชุดข้อมูลที่สนใจ เพื่อนำข้อมูลความหลากหลายไป วิเคราะห์และวางแผนการบริหารจัดการชุมชนได้ตามวัตถุประสงค์

1.3 ขอบเขต

วิทยานิพนธ์ฉบับนี้ ได้นำข้อมูลซึ่งได้มาจากการสำรวจภายใต้โครงการมหาวิทยาลัยสู่ตำบล (U2T) ตั้งแต่ มีนาคม 2564 ถึง สิงหาคม 2565

1.4 ขั้นตอนและวิธีการดำเนินงาน

1. รวบรวมลักษณะข้อมูลจากหลากหลายชุมชน เพื่อทำความเข้าใจถึงลักษณะหน้าตาของข้อมูลที่จะต้องนำมาจัดเก็บ
2. รวบรวมวรรณกรรมที่เกี่ยวข้องกับการออกแบบและพัฒนาโครงสร้างการจัดเก็บหรือสถาปัตยกรรมของการจัดการข้อมูลหลากหลาย
3. รวบรวมและศึกษาเครื่องมือที่เหมาะสมจะนำมาใช้พัฒนา
4. ออกแบบสถาปัตยกรรมการจัดการข้อมูลหลากหลาย
5. ออกแบบโครงสร้างการจัดเก็บข้อมูลหรือรูปแบบฐานข้อมูล
6. ออกแบบระบบอำนวยความสะดวกในการสร้างชุดคำถามและระบบอำนวยความสะดวกในการบันทึกข้อมูลแต่ละชุมชน
7. สร้างระบบฐานข้อมูลตามโครงสร้างข้อมูลที่ได้ออกแบบไว้
8. เตรียมเครื่องประมวลผลสำหรับรองรับการประมวลข้อมูลก่อนจัดเก็บลงฐานข้อมูล
9. สร้างระบบอำนวยความสะดวกในการสร้างชุดคำถามและระบบอำนวยความสะดวกในการบันทึกข้อมูลแต่ละชุมชน
10. ทดสอบสร้างฐานข้อมูลของหมวดหมู่ชุดคำถามสำหรับเก็บข้อมูลในชุมชนและทดสอบป้อนข้อมูลผ่านระบบบันทึกข้อมูลชุมชน
11. เปิดให้บุคคลอาสาสมัครตัวแทนของแต่ละชุมชนช่วยกันรวบรวมข้อมูลและบันทึกผ่านระบบบันทึกข้อมูลชุมชน
12. ทดสอบประสิทธิภาพความเร็วในการสืบค้นข้อมูลที่สนใจ
13. วิเคราะห์ผลการทดสอบ ปัญหาอุปสรรคที่พบ และสรุปผล

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1. นักวิจัยหรือนักพัฒนาสามารถนำรูปแบบโครงสร้างของระบบจัดการข้อมูลหลากหลายนี้เป็นแนวทางในการจัดการข้อมูลขนาดใหญ่ต่อไป

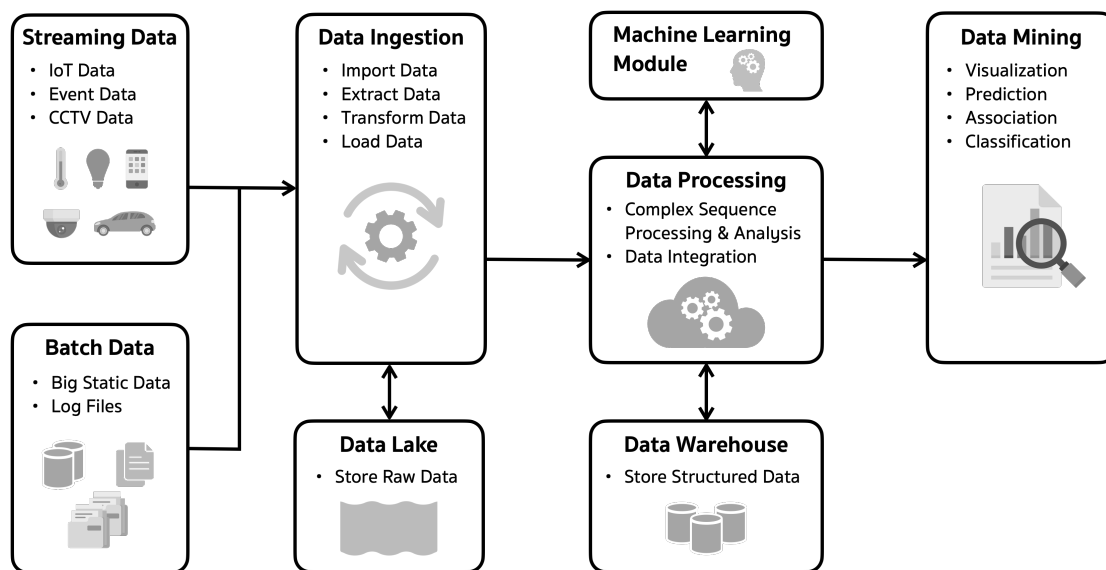
2. ผู้ใช้งานข้อมูลสามารถนำข้อมูลไปใช้ประโยชน์ เพื่อการวางแผนและบริหารจัดการ
สำหรับการพัฒนาชุมชน ตำบล หรือระดับประเทศได้

บทที่ 2 เอกสารงานวิจัยที่เกี่ยวข้อง/วิธีการวิจัย

2.1 ตรวจเอกสาร

การเก็บรวบรวมข้อมูลซึ่งมีปริมาณมากขึ้นเรื่อย ๆ จากหลากหลายแหล่ง มีความจำเป็นอย่างยิ่งและส่งผลกระทบต่อประสิทธิภาพของชุมชน เมือง และประเทศ นักวิจัยจึงพยายามรวบรวมข้อมูลเพื่อวิเคราะห์ปัญหาสำหรับการบริหารจัดการเมือง โดยพบว่าข้อมูลส่วนใหญ่ยังไม่ถูกแปลงสู่ระบบดิจิทัล และมีการเก็บข้อมูลกระจัดกระจายแยกออกจากกัน แต่ละเมืองจะมีวิธีการจัดเก็บข้อมูลเป็นของตนเอง จึงทำให้ไม่สามารถเชื่อมโยงและแลกเปลี่ยนข้อมูลระหว่างกันได้ หลายภาคส่วนมีการรวบรวมข้อมูลเดียวกันแต่ใช้วิธีการที่ต่างกัน โดยวิธีการจัดเก็บข้อมูลที่แตกต่างกันออกไปตั้งแต่การจัดเก็บในตาราง เอกสาร คอมพิวเตอร์ โดยปัญหาที่พบคือข้อมูลมีความซ้ำซ้อนและไม่สอดคล้องกัน จึงส่งผลให้ยังไม่สามารถเชื่อมโยงข้อมูลเพื่อสืบค้นและนำมาใช้ในการวางแผนบริหารจัดการเมืองทั้งหมดได้ จึงยังมีความต้องการเครื่องมือที่ช่วยเก็บข้อมูลความหลากหลายของแต่ละเมืองเพื่อสืบค้นร่วมกันและนำมาประยุกต์สำหรับการพัฒนาเมืองได้อย่างมีประสิทธิภาพ [7] โดยการเก็บรวบรวมข้อมูลในด้านอื่น ๆ เช่นด้านการเกษตร ได้มีผู้วิจัยการออกแบบและพัฒนาระบบเก็บข้อมูลด้านการเกษตรขนาดใหญ่ โดยได้รวบรวมข้อมูลจากหลากหลายแหล่ง เช่น เซนเซอร์ชนิดต่าง ๆ อุปกรณ์ระบุตำแหน่ง เครื่องมือทางการเกษตร ชาวนา และธุรกิจด้านการเกษตร เป็นต้น ซึ่งข้อมูลที่ได้มีหลากหลายชนิดข้อมูล มีขนาดใหญ่ และมีความซับซ้อน มีทั้งข้อมูลที่มีโครงสร้างและไร้โครงสร้าง จึงได้เลือกเครื่องมือที่เก็บข้อมูลได้อย่างยืดหยุ่น และสามารถอ่านเขียนข้อมูลได้รวดเร็ว จึงเลือกใช้เครื่องมือสำหรับจัดการคลังข้อมูล (Data Warehouse) [8] ประเภท NoSQL [9] โดยระบบที่ออกแบบสามารถสืบค้นข้อมูลเพื่อนำมาบริหารจัดการได้โดยมีประสิทธิภาพด้านความเร็วที่ดีกว่าฐานข้อมูลเชิงสัมพันธ์มาก โดยใช้การออกแบบโครงสร้างข้อมูลและการเชื่อมโยงความสัมพันธ์ระหว่างกันโดยผู้เชี่ยวชาญ โดยเฉพาะอย่างยิ่งไม่มีระบบช่วยอำนวยความสะดวกในการจัดทำโครงสร้างการจัดเก็บลงฐานข้อมูล และการสืบค้นข้อความจำเป็นต้องป้อนคำหรือข้อความที่มีความหมายจะทำให้สามารถสืบค้นข้อมูลออกมาได้ [10] นอกจากนั้นการเก็บข้อมูลสำหรับด้านการแพทย์ได้มีผู้ออกแบบและพัฒนาระบบจัดการข้อมูลเกี่ยวกับโรคมะเร็ง ซึ่งมีข้อมูลหลากหลายชนิดทั้งที่มีโครงสร้างและไม่มีโครงสร้าง โดยได้นำข้อมูลมาจากหลากหลายแหล่งซึ่งมีขนาดใหญ่มาจัดเก็บรวบรวมกัน เพื่อนำมาประมวลผลและแสดงรายงานให้บุคลากรผู้เกี่ยวข้องสามารถติดตามและวิเคราะห์ร่วมกันได้ โดยพบปัญหาข้อมูลบางส่วนยังมีความผิดพลาด เนื่องจากยังต้องป้อนเข้ามาด้วยมนุษย์ ซึ่งยังคงใช้เจ้าหน้าที่ในการตรวจสอบความถูกต้องด้วยตนเอง จึงทำให้เกิดความไม่สะดวกในประการนี้ ซึ่งส่งผลต่อการสืบค้นเพื่อนำข้อมูลไปใช้ประโยชน์ได้ไม่ครอบคลุม [11]

ด้านการเลือกใช้ระบบเก็บข้อมูลหรือเทคนิคที่เหมาะสมได้มีผู้ออกแบบและพัฒนา ระบบสำหรับวิเคราะห์การบริโภคภายในมหาวิทยาลัย โดยใช้ฐานข้อมูลชนิด NoSQL [9] เนื่องจากเจอปัญหาความเร็วในการวิเคราะห์ข้อมูลขนาดใหญ่บนฐานข้อมูลเชิงสัมพันธ์ (Relational Database) [12] ซึ่งใช้เวลานานในการประมวลผลและการค้นหาข้อมูล จึงได้นำฐานข้อมูลชนิด NoSQL [9] มาจัดเก็บข้อมูลการบริโภคแทน พบว่าสามารถประมวลผลการค้นหาข้อมูลและตอบสนองได้ไวกว่าฐานข้อมูลเชิงสัมพันธ์เป็นอย่างมาก และรองรับการขยายตัวของข้อมูลเป็นอย่างดี [13] ส่วนเทคนิคการสืบค้นให้ได้ผลลัพธ์ที่ถูกต้องสอดคล้องนั้นได้มีผู้วิจัยการออกแบบระบบสำหรับการค้นผลลัพธ์ด้วยภาษาพูดหรือภาษาที่ไม่เป็นทางการ โดยใช้การเก็บข้อมูลข้อมูลคำหรือข้อความ ซึ่งมีทั้งที่สะกดถูกต้องและสะกดผิด เพื่อทดสอบว่าหากใช้ภาษาพูดในการสืบค้นข้อมูลจะได้ผลลัพธ์ที่สอดคล้องตรงกันกับคำค้นหามากน้อยเพียงใด โดยผลลัพธ์ที่ได้จากการทดสอบยังไม่ค่อยถูกต้องและไม่สอดคล้องกับข้อมูลที่ต้องการค้นหา และมีหลายครั้งที่ทำการสืบค้นแล้วไม่ได้ผลลัพธ์ตอบสนองกลับมา [14] จึงได้ศึกษาสถาปัตยกรรมหรือระบบจัดการข้อมูลความหลากหลายขนาดใหญ่พบว่ามี การออกแบบสถาปัตยกรรมโดยแบ่งการจัดการออกเป็นสามส่วนหลัก ประกอบไปด้วยส่วนจัดเก็บ ข้อมูลขนาดใหญ่ ส่วนประมวลผล และส่วนจัดการการเข้าถึงข้อมูลที่มีการประมวลผลแล้ว โดยได้มีการสำรวจแหล่งข้อมูลขนาดใหญ่ เช่น Walmart, Amazon, Google และ Facebook มีปริมาณ ข้อมูลขนาดใหญ่ (PB) และมีการประมวลผลข้อมูลที่รวดเร็วเพื่อรองรับผู้ใช้จำนวนมาก ข้อมูลในระบบ เหล่านี้มีความแม่นยำสูงและรองรับการประมวลผลแบบกึ่งโครงสร้างหรือไม่มีโครงสร้างตามลักษณะ ของข้อมูล [15] และได้ศึกษาสถาปัตยกรรมที่มีการจัดการข้อมูลดิบและคลังข้อมูลสำหรับใช้งานเชิง ธุรกิจ โดยพบว่าการแปลงข้อมูลตามชนิดหรือหมวดหมู่ต่าง ๆ เพื่อจัดเก็บยัง Data Warehouse นั้น ต้องใช้ผู้เชี่ยวชาญและใช้เวลาในการจัดการ โดยยังไม่มีส่วนของแปลงข้อมูล เพื่อการค้นหาได้อย่าง รวดเร็ว [8] บางสถาปัตยกรรมได้ออกแบบเพื่อจัดเก็บข้อมูลหลากหลายรูปแบบทั้งข้อมูลดิบ ข้อมูลเชิง โครงสร้าง กึ่งโครงสร้าง และไม่มีโครงสร้าง โดยไม่มีการแปลงข้อมูลก่อนจัดเก็บ เมื่อต้องการใช้งาน กลุ่มข้อมูล จึงค่อยนำข้อมูลดิบเหล่านั้นมาประมวลผลตามความต้องการ โดยการค้นหาข้อมูลขึ้นอยู่กับ การจัดการส่วนอธิบายข้อมูล (Metadata) ซึ่งอาจทำให้ยังค้นหาได้ยากและไม่ครอบคลุม และใช้ เวลานานในการประมวลผลค้นหา [16] จากการศึกษาสถาปัตยกรรมการจัดการข้อมูลหลากหลาย ชนิด ซึ่งประกอบไปด้วยข้อมูลดิบ ข้อมูลที่ถูกจัดระเบียบตามโครงสร้างการเก็บข้อมูล ข้อมูลขนาดใหญ่ ข้อมูลที่มีการป้อนเข้าสู่ระบบอย่างต่อเนื่องตลอดเวลา [17] จึงได้นำมาออกแบบเป็นภาพรวม ของสถาปัตยกรรมการจัดการข้อมูลขนาดใหญ่ดังรูปที่ 1

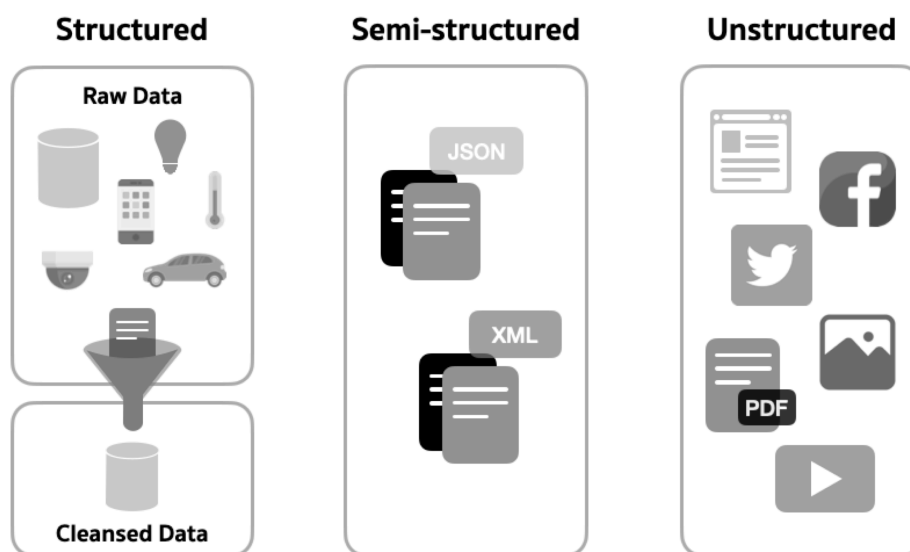


รูปที่ 1 ภาพรวมของสถาปัตยกรรมการจัดการข้อมูลขนาดใหญ่

จากรูปที่ 1 แสดงภาพรวมของสถาปัตยกรรมการจัดการข้อมูลขนาดใหญ่ ซึ่งมีส่วนประกอบหลักดังต่อไปนี้ คือ ชนิดของข้อมูลที่ได้แบ่งออกเป็นสองกลุ่มใหญ่ประกอบไปด้วย กลุ่มข้อมูลขนาดใหญ่ (Batch Data) และข้อมูลที่มีการป้อนเข้าสู่ระบบอย่างต่อเนื่องเวลา (Streaming Data) โดยจะมีส่วนที่คอยรับข้อมูลทั้งสองกลุ่มนี้เข้ามาประมวลผลและแปลงเพื่อจัดเก็บอย่างเหมาะสม (Data Ingestion) โดยไฟล์ข้อมูลดิบจะถูกส่งไปจัดเก็บยัง Data Lake และข้อมูลส่วนที่ได้รับการประมวลผลผ่านส่วน Data Processing และแปลงข้อมูลตามโครงสร้างที่กำหนดแล้วจะถูกจัดเก็บลงในคลังข้อมูล (Data Warehouse) โดยข้อมูลทั้งหมดที่ได้รับการจัดเก็บผ่านระบบรูปแบบดังกล่าว สามารถนำไปวิเคราะห์และประมวลผลในเชิงธุรกิจตามความต้องการผ่านส่วน Data Mining หรือนำกลุ่มข้อมูลไปวิเคราะห์เพื่อพัฒนาโมเดลด้านปัญญาประดิษฐ์ (Machine Learning Module) เพื่อการประยุกต์ใช้งานข้อมูลอย่างมีประสิทธิภาพมากที่สุด

2.2 ชนิดของข้อมูล

จากการศึกษาชนิดของข้อมูล สามารถจัดแบ่งกลุ่มของข้อมูลออกเป็นสามชนิดหลักดังต่อไปนี้



รูปที่ 2 ชนิดของข้อมูล

จากรูปที่ 2 แสดงชนิดของข้อมูล ซึ่งจัดแบ่งเป็นกลุ่มใหญ่ อันประกอบไปด้วย ข้อมูลที่มีโครงสร้าง (Structured Data) [18] ข้อมูลกึ่งโครงสร้าง (Semi-structured Data) [18] และข้อมูลแบบไร้โครงสร้าง (Unstructured Data) [18] ซึ่งข้อมูลแต่ละรูปแบบนี้ จำเป็นต้องออกแบบวิธีจัดเก็บข้อมูลอย่างเหมาะสม เพื่อการอ่าน การเขียน การค้นหาข้อมูล กลับมาใช้งานได้อย่างมีประสิทธิภาพ โดยชนิดข้อมูลเหล่านี้มีความหมายและลักษณะข้อมูลดังต่อไปนี้

ข้อมูลที่มีโครงสร้าง (Structured Data) [18] คือ ข้อมูลที่มีรูปแบบชัดเจน ช่วยให้ระบบและผู้ใช้สามารถเข้าถึงได้อย่างมีประสิทธิภาพ รูปแบบของข้อมูลมักเป็นตาราง ซึ่งมีแถวและคอลัมน์ในจำนวนที่แน่นอน

ข้อมูลกึ่งโครงสร้าง (Semi-structured Data) [18] คือ ข้อมูลที่มีโครงสร้างการเก็บข้อมูลไม่เหมือนกันได้ รูปแบบข้อมูลอาจเป็นชนิด XML [19] หรือ JSON [19] เป็นต้น

ข้อมูลแบบไร้โครงสร้าง (Unstructured Data) [18] คือ ข้อมูลที่ไม่มีโครงสร้างข้อมูล มักเป็นรูปภาพ วิดีโอ ไฟล์เอกสาร ข้อมูลสื่อสังคมออนไลน์ เป็นต้น

การจัดเก็บข้อมูลซึ่งมีความแตกต่างกันไปในแต่ละชุมชน อาจใช้วิธีการจัดแบ่งหมวดหมู่ของข้อมูลเพื่อจัดเก็บชุดข้อมูลแต่ละสาขา โดยมีการจัดเก็บชนิดของข้อมูลเบื้องต้นซึ่งประกอบไปด้วย ตัวเลข วันที่และเวลา พิกัดตำแหน่ง ข้อความ โดยข้อมูลเหล่านี้ เหมาะสำหรับการจัดเก็บในลักษณะโครงสร้างข้อมูลแบบกึ่งโครงสร้าง (Semi-structured Data) [18] ซึ่งสามารถจัดเก็บข้อมูลได้อย่างยืดหยุ่น โดยแต่ละหมวดหมู่ข้อมูลไม่จำเป็นต้องมีลักษณะข้อมูลที่เหมือนกัน ซึ่งข้อมูลกึ่งโครงสร้างนี้ สามารถอยู่ในเอกสารรูปแบบ XML [19] หรือ JSON [19] ซึ่งเหมาะกับการใช้

งานฐานข้อมูลชนิด NoSQL [9] โดยการอธิบายลักษณะของระบบจัดเก็บข้อมูลชนิดต่าง ๆ จะกล่าวถึงในหัวข้อถัดไป

2.3 ชนิดของระบบจัดเก็บข้อมูล

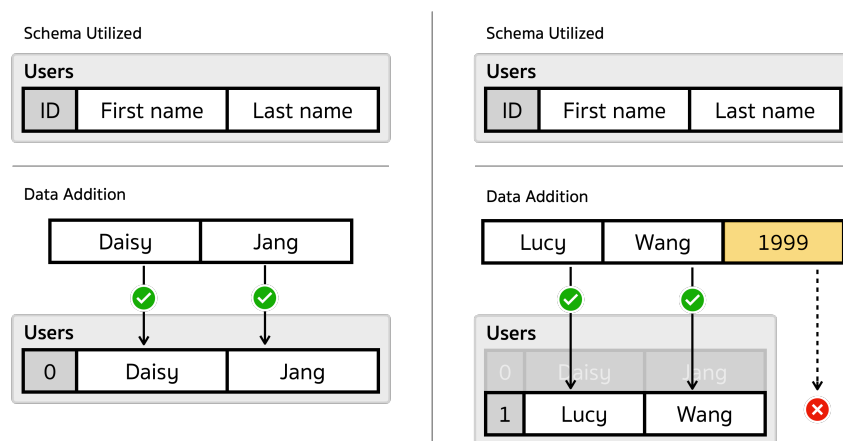
ระบบจัดเก็บข้อมูลสามารถแบ่งออกเป็นสองชนิดหลัก ซึ่งประกอบไปด้วย Relational Database [12] และ NoSQL [9] โดยแต่ละชนิดมีความหมายดังต่อไปนี้

Relational Database [12] หมายถึง ฐานข้อมูลเชิงสัมพันธ์ ซึ่งจัดเก็บข้อมูลอย่างเป็นระบบ มีโครงสร้างข้อมูลที่แน่นอน โดยรูปแบบมักจะเป็นลักษณะตาราง ซึ่งประกอบไปด้วยแถวและคอลัมน์ โดยคอลัมน์จะมีการกำหนดชื่อและชนิดของข้อมูลที่จะบันทึกลงไป ว่าเป็นชนิดข้อความ (String) หรือตัวเลข (Number) พร้อมกำหนดขนาดความที่จำกัด หรือเป็นชนิดวันที่ (Date) เป็นต้น โดยมีระบบเข้ามาช่วยจัดการฐานข้อมูลเชิงสัมพันธ์นี้ ซึ่งมีชื่อเรียกว่า RDBMS ย่อมาจาก Relational Database Management System ซึ่งจะใช้ภาษา SQL ในการจัดการเป็นส่วนใหญ่

NoSQL (Non-SQL) [9] หมายถึง ระบบฐานข้อมูลที่ไม่ใช่ SQL (Structured Query Language) ซึ่งมีการเก็บข้อมูลในรูปแบบอื่น ๆ ที่ไม่ใช่ตาราง โดยถูกออกแบบมาให้รองรับกับการอ่านข้อมูลปริมาณมาก โดยรองรับข้อมูลประเภท Semi-structured และ Unstructured นอกจากนั้นยังรองรับการขยายตัวของข้อมูลได้โดยไม่กระทบกับโครงสร้างการเก็บข้อมูล NoSQL [9] สามารถแบ่งเป็นประเภทใหญ่ 4 ประเภท ซึ่งประกอบไปด้วย

1. Document Database มีโครงสร้างการเก็บข้อมูลในรูปแบบ JSON [19]
2. Key-value Store มีการเก็บข้อมูลในรูปแบบความสัมพันธ์ของ Key และ Value ซึ่งการใช้งานทำได้โดยระบุชื่อ Key จะได้ค่า Value ที่ต้องการทันที มีความรวดเร็วและใช้งานง่ายแม้จะมีข้อมูลจำนวนมาก
3. Wide-column Store เป็นการเก็บข้อมูลในรูปแบบตารางซึ่งมีความยืดหยุ่นมากกว่าฐานข้อมูลเชิงสัมพันธ์ เนื่องจากแต่ละแถวไม่จำเป็นต้องมีจำนวนคอลัมน์ที่เท่ากัน และรองรับการเก็บข้อมูลปริมาณมาก
4. Graph Database เป็นการเก็บข้อมูลในรูปแบบของ Node และ Edge ซึ่งแต่ละ Node จะเก็บค่าต่าง ๆ ที่ต้องการ ส่วนของ Edge จัดเก็บความสัมพันธ์ระหว่าง Node ต่าง ๆ จึงเหมาะกับการเก็บข้อมูลที่มีความสัมพันธ์เกี่ยวข้องกันจำนวนมาก

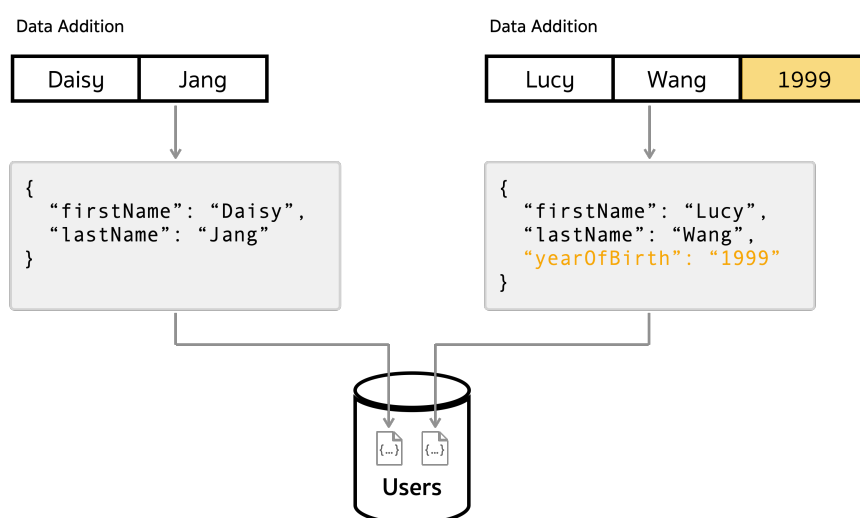
ลักษณะโครงสร้างการจัดเก็บข้อมูลของ Relational Database [12] หรือฐานข้อมูลเชิงสัมพันธ์ สามารถแสดงได้ดังรูปที่ 3



รูปที่ 3 ลักษณะการจัดเก็บข้อมูลของ Relational Database [12]

จากรูปที่ 3 แสดงลักษณะโครงสร้างการเก็บข้อมูลของ Relational Database [12] ซึ่งจากรูปในครึ่งซ้ายแสดงให้เห็นโครงสร้างข้อมูล ซึ่งประกอบไปด้วยตารางข้อมูลชื่อ Users ซึ่งมีคอลัมน์สำหรับเก็บข้อมูลชื่อ ID (รหัสที่ไม่ซ้ำกันของแต่ละแถว) First name (ชื่อจริง) และ Last name (สกุล) ทั้งหมดสามคอลัมน์ เมื่อมีการเพิ่มข้อมูล User ใหม่ โดยระบุค่า First name เป็น Daisy และ Last name เป็น Jang จึงสามารถบันทึกผลได้สำเร็จ ส่วนรูปในครึ่งขวาแสดงให้เห็นการเพิ่มข้อมูล User โดยระบุค่า First name เป็น Lucy ค่า Last name เป็น Wang และระบุคอลัมน์เพิ่มเติมเป็น 1999 เพิ่มอีกหนึ่งช่อง ผลลัพธ์คือไม่สามารถบันทึกค่า 1999 ได้

ลักษณะโครงสร้างการจัดเก็บข้อมูลของ NoSQL [9] ในรูปแบบของ JSON [19] สามารถแสดงได้ดังรูปที่ 4



รูปที่ 4 ลักษณะการจัดเก็บข้อมูลของ NoSQL [9]

จากรูปที่ 4 แสดงลักษณะโครงสร้างการจัดเก็บข้อมูลของ NoSQL [9] ในรูปแบบ JSON [19] ซึ่งแสดงให้เห็นว่าไม่ว่าข้อมูลในแต่ละ Object [20] จะมีกี่หัวข้อการเก็บข้อมูล (Property) ก็สามารถจัดเก็บลงฐานข้อมูล NoSQL [9] ได้โดยไม่กระทบกับโครงสร้างข้อมูลส่วนใด

นอกเหนือจากชนิดข้อมูลซึ่งเป็นข้อความ ตัวเลข วันที่ หรือพิกัดตำแหน่งแล้ว ยังมีข้อมูลในรูปแบบไฟล์ ซึ่งอาจเป็นรูปภาพ วิดีโอ หรือไฟล์เอกสารหลากหลายชนิด รวมถึงระบบจัดเรียงข้อมูลเพื่อการสืบค้นอย่างรวดเร็ว เมื่อพิจารณาตามชนิดข้อมูลทั้งหมดนี้ สามารถจัดแบ่งประเภทของระบบจัดเก็บข้อมูลพร้อมยกตัวอย่างระบบฐานข้อมูลที่เกี่ยวข้อง ดังรายละเอียดต่อไปนี้

1. RDBMS [12]

ระบบสำหรับการจัดการฐานข้อมูลเชิงสัมพันธ์ (RDBMS) ซึ่งใช้จัดการข้อมูลที่มีโครงสร้างแน่นอน มีดังต่อไปนี้

- MySQL [21] เป็นระบบจัดการฐานข้อมูลเชิงสัมพันธ์แบบ Open source [22] ซึ่งได้รับความนิยม สามารถเรียนรู้การใช้งานได้ง่าย แต่ไม่เหมาะกับระบบที่มีการเก็บข้อมูลขนาดใหญ่ หรือมีปริมาณข้อมูลเข้ามาจำนวนมากอย่างต่อเนื่อง
- POSTGRESQL [23] เป็นระบบจัดการฐานข้อมูลเชิงสัมพันธ์แบบ Open source [22] โดยมีจุดเด่นที่การตั้งค่าการเชื่อมโยงความสัมพันธ์ไม่ซับซ้อน แต่ประสิทธิภาพยังช้ากว่า MySQL [21]
- SQLite [24] เป็นระบบจัดการฐานข้อมูลเชิงสัมพันธ์แบบ Open source [22] ถูกพัฒนาให้สามารถเก็บข้อมูลได้โดยไม่ต้องเชื่อมต่อกับฐานข้อมูลกับเครื่องประมวลผล (Server) โดยสามารถเก็บข้อมูลได้ในไฟล์เดียว
- SQL Server [25] เป็นระบบจัดการฐานข้อมูลเชิงสัมพันธ์แบบ Closed source [26] โดยเน้นให้บริการบริษัทรายใหญ่ และมีค่าใช้จ่ายสูง
- Oracle Database [27] เป็นระบบจัดการฐานข้อมูลเชิงสัมพันธ์แบบ Closed source [26] เช่นเดียวกับ SQL Server โดยเน้นให้บริการบริษัทรายใหญ่ เช่นเดียวกัน และมีค่าใช้จ่ายสูง
- Amazon RDS หรือ Amazon Relational Database Service [28] เป็นบริการฐานข้อมูลเชิงสัมพันธ์ในรูปแบบ Web Service ซึ่งอำนวยความสะดวกในการจัดการ การปรับขนาดตามความต้องการ

2. NoSQL [9]

ระบบซึ่งมีการเก็บข้อมูลในรูปแบบอื่น ๆ ที่ไม่ใช่ตาราง โดยถูกออกแบบมาให้รองรับกับการอ่านข้อมูลปริมาณมาก โดยรองรับข้อมูลประเภท Semi-structured และ Unstructured มีดังต่อไปนี้

- Redis [29] เป็นระบบจัดการฐานข้อมูลในหน่วยความจำ (In-memory Database) ซึ่งเป็น Open source [22] และมีความเร็วสูง โดยลักษณะรูปแบบการจัดเก็บเป็น Key-value ซึ่งสามารถตอบสนองได้เร็วระดับมิลลิวินาที มีการทำงานอยู่ในหน่วยความจำแบบชั่วคราว (RAM) [30] จึงมักนำไปใช้สำหรับเก็บข้อมูลชั่วคราว
- Amazon DynamoDB [31] เป็นระบบจัดการฐานข้อมูลที่มีรูปแบบการจัดเก็บลักษณะ Key-value ที่มีการจัดการอย่างเต็มรูปแบบ โดยไม่จำเป็นต้องติดตั้งเครื่องประมวลผลเอง (Server less) ซึ่งมีคุณสมบัติอย่างการเก็บข้อมูลในหน่วยความจำ การประมวลผลแบบ Real-time สามารถขยายพื้นที่การเก็บข้อมูลได้โดยไม่กระทบโครงสร้างข้อมูล
- Cassandra [32] เป็นระบบจัดการฐานข้อมูลแบบ Wide-column ซึ่งเป็น Open source [22] โดยมีคุณสมบัติเด่นคือ รองรับการขยายตัวของข้อมูลสูง (High Scalability) และมีการกระจายฐานการเก็บข้อมูลอย่างมีประสิทธิภาพ (Distributed) และมีความทนทานสูง (Fault Tolerance)
- MongoDB [33] เป็นระบบจัดการฐานข้อมูลรูปแบบ Document-based ซึ่งเป็น Open source [22] โดยเก็บข้อมูลในรูปแบบ JSON [19] และรองรับการขยายของข้อมูลขนาดใหญ่
- CouchDB [34] เป็นระบบจัดการฐานข้อมูลรูปแบบ Document-based ซึ่งเป็น Open source [22] โดยเก็บข้อมูลในรูปแบบ JSON [19] และรองรับการขยายของข้อมูลปริมาณมาก
- Neo4j หรือ Native Graph Database [35] ซึ่งเป็น Open source [22] โดยมีการเก็บความสัมพันธ์ระหว่างข้อมูล ซึ่งสามารถสืบค้นได้

3. Cloud Storage [36]

ระบบสำหรับจัดการไฟล์รูปภาพ วิดีโอ เอกสารชนิดต่าง ๆ หรือใช้จัดเก็บข้อมูลแบบไร้โครงสร้าง (Unstructured Data) มีดังต่อไปนี้

- Amazon S3 [37] ย่อมาจาก Simple Storage Service เป็นบริการเก็บข้อมูลแบบวัตถุ (Object Storage) บนคลาวด์ของ Amazon ซึ่งเป็นการเก็บวัตถุเป็นชิ้นหรือเป็นไฟล์ตามต้องการ โดยรองรับข้อมูลจำนวนมาก (Scalability)
- Azure Storage [38] เป็นบริการเก็บข้อมูลบนคลาวด์ของ Microsoft ซึ่งจัดเก็บไฟล์แบบไม่มีโครงสร้างแน่นอน (Blob Storage) ขนาดใหญ่ได้ โดยรองรับการขยายตัวของข้อมูล และมีความทนทานต่อความผิดพลาด
- Google Cloud Firestore [3] เป็นบริการจัดเก็บข้อมูลบนคลาวด์ของ Google ซึ่งสามารถจัดเก็บไฟล์ไม่มีโครงสร้างต่าง ๆ ได้ตามต้องการ (Object Storage) โดยสามารถจัดเก็บข้อมูลได้อย่างยืดหยุ่น (Schema-less) และเหมาะจะใช้งานกับระบบที่เน้นการอัปเดตข้อมูลแบบ Real-time โดยรองรับการขยายขนาดการเก็บข้อมูลให้อัตโนมัติตามการใช้งานจริง

4. Full-text Search Database [39]

ระบบซึ่งถูกออกแบบสำหรับการค้นหาข้อความซึ่งมีความซับซ้อนได้ (Full-text Search) ได้อย่างรวดเร็ว มีดังต่อไปนี้

- Apache Solr [40] เป็นแพลตฟอร์มสำหรับให้บริการการค้นหาข้อความเต็มรูปแบบ โดยสามารถค้นหาข้อมูลที่เก็บไว้ในระบบจัดการฐานข้อมูลขนาดใหญ่ (HDFS) [41] ได้อย่างรวดเร็ว โดยรองรับการขยายตัวของข้อมูลและทนทานต่อความผิดพลาดสูง
- MySQL Full-Text Search [42] เป็นระบบเพิ่มความสามารถให้กับ MySQL [21] ซึ่งเพิ่มประสิทธิภาพโดยการเรียงลำดับข้อมูล (Index) [43] ในคอลัมน์ที่ต้องการค้นหา ซึ่งส่งผลให้ค้นหาข้อมูลได้รวดเร็วยิ่งขึ้น
- Elasticsearch [4] เป็นได้ทั้งฐานข้อมูล (Database) และระบบสืบค้นข้อมูล (Search Engine) ในเวลาเดียวกัน ซึ่งสามารถใช้เก็บข้อมูลในปริมาณมาก โดยการกระจายข้อมูลไปยังหลายเครื่องได้ ซึ่งจัดเก็บในรูปแบบ JSON Document และผู้ใช้งานสามารถค้นหาด้วยคำง่าย ๆ ไปจนถึงข้อความที่ซับซ้อนโดยใช้เวลารวดเร็ว

5. Data Analytics Tools [44]

เมื่อได้เลือกกระบวนจัดการฐานข้อมูล เพื่อจัดเก็บชนิดข้อมูลที่ต้องการอย่างเหมาะสมแล้ว หากต้องการเข้าถึงข้อมูลเชิงลึก ข้อมูลเชิงสถิติ ข้อมูลเชิงธุรกิจ หรือข้อสรุปเพื่อการบริหาร

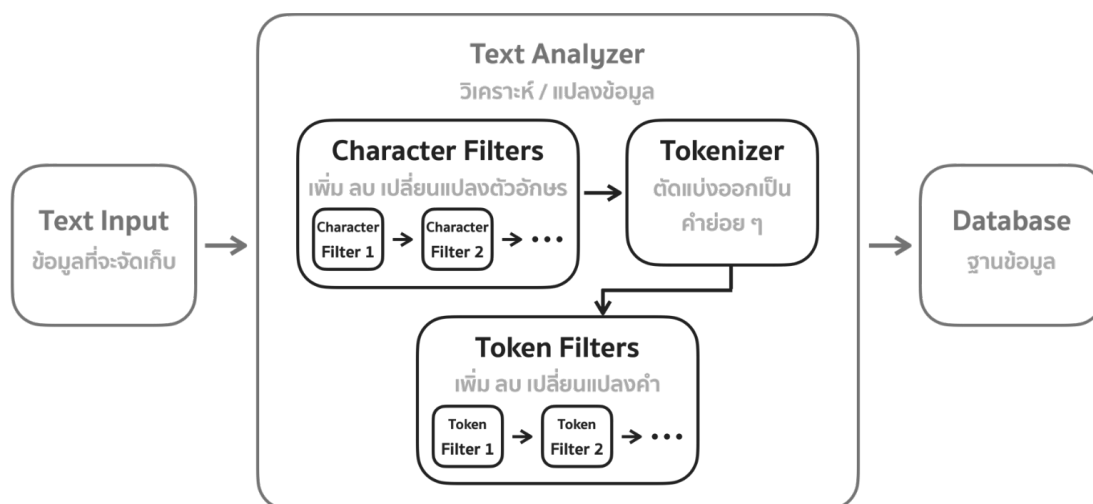
จำเป็นต้องมีระบบช่วยวิเคราะห์เชิงสถิติและแสดงสรุปข้อมูลตามช่วงเวลาที่น่าสนใจ (Data Analytics Tools หรือ Data Visualization Tools) ดังต่อไปนี้

- Microsoft Power BI [45] เป็นเครื่องมือสำหรับวิเคราะห์ข้อมูลธุรกิจแบบโต้ตอบได้ (Interactive Data Visualization) จากบริษัท Microsoft [46] การสร้างรายงานหรือการสร้างกระดานสถิติ (Dashboard) ซึ่งรองรับการสร้างส่วนแสดงผลได้หลากหลายรูปแบบ โดยเลือกกรอง ระบุค่าค้นหา ระบุช่วงเวลาที่น่าสนใจได้ โดยจำเป็นต้องติดตั้งเพื่อใช้งานบนเครื่องคอมพิวเตอร์ด้วยตนเอง (On premise) [47]
- Tableau [48] เป็นเครื่องมือสำหรับวิเคราะห์ข้อมูลธุรกิจแบบโต้ตอบได้ (Interactive Data Visualization) สามารถใช้งานแบบลากวางซึ่งเรียนรู้ได้ง่าย โดยมีค่าใช้จ่ายที่แพงกว่า Power BI [45] สามารถเลือกใช้งานได้แบบ On premise และผ่านคลาวด์
- Grafana [49] เป็นเครื่องมือวิเคราะห์ข้อมูลรูปแบบ Time Series [50] เพื่อสรุปข้อมูลเชิงสถิติสำหรับการตัดสินใจ (Interactive Data Visualization) เพื่อช่วยในการบริหารจัดการตามต้องการ
- Kibana [51] เป็นเครื่องมือสำหรับวิเคราะห์ข้อมูลจาก Elasticsearch [4] เพื่อแสดงผลสรุปในหลากหลายรูปแบบ (Interactive Data Visualization)

2.4 วิธีการวิเคราะห์ข้อความ (Text Analyzer)

เนื่องจากข้อมูลส่วนใหญ่ถูกจัดเก็บโดยคน ข้อมูลบางส่วนมาจากการพิมพ์อธิบายลงไป ทำให้ต้องมีกระบวนการจัดการและวิเคราะห์ข้อความโดยเฉพาะ ซึ่งการจะนำข้อมูลชนิดข้อความไปใช้ประโยชน์ได้นั้น ต้องผ่านการวิเคราะห์และแปลงข้อมูลอย่างเหมาะสม เพื่อจัดเก็บลงฐานข้อมูลอย่างเป็นระบบ ซึ่งข้อมูลที่ถูกจัดเก็บอย่างเหมาะสมแล้วจะสามารถค้นหาและนำกลับมาใช้ได้อย่างครบถ้วนและมีประสิทธิภาพ จึงได้มีการปรับปรุงระบบวิเคราะห์ข้อความจากเดิมซึ่งใช้ รูปแบบการตัดคำภาษาไทย (Thai Tokenizer) [49] เปลี่ยนเป็น รูปแบบการตัดคำชนิดเอ็นแกรม (N-gram Tokenizer) [53]

หลักการจัดเก็บข้อมูล ซึ่งเป็นข้อความ เริ่มตั้งแต่การรับข้อความเข้ามา แล้วส่งต่อไปวิเคราะห์โดยใช้อัลกอริทึมที่เหมาะสมตามการตั้งค่าของวิศวกรข้อมูล (Data Engineer) เพื่อให้ได้ผลลัพธ์ที่ดีที่สุด ตามขั้นตอนดังต่อไปนี้



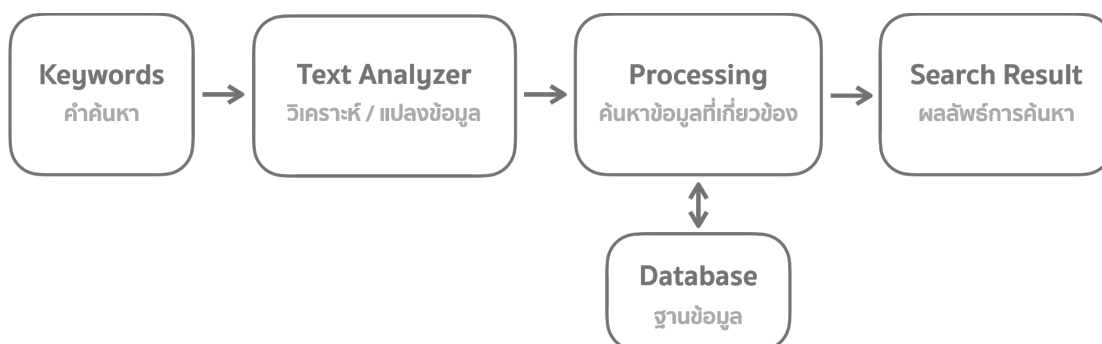
รูปที่ 5 หลักการจัดเก็บข้อมูล ชนิดข้อความ

จากรูปที่ 5 แสดงหลักการจัดเก็บข้อมูล โดยเริ่มจากการรับข้อมูลหรือข้อความที่ต้องการจัดเก็บ (Text Input) เพื่อส่งต่อให้ส่วนวิเคราะห์และแปลงข้อมูล (Text Analyzer) ดำเนินการเพิ่ม ลบ หรือเปลี่ยนแปลงตัวอักษร (Character Filters) ตัดแบ่งข้อความออกเป็นคำย่อย ๆ (Tokenizer) และตัดเชื่อมออกไป (Token Filters) จากนั้นจึงจัดเก็บลงฐานข้อมูล (Database)

ส่วนของ Text Analyzer เป็นเครื่องมือสำคัญที่ทำหน้าที่วิเคราะห์และแปลงข้อมูล ซึ่งมีส่วนประกอบหลักและรายละเอียดดังต่อไปนี้

- **Character filter** ทำหน้าที่ เพิ่ม ลบ หรือเปลี่ยนแปลงตัวอักษรใน Text Input ซึ่งเป็นส่วนประกอบ (block) ที่มีหรือไม่มีก็ได้ ขึ้นอยู่กับชนิดของ Text Analyzer ที่ตั้งค่าไว้
- **Tokenizer** ทำหน้าที่ ตัดแบ่ง Text Input ออกเป็นคำย่อย ๆ เรียกว่า Token (ต้องมี 1 block เท่านั้น) และมีการบันทึกลำดับของคำที่อยู่ใน Text Input เพื่อใช้ในการค้นหา
- **Token filter** ทำหน้าที่ เพิ่ม ลบ หรือเปลี่ยนแปลงคำ (Token) ที่ผ่าน Tokenizer มาแล้ว (ส่วนประกอบนี้ จะไม่มีก็ได้ หรือมีมากกว่า 1 ก็ได้)

เมื่อมีการจัดเก็บข้อมูลอย่างเป็นระบบและมีการเลือกใช้ Text Analyzer อย่างเหมาะสมแล้ว ทำให้สามารถค้นหากลุ่มข้อมูลที่สอดคล้องกับคำค้นหา (Keywords) ที่ป้อนเข้าไปได้



รูปที่ 6 หลักการค้นหาข้อมูล ชนิดข้อความ

จากรูปที่ 6 แสดงหลักการจัดเก็บข้อมูล โดยเริ่มจากการป้อนคำค้นหา (Keywords) เข้าไป จากนั้นจึงส่งต่อให้ Text Analyzer (มักเป็น Text Analyzer ชนิดเดียวกับที่ใช้วิเคราะห์ในขั้นตอนจัดเก็บข้อมูล) ดำเนินการวิเคราะห์และแปลงข้อมูล เมื่อวิเคราะห์คำค้นหาเรียบร้อยแล้ว จึงดำเนินการค้นหาข้อมูลที่สอดคล้องกันจากในฐานข้อมูล โดยผลลัพธ์ที่ได้จากการค้นหานี้ สามารถนำไปใช้งานต่อไปได้ตามความต้องการ

หลักการวิเคราะห์ข้อความ โดยใช้รูปแบบการตัดคำภาษาไทย (Thai Tokenizer) แสดงดังรูปที่ 7



รูปที่ 7 การวิเคราะห์ข้อความด้วย Thai Tokenizer

จากรูปที่ 7 แสดงหลักการวิเคราะห์ข้อมูลด้วย Text Analyzer ประเภทที่ 2 โดยเริ่มจากการรับคำค้นหา (Keywords) เพื่อส่งต่อให้ Character Filters ทำหน้าที่ แปลงเลขไทยเป็นเลขอารบิก จากนั้นจึงใช้ Thai Tokenizer ดำเนินการตัดคำสำหรับภาษาไทย โดยใช้รูปแบบตามฐานคำศัพท์ในพจนานุกรม ตัวอย่างการใช้งาน แสดงดังนี้

ตัวอย่างที่ 1

หากต้องการจัดเก็บและค้นหาคำว่า “ยินดีต้อนรับสู่ประเทศไทย!” จะได้ผลลัพธ์จากแต่ละกระบวนการดังรายละเอียดต่อไปนี้

กระบวนการจัดเก็บข้อความว่า “ยินดีต้อนรับสู่ประเทศไทย!” มีดังนี้

1. เมื่อผ่าน Character Filter จะแปลงเลขไทยเป็นเลขอารบิก จากนั้น จะได้คำว่า “ยินดีต้อนรับสู่ประเทศไทย!”
2. เมื่อผ่าน Thai Tokenizer จะได้คำว่า “ยินดี” “ต้อนรับ” “สู่” “ประเทศ” และ “ไทย” จากนั้นจึงเก็บลงฐานข้อมูล

กระบวนการค้นหาข้อความว่า “ยินดีต้อนรับสู่ประเทศไทย!” มีดังนี้

1. หากป้อนคำค้นหาว่า “ประเทศไทย” และส่งต่อให้ Thai Tokenizer จะได้ผลลัพธ์ คำว่า “ประเทศ” และ “ไทย” (สอดคล้อง)
2. หากป้อนคำค้นหาว่า “ประเทศไทย!” และส่งต่อให้ Thai Tokenizer จะได้ผลลัพธ์ คำว่า “ประเทศ” และ “ไทย” (สอดคล้อง)
3. หากป้อนคำค้นหาว่า “ยินดีต้อนรับ” และส่งต่อให้ Thai Tokenizer จะได้ผลลัพธ์ คำว่า “ยินดี” และ “ต้อนรับ” (สอดคล้อง)
4. หากป้อนคำค้นหาว่า “ยินดีต้อนรับสู่ประเทศไทย” และส่งต่อให้ Thai Tokenizer จะได้ผลลัพธ์คำว่า “ยินดี” “ต้อนรับ” “สู่” “ประเทศ” และ “ไทย” (สอดคล้อง)

ตัวอย่างที่ 2

หากต้องการจัดเก็บและค้นหาคำว่า “ข้าวหอมมะลิ นครราชสีมา๑๐๑” จะได้ผลลัพธ์จากแต่ละกระบวนการดังรายละเอียดต่อไปนี้

กระบวนการจัดเก็บข้อความว่า “ข้าวหอมมะลิ นครราชสีมา๑๐๑” มีดังนี้

1. เมื่อผ่าน Character Filter จะแปลงเลขไทยเป็นเลขอารบิก จะได้คำว่า “ข้าวหอมมะลิ นครราชสีมา101”
2. เมื่อผ่าน Thai Tokenizer จะได้คำว่า “ข้าว” “หอม” “มะลิ” “นครราชสีมา” และ “101” จากนั้นจึงเก็บลงฐานข้อมูล

กระบวนการค้นหาข้อความว่า “ข้าวหอมมะลิ นครราชสีมา๑๐๑” มีดังนี้

1. หากป้อนคำค้นหาว่า “ข้าวหอมมะลิ” และส่งต่อให้ Thai Tokenizer จะได้ผลลัพธ์ คำว่า “ข้าว” “หอม” และ “มะลิ” (สอดคล้อง)
2. หากป้อนคำค้นหาว่า “ข้าวหอมนครราชสีมา” และส่งต่อให้ Thai Tokenizer จะได้ผลลัพธ์คำว่า “ข้าว” “หอม” และ “นครราชสีมา” (สอดคล้อง)

3. หากป้อนคำค้นหาว่า “ข้าวหอม101” และส่งต่อให้ Thai Tokenizer จะได้ผลลัพธ์คำว่า “ข้าว” “หอม” และ “101” (สอดคล้อง)
4. หากป้อนคำค้นหาว่า “ราชสีมา” และส่งต่อให้ Thai Tokenizer จะได้ผลลัพธ์คำว่า “ราช” และ “สีมา” (ไม่สอดคล้อง)

จากทั้งสองตัวอย่าง แสดงให้เห็นว่า Thai Tokenizer สามารถใช้งานกับภาษาไทยได้ แต่ยังค้นหาบางส่วนของคำที่เป็นชื่อเฉพาะไม่ได้

หลักการวิเคราะห์ข้อความ ชนิดเอ็นแกรม (N-gram Tokenizer) แสดงดังรูปที่ 8



รูปที่ 8 การวิเคราะห์ข้อความด้วย N-gram Analyzer

จากรูปที่ 8 แสดงหลักการวิเคราะห์ข้อมูลด้วย N-gram Analyzer โดยเริ่มจากการรับคำค้นหา (Keywords) เพื่อส่งต่อให้ Character Filters ทำหน้าที่แปลงเลขไทยเป็นเลขอารบิก จากนั้นจึงใช้ Tokenizer ชนิด N-gram ดำเนินการตัดคำตามการตั้งค่าลำดับตัวอักษร

N-gram Analyzer ใช้วิธีการตัดคำแบบ N-gram ซึ่งเป็นการตัดคำตามลำดับตัวอักษร โดยสามารถระบุตัวเลขจำนวนตัวอักษรตามหลังได้ เช่น N-gram2,3 เป็นต้น

ตัวอย่างการตัดคำว่า “ข้าวหอมมะลิ” โดยใช้วิธีการตัดคำแบบ N-gram2,3 แสดงผลลัพธ์ที่ได้จากการวิเคราะห์จำนวน 19 คำ ดังรูปที่ 9

ชั	ข้า	้า	้าว	าว	าวห	วห	วหอ	หอ	หอม	อม	อมม	มม	มมะ	มะ	มะล	ะล	ะลิ	ลิ
----	-----	----	-----	----	-----	----	-----	----	-----	----	-----	----	-----	----	-----	----	-----	----

รูปที่ 9 ผลลัพธ์จากการใช้ N-gram Analyzer (วิธีการตัดคำแบบ N-gram2,3)

จากตัวอย่างข้างต้น แสดงให้เห็นว่า N-gram Analyzer สามารถใช้งานได้กับภาษาไทยและกับทุกภาษา แต่จะใช้พื้นที่เก็บข้อมูลสำหรับการค้นหาเยอะกว่า Thai Analyzer และอาจจะได้ผลลัพธ์ที่จำนวนมากเกินไป หากกำหนดขนาดต่ำสุด (Min) และสูงสุด (Max) ของจำนวนตัวอักษรไม่เหมาะสม

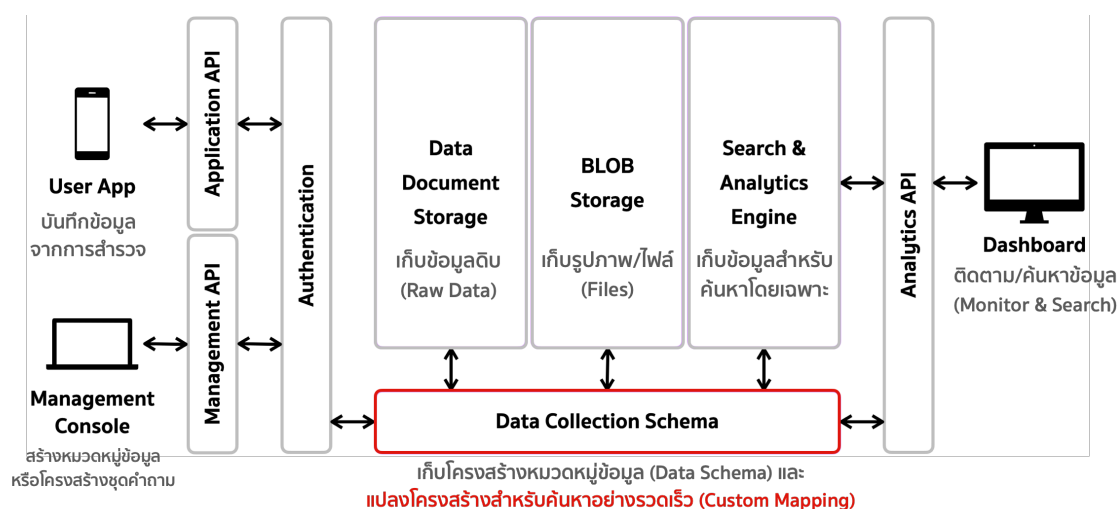
บทที่ 3 วิธีดำเนินการวิจัย/วิธีการวิจัย

ในบทนี้จะอธิบายถึงการออกแบบและวิธีการทดสอบระบบที่เกี่ยวข้องกับวิทยานิพนธ์ฉบับนี้ ซึ่งประกอบด้วย สถาปัตยกรรมของระบบจัดการข้อมูลความหลากหลาย กระบวนการจัดการข้อมูล โครงสร้างการเก็บข้อมูล เครื่องมือที่ใช้ ระบบอำนวยความสะดวกสำหรับจัดการข้อมูลความหลากหลาย การออกแบบโครงสร้างข้อมูลตามหมวดหมู่ และการทดสอบความเร็วในการค้นหาด้วยคำหรือข้อความ

3.1 สถาปัตยกรรมของระบบจัดการข้อมูลความหลากหลาย

การจัดการข้อมูลความหลากหลาย ซึ่งมีปริมาณมากขึ้นอยู่ตลอดเวลา จำเป็นต้องใช้แพลตฟอร์มสำหรับบริหารจัดการข้อมูลชุมชนขนาดใหญ่ ซึ่งได้รับการออกแบบและพัฒนาขึ้นมาโดยบุคลากรผู้เชี่ยวชาญเฉพาะด้าน เช่น วิศวกรข้อมูล (Data Engineer)

การจัดเก็บข้อมูลอย่างมีประสิทธิภาพ จำเป็นต้องออกแบบตั้งแต่โครงสร้างการเก็บข้อมูลให้มีความเหมาะสมสอดคล้องกับข้อมูลความหลากหลาย รวมถึงการตรวจสอบความถูกต้องของข้อมูล เพื่อให้ข้อมูลที่ถูกจัดเก็บสามารถนำไปใช้ประโยชน์ได้ทุกส่วน จึงได้ออกแบบโครงสร้างของระบบจัดเก็บข้อมูลความหลากหลายขึ้นมา ดังรูปที่ 10



รูปที่ 10 โครงสร้างของระบบจัดการข้อมูลความหลากหลาย

จากรูปที่ 10 แสดงโครงสร้างของระบบจัดการข้อมูลความหลากหลาย ซึ่งใช้สำหรับการจัดการข้อมูลแต่ละหมวดหมู่ที่ต้องการจัดเก็บ โดยสามารถสร้างและจัดการโครงสร้างข้อมูลตามหมวดหมู่หรือโครงสร้างของชุดคำถามจากส่วน Management Console ซึ่งแลกเปลี่ยนข้อมูลผ่าน

Management API และยืนยันสิทธิ์การเข้าถึงข้อมูลผ่าน Authentication เพื่อเชื่อมต่อกับส่วนของ Data Collection Schema สำหรับบันทึกโครงสร้างข้อมูล ซึ่งส่วนนี้จะเชื่อมโยงกับ Data Document Storage สำหรับเก็บข้อมูลดิบ และ BLOB Storage สำหรับเก็บข้อมูลไฟล์ เช่น รูปภาพ เป็นต้น

เมื่อสร้างหมวดหมู่ข้อมูลเรียบร้อยแล้ว ผู้สำรวจและบันทึกข้อมูลในแต่ละชุมชนสามารถบันทึกและจัดการข้อมูลได้ผ่าน User App ซึ่งแลกเปลี่ยนข้อมูลผ่าน Application API และยืนยันสิทธิ์การเข้าถึงข้อมูลผ่าน Authentication เพื่อเชื่อมต่อกับ Data Collection Schema เพื่อนำโครงสร้างข้อมูลของหมวดหมู่ที่มีการบันทึกมาแสดงผล ซึ่งส่วนนี้เชื่อมต่อกับ Data Document Storage และ BLOB Storage เพื่อนำข้อมูลดิบและไฟล์ที่สัมพันธ์กันนำกลับไปแสดงผลบน User App

ส่วนของ Data Collection Schema เป็นส่วนที่ทำหน้าที่เก็บโครงสร้างตามหมวดหมู่ข้อมูล (Data Schema) และเชื่อมโยงการเข้าถึงข้อมูลดิบซึ่งจัดเก็บใน Data Document Storage และเชื่อมโยงการเข้าถึงไฟล์ข้อมูล ซึ่งจัดเก็บใน BLOB Storage นอกจากนั้น ยังทำหน้าที่แปลงและจัดเก็บโครงสร้างข้อมูลแต่ละหมวดหมู่ (Data Schema) และทำหน้าที่สำคัญในการแปลงให้เป็นโครงสร้างสำหรับการค้นหาได้อย่างรวดเร็ว (Custom Mapping) ซึ่งเป็นส่วนที่ได้ออกแบบขึ้นมาและถูกใช้งานในส่วนของ Search & Analytics Engine

เมื่อมีการบันทึกข้อมูลซึ่งได้จากการสำรวจแล้ว สามารถติดตามหรือค้นหาข้อมูล ได้ผ่านส่วนของ Dashboard ซึ่งแลกเปลี่ยนข้อมูลผ่าน Analytics API เพื่อเข้าถึงฐานข้อมูลสำหรับการค้นหาหรือส่วนของ Search & Analytics Engine ซึ่งจะทำหน้าที่ประมวลผลตามข้อกำหนดของ Custom Mapping และนำผลลัพธ์ที่ได้กลับไปแสดงผลยัง Dashboard

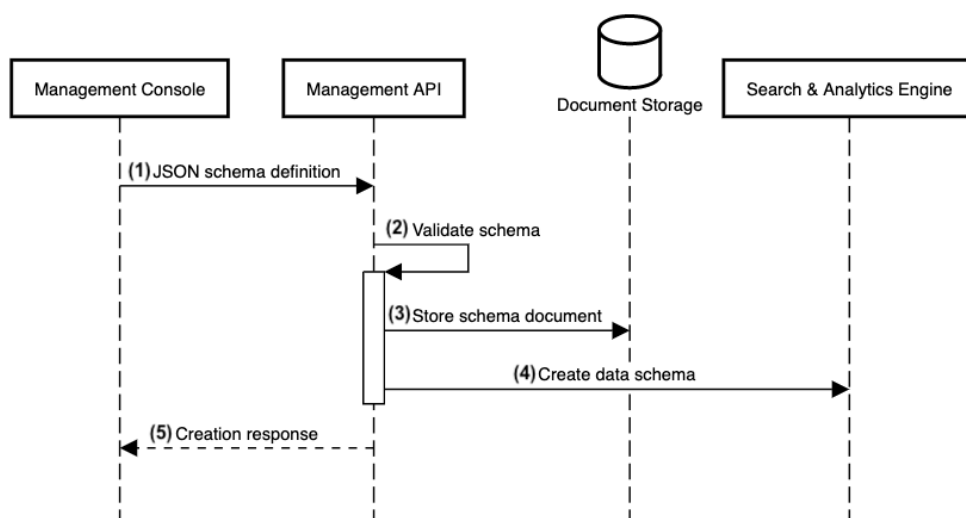
3.2 การออกแบบกระบวนการจัดการข้อมูล

การออกแบบกระบวนการจัดการข้อมูล แสดงให้เห็นถึงลำดับขั้นตอนการจัดการของแต่ละกระบวนการโดยละเอียด ซึ่งสามารถอธิบายด้วยแผนภาพแบบ Sequence Diagram เพื่อแสดงขั้นตอนการทำงานอย่างง่าย โดยกระบวนการจัดการข้อมูลที่จะกล่าวถึงในส่วนถัดไป ประกอบไปด้วย

1. การสร้างหมวดหมู่ข้อมูล (Category Schema Creation)
2. การเพิ่มข้อมูลในแต่ละหมวดหมู่ (Data Submission)
3. การลบข้อมูล (Data Deletion)
4. การแก้ไขข้อมูล (Data Modification)
5. การค้นหาข้อมูล (Data Searching and Retrieval)

การสร้างหมวดหมู่ข้อมูล (Category Schema Creation)

การสร้างหมวดหมู่ข้อมูล คือ การกำหนดโครงสร้างการเก็บข้อมูลในแต่ละหมวดหมู่ ซึ่งออกแบบโดยผู้เชี่ยวชาญจากหลากหลายสาขา ทำหน้าที่กำหนดหรือตั้งค่าโครงสร้างข้อมูลอย่างเหมาะสม สอดคล้องกับการเก็บรวบรวมข้อมูลในแต่ละชุมชน โดยผู้เชี่ยวชาญหรือผู้ใช้งานดังกล่าวนี้สามารถเข้าไปสร้างหมวดหมู่ข้อมูลผ่าน Management Console ซึ่งได้ออกแบบและพัฒนากระบวนการสร้างหมวดหมู่ข้อมูล (Category Schema Creation) ดังนี้



รูปที่ 11 กระบวนการสร้างหมวดหมู่ข้อมูล (Category Schema Creation)

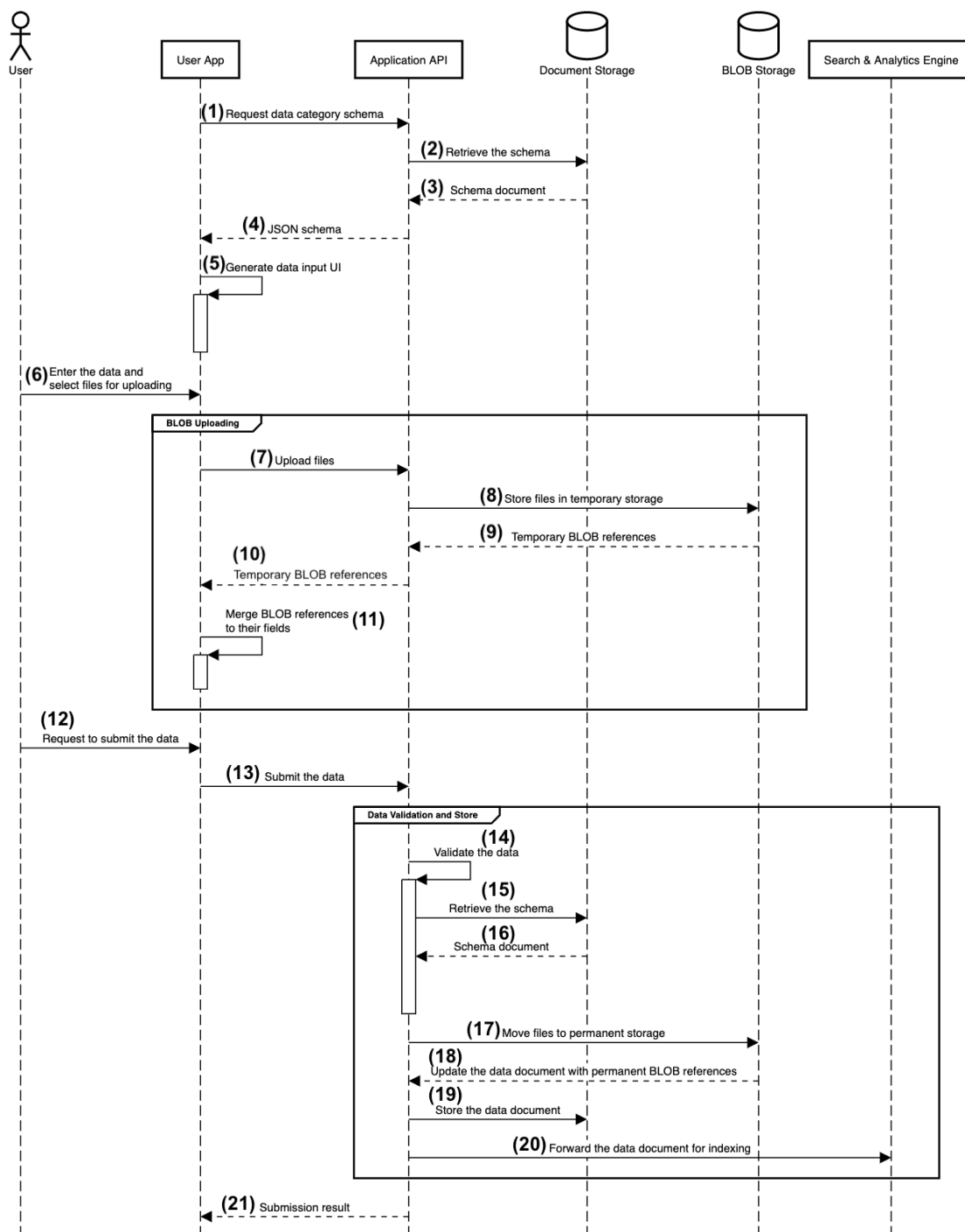
จากรูปที่ 11 แสดงแผนภาพการสร้างหมวดหมู่สำหรับเก็บข้อมูล ตามลำดับดังนี้

- (1) ผู้ใช้งานกำหนดโครงสร้างข้อมูลผ่านทาง Management Console ซึ่งจัดการโครงสร้างนี้ให้อยู่ในรูปแบบ JSON [19] ตามโครงสร้างของ JSON Schema Definition ซึ่งได้อธิบายเพิ่มเติมในหัวข้อ 0 โดยโครงสร้างข้อมูลนี้จะถูกส่งต่อไปยัง Management API
- (2) เมื่อส่วน Management API ได้รับข้อมูลแล้ว จึงดำเนินการตรวจสอบความถูกต้องของโครงสร้างข้อมูล (Validate Schema)
- (3) เมื่อตรวจสอบแล้วว่าโครงสร้างข้อมูลถูกต้อง จึงนำโครงสร้างนี้ไปเก็บไว้ที่ Document Storage
- (4) จากนั้น จึงเริ่มสร้างโครงสร้างข้อมูลอีกชุดหนึ่ง เพื่อส่งให้กับส่วนของ Search & Analytics Engine ซึ่งเป็นส่วนที่เก็บโครงสร้างข้อมูลเพื่อรองรับการเก็บข้อมูลในรูปแบบที่จะนำมาใช้ในการค้นหาโดยเฉพาะ

- (5) เมื่อดำเนินการทั้งหมดเสร็จเรียบร้อยแล้ว จึงส่งข้อมูลตอบกลับไปยังส่วน Management Console เพื่อยืนยันว่าบันทึกโครงสร้างเหล่านั้นเรียบร้อยแล้ว

การบันทึกข้อมูลแต่ละหมวดหมู่ (Data Submission)

เมื่อสร้างหมวดหมู่ข้อมูลเรียบร้อยแล้ว ผู้บันทึกข้อมูลสามารถเข้าไปเพิ่มข้อมูลของหมวดหมู่ที่สนใจได้ ผ่าน User App ซึ่งกระบวนการการบันทึกข้อมูล (Data Submission) แสดงดังรูปที่ 12



รูปที่ 12 กระบวนการบันทึกข้อมูลแต่ละหมวดหมู่ (Data Submission)

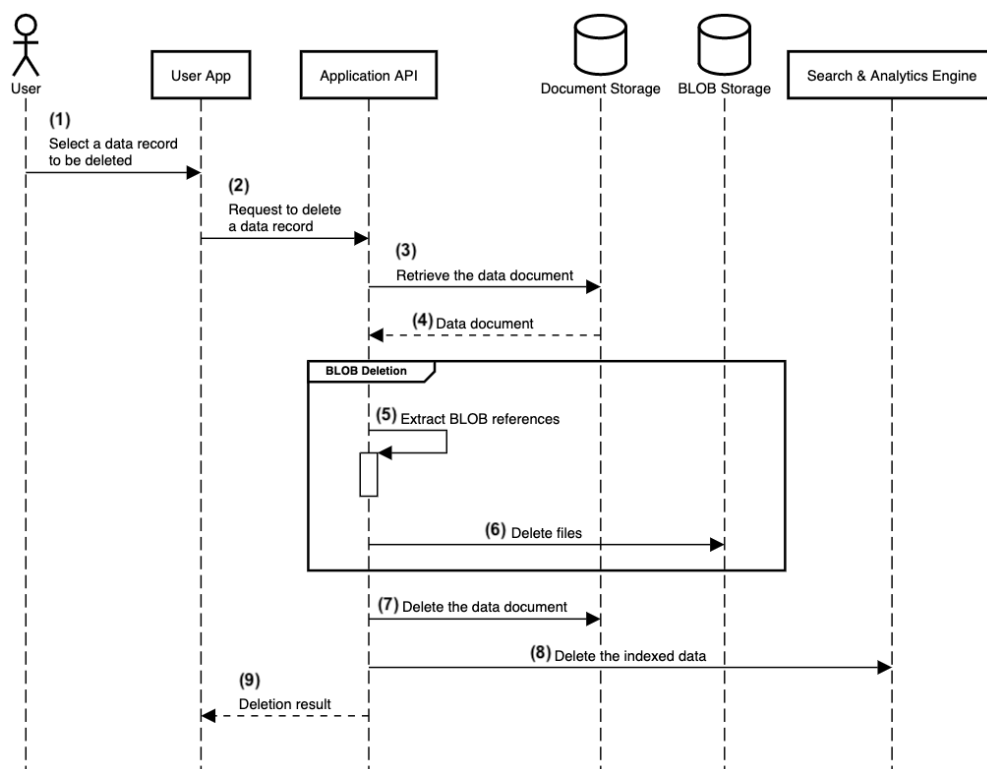
จากรูปที่ 12 แสดงแผนภาพการบันทึกข้อมูลจากการสำรวจ ตามลำดับขั้นตอนดังนี้

- (1) ผู้ใช้งานร้องขอข้อมูลโครงสร้างของหมวดหมู่ที่ต้องการบันทึกข้อมูล (Data Category Schema) ผ่านการใช้งาน User App โดยคำร้องขอจะถูกส่งไปยังส่วน Application API
- (2) จากนั้น Application API จะไปดึงข้อมูลโครงสร้างมาจากส่วนของ Document Storage
- (3) เมื่อ Document Storage เจอข้อมูลโครงสร้างที่ต้องการแล้วนั้น จะส่งข้อมูลโครงสร้าง (Schema Document) กลับมายัง Application API
- (4) จากนั้น Application API จึงส่งข้อมูลโครงสร้างในรูปแบบ JSON [19] (JSON Schema) กลับไปยัง User App
- (5) User App จะดำเนินการสร้างแบบฟอร์มสำหรับกรอกข้อมูล (Data Input UI) ตามโครงสร้างข้อมูลที่ได้รับมา
- (6) ผู้ใช้งาน (User) เริ่มกรอกข้อมูล (Input Data) และอัปโหลดไฟล์ (Upload Files) ตามโครงสร้างของข้อมูล
- (7) หากมีการอัปโหลดไฟล์ User App จะส่งไฟล์เหล่านั้นไปยัง Application API
- (8) จากนั้น Application API จะส่งไฟล์ไปเก็บยังที่เก็บไฟล์แบบชั่วคราว (Temporary Storage) ซึ่งอยู่ในส่วนของ BLOB Storage
- (9) จากนั้น BLOB Storage จะส่งที่อยู่สำหรับอ้างอิงไฟล์เหล่านั้น (Temporary BLOB references) กลับไปยัง Application API
- (10) Application API ดำเนินการส่ง Temporary BLOB references กลับไปยัง User App
- (11) User App ดำเนินการจัดเก็บ BLOB references เหล่านั้นไว้กับระบบก่อน
- (12) เมื่ออัปโหลดไฟล์ประกอบ พร้อมกรอกข้อมูลครบถ้วนตามโครงสร้างแล้ว ผู้ใช้งาน (User) จึงส่งคำสั่งขอบันทึกข้อมูล (Submit Data) ผ่าน User App
- (13) จากนั้น User App จะส่งข้อมูลต่อไปยัง Application API
- (14) Application API ดำเนินการตรวจสอบความถูกต้องของข้อมูล
- (15) เมื่อ Application API ตรวจสอบแล้วว่าข้อมูลที่กรอกมาถูกต้องครบถ้วน จึงส่งคำสั่งขอโครงสร้างข้อมูลไปยัง Document Storage
- (16) Document Storage ส่งโครงสร้างข้อมูลกลับมาให้ยัง Application API

- (17) Application API ดำเนินการย้ายไฟล์ทั้งหมดมาเก็บแบบถาวร (Permanent Storage) ซึ่งพื้นที่เก็บไฟล์อยู่ภายใน BLOB Storage
- (18) BLOB Storage ส่งที่อยู่อ้างอิงสำหรับไฟล์ต่าง ๆ (Permanent BLOB References) กลับมายัง Application API
- (19) จากนั้น Application จึงนำข้อมูลทั้งหมดพร้อมอ้างอิงที่อยู่ไฟล์เรียบร้อย ส่งไปเก็บไว้ใน Document Storage
- (20) Application API ดำเนินการส่งรูปแบบข้อมูลที่มีไว้สำหรับการค้นหา โดยเฉพาะไปยัง Search & Analytics Engine
- (21) เมื่อดำเนินการบันทึกข้อมูลทั้งหมดเรียบร้อยแล้ว Application API จึงส่งผลลัพธ์การบันทึกกลับไปยัง User App

การลบข้อมูล (Data Deletion)

เมื่อบันทึกข้อมูลจากการสำรวจตามหมวดหมู่ที่สนใจเรียบร้อยแล้ว หากมีการกรอกข้อมูลผิดพลาดและต้องการลบข้อมูลชุดนั้นทิ้งไป จึงได้ออกแบบกระบวนการสำหรับการลบข้อมูล (Data Deletion) ดังรูปที่ 13



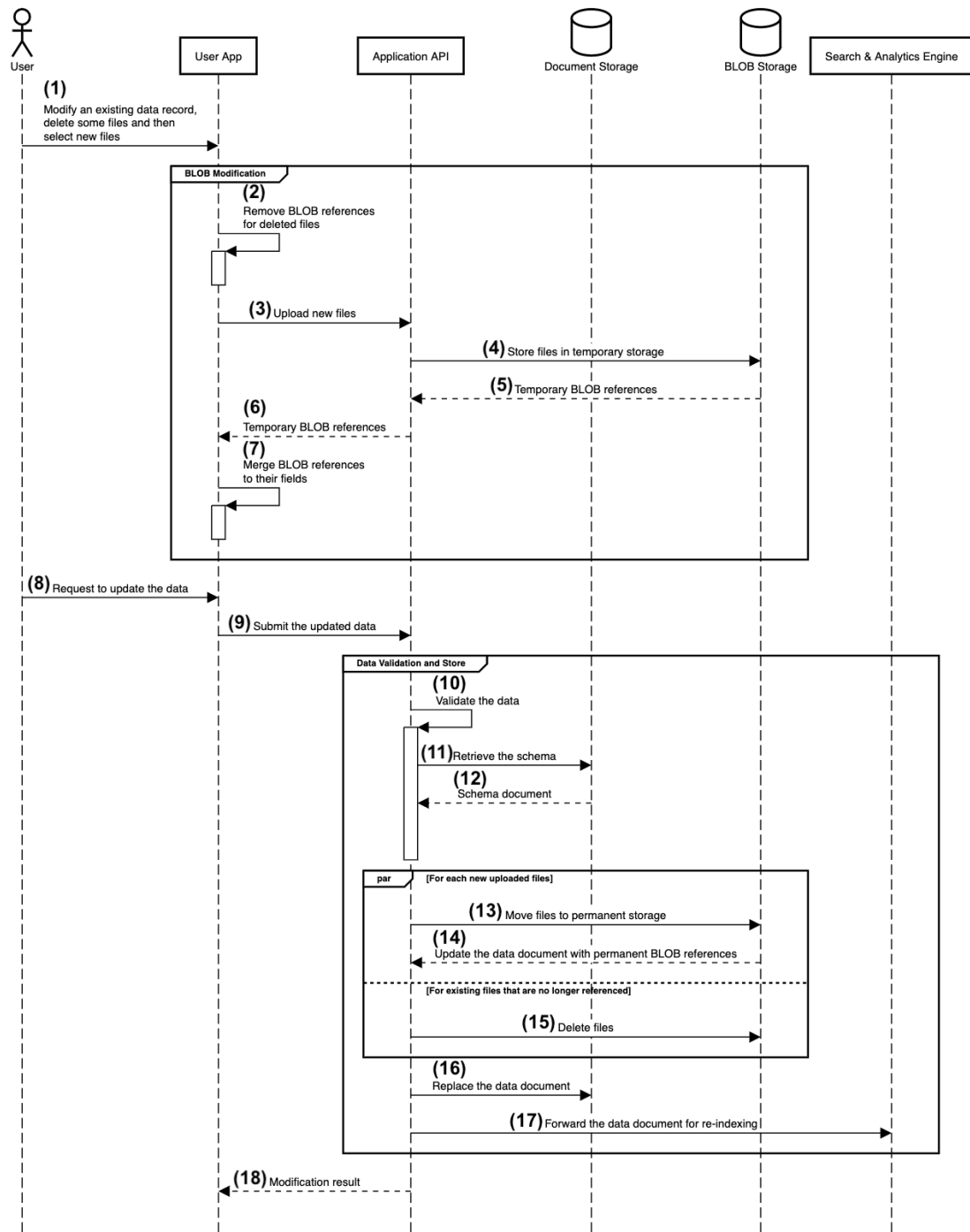
รูปที่ 13 กระบวนการลบข้อมูล (Data Deletion)

จากรูปที่ 13 แสดงกระบวนการลบข้อมูล ตามลำดับขั้นตอนดังนี้

- (1) ผู้ใช้งาน (User) เลือกรายการข้อมูลที่ต้องการลบผ่าน User App
- (2) จากนั้น User App ส่งคำสั่งลบรายการดังกล่าวไปยัง Application API
- (3) ต่อมา Application API ไปดึงข้อมูลรายการที่ต้องการลบนั้นจาก Document Storage
- (4) Document Storage ส่งข้อมูลของรายการที่ต้องการ (Data Document) กลับมาให้ Application API
- (5) Application API สกัดที่อยู่อ้างอิงการเก็บของแต่ละไฟล์ออกมา (Extract BLOB References)
- (6) จากนั้น Application API จึงดำเนินการลบไฟล์ทั้งหมด
- (7) ต่อมา Application API ดำเนินการลบรายการข้อมูลออกจาก Document Storage
- (8) จากนั้น Application API จึงดำเนินการลบข้อมูลส่วนที่ใช้สำหรับการค้นหา โดยเฉพาะ ออกจากส่วนของ Search & Analytics Engine
- (9) Application API ส่งข้อความตอบกลับผลลัพธ์การดำเนินการลบทั้งหมดไปยัง User App

การแก้ไขข้อมูล (Data Modification)

เมื่อบันทึกข้อมูลในหมวดหมู่ที่สนใจเรียบร้อยแล้ว หากต้องการแก้ไขข้อมูลบางส่วน สามารถทำได้ตามกระบวนการแก้ไขรายการข้อมูลที่ต้องการ (Data Modification) ดังรูปที่ 14



รูปที่ 14 กระบวนการแก้ไขข้อมูล (Data Modification)

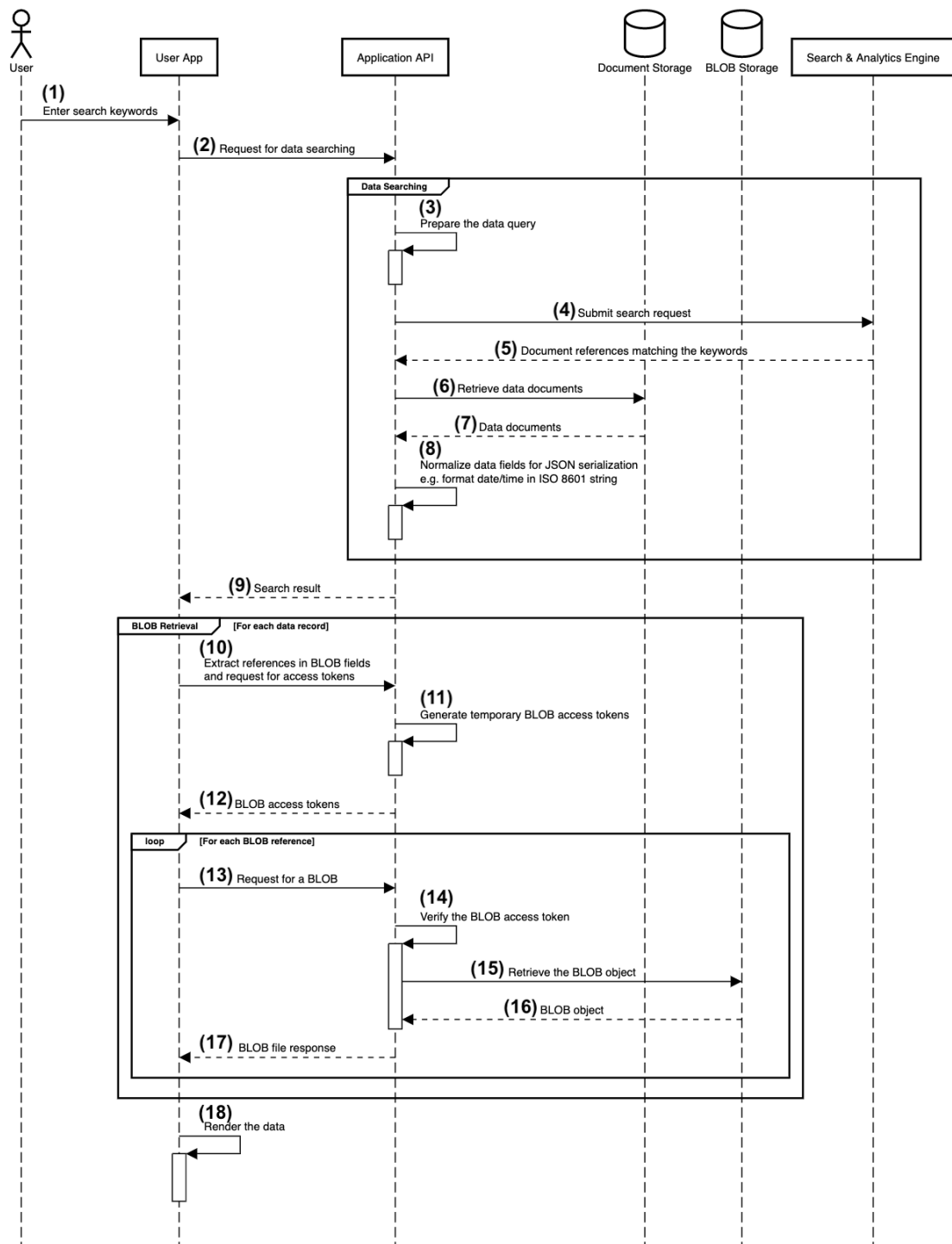
จากรูปที่ 14 แสดงกระบวนการแก้ไขข้อมูล ตามลำดับขั้นตอนดังนี้

- (1) ผู้ใช้งาน (User) ดำเนินการอัปเดตข้อมูลของรายการที่ต้องการผ่าน User App
- (2) จากนั้น User App ดำเนินการลบที่อยู่อ้างอิงของแต่ละไฟล์ออกไป

- (3) ต่อมา User App ดำเนินการส่งไฟล์ที่ต้องการอัปโหลดใหม่ไปยัง Application API
- (4) Application API ดำเนินการจัดเก็บไฟล์เหล่านั้นลงในที่จัดเก็บชั่วคราว (Temporary Storage) ซึ่งอยู่ภายใน BLOB Storage
- (5) จากนั้น BLOB Storage จะส่งที่อยู่อ้างอิงไฟล์เหล่านั้น (Temporary BLOB References) กลับมายัง Application API
- (6) Application API ส่ง Temporary BLOB References กลับไปยัง User App
- (7) User App ดำเนินการจัดเก็บ BLOB references เหล่านั้นไว้กับระบบก่อน
- (8) เมื่ออัปโหลดไฟล์ประกอบ พร้อมกรอกข้อมูลครบถ้วนตามโครงสร้างแล้ว ผู้ใช้งาน (User) จึงส่งคำสั่งขอแก้ไขข้อมูล ผ่าน User App
- (9) จากนั้น User App จะส่งข้อมูลต่อไปยัง Application API
- (10) Application API ดำเนินการตรวจสอบความถูกต้องของข้อมูล
- (11) เมื่อ Application API ตรวจสอบแล้วว่าข้อมูลที่กรอกมาถูกต้องครบถ้วน จึงส่งคำสั่งขอโครงสร้างข้อมูลไปยัง Document Storage
- (12) Document Storage ส่งโครงสร้างข้อมูลกลับมาให้ยัง Application API
- (13) Application API ดำเนินการย้ายไฟล์ทั้งหมดมาเก็บแบบถาวร (Permanent Storage) ซึ่งพื้นที่เก็บไฟล์อยู่ภายใน BLOB Storage
- (14) BLOB Storage ส่งที่อยู่อ้างอิงสำหรับไฟล์ต่าง ๆ (Permanent BLOB References) กลับมายัง Application API
- (15) จากนั้น Application API จึงดำเนินการลบไฟล์ทั้งหมด
- (16) ถัดมา Application API ดำเนินการอัปเดตหรือแทนที่ข้อมูลไปยัง Document Storage
- (17) Application API ดำเนินการอัปเดตรูปแบบข้อมูลสำหรับการค้นหาโดยเฉพาะไปยังส่วน Search & Analytics Engine
- (18) เมื่อดำเนินการอัปเดตข้อมูลทั้งหมดเรียบร้อยแล้ว Application API จึงส่งผลลัพธ์การอัปเดตกลับไปยัง User App

การค้นหาข้อมูล (Data Searching and Retrieval)

เมื่อบันทึกข้อมูลแต่ละหมวดหมู่แล้ว เพื่อการเรียกดูประวัติหรือค้นหาข้อมูลที่ต้องการ จึงออกแบบกระบวนการค้นหาข้อมูล (Data Searching and Retrieval) เพื่อสืบค้นผลลัพธ์ที่ตรงกับความต้องการมาแสดงผล ซึ่งกระบวนการดังกล่าวแสดงดังรูปที่ 15



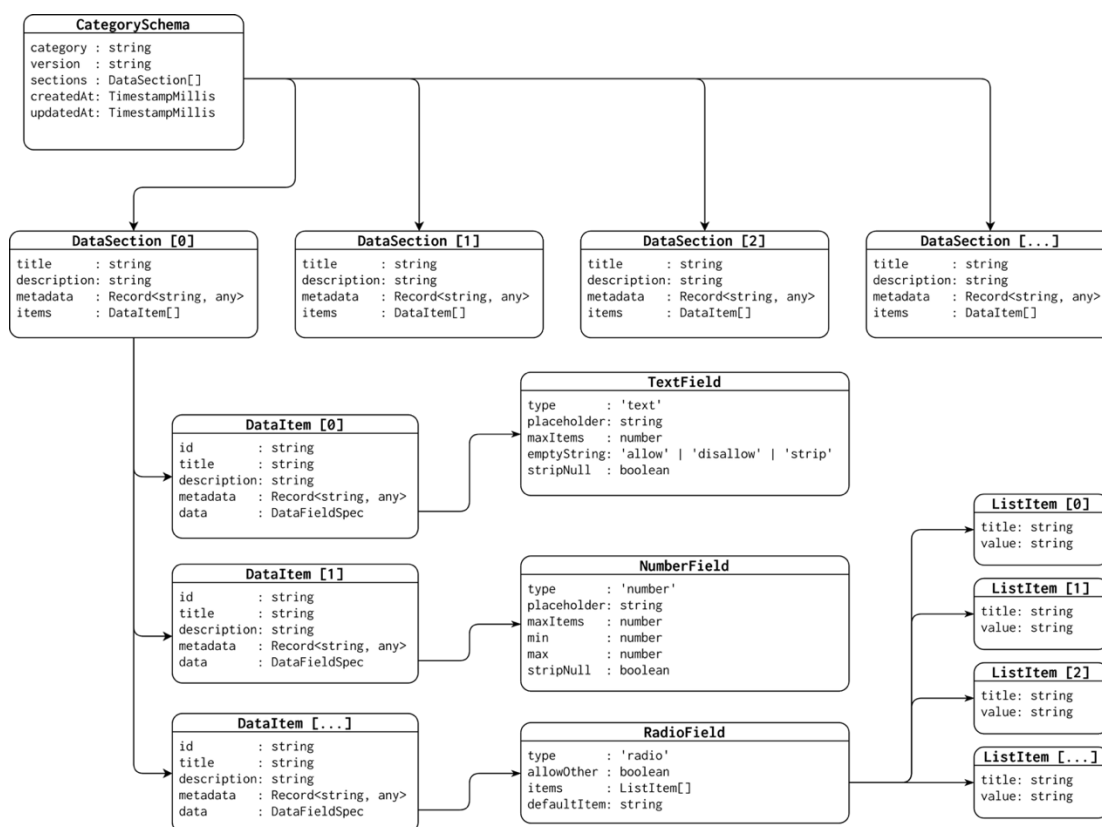
รูปที่ 15 กระบวนการค้นหาข้อมูล (Data Searching and Retrieval)

จากรูปที่ 15 แสดงกระบวนการค้นหาข้อมูลที่ต้องการ ลำดับขั้นตอนดังนี้

- (1) ผู้ใช้งาน (User) ระบุข้อมูลที่ต้องการค้นหาผ่าน User App
- (2) User App ดำเนินการส่งคำร้องขอไปยัง Application API
- (3) จากนั้น Application API เตรียมข้อมูลที่จะนำไปประมวลผล
- (4) ถัดมา Application API ส่งคำร้องขอการค้นหาไปยังส่วน Search & Analytics Engine
- (5) จากนั้น Search & Analytics Engine ส่งผลลัพธ์ซึ่งสอดคล้องกับคำค้นหา กลับมายัง Application API
- (6) ต่อมา Application API ส่งข้อความขอข้อมูลดิบจาก Document Storage
- (7) Document Storage ส่งข้อมูลตอบกลับมาให้ Application API
- (8) จากนั้น Application API ดำเนินการจัดระเบียบข้อมูลและแปลงข้อมูลให้ถูกต้องตามมาตรฐาน เช่น จัดรูปแบบของวันที่ให้ตรงตามมาตรฐาน ISO 8601
- (9) ถัดมา Application API ดำเนินการส่งผลลัพธ์การค้นหากลับมายัง User App
- (10) จากนั้น User App จึงสกัดที่อยู่อ้างอิงของไฟล์ต่าง ๆ ออกมา และส่งคำร้องขอ BLOB Access Tokens (คีย์ที่ใช้สำหรับการเข้าถึงไฟล์) ไปยัง Application API
- (11) Application API ดำเนินการสร้าง BLOB Access Tokens แบบชั่วคราว
- (12) จากนั้น Application API ส่ง BLOB Access Tokens กลับมาให้ User App
- (13) User App ดำเนินการส่งคำร้องขอไฟล์ไปยัง Application API
- (14) Application API ดำเนินการตรวจสอบสิทธิ์ในการเข้าถึงจาก BLOB Access Token ที่แนบไปด้วย
- (15) จากนั้น Application API ส่งคำสั่งขอข้อมูลไฟล์ไปยัง BLOB Storage
- (16) BLOB Storage ส่งข้อมูลไฟล์กลับมายัง Application API
- (17) Application API ส่งไฟล์ข้อมูลนั้นกลับไปให้ User App
- (18) จากนั้น User App จึงแสดงผลข้อมูลที่ได้รับ

3.3 รูปแบบโครงสร้างการเก็บข้อมูลแต่ละหมวดหมู่ (JSON Schema Definition)

รูปแบบโครงสร้างการเก็บข้อมูลที่เหมาะสม สอดคล้องกัน โดยมีความยืดหยุ่นในการปรับเพิ่มหรือลดข้อมูล เป็นสิ่งสำคัญและจำเป็นอย่างยิ่ง เนื่องจากต้องรองรับกับข้อมูลแต่ละหมวดหมู่ที่มีความหลากหลายและมีการเพิ่มหมวดหมู่ข้อมูลใหม่ ๆ เข้ามาในอนาคตอย่างต่อเนื่อง เพื่อให้สะดวกต่อการจัดการ การต่อเติม การแก้ไข ข้อมูลหลากหลายสาขานี้ จึงได้ออกแบบโครงสร้างการเก็บข้อมูลตามหมวดหมู่ (Data Schema) ซึ่งอ้างอิงรูปแบบโครงสร้างตาม JSON Schema Definition ซึ่งได้ออกแบบและพัฒนาขึ้นมาดังรูปที่ 16 นี้ โดยสามารถศึกษาลักษณะโครงสร้างตามหมวดหมู่ข้อมูล (Data Schema) ที่สร้างเสร็จแล้วเพิ่มเติมในหัวข้อ 0 ซึ่งแต่ละ Data Schema นี้จะถูกบันทึกโครงสร้างไว้ใน Data Collection Schema ซึ่งเป็นส่วนประกอบที่ระบุไว้ในรูปที่ 10



รูปที่ 16 JSON Schema Definition

จากรูปที่ 16 แสดงโครงสร้างข้อมูลในรูปแบบ JSON [19] (JSON Schema Definition) ซึ่งใช้สำหรับการสร้างหมวดหมู่สำหรับการเก็บข้อมูลชนิดใหม่ (CategorySchema) ซึ่งประกอบด้วย 5 Fields คือ

- category สำหรับเก็บ ID ของหมวดหมู่ข้อมูล
- version สำหรับระบุเวอร์ชันของหมวดหมู่ข้อมูล

- createdAt สำหรับระบุเวลาที่หมวดหมู่ข้อมูลถูกสร้างขึ้น
- updatedAt สำหรับระบุเวลาที่หมวดหมู่ข้อมูลถูกแก้ไขล่าสุด
- sections เป็นรายการของ **DataSection** ซึ่งจะอธิบายในส่วนถัดไปนี้

DataSection ประกอบไปด้วย 4 Fields ดังนี้

- title สำหรับเก็บชื่อของ Section
- description สำหรับเก็บคำอธิบายของ Section
- metadata สำหรับเก็บข้อมูลอื่น ๆ เพิ่มเติมที่ต้องการ
- items เป็นรายการของ **DatalItem** ซึ่งจะอธิบายในส่วนถัดไปนี้

DatalItem ประกอบไปด้วย 5 Fields ดังนี้

- id สำหรับเก็บรหัสของ DatalItem
- title สำหรับเก็บชื่อของ DatalItem
- description สำหรับเก็บคำอธิบายของ DatalItem
- metadata สำหรับเก็บข้อมูลอื่น ๆ เพิ่มเติมที่ต้องการ
- data เป็นรายการของ **DataFieldSpec** ซึ่งจะอธิบายในส่วนถัดไปนี้

DataFieldSpec คือ ชนิดของข้อมูลที่จัดเก็บ ที่นอกจากจะใช้ในการระบุชนิดของข้อมูลแล้ว ยังใช้ในการตรวจสอบความถูกต้องของข้อมูลด้วย (Validation) โดยในปัจจุบันได้ออกแบบให้รองรับข้อมูล 11 ชนิด ชนิดของข้อมูลดังนี้

- **Text Field** เก็บข้อมูลที่เป็นข้อความ
- **Number Field** เก็บข้อมูลที่เป็นเชิงตัวเลข
- **Date Field** เก็บข้อมูลวันที่
- **Time Field** เก็บข้อมูลเวลา
- **Date and Time Field** เก็บข้อมูลวันที่และเวลา
- **Geo Point Field** เก็บข้อมูลพิกัด เช่น ละติจูด ลองจิจูด เป็นต้น
- **Address Field** เก็บข้อมูลที่อยู่
- **Drop Down Field** เก็บข้อมูล 1 อย่าง จากรายการตัวเลือกที่กำหนดไว้

- **Radio Field** เก็บข้อมูล 1 อย่าง จากรายการตัวเลือกที่กำหนดไว้ หรือ ข้อมูลอื่น ๆ นอกเหนือจากรายการตัวเลือกได้
- **Check Boxes Field** เก็บข้อมูลได้หลายอย่าง จากรายการตัวเลือกที่กำหนดไว้ หรือ ข้อมูลอื่น ๆ นอกเหนือจากรายการตัวเลือกได้
- **File Field** เก็บข้อมูลไฟล์ เช่น รูปภาพ ไฟล์เอกสาร เป็นต้น

3.4 เครื่องมือที่ใช้

จากสถาปัตยกรรมของระบบจัดการข้อมูลความหลากหลาย ซึ่งแสดงไว้ดังรูปที่ 10 แล้วนั้น เครื่องมือที่ใช้ในการพัฒนาของแต่ละองค์ประกอบสำคัญ แสดงดังต่อไปนี้

Application API และ Management API

ส่วนของ Application API และ Management API จำเป็นต้องใช้สถาปัตยกรรมคลาวด์แบบ Serverless ซึ่งไม่ต้องเปิดเครื่อง Server ทิ้งไว้ และจะทำงานต่อเมื่อ User App หรือ Management Console ร้องขอเข้ามาเท่านั้น จึงได้เลือกใช้เครื่องมือ Google Cloud Functions ซึ่งเป็นบริการที่คิดค่าใช้จ่ายตามการใช้งานจริงตามการร้องขอของผู้ใช้งานและรองรับการขยายตัวของทรัพยากรได้โดยอัตโนมัติ ซึ่งหากไม่ใช้บริการของ Google Cloud Functions นี้ สามารถเลือกใช้บริการลักษณะเดียวกันจาก Amazon Web Services โดยใช้บริการที่ชื่อว่า AWS Lambda

Data Collection Schema และ Data Document Storage

ส่วนของ Data Collection Schema และ Data Document Storage ได้เลือกใช้เครื่องมือ Google Firestore [2] ซึ่งเป็นระบบฐานข้อมูลชนิด NoSQL [9] มีความยืดหยุ่นสามารถรองรับการเก็บข้อมูลหลากหลายชนิด โดยไม่ต้องอยู่ในรูปแบบความสัมพันธ์เชิงโครงสร้าง (Schema-less) คุณสมบัติเด่นคือการอัปเดตข้อมูลแบบ Real-time ฐานข้อมูลชนิดนี้รองรับการขยายขนาดการเก็บข้อมูลโดยอัตโนมัติตามการใช้งานจริง และมีคุณสมบัติของ Serverless ซึ่งทำให้ไม่ต้องติดตั้งหรือบริหารจัดการเครื่องประมวลผล (Server) ด้วยตนเอง นอกจากนั้น สามารถรองรับการเข้าใช้งาน (Concurrent Connection) จำนวน 1,000,000 request พร้อมกัน

BLOB Storage

ส่วนของ BLOB Storage ใช้งานสำหรับเก็บข้อมูลชนิดไฟล์ เช่น รูปภาพ เอกสาร เป็นต้น เหมาะสมที่กับเครื่องมือซึ่งมีความยืดหยุ่นสามารถขยายพื้นที่เก็บข้อมูลได้โดยอัตโนมัติ โดยคิดค่าใช้จ่ายตามพื้นที่เก็บข้อมูลที่บันทึก และการร้องขอข้อมูลตามปริมาณการใช้งานจริง โดยต้องการคุณสมบัติของ Serverless จึงได้เลือกใช้เครื่องมือ Google Cloud Storage ซึ่งรองรับการเก็บไฟล์ทุกรูปแบบตามต้องการและมีคุณสมบัติตรงตามวัตถุประสงค์ โดยหากไม่ใช้บริการของ Google Cloud Storage ยังสามารถเลือกใช้บริการลักษณะเดียวกันได้จากผู้ให้บริการอื่น ๆ เช่น Amazon Web Services ได้ โดยใช้บริการที่ชื่อว่า AWS S3 เป็นต้น

Search & Analytics Engine

ส่วนของ Search & Analytics Engine ซึ่งใช้สำหรับเก็บข้อมูลในรูปแบบที่เหมาะสมสำหรับการค้นหาโดยเฉพาะ โดยเหมาะสมกับเครื่องมือที่สามารถรองรับการขยายแบบแนวนอน และรองรับ Full-text Search ซึ่งเป็นคุณสมบัติที่ช่วยในการค้นหาข้อความที่ต้องการได้อย่างรวดเร็วและมีประสิทธิภาพ จึงได้เลือกใช้เครื่องมือ Elasticsearch [4] ซึ่งมีคุณสมบัติตรงกับความต้องการ

User App

ส่วนของ User App ซึ่งใช้สำหรับอำนวยความสะดวกให้กับผู้สำรวจและเก็บบันทึกข้อมูลชุมชน ได้เลือกใช้เครื่องมือสำหรับการพัฒนาเว็บแอปพลิเคชัน (Web Application) ซึ่งมีความสะดวกต่อผู้ใช้งานผ่านระบบปฏิบัติหลากหลาย เช่น iOS Android เป็นต้น โดยกลุ่มของเครื่องมือที่เลือกใช้พัฒนาส่วนติดต่อกับผู้ใช้งาน User App นี้ คือ Ionic Framework ซึ่งเป็น Hybrid Framework โดยรองรับการพัฒนา Mobile Application และ Web Application ซึ่งพัฒนาโดย Angular เป็นหลัก ยังมีอีกหลากหลาย framework ซึ่งเป็นเครื่องมือช่วยอำนวยความสะดวกในการพัฒนาได้ เช่น React Native หรือ Flutter เป็นต้น

Management Console

ส่วนของ Management Console ซึ่งใช้สำหรับอำนวยความสะดวกในการสร้างโครงสร้างการเก็บข้อมูลสำหรับแต่ละหมวดหมู่สำหรับแต่ละชุมชน ซึ่งเป็น Web Application ส่วนติดต่อกับผู้ใช้ที่ช่วยออกแบบและสร้างโครงการการเก็บข้อมูลอย่างเหมาะสมให้โดยอัตโนมัติ โดยเครื่องมือที่ใช้ในการพัฒนา คือ Ionic Framework ซึ่งเป็นเครื่องมือที่ช่วยให้การพัฒนา Web

Application ทำได้อย่างสะดวกและรวดเร็วมากยิ่งขึ้น โดยยังมีอีกหลากหลาย framework ซึ่งเป็นเครื่องมือช่วยอำนวยความสะดวกในการพัฒนาได้ เช่น React Native หรือ Flutter เป็นต้น

Dashboard

ส่วนของ Dashboard หรือ Data Visualization เป็นส่วนที่ช่วยให้ติดตามข้อมูลสถิติหรือข้อมูลสรุปสำหรับผู้บริหารจัดการพื้นที่หรือชุมชนระดับต่าง ๆ ได้ในรูปแบบหลากหลาย โดยสามารถเลือกกรองเฉพาะชุดข้อมูลที่น่าสนใจ ตามวันเวลาที่ต้องการได้ โดยส่วนของ Dashboard นี้ได้เลือกใช้เครื่องมือที่เหมาะสม เพื่อช่วยในการวิเคราะห์ข้อมูลในหลากหลายมุมมอง ซึ่งประกอบไปด้วย Ionic Framework สำหรับพัฒนาระบบติดตามข้อมูลสถิติ เพื่อเข้าถึงข้อมูลสรุป ข้อมูลดิบ การนำออก และการปรับปรุงข้อมูลเพิ่มเติม สำหรับบริหารจัดการแต่ละชุมชน และใช้เครื่องมือ Kibana สำหรับสร้าง Dashboard ในรูปแบบที่หลากหลายมากยิ่งขึ้น เช่น กราฟเส้น (Line) กราฟวงกลม (Pie) ความหนาแน่นของกลุ่มข้อมูล (Heat Map) เป็นต้น โดย Kibana เป็น Data Visualization ที่เชื่อมต่อกับ Elasticsearch [4] เพื่อนำมาจัดแสดงข้อมูลสรุปได้ทันที นอกจากนั้น ยังได้ใช้เครื่องมือ Microsoft Power BI เนื่องจากมีชนิดของกราฟที่มากกว่า Kibana และสามารถจัดวางโครงสร้างหน้าตาการนำเสนอได้ยืดหยุ่นมากกว่า Kibana ซึ่งเป็นประโยชน์ต่อการพิจารณาข้อมูลได้สะดวกมากยิ่งขึ้นต่อผู้บริหาร

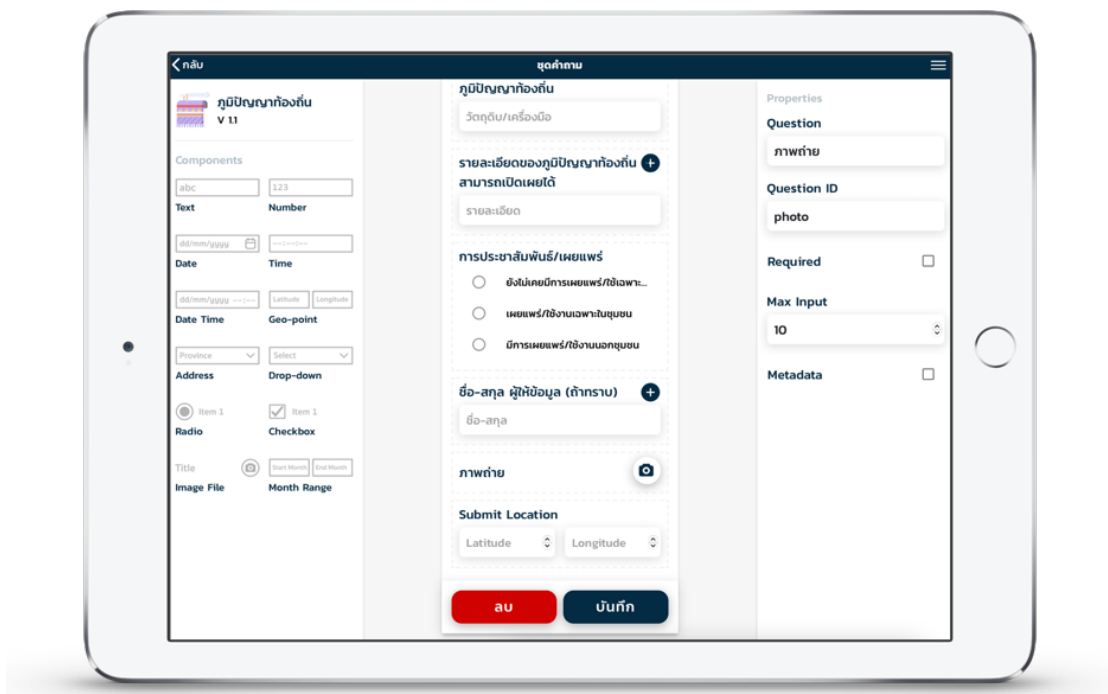
3.5 ระบบอำนวยความสะดวกในการจัดการข้อมูลความหลากหลาย

ระบบอำนวยความสะดวกในการจัดการข้อมูลความหลากหลาย ประกอบไปด้วยสามระบบ คือ ระบบสร้างชุดคำถาม ระบบบันทึกข้อมูล และระบบติดตามข้อมูลสถิติ

ระบบสร้างชุดคำถาม (Management Console)

ระบบสร้างชุดคำถามช่วยอำนวยความสะดวกให้ผู้เชี่ยวชาญแต่ละสาขาจัดการโครงสร้างชุดข้อมูลของหมวดหมู่ที่ชุมชนหรือเมืองสนใจ ความสะดวกในการสร้างโครงสร้างการจัดเก็บข้อมูลที่หลากหลายแบบอัตโนมัติ เพื่อให้สอดคล้องกับความต้องการของแต่ละพื้นที่ โดยได้ออกแบบให้สามารถลากวางส่วนประกอบของช่องบันทึกประเภทต่าง ๆ (Input Components) ได้ โดย Input Components ที่สามารถรองรับ ได้แก่ ช่องบันทึกข้อความ (Text) วันที่ (Date) ตัวเลือกแบบเลือกได้รายการเดียว (Radio) ตัวเลือกแบบเลือกได้หลายรายการ (Checkbox) ช่องสำหรับกรอกที่อยู่ (Address) ช่องแนบรูปภาพ (Image) พิกัดตำแหน่ง (Geo Point) เป็นต้น เมื่อผู้สร้างชุดคำถามได้ลากวางและพิมพ์รายละเอียดที่จำเป็นตามที่ระบบแนะนำเรียบร้อยแล้ว ระบบจะดำเนินการ

จัดทำเป็นฐานข้อมูลโดยอัตโนมัติให้พร้อมสำหรับให้ผู้ลงพื้นที่เก็บข้อมูลนำไปใช้งานได้ทันที หน้าตาของระบบสร้างชุดคำถามแสดงดังรูปที่ 17



รูปที่ 17 ระบบสร้างชุดคำถาม (Management Console)

ระบบบันทึกข้อมูล (User App)

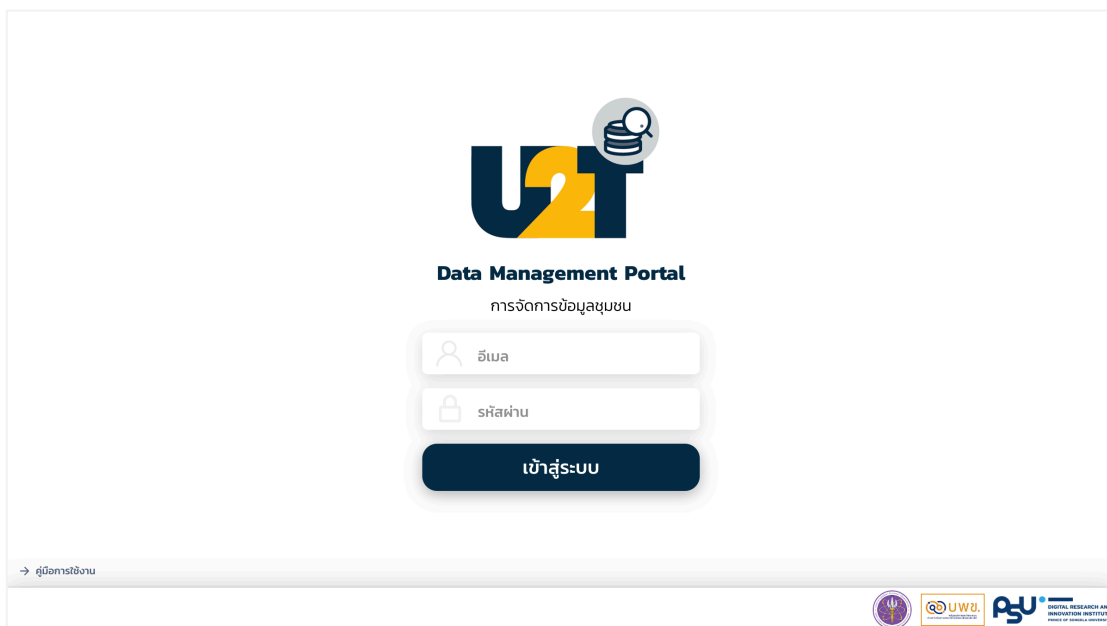
ผู้สำรวจหรือผู้บันทึกข้อมูลสามารถบันทึกผ่านระบบบันทึกข้อมูล ซึ่งเข้ามาอำนวยความสะดวกสำหรับผู้ลงพื้นที่รวบรวมและบันทึกข้อมูลที่มีความเฉพาะเจาะจงกับแต่ละชุมชน โดยผู้สำรวจสามารถใช้งานระบบบันทึกข้อมูลนี้ได้ผ่านอุปกรณ์สมาร์ทโฟน โดยสามารถเข้าใช้งานผ่านเว็บเบราว์เซอร์ที่ถูกติดตั้งอยู่บนระบบปฏิบัติการบนสมาร์ตโฟนยอดนิยม เช่น iOS Android เป็นต้น ซึ่งผู้สำรวจสามารถจัดการเพิ่ม แก้ไข หรือลบข้อมูล ที่ตนเป็นผู้บันทึกได้ทั้งหมด ลักษณะของระบบบันทึกข้อมูล แสดงดังรูปที่ 18 ซึ่งแสดงหน้าสำหรับเข้าสู่ระบบ หน้าแสดงหมวดหมู่ของข้อมูล ซึ่งผู้สำรวจสามารถคลิกเลือกหมวดหมู่ที่ต้องการเพิ่มข้อมูล เพื่อเข้าสู่ยังหน้าบันทึกข้อมูลต่อไป โดยผู้สำรวจจำเป็นต้องกรอกข้อมูลอย่างถูกต้องครบถ้วนตามระบบแนะนำ เมื่อต้องการกลับมาดูข้อมูลที่เคยบันทึกไปแล้ว เพื่อดำเนินการแก้ไขหรือลบข้อมูลออกไป สามารถคลิกยังเมนูประวัติ เพื่อเข้าไปค้นหาข้อมูลที่ต้องการจัดการได้ทันที



ข้อมูลดิบทั้งหมดซึ่งมาจากการลงพื้นที่สำรวจในชุมชนจากอาสาสมัคร นักศึกษา พร้อมทีมคณาจารย์ซึ่งคอยอบรม ให้คำแนะนำ และคอยตรวจสอบความถูกต้องของข้อมูล ข้อมูลที่ได้จึงสามารถนำมาต่อยอดเพื่อวิเคราะห์ข้อมูลของแต่ละชุมชนในมิติต่าง ๆ เช่น ด้านสุขภาพ ความเป็นอยู่ การศึกษา รายได้ การวิเคราะห์จุดแข็งจุดอ่อนของแต่ละชุมชน เป็นต้น เพื่อวางแผนพัฒนาระดับทั้งด้านเศรษฐกิจ สังคม หรือสิ่งแวดล้อมได้ตามต้องการ

ระบบติดตามข้อมูลสถิติ

ระบบติดตามข้อมูลสถิติช่วยอำนวยความสะดวกในการติดตามข้อมูลภาพรวมของแต่ละพื้นที่ ช่วยให้ผู้บริหารจัดการชุมชนของตนเอง หรือผู้บริหารในระดับตำบล จังหวัด หรือประเทศสามารถเข้าถึงข้อมูลสถิติและกรองข้อมูล หรือค้นหาข้อมูลตามความสนใจได้ โดยลักษณะระบบติดตามข้อมูลสถิตินี้ แสดงดังรูปที่ 19 ถึงรูปที่ 22




Data Management Portal
 การจัดการข้อมูลชุมชน

→ คู่มือการใช้งาน



รูปที่ 19 หน้าเข้าสู่ระบบ ของระบบจัดการข้อมูลชุมชน (Data Management Portal)

จากรูปที่ 19 แสดงหน้าสำหรับเข้าสู่ระบบ ของระบบจัดการข้อมูลชุมชน (Data Management Portal) ซึ่งผู้บริหารจัดการข้อมูลแต่ละชุมชนสามารถเข้าใช้งานผ่านระบบนี้ได้ โดยวิธีการใช้งานอยู่ในลิงค์คู่มือการใช้งานบริเวณมุมล่างซ้าย

สถิติภาพรวม วันที่เริ่มต้น: 1/3/2564 วันที่สิ้นสุด: 16/6/2564

มหาวิทยาลัยสงขลานครินทร์		จำนวนข้อมูลที่เป็นที่กัก	ตำบลที่สำรวจ	ค่าเฉลี่ยการบันทึกต่อตำบล
☒ ทุกตำบล จังหวัด: ☐ ทุกหมวดหมู่ ผู้ที่ย้ายกลับบ้าน เนื่องจาก:		81 รายการ	8/90 8.89 %	10.13 รายการ
จังหวัด :	อำเภอ :	ตำบล :	จำนวนข้อมูลที่เป็นที่กัก :	จำนวนผู้ที่บันทึก :
สงขลา	จะนะ	ตลิ่งชัน	66	1
นราธิวาส	รือเสาะ	สามัคคี	4	1
สงขลา	ระโนด	ตะเครียะ	3	1
สตูล	ละงู	ละงู	3	1
นราธิวาส	รือเสาะ	โคกสะตอ	2	1
ปัตตานี	ยะหริ่ง	จะรัง	1	1
ปัตตานี	เมืองปัตตานี	บาราโหม	1	1

8 รายการ CSV

รูปที่ 20 หน้าสถิติภาพรวม ของระบบจัดการข้อมูลชุมชน (Data Management Portal)

จากรูปที่ 20 แสดงหน้าข้อมูลสถิติภาพรวม ของระบบจัดการข้อมูลชุมชน (Data Management Portal) ซึ่งผู้บริหารจัดการสำหรับแต่ละหน่วยงานหรือแต่ละชุมชน สามารถเข้าดูจำนวนข้อมูลที่บันทึกในระบบบริหารจัดการข้อมูลความหลากหลายได้ โดยสามารถเลือกกรองข้อมูลตามความสนใจ เช่น การระบุตำบล ระบุหมวดหมู่ของข้อมูล การเลือกวันที่ที่บันทึกข้อมูล เป็นต้น

The screenshot displays the 'ข้อมูลดิบ' (Raw Data) section of the Data Management Portal. The interface includes a sidebar with filters for 'มหาวิทยาลัยธรรมศาสตร์' (Thammasat University), 'พืชในท้องถิ่น' (Local Plants), 'จังหวัด' (Province), 'ชื่อผู้บันทึกข้อมูล' (Data Recorder Name), and 'ชื่อ-สกุล' (Name-Surname). The main area shows a table of data records with columns for 'รูปภาพ' (Image), 'วันที่บันทึกข้อมูล' (Data Recording Date), 'รหัสรายการ' (Record ID), 'ชื่อ-สกุลผู้บันทึกข้อมูล' (Data Recorder Name), 'ชื่อเรียกของพืช' (Plant Name), and 'สถานที่พบ' (Location Found). The table lists four records, each with a corresponding image of a plant. The total number of records is 1,336, and there is a 'CSV' export button at the bottom right.

รูปภาพ	วันที่บันทึกข้อมูล	รหัสรายการ	ชื่อ-สกุลผู้บันทึกข้อมูล	ชื่อเรียกของพืช	สถานที่พบ
	4/28/2021 8:45:06 PM	VKSBCU-522441	[Name]	- ผักหวานบ้าน - ผักห:	ต.ป่าคาหลวง อ.บ้าน
	4/28/2021 8:59:01 PM	XQNZD-268060	[Name]	- ผักกูด	ต.ป่าคาหลวง อ.บ้าน
	5/7/2021 3:40:50 PM	BHP5MH-672415	[Name]	- ข้าว	ต.ปางแสนทอง อ.เมือง
	5/8/2021 4:50:45 PM	LOBDCX-243611	[Name]	- มะเขือแฉะ	ต.ปางตอน อ.แจ้ห่ม จ.

รูปที่ 21 หน้าข้อมูลดิบ ของระบบจัดการข้อมูลชุมชน (Data Management Portal)

จากรูปที่ 21 แสดงหน้าข้อมูลดิบ ของระบบจัดการข้อมูลชุมชน (Data Management Portal) ซึ่งผู้บริหารจัดการข้อมูลของแต่ละหน่วยงานหรือแต่ละชุมชน สามารถเข้าถึงข้อมูลแต่ละหมวดหมู่ได้โดยละเอียด สามารถดูข้อมูลดิบของแต่ละเอกสารได้ เพื่อการวิเคราะห์ข้อมูลตามวัตถุประสงค์การใช้งานของแต่ละชุมชน

ข้อมูลรายจังหวัด

หมวดหมู่ข้อมูล

แหล่งท่องเที่ยว

รายชื่อจังหวัด

กระบี่

อำเภอ

ค้นหา

เวลาเริ่มต้น

1/3/2564

เวลาสิ้นสุด

24/11/2564

95รายการ

จำนวนข้อมูลทั้งหมด

วันที่บันทึกข้อมูล : ชื่อสถานที่ท่องเที่ยว : ประเภทของสถานที่ท่องเที่ยว : ที่อยู่ : พิกัด : สิ่งอำนวยความสะดวก : จำนวนที่จอดรถ (คัน) :

5/10/2021 5:21:47 PM เกาะโพธิ์

เชิงธรรมชาติ

ถ.เกาะลันตาใหญ่ อ.เกาะ

15.8700, 100.9925

มีร้านอาหาร, มีห้องน้ำ,

-

5/10/2021 5:28:59 PM ผืนเมฆ ผืนแดง

เชิงธรรมชาติ

ถ.เกาะลันตาใหญ่ อ.เกาะ

15.8700, 100.9925

-

-

5/15/2021 11:35:04 AM เกาะพีพี

เชิงธรรมชาติ

ถ.สายไทย อ.เมืองกระบี่ จ.

13.7419, 100.5215

มีบริการฟรีอินเทอร์เน็ต

-

5/17/2021 5:49:24 PM หาดทอง

เชิงธรรมชาติ

ถ.เกาะลันตาใหญ่ อ.เกาะ

7.3409, 100.3216

มีบริการฟรีอินเทอร์เน็ต

-

6/10/2021 10:54:09 A ท่าเรือแจ้หิ

เชิงธรรมชาติ

ถ.เกาะลันตาใหญ่ อ.เกาะ

7.5632, 99.0692

มีร้านอาหาร, มีห้องน้ำ,

15

6/10/2021 10:59:03 A ท่าเรือแจ้หิ

เชิงธรรมชาติ

ถ.เกาะลันตาใหญ่ อ.เกาะ

7.5632, 99.0692

มีบริการฟรีอินเทอร์เน็ต

15

6/10/2021 11:09:02 A ท่าเรือแจ้หิ

เชิงธรรมชาติ

ถ.เกาะลันตาใหญ่ อ.เกาะ

7.5632, 99.0692

มีบริการฟรีอินเทอร์เน็ต

15

6/10/2021 11:40:05 A หาดคลองไธ

เชิงธรรมชาติ

ถ.เกาะลันตาใหญ่ อ.เกาะ

7.5410, 99.0490

มีร้านอาหาร, มีห้องน้ำ,

-

6/10/2021 1:43:30 PM หาดคลองไธ

เชิงธรรมชาติ

ถ.เกาะลันตาใหญ่ อ.เกาะ

7.5439, 99.0468

มีบริการฟรีอินเทอร์เน็ต

-

6/18/2021 10:05:17 AM เกาะเหล็ก

เชิงธรรมชาติ, เชิงประวัติ

ถ.เกาะลันตาใหญ่ อ.เกาะ

7.4906, 99.1103

มีร้านอาหาร

-

รูปที่ 22 หน้าข้อมูลรายจังหวัด ของระบบจัดการข้อมูลชุมชน (Data Management Portal)

จากรูปที่ 22 แสดงหน้าข้อมูลรายจังหวัด ของระบบจัดการข้อมูลชุมชน (Data Management Portal) ซึ่งผู้บริหารข้อมูลแต่ละจังหวัด สามารถเข้าถึงเพื่อการนำไปวิเคราะห์และวางแผนการบริหารจัดการภายในจังหวัดที่ดูแลรับผิดชอบได้

3.6 การออกแบบโครงสร้างข้อมูลตามหมวดหมู่

การออกแบบและกำหนดโครงสร้างข้อมูลอย่างเหมาะสมตามหมวดหมู่ข้อมูลสำคัญ เพื่อใช้สำหรับเก็บข้อมูลในแต่ละชุมชนนั้น สามารถทำได้ผ่านระบบสร้างชุดคำถาม (Management Console) ซึ่งระบบนี้ จะอำนวยความสะดวกในการสร้างโครงสร้างข้อมูลของแต่ละหมวดหมู่ลงใน Google Cloud Firestore [2] และมีการเชื่อมโยงกับฐานข้อมูลสำหรับจัดเก็บรูปภาพจะผ่าน Google Cloud Storage พร้อมแปลงโครงสร้างข้อมูลสำหรับ Elasticsearch [4] ซึ่งช่วยให้สามารถสืบค้นข้อมูลได้อย่างรวดเร็ว

3.6.1 โครงสร้างข้อมูลของแต่ละหมวดหมู่ที่ถูกสร้างผ่าน Management Console

ลักษณะโครงสร้างข้อมูลของแต่ละหมวดหมู่ (Data Schema) ซึ่งถูกออกแบบและสร้างผ่านระบบสร้างชุดคำถาม (Management Console) โดยได้สรุปการออกแบบโครงสร้างข้อมูลจำนวน 10 หมวดหมู่ในรูปแบบตาราง เพื่อสะดวกต่อการอธิบายความหมายในแต่ละข้อคำถามดังต่อไปนี้

1. ผู้ย้ายกลับบ้าน เนื่องจากสถานการณ์โควิด

รูปแบบโครงสร้างข้อมูลหมวดหมู่ผู้ย้ายกลับบ้าน เนื่องจากสถานการณ์โควิด แสดงดังตารางที่ 1

ตารางที่ 1 ลักษณะข้อมูลหมวดหมู่ผู้ย้ายกลับบ้าน เนื่องจากสถานการณ์โควิด

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
ชื่อ-สกุล	-	Text	-	ชื่อ-สกุล ของผู้ย้ายกลับบ้าน เนื่องจากสถานการณ์โควิด
เบอร์ติดต่อ	-	Text	-	กรอกเฉพาะตัวเลขเท่านั้น
เพศ	-	Text	○ ชาย ○ หญิง ○ ไม่ระบุ	เลือกได้ 1 ตัวเลือก
อายุ (ปี)	-	Number	-	กรอกเฉพาะตัวเลขเท่านั้น
ระดับการศึกษา	ระดับการศึกษา สูงสุด หรือ ปัจจุบัน	Text	○ ประถมศึกษา ○ มัธยมศึกษา/ปวช. ○ ปวส. ○ ปริญญาตรี ○ ปริญญาโท ○ ปริญญาเอก ○ อื่น ๆ	เลือกได้ 1 ตัวเลือก ถ้าเลือก อื่น ๆ จะปรากฏ ช่องกรอกเพิ่มเติม 1 ช่อง
สถานภาพสมรส	การจดทะเบียน ตามกฎหมาย	Text	○ โสด ○ สมรส ○ อื่น ๆ	เลือกได้ 1 ตัวเลือก ถ้าเลือก อื่น ๆ จะปรากฏ ช่องกรอกเพิ่มเติม 1 ช่อง
จำนวนบุตร (ถ้ามี)	-	Number	-	กรอกเฉพาะตัวเลขเท่านั้น
ที่อยู่ปัจจุบัน	-	Text	○ จังหวัด ○ อำเภอ ○ ตำบล	แสดงตัวเลือกแบบค้นหาได้ 3 ช่อง สำหรับระบุ จังหวัด อำเภอ และตำบล *ระบบจะกรอกให้อัตโนมัติ เป็นค่าเริ่มต้น สามารถแก้ไข ได้
ย้ายมาพร้อม ครอบครัวหรือไม่	-	Text	○ ใช่ ○ ไม่ใช่	เลือกได้ 1 ตัวเลือก

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
ย้ายมาจากที่ไหน	-	Text	- จังหวัด - อำเภอ - ตำบล	แสดงตัวเลือกแบบค้นหาได้ 3 ช่อง สำหรับระบุ จังหวัด อำเภอ และตำบล
สาเหตุที่ย้ายกลับบ้าน	-	Text Array	- ตงงาน - ต้องการลดค่าใช้จ่าย - ลาออก/เปลี่ยนงาน - กังวลการระบาดของโควิดในพื้นที่เดิม - ต้องการทำงานจากที่บ้าน (Work from Home) - อื่น ๆ	เลือกได้หลายตัวเลือก ถ้าเลือก อื่น ๆ จะปรากฏ ช่องให้กรอกเพิ่มเติมได้ หลายช่อง
แผนในชีวิตที่จะทำเมื่อกลับมาอยู่บ้าน	-	Text Array	- ทำการเกษตร - ทำธุรกิจส่วนตัว - สานต่อกิจการครอบครัว - ค้าขาย - รับจ้างทั่วไป (งานไม่ประจำ) - ลูกจ้าง/พนักงาน (งานประจำ) - ยังไม่มีแผน - อื่น ๆ	เลือกได้หลายตัวเลือก ถ้าเลือก อื่น ๆ จะปรากฏ ช่องให้กรอกเพิ่มเติมได้ หลายช่อง
ความช่วยเหลือที่อยากได้รับ	-	Text Array	- หางานให้ - พักชั่วคราว - เงินกู้ดอกเบี้ยต่ำพิเศษ - เงินเยียวยา - ประกัน COVID-19 - ต้องการวัคซีนป้องกัน COVID-19 - อื่น ๆ	เลือกได้หลายตัวเลือก ถ้าเลือก อื่น ๆ จะปรากฏ ช่องให้กรอกเพิ่มเติมได้ หลายช่อง

2. แหล่งท่องเที่ยว

รูปแบบโครงสร้างข้อมูลหมวดหมู่แหล่งท่องเที่ยว แสดงดัง ตารางที่ 2

ตารางที่ 2 ลักษณะข้อมูลหมวดหมู่แหล่งท่องเที่ยว

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
ชื่อสถานที่ท่องเที่ยว	-	Text	-	
ประเภทของสถานที่ท่องเที่ยว	จุดเด่นของสถานที่	Text	○ เชิงธรรมชาติ (อุทยาน, สวนสาธารณะ, ฯลฯ) ○ เชิงประวัติศาสตร์/วัฒนธรรม ○ เชิงสุขภาพ (สปา, นวดแผนไทย, ฯลฯ) ○ เชิงกีฬา ○ เชิงการเรียนรู้ (พิพิธภัณฑ์, สวนสัตว์, ฯลฯ) ○ เมืองเก่า ○ มรดกโลก (UNESCO/Geopark) ○ อื่น ๆ	เลือกได้หลายตัวเลือก ถ้าเลือก อื่น ๆ จะปรากฏช่องให้กรอกเพิ่มเติมได้หลายช่อง
ที่อยู่	-	Text	○ จังหวัด ○ อำเภอ ○ ตำบล	แสดงตัวเลือกแบบค้นหาได้ 3 ช่อง สำหรับระบุ จังหวัด อำเภอ และตำบล *ระบบจะกรอกให้อัตโนมัติเป็นค่าเริ่มต้น
พิกัด	-	Geo Point	-	แสดงช่องกรอก Latitude, Longitude *ระบบจะกรอกให้อัตโนมัติเป็นค่าเริ่มต้น

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
สิ่งอำนวยความสะดวก	-	Text Array	○ มีบริการฟรี อินเทอร์เน็ต/Wi-Fi ○ มีร้านอาหาร ○ มีร้านเครื่องดื่ม/คาเฟ่ ○ มีห้องน้ำ ○ มีที่จอดรถ ○ มีรถโดยสารสาธารณะ ○ นำสัตว์เลี้ยงเข้าไปได้ ○ มีพื้นที่เฉพาะสำหรับเด็ก ○ มีสิ่งอำนวยความสะดวกผู้พิการ ○ มีพาหนะให้เช่า (รถกอล์ฟ, จักรยาน, ฯลฯ) ○ มีที่ชาร์จรถยนต์ไฟฟ้า ○ อื่น ๆ	เลือกได้หลายตัวเลือก ถ้าเลือก อื่น ๆ จะปรากฏช่องให้กรอกเพิ่มเติมได้หลายช่อง
จำนวนที่จอดรถ (คัน)	-	Number	-	ปรากฏเมื่อเลือก มีที่จอดรถ
เป็นสถานที่ท่องเที่ยวตามฤดูกาลหรือไม่	-	Text	○ ใช่ ○ ไม่ใช่ (ท่องเที่ยวได้ทุกฤดูกาล)	เลือกได้ 1 ตัวเลือก
ช่วงเดือนท่องเที่ยว	-	Text	-	ปรากฏเมื่อตอบ ใช่ (เป็นสถานที่ท่องเที่ยวตามฤดูกาล) โดยจะปรากฏตัวเลือก 2 ช่อง สำหรับระบุ เดือนเริ่มต้น และ เดือนสิ้นสุด
จำนวนนักท่องเที่ยวเฉลี่ยต่อปี (คน)	-	Number	-	กรอกเฉพาะตัวเลขเท่านั้น

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
จำนวนพนักงาน/ เจ้าหน้าที่ (คน)	-	Number	-	กรอกเฉพาะตัวเลขเท่านั้น
รายได้เฉลี่ยต่อปี (บาท)	-	Number	-	กรอกเฉพาะตัวเลขเท่านั้น
ผู้ดูแล/รับผิดชอบ สถานที่ (ถ้ามี)	-	Text Array	-	
เบอร์ติดต่อ (ถ้ามี)	-	Text	-	กรอกเฉพาะตัวเลขเท่านั้น สามารถเพิ่มได้หลายช่อง
เว็บไซต์ (ถ้ามี)	-	Text	-	สามารถเพิ่มได้หลายช่อง
ภาพถ่ายสถานที่ (อย่างน้อย 5 ภาพ)	-	Text Array	-	เพิ่มได้หลายรูปภาพ

3. ที่พักหรือโรงแรม

รูปแบบโครงสร้างข้อมูลหมวดหมู่ที่พักหรือโรงแรม แสดงดังตารางที่ 3

ตารางที่ 3 ลักษณะข้อมูลหมวดหมู่ที่พักหรือโรงแรม

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
ชื่อที่พัก	-	Text	-	
ประเภทที่พัก	-	Text	○ โรงแรม ○ โฮสเทล/เกสต์เฮาส์ ○ วิลล่า/รีสอร์ท/ บังกะโล ○ โฮมสเตย์ ○ พื้นที่ตั้งแคมป์ ○ อพาร์ทเมนต์/ คอนโดมิเนียม ○ บ้านพักรับรอง ○ อื่น ๆ	เลือกได้ 1 ตัวเลือก ถ้าเลือก อื่น ๆ จะปรากฏ ช่องกรอกเพิ่มเติม 1 ช่อง
ที่อยู่	-	Text	○ จังหวัด ○ อำเภอ ○ ตำบล	แสดงตัวเลือกแบบค้นหาได้ 3 ช่อง สำหรับระบุ จังหวัด อำเภอ และตำบล

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
				*ระบบจะกรอกให้อัตโนมัติ เป็นค่าเริ่มต้น
พิกัด	-	Geo Point	-	แสดงช่องกรอก Latitude, Longitude *ระบบจะกรอกให้อัตโนมัติ เป็นค่าเริ่มต้น
สิ่งอำนวยความสะดวก	-	Text Array	○ มีแผนกต้อนรับ บริการ 24 ชม. ○ มีบริการอาหาร ○ ฟรีอินเทอร์เน็ต/Wi-Fi ○ มีเครื่องปรับอากาศ ○ มีเครื่องทำน้ำอุ่น ○ รองรับการเข้าพัก พร้อมเด็กเล็ก ○ นำสัตว์เลี้ยงเข้าพัก ได้ ○ มีสิ่งอำนวยความสะดวก ผู้พิการ ○ มีบริการรถรับ-ส่ง ○ มีที่จอดรถ ○ มีห้องประชุม ○ มีพื้นที่ออกกำลังกาย ○ มีสระว่ายน้ำ ○ จองล่องหน้าแบบ ออนไลน์ได้ ○ อื่น ๆ	เลือกได้หลายตัวเลือก ถ้าเลือก อื่น ๆ จะปรากฏ ช่องให้กรอกเพิ่มเติมได้ หลายช่อง
จำนวนที่จอดรถ (คัน)	-	Number	-	ปรากฏเมื่อตอบ มีที่จอดรถ

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
ได้รับมาตรฐาน SHA	มาตรฐานที่פקกำหนดโดยกระทรวงการท่องเที่ยวและกีฬา	Boolean	○ ได้รับ ○ ไม่ได้รับ	
จำนวนผู้เข้าพักเฉลี่ยต่อปี (คน)	-	Number	-	กรอกเฉพาะตัวเลขเท่านั้น
จำนวนห้อง	-	Number	-	กรอกเฉพาะตัวเลขเท่านั้น
จำนวนพนักงาน/เจ้าหน้าที่ (คน)	-	Number	-	กรอกเฉพาะตัวเลขเท่านั้น
รายได้เฉลี่ยต่อปี (บาท)	-	Number	-	กรอกเฉพาะตัวเลขเท่านั้น
เบอร์ติดต่อ	-	Text Array	-	กรอกเฉพาะตัวเลขเท่านั้น สามารถเพิ่มได้หลายช่อง
เว็บไซต์ (ถ้ามี)	-	Text	-	สามารถเพิ่มได้หลายช่อง
ภาพถ่ายสถานที่ (อย่างน้อย 5 ภาพ)	-	Text Array	-	เพิ่มได้หลายรูปภาพ

4. ร้านอาหารในท้องถิ่น

รูปแบบโครงสร้างข้อมูลหมวดหมู่ร้านอาหารในท้องถิ่น แสดงดังตารางที่ 4

ตารางที่ 4 ลักษณะข้อมูลหมวดหมู่ร้านอาหารในท้องถิ่น

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
ชื่อร้านอาหาร	-	Text	-	
ที่ตั้งร้านอาหาร	-	Text	○ จังหวัด ○ อำเภอ ○ ตำบล	แสดงตัวเลือกแบบค้นหาได้ 3 ช่อง สำหรับระบุ จังหวัด อำเภอ และตำบล *ระบบจะกรอกให้อัตโนมัติ เป็นค่าเริ่มต้น
ตำแหน่งร้านอาหาร	-	Geo Point	-	แสดงช่องกรอก Latitude, Longitude

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
				*ระบบจะกรอกให้อัตโนมัติ เป็นค่าเริ่มต้น
เมนูที่มีเฉพาะใน ท้องถิ่น	อาหารพื้นเมือง อาหารที่เป็น เอกลักษณ์ของ ท้องถิ่น	Text Array	-	สามารถเพิ่มได้หลายช่อง
เมนูแนะนำอื่น ๆ	เมนูเด่น เมนูที่ ได้รับความนิยม ของร้าน	Text Array	-	สามารถเพิ่มได้หลายช่อง
สัญชาติอาหาร	-	Text Array	○ อาหารใต้ ○ อาหารอีสาน ○ อาหารเหนือ ○ อาหารภาคกลาง ○ อาหารมุสลิม/ฮาลาล ○ อาหารจีน ○ อาหารญี่ปุ่น ○ อาหารเกาหลี ○ อาหารเวียดนาม ○ อาหารอิตาลี ○ อาหารอินเดีย ○ อาหารนานาชาติ ○ อาหารสแกนดิเนเวีย ○ อื่น ๆ	เลือกได้หลายตัวเลือก ถ้าเลือก อื่น ๆ จะปรากฏ ช่องให้กรอกเพิ่มเติมได้ หลายช่อง
วันที่เปิดให้บริการ	-	Text Array	○ จันทร์ ○ อังคาร ○ พุธ ○ พฤหัส ○ ศุกร์ ○ เสาร์ ○ อาทิตย์	เลือกได้หลายตัวเลือก
จำนวนพนักงาน/ เจ้าหน้าที่ (คน)	-	Number	-	

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
รายได้เฉลี่ยต่อปี (บาท)	-	Number	-	
เบอร์ติดต่อ (ถ้ามี)	-	Text	-	กรอกเฉพาะตัวเลขเท่านั้น สามารถเพิ่มได้หลายช่อง
เว็บไซต์ (ถ้ามี)	-	Text	-	สามารถเพิ่มได้หลายช่อง
ข้อมูลเพิ่มเติมอื่น ๆ (ถ้ามี)	-	Text	-	สามารถใส่คำอธิบายข้อมูลเพิ่มเติมได้
ภาพบรรยากาศร้าน (อย่างน้อย 5 ภาพ)	-	Text Array	-	เพิ่มได้หลายรูปภาพ
ภาพอาหาร (อย่างน้อย 5 ภาพ)	-	Text Array	-	เพิ่มได้หลายรูปภาพ

5. อาหารที่น่าสนใจประจำถิ่น

รูปแบบโครงสร้างข้อมูลหมวดหมู่อาหารที่น่าสนใจประจำถิ่น แสดงดังตารางที่ 5

ตารางที่ 5 ลักษณะข้อมูลหมวดหมู่อาหารที่น่าสนใจประจำถิ่น

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
ชื่ออาหาร/เครื่องดื่ม	-	Text	-	
ประเภทของอาหาร/เครื่องดื่ม	-	Text	○ อาหารคาว ○ อาหารหวาน ○ เครื่องดื่ม ○ อื่น ๆ	เลือกได้ 1 ตัวเลือก ถ้าเลือก อื่น ๆ จะปรากฏช่องให้กรอก 1 ช่อง
ชื่อร้านที่จำหน่าย/ผลิต (ถ้ามี)	-	Text	-	
สถานที่ที่จำหน่าย/ผลิต	-	Text	○ จังหวัด ○ อำเภอ ○ ตำบล	แสดงตัวเลือกแบบค้นหาได้ 3 ช่อง สำหรับระบุ จังหวัด อำเภอ และตำบล *ระบบจะกรอกให้อัตโนมัติ เป็นค่าเริ่มต้น
พิกัดที่จำหน่าย/ผลิต	-	Geo Point	-	แสดงช่องกรอก Latitude, Longitude

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
				*ระบบจะกรอกให้อัตโนมัติเป็นค่าเริ่มต้น
มีที่จอดรถหรือไม่ (กรณีเป็นร้าน)	-	Text	○ มี ○ ไม่มี ○ ไม่ทราบ	
จำนวนที่จอดรถ (คัน)	-	Number	-	ปรากฏกรณีตอบว่า มีที่จอดรถ
กำลังการผลิต กิโลกรัมต่อวัน (ถ้า ทราบ)	ผลผลิตที่ได้ใน 1 วัน (กรณีเป็น ร้าน/มีการผลิต ในครัวเรือน)	Number	-	
ส่วนผสมหลัก	ซึ่งเป็นจุดเด่น ของเมนูนี้	Text Array	-	สามารถเพิ่มได้หลายช่อง
เรื่องราว ความเชื่อ ความเป็นมาของ อาหาร (ถ้ามี)	-	Text	-	
รูปภาพอาหาร (อย่างน้อย 5 ภาพ)	-	Text Array	-	เพิ่มได้หลายรูปภาพ

6. เกษตรกรในท้องถิ่น

รูปแบบโครงสร้างข้อมูลหมวดหมู่เกษตรกรในท้องถิ่น แสดงดังตารางที่ 6

ตารางที่ 6 ลักษณะข้อมูลหมวดหมู่เกษตรกรในท้องถิ่น

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
ชื่อเกษตรกร	-	Text	-	
ที่อยู่	-	Text	○ จังหวัด ○ อำเภอ ○ ตำบล	แสดงตัวเลือกแบบค้นหาได้ 3 ช่อง สำหรับระบุ จังหวัด อำเภอ และตำบล *ระบบจะกรอกให้อัตโนมัติ เป็นค่าเริ่มต้น

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
พิกัด	-	Geo Point	-	แสดงช่องกรอก Latitude, Longitude *ระบบจะกรอกให้อัตโนมัติ เป็นค่าเริ่มต้น
เบอร์ติดต่อ เกษตรกร (ถ้ามี)	-	Text	-	กรอกเฉพาะตัวเลขเท่านั้น
LINE ID (ถ้ามี)	ไอดีไลน์ของ เกษตรกร	Text	-	
Facebook (ถ้ามี)	เฟสบุ๊คของ เกษตรกร	Text	-	
เว็บไซต์ (ถ้ามี)	เว็บไซต์ส่วนตัว ของเกษตรกร	Text	-	
จำนวนแรงงาน ทั้งหมดที่ใช้ (คน)	-	Number	-	กรอกเฉพาะตัวเลขเท่านั้น
รายได้เฉลี่ยต่อปี (บาท)	-	Number	-	กรอกเฉพาะตัวเลขเท่านั้น
ใช้ระบบหรือ เทคโนโลยีทุนแรง หรือไม่	เช่น เครื่องจักร หุ่นยนต์ โดรน หรือระบบดิจิทัล สำหรับเก็บ/ วิเคราะห์ข้อมูล	Boolean	☐ ใช้ ☐ ไม่ใช่	เลือกได้ 1 ตัวเลือก
ระบบหรือ เทคโนโลยีที่ใช้	-	Text Array	-	ปรากฏเมื่อตอบว่า ใช้ระบบ หรือเทคโนโลยีทุนแรง สามารถเพิ่มได้หลายช่อง
เปิดให้บุคคลทั่วไป เข้าชมได้หรือไม่	-	Boolean	☐ เปิดให้เข้าชม ☐ ไม่เปิดให้เข้าชม	เลือกได้ 1 ตัวเลือก
พื้นที่ทำการเกษตร ทั้งหมด (ไร่ งาน วา)	กรอกจำนวนไร่ งาน และวา ตามลำดับ	Number	-	ปรากฏช่องกรอก 3 ช่อง ตามลำดับ สำหรับกรอก ไร่ งาน และวา โดยให้กรอกเฉพาะตัวเลข เท่านั้น

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
รูปแบบการใช้งานพื้นที่ทำการเกษตร	-	Text Array	○ สวน ○ ไร่ ○ นา ○ ปศุสัตว์ ○ ประมง/บ่อเลี้ยงสัตว์ ○ สระน้ำ ○ อื่น ๆ	เลือกได้หลายตัวเลือก ถ้าเลือก อื่น ๆ จะปรากฏช่องให้กรอกเพิ่มเติมได้หลายช่อง
ขนาดพื้นที่ทำการเกษตร (ไร่ งาน วา)	-	Number	-	ปรากฏช่องกรอก สอดคล้องตามการเลือกรูปแบบการใช้งานพื้นที่ทำการเกษตร
ผลผลิตที่ได้ต่อปี (ตัน, ตัว)	-	Number	-	
ต้นทุนในการผลิตต่อปี (บาท)	-	Number	-	
รายการพืช/สัตว์ที่ทำการเกษตร	ระบุชื่อชนิดของพืชที่ปลูก และ/หรือ สัตว์ที่เลี้ยง	Text Array	-	ปรากฏสอดคล้องตามการเลือกรูปแบบการใช้งานพื้นที่ทำการเกษตร มี Autocomplete ในการแนะนำชื่อที่สอดคล้อง
ภาพถ่ายพื้นที่/ผลผลิต/เกษตรกร	-	Text Array	-	เพิ่มได้หลายรูปภาพ

7. พืชในท้องถิ่น

รูปแบบโครงสร้างข้อมูลหมวดหมู่พืชในท้องถิ่น แสดงดังตารางที่ 7

ตารางที่ 7 ลักษณะข้อมูลหมวดหมู่พืชในท้องถิ่น

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
ชื่อชนิดของพืช	เช่น ผักเห็ด ฟ้าทะลายโจร มะรุม	Text	-	

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
สถานที่พบ	-	Text	- จังหวัด - อำเภอ - ตำบล	แสดงตัวเลือกแบบค้นหาได้ 3 ช่อง สำหรับระบุ จังหวัด อำเภอ และตำบล *ระบบจะกรอกให้อัตโนมัติ เป็นค่าเริ่มต้น
ตำแหน่งที่พบ	-	Geo Point	-	แสดงช่องกรอก Latitude, Longitude *ระบบจะกรอกให้อัตโนมัติ เป็นค่าเริ่มต้น
ลักษณะการใช้งาน	-	Text Array	- เป็นอาหารสัตว์ - เป็นอาหารมนุษย์ - ประดับตกแต่ง - เพื่อจำหน่าย - ใช้เป็นยา - ใช้บำรุงผิว - ใช้บำรุงเส้นผม - ใช้ไล่แมลง - ใช้ในพิธีกรรม - ใช้กลั่น - ใช้ทำสี - อื่น ๆ	เลือกได้หลายตัวเลือก ถ้าเลือก อื่น ๆ จะปรากฏ ช่องให้กรอกเพิ่มเติมได้ หลายช่อง
จำนวนต้นที่พบ โดยประมาณ (ถ้า ทราบ)	-	Number	-	กรอกเฉพาะตัวเลขเท่านั้น
ช่วงเดือนที่พบได้ มาก (ถ้าทราบ)	-	Text	-	แสดงตัวเลือก 2 ช่อง คือ เดือนเริ่มต้น และ เดือน สิ้นสุด
มีเจ้าของหรือไม่	-	Text	- ไม่มี - มี - ไม่ทราบ	
ชื่อ-สกุล เจ้าของ (ถ้าทราบ)	-	Text	-	ปรากฏกรณีตอบว่า มี เจ้าของ
ชื่อ-สกุล ผู้ให้ข้อมูล (ถ้าทราบ)	-	Text	-	

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
ภาพรวม	ภาพถ่ายทั้งต้น/ บริเวณที่ปลูก	Text Array	-	
ใบ	-	Text Array	-	
ต้น	-	Text Array	-	
ดอก	-	Text Array	-	
ผล	-	Text Array	-	

8. สัตว์ในท้องถิ่น

รูปแบบโครงสร้างข้อมูลหมวดหมู่สัตว์ในท้องถิ่น แสดงดังตารางที่ 8

ตารางที่ 8 ลักษณะข้อมูลหมวดหมู่สัตว์ในท้องถิ่น

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
ชื่อชนิดของสัตว์	เช่น นกเงือก เสือดาว แมว วิเชียรมาศ	Text	-	
ประเภทของสัตว์	-	Text	- สัตว์เลี้ยงลูกด้วยนม - สัตว์น้ำ - สัตว์ปีก - สัตว์เลื้อยคลาน - สัตว์ครึ่งบกครึ่งน้ำ - สัตว์ไม่มีกระดูกสันหลัง	เลือกได้ 1 ตัวเลือก
สถานที่พบ	-	Text	- จังหวัด - อำเภอ - ตำบล	แสดงตัวเลือกแบบค้นหาได้ 3 ช่อง สำหรับระบุ จังหวัด อำเภอ และตำบล *ระบบจะกรอกให้อัตโนมัติ เป็นค่าเริ่มต้น สามารถแก้ไขได้
ตำแหน่งที่พบ	-	Geo Point	-	แสดงช่องกรอก Latitude, Longitude

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
				*ระบบจะกรอกให้อัตโนมัติ เป็นค่าเริ่มต้น สามารถแก้ไข ได้
จำนวนสัตว์เพศผู้	จำนวนที่พบใน พื้นที่	Number	-	มากกว่าหรือเท่ากับศูนย์
จำนวนสัตว์เพศเมีย	จำนวนที่พบใน พื้นที่	Number	-	มากกว่าหรือเท่ากับศูนย์
จำนวนสัตว์ไม่ทราบ เพศ	จำนวนที่พบใน พื้นที่	Number	-	มากกว่าหรือเท่ากับศูนย์
ลักษณะการใช้งาน	-	Text Array	○ เพื่อจำหน่าย ○ ใช้เป็นพ่อพันธุ์/แม่พันธุ์ ○ เป็นสัตว์เลี้ยง ○ ใช้แรงงาน ○ เป็นอาหารมนุษย์ ○ เป็นอาหารสัตว์ ○ อื่น ๆ	เลือกได้หลายตัวเลือก ถ้าเลือก อื่น ๆ จะปรากฏ ช่องให้กรอกเพิ่มเติมได้ หลายช่อง
ช่วงเดือนที่พบได้ มาก (ถ้าทราบ)	-	Text	-	
สัตว์มีเจ้าของหรือไม่	-	Text	○ ไม่มี ○ มี ○ ไม่ทราบ	
ชื่อ-สกุล เจ้าของ (ถ้าทราบ)	-	Text	-	
ภาพถ่ายของสัตว์ (อย่างน้อย 5 ภาพ)	-	Text Array	-	

9. ภูมิปัญญาท้องถิ่น

รูปแบบโครงสร้างข้อมูลหมวดหมู่ภูมิปัญญาท้องถิ่น แสดงดังตารางที่ 9

ตารางที่ 9 ลักษณะข้อมูลหมวดหมู่ภูมิปัญญาท้องถิ่น

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
ชื่อภูมิปัญญาท้องถิ่น	-	Text	-	
ประเภทของภูมิปัญญาท้องถิ่น	-	Text	○ การเกษตรเกี่ยวกับพืช ○ การเกษตรเกี่ยวกับสัตว์ ○ การเกษตรเกี่ยวกับการประมง ○ การเกษตรเกี่ยวกับป่า ○ เครื่องมือเครื่องใช้ทางการเกษตร ○ การทำอาหาร ○ การเก็บรักษาและถนอมอาหาร ○ การแพทย์และสมุนไพร ○ งานอุตสาหกรรมและหัตถกรรม ○ ทรัพยากรธรรมชาติและสิ่งแวดล้อม ○ ศิลปะพื้นบ้าน/วัฒนธรรม ○ ภาษาและวรรณกรรม ○ ประเพณี ความเชื่อ พิธีกรรม ศาสนา ○ อื่น ๆ	เลือกได้ 1 ตัวเลือก ถ้าเลือก อื่น ๆ จะ ปรากฏช่องให้กรอก 1 ช่อง
สถานที่สำรวจ	-	Text	○ จังหวัด ○ อำเภอ ○ ตำบล	แสดงตัวเลือกแบบ ค้นหาได้ 3 ช่อง สำหรับระบุ จังหวัด อำเภอ และตำบล *ระบบจะกรอกให้อัตโนมัติเป็นค่าเริ่มต้น
ตำแหน่งที่สำรวจ	-	Geo Point	-	แสดงช่องกรอก Latitude, Longitude *ระบบจะกรอกให้อัตโนมัติเป็นค่าเริ่มต้น

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
สถานะของภูมิ ปัญหาท้องถิ่น	-	Text	- ภูมิปัญหาท้องถิ่นดั้งเดิม - ภูมิปัญหาท้องถิ่นที่ได้รับการพัฒนาและต่อยอด - นวัตกรรมที่คิดขึ้นมาใหม่	
ลักษณะเด่นหรือ เอกลักษณ์ของภูมิ ปัญหาท้องถิ่น	ระบุ 1 ลักษณะเด่น ต่อ 1 ช่อง	Text Array	-	เพิ่มได้หลายช่อง
วัตถุดิบ/เครื่องมือ ที่ นำมาใช้ในภูมิ ปัญหาท้องถิ่น (ถ้า มี)	ระบุ 1 วัตถุดิบ/ เครื่องมือ ต่อ 1 ช่อง	Text Array	-	เพิ่มได้หลายช่อง
รายละเอียดของภูมิ ปัญหาท้องถิ่น ที่ สามารถเปิดเผยได้	-	Text Array	-	เพิ่มได้หลายช่อง
การประชาสัมพันธ์/ เผยแพร่	-	Text	- ยังไม่เคยมีการเผยแพร่/ใช้เฉพาะบุคคล - เผยแพร่/ใช้งานเฉพาะในชุมชน - มีการเผยแพร่/ใช้งานนอกชุมชน	เลือกได้ 1 ตัวเลือก
การจัดสิทธิบัตร	จดทะเบียน แสดงความ เป็นเจ้าของ สิ่งประดิษฐ์/ ทรัพย์สิน ทางปัญญา	Text	- จดสิทธิบัตรแล้ว - ไม่ได้จดสิทธิบัตร - ไม่ทราบ	เลือกได้ 1 ตัวเลือก
ชื่อ-สกุล ผู้ให้ข้อมูล (ถ้าทราบ)	-	Text	-	
ภาพถ่าย (อย่างน้อย 5 ภาพ)	ผลงาน/ผู้ คิดค้น/ สถานที่	Text Array	-	เพิ่มได้หลาย รูปภาพ

10. แหล่งน้ำในท้องถิ่น

รูปแบบโครงสร้างข้อมูลหมวดหมู่แหล่งน้ำในท้องถิ่น แสดงดังตารางที่ 10

ตารางที่ 10 ลักษณะข้อมูลหมวดหมู่แหล่งน้ำในท้องถิ่น

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
ชื่อแหล่งน้ำ	-	Text	-	
ที่อยู่	-	Text	○ จังหวัด ○ อำเภอ ○ ตำบล	แสดงตัวเลือกแบบค้นหาได้ 3 ช่อง สำหรับระบุ จังหวัด อำเภอ และตำบล *ระบบจะกรอกให้อัตโนมัติเป็นค่าเริ่มต้น
ตำแหน่ง	-	Geo Point	-	แสดงช่องกรอก Latitude, Longitude *ระบบจะกรอกให้อัตโนมัติเป็นค่าเริ่มต้น
ประเภทของแหล่งน้ำ	-	Text	○ แม่น้ำ ○ ห้วย ○ หนอง ○ คลอง ○ บึง ○ เชื่อน ○ อ่างเก็บน้ำ ○ ฝาย	เลือกได้ 1 ตัวเลือก
ช่วงเวลาที่เริ่มน้ำ	-	Text	-	แสดงตัวเลือก 2 ช่อง คือ เดือนเริ่มต้น และ เดือนสิ้นสุด
ช่วงเวลาที่น้ำแห้ง (ถ้ามี)	-	Text	-	แสดงตัวเลือก 2 ช่อง คือ เดือนเริ่มต้น และ เดือนสิ้นสุด
ช่วงเวลาที่น้ำท่วม (ถ้ามี)	-	Text	-	แสดงตัวเลือก 2 ช่อง คือ เดือนเริ่มต้น และ เดือนสิ้นสุด
คุณภาพของน้ำ	-	Text	○ ดีมาก (ใช้อุปโภคบริโภคได้) ○ ปานกลาง (ใช้ทำการประมง/กักทางน้ำได้) ○ น้อย (ใช้ทำการเกษตร/อุตสาหกรรมได้)	เลือกได้ 1 ตัวเลือก

คำถาม	คำอธิบาย	ชนิดข้อมูล	ตัวเลือก	หมายเหตุ
			○ เน่าเสีย (ใช้เฉพาะการคมนาคม)	
อุณหภูมิน้ำ ตอนเช้า (°C)	(5.00 - 7.00 น.)	Number	-	ไม่บังคับกรอก ผู้กรอกต้องศึกษาวิธีการตรวจวัดแหล่งน้ำแต่ละประเภทอย่างถูกต้อง หรือตรวจวัดโดยผู้เชี่ยวชาญเฉพาะด้าน
อุณหภูมิน้ำ ตอนเที่ยง (°C)	(11.00 - 13.00 น.)	Number	-	
อุณหภูมิน้ำ ตอนเย็น (°C)	(16.00 - 18.00 น.)	Number	-	
อุณหภูมิน้ำ สูงสุด (°C)	อุณหภูมิร้อนที่สุด	Number	-	
อุณหภูมิน้ำ ต่ำสุด (°C)	อุณหภูมิต่ำสุด	Number	-	
ขนาดพื้นที่แหล่งน้ำ กรณีประเภทผิวดิน (ตร.ม.) (ถ้าทราบ)	-	Number	-	
ชื่อ-สกุล ผู้ให้ข้อมูล (ถ้ามี)	-	Text	-	เพิ่มได้หลายช่อง
ภาพถ่ายแหล่งน้ำ (อย่างน้อย 5 ภาพ)	-	Text Array	-	เพิ่มได้หลายรูปภาพ

จากลักษณะโครงสร้างการเก็บข้อมูลซึ่งได้ออกแบบให้เหมาะสมกับการเก็บข้อมูลในแต่ละหมวดหมู่ แสดงให้เห็นว่าระบบจัดการข้อมูลความหลากหลายนี้ รองรับการสร้างและจัดเก็บข้อมูลตาม JSON Schema Definition ดังรายละเอียดในหัวข้อ 0 ซึ่งมีการเชื่อมโยงเพื่อแสดงผลข้อมูลผ่าน User App เพื่อบันทึกข้อมูล และ Management Console เพื่อสร้างหรือปรับปรุงข้อมูลแต่ละหมวดหมู่ รวมถึงได้แปลงรูปแบบเพื่อจัดเก็บลง Elasticsearch [4] เพื่อการสืบค้นย้อนหลังอย่างรวดเร็ว

3.6.2 โครงสร้างข้อมูลซึ่งจัดเก็บลงใน Google Firestore

โครงสร้างข้อมูล (Data Schema) ที่มีการจัดเก็บลงใน Google Firestore [2] ซึ่งอยู่ในรูปแบบของ JSON [19] โดยมีความยืดหยุ่นในการบันทึกหมวดหมู่ข้อมูลซึ่งมีจำนวนข้อมูลที่ไม่เท่ากันได้ โดยหน้าตาข้อมูลแต่ละหมวดหมู่ จัดแบ่งโครงสร้างเป็นสองกลุ่ม ประกอบไปด้วย โครงสร้างข้อมูลสำหรับจัดเก็บคำถาม และโครงสร้างข้อมูลสำหรับจัดเก็บคำตอบ จึงขอนำโครงสร้างของหมวดพืชในท้องถิ่น ซึ่งมีรายละเอียดดังต่อไปนี้

3.6.2.1 โครงสร้างข้อมูลสำหรับจัดเก็บคำถาม

โครงสร้างข้อมูลสำหรับจัดเก็บคำถามสำหรับหมวดหมู่พืชในท้องถิ่น ซึ่งอยู่ในรูปแบบของ JSON [19] โดยจัดเก็บลงใน Google Firestore [2] มีลักษณะดังต่อไปนี้

ตารางที่ 11 โครงสร้างข้อมูลคำถาม หมวดหมู่พืชในท้องถิ่น ซึ่งจัดเก็บใน Google Firestore

```
{
  "result": {
    "createdAt": 1657700752245,
    "category": "--public--local-plant",
    "version": "2.0",
    "sections": [
      {
        "metadata": null,
        "questions": [
          {
            "metadata": {
              "visibility": {
                "action": "hide",
                "dependencies": []
              },
              "autocomplete": {
                "questionId": "plantName",
                "questionVersion": "2.0",
                "questionCategory": "--public--local-plant",
                "userScope": "all",
                "limit": 20,
                "dataSources": [
                  "external",
                  "submission"
                ]
              }
            },
            "question": "ชื่อชนิดของพืช",
            "answer": {
              "maxItems": 1,
              "metadata": {
                "regex": ""
              },
              "stripNull": true,
              "emptyString": "strip",
              "placeholder": "ชื่อชนิดของพืช",
            }
          }
        ]
      }
    ]
  }
}
```

```

        "type": "text",
        "required": true
    },
    "imageUrl": null,
    "description": "เช่น ผักเหลียง ฟ้าทะลายโจร มะรุม",
    "id": "plantName"
  }
],
"description": "เช่น ผักเหลียง ฟ้าทะลายโจร มะรุม",
"title": "ชื่อชนิดของพืช"
},
{
  "metadata": null,
  "questions": [
    {
      "metadata": null,
      "question": "สถานที่พบ",
      "answer": {
        "metadata": {
          "useLocationService": true
        },
        "type": "address",
        "fields": {
          "country": {
            "required": false
          },
          "province": {
            "required": false
          },
          "street": {
            "required": false
          },
          "district": {
            "required": false
          },
          "postalCode": {
            "required": false
          },
          "village": {
            "required": false
          },
          "subDistrict": {
            "required": false
          }
        }
      },
      "required": true
    },
    "imageUrl": null,
    "description": null,
    "id": "address"
  ]
},
"description": null,
"title": "สถานที่พบ"
},
{
  "metadata": null,
  "questions": [
    {
      "metadata": null,

```

```

        "question": "ตำแหน่งที่พบ",
        "answer": {
            "metadata": {
                "useLocationService": true
            },
            "type": "geo-point",
            "required": true
        },
        "imageUrl": null,
        "description": null,
        "id": "location"
    }
],
"description": null,
"title": "ตำแหน่งที่พบ"
},
{
    "metadata": null,
    "questions": [
        {
            "metadata": null,
            "question": "ลักษณะการใช้งาน",
            "answer": {
                "allowOther": true,
                "metadata": null,
                "type": "checkboxes",
                "items": [
                    {
                        "checked": false,
                        "title": "เป็นอาหารสัตว์",
                        "value": "เป็นอาหารสัตว์"
                    },
                    {
                        "checked": false,
                        "title": "เป็นอาหารมนุษย์",
                        "value": "เป็นอาหารมนุษย์"
                    },
                    {
                        "checked": false,
                        "title": "ประดับตกแต่ง",
                        "value": "ประดับตกแต่ง"
                    },
                    {
                        "checked": false,
                        "title": "เพื่อจำหน่าย",
                        "value": "เพื่อจำหน่าย"
                    },
                    {
                        "checked": false,
                        "title": "ใช้เป็นยา",
                        "value": "ใช้เป็นยา"
                    },
                    {
                        "checked": false,
                        "title": "ใช้บำรุงผิว",

```

```

        "value": "ใช้บำรุงผิว"
    },
    {
        "checked": false,
        "title": "ใช้บำรุงเส้นผม",
        "value": "ใช้บำรุงเส้นผม"
    },
    {
        "checked": false,
        "title": "ใช้โล่แมลง",
        "value": "ใช้โล่แมลง"
    },
    {
        "checked": false,
        "title": "ใช้ในพิธีกรรม",
        "value": "ใช้ในพิธีกรรม"
    },
    {
        "checked": false,
        "title": "ใช้กลิ่น",
        "value": "ใช้กลิ่น"
    },
    {
        "checked": false,
        "title": "ใช้ทำสี",
        "value": "ใช้ทำสี"
    }
    ],
    "required": true
},
"imageUrl": null,
"description": null,
"id": "application"
}
],
"description": null,
"title": "ลักษณะการใช้งาน"
},
{
    "metadata": null,
    "questions": [
        {
            "metadata": null,
            "answer": {
                "maxItems": 1,
                "metadata": null,
                "min": 0,
                "max": 999999,
                "stripNull": true,
                "placeholder": "จำนวนต้น",
                "type": "number",
                "required": false
            },
            "question": "จำนวนต้นที่พบโดยประมาณ (ถ้าทราบ)",
            "imageUrl": null,
            "description": null,

```

```

        "id": "plantCount"
      },
    ],
    "description": null,
    "title": "จำนวนต้นที่พบโดยประมาณ (ถ้าทราบ)"
  },
  {
    "metadata": {
      "uiComponentParams": {
        "endMonth": {
          "questionId": "endAvailableMonth"
        },
        "beginMonth": {
          "questionId": "beginAvailableMonth"
        }
      },
      "uiComponent": "MonthRange"
    },
    "questions": [
      {
        "metadata": null,
        "question": "ช่วงเดือนที่พบได้มาก (ถ้าทราบ)",
        "answer": {
          "maxItems": 1,
          "metadata": null,
          "min": 0,
          "max": 12,
          "stripNull": true,
          "placeholder": null,
          "type": "number",
          "required": false
        },
        "imageUrl": null,
        "description": null,
        "id": "beginAvailableMonth"
      },
      {
        "metadata": null,
        "question": "ช่วงเดือนที่พบได้มาก (ถ้าทราบ)",
        "answer": {
          "maxItems": 1,
          "metadata": null,
          "min": 0,
          "max": 12,
          "stripNull": true,
          "placeholder": null,
          "type": "number",
          "required": false
        },
        "imageUrl": null,
        "description": null,
        "id": "endAvailableMonth"
      }
    ],
    "description": null,
    "title": "ช่วงเดือนที่พบได้มาก (ถ้าทราบ) "
  },
  {
    "metadata": null,
    "questions": [

```

```

        "metadata": null,
        "answer": {
            "allowOther": false,
            "metadata": null,
            "defaultItem": null,
            "type": "radio",
            "items": [
                {
                    "title": "ไม่มี",
                    "value": "ไม่มี"
                },
                {
                    "title": "มี",
                    "value": "มี"
                },
                {
                    "title": "ไม่ทราบ",
                    "value": "ไม่ทราบ"
                }
            ],
            "required": true
        },
        "question": "มีเจ้าของหรือไม่",
        "imageUrl": null,
        "description": null,
        "id": "haveOwner"
    }
],
"description": null,
"title": "มีเจ้าของหรือไม่"
},
{
    "metadata": null,
    "questions": [
        {
            "metadata": {
                "visibility": {
                    "action": "show",
                    "dependencies": [
                        {
                            "questionId": "haveOwner",
                            "value": "มี"
                        }
                    ]
                }
            },
            "requireWhenShow": false
        },
        "answer": {
            "maxItems": 1,
            "metadata": {
                "regex": ""
            },
            "stripNull": true,
            "emptyString": "strip",
            "placeholder": "ชื่อ-สกุล เจ้าของ",
            "type": "text",
            "required": false
        },
    ],

```

```

        "question": "ชื่อ-สกุล เจ้าของ (ถ้าทราบ)",
        "imageUrl": null,
        "description": null,
        "id": "owner"
    },
    ],
    "description": null,
    "title": "ชื่อ-สกุล เจ้าของ (ถ้าทราบ)"
},
{
    "metadata": null,
    "questions": [
        {
            "metadata": null,
            "question": "ชื่อ-สกุล ผู้ให้ข้อมูล (ถ้าทราบ)",
            "answer": {
                "maxItems": 1,
                "metadata": {
                    "regex": ""
                },
                "stripNull": true,
                "emptyString": "strip",
                "placeholder": "ชื่อ-สกุล ผู้ให้ข้อมูล",
                "type": "text",
                "required": false
            },
            "imageUrl": null,
            "description": null,
            "id": "informant"
        }
    ],
    "description": null,
    "title": "ชื่อ-สกุล ผู้ให้ข้อมูล (ถ้าทราบ)"
},
{
    "metadata": null,
    "questions": [
        {
            "metadata": null,
            "question": "ภาพรวม",
            "answer": {
                "maxItems": 10,
                "viewer": null,
                "metadata": null,
                "type": "file",
                "required": true,
                "accept": [
                    "image/*"
                ]
            },
            "imageUrl": null,
            "description": "ภาพถ่ายทั้งต้น/บริเวณที่ปลูก",
            "id": "overviewPhoto"
        }
    ],
    "description": "ภาพถ่ายทั้งต้น/บริเวณที่ปลูก",
    "title": "ภาพรวม"
},
{

```

```

"metadata": null,
"questions": [
  {
    "metadata": null,
    "answer": {
      "maxItems": 10,
      "viewer": null,
      "metadata": null,
      "type": "file",
      "required": false,
      "accept": [
        "image/*"
      ]
    },
    "question": "ใบ",
    "imageUrl": null,
    "description": null,
    "id": "leafPhoto"
  }
],
"description": null,
"title": "ใบ"
},
{
  "metadata": null,
  "questions": [
    {
      "metadata": null,
      "answer": {
        "viewer": null,
        "maxItems": 10,
        "metadata": null,
        "type": "file",
        "required": false,
        "accept": [
          "image/*"
        ]
      },
      "question": "ต้น",
      "imageUrl": null,
      "description": null,
      "id": "trunkPhoto"
    }
  ],
  "description": null,
  "title": "ต้น"
},
{
  "metadata": null,
  "questions": [
    {
      "metadata": null,
      "answer": {
        "maxItems": 10,
        "viewer": null,
        "metadata": null,
        "type": "file",
        "required": false,
        "accept": [
          "image/*"
        ]
      }
    }
  ]
}

```

```

    },
    "question": "ดอก",
    "imageUrl": null,
    "description": null,
    "id": "flowerPhoto"
  }
],
"description": null,
"title": "ดอก"
},
{
  "metadata": null,
  "questions": [
    {
      "metadata": null,
      "question": "ผล",
      "answer": {
        "viewer": null,
        "maxItems": 10,
        "metadata": null,
        "type": "file",
        "required": false,
        "accept": [
          "image/*"
        ]
      },
      "imageUrl": null,
      "description": null,
      "id": "fruitPhoto"
    }
  ],
  "description": null,
  "title": "ผล"
},
{
  "metadata": null,
  "questions": [
    {
      "metadata": {
        "visibility": {
          "action": "show",
          "dependencies": []
        }
      },
      "question": "Submit Location",
      "answer": {
        "metadata": {
          "useLocationService": true
        },
        "type": "geo-point",
        "required": false
      },
      "imageUrl": null,
      "description": null,
      "id": "submitLocation"
    }
  ],
  "description": null,
  "title": "Submit Location"
}
],

```

```

    "updatedAt": 1657700752245
  }
}

```

จากตารางที่ 11 แสดงลักษณะโครงการการเก็บข้อมูลของหมวดหมู่พีชในท้องถิ่น ซึ่งสามารถจัดแบ่งชุดถามเป็น section ได้ โดยแต่ละ section สามารถกำหนดข้อมูลคำถามและกำหนดลักษณะของคำตอบได้ตามต้องการ โดยรองรับชนิดข้อมูลหลากหลายรูปแบบ เช่น ข้อความ (Text string) วันที่และเวลา (Date and time) ข้อมูลไฟล์ (Bytes) ตัวเลข (Integer/Floating-point) พิกัดตำแหน่ง (Geographical point หรือ Geo-point) ข้อมูลชนิด Object (Map) เป็นต้น โดยลักษณะข้อมูลรูปแบบ JSON [19] จากตารางที่ 11 นี้ สามารถสรุปเป็นตารางอย่างง่ายได้ดังตารางที่ 12

ตารางที่ 12 คำอธิบายโครงสร้างข้อมูลคำถามซึ่งจัดเก็บใน Google Firestore

ชื่อข้อมูล	ชนิดข้อมูล	คำอธิบาย
createdAt	Date and time	วันและเวลาที่สร้างเอกสารหรือสร้างคำถาม
category	Text string	รหัสประจำหมวดหมู่ข้อมูล
version	Text string	หมายเลขเวอร์ชันของหมวดหมู่ข้อมูล
sections	Array	กลุ่มของ section ใช้สำหรับจัดกลุ่มคำถามตามต้องการ
โครงสร้างแต่ละ section ประกอบไปด้วย		
title	Text string	คำอธิบายเพิ่มเติมสำหรับข้อคำถาม (question)
description	Text string	คำอธิบายประกอบ section
metadata	Map	คำอธิบายเพิ่มเติม (ระบุ Key และ Value)
questions	Array	ชุดของคำถาม
โครงสร้างแต่ละ question ประกอบไปด้วย		
question	Text string	ระบุคำถาม
description	Text string	คำอธิบายประกอบ question
imageUrl	Text string	ที่อยู่ไฟล์รูปภาพ ซึ่งเชื่อมโยงกับ Google Cloud Storage
metadata	Map	คำอธิบายเพิ่มเติม (ระบุ Key และ Value)
visibility	Map	Property ของ metadata สำหรับตั้งค่าการแสดงผลคำถาม
action	Boolean	Property ของ visibility ใช้ระบุว่าแสดงหรือซ่อนคำถาม
dependencies	Array	Property ของ visibility ใช้ระบุเงื่อนไขการแสดงผลหรือซ่อนคำถาม
autocomplete	Map	Property ของ metadata ใช้กำหนดคุณสมบัติสำหรับ autocomplete

ชื่อข้อมูล	ชนิดข้อมูล	คำอธิบาย
questionId	Text string	Property ของ autocomplete ใช้กำหนดรหัสของคำถาม
questionVersion	Text string	Property ของ autocomplete สำหรับระบุเวอร์ชันของคำถาม
questionCategory	Text string	Property ของ autocomplete สำหรับหมวดหมู่ของคำถาม
userScope	Text string	Property ของ autocomplete สำหรับระบุประเภทของผู้ใช้งาน
limit	Integer	Property ของ autocomplete สำหรับระบุจำนวนผลลัพธ์
dataSources	Array	Property ของ autocomplete สำหรับระบุกลุ่มข้อมูลที่ใช้ค้นหา
answer	Map	กำหนดลักษณะของคำตอบ
type	Text string	Property ของ answer สำหรับระบุชนิดข้อมูลของคำตอบ
placeholder	Text string	Property ของ answer สำหรับระบุคำอธิบายในช่องคำตอบ
required	Boolean	Property ของ answer สำหรับตั้งค่าการบังคับตอบคำถาม
maxItems	Integer	Property ของ answer สำหรับจำกัดจำนวนคำตอบ
metadata	Map	Property ของ answer สำหรับระบุข้อมูลเพิ่มเติม
regex	Text string	Property ของ metadata ใน answer สำหรับกำหนดรูปแบบการพิมพ์คำตอบ
stripNull	Boolean	Property ของ answer ใช้ตั้งค่ากรณีคำตอบเป็นค่าว่าง (Null) สามารถกำหนดให้ลบทิ้งได้
emptyString	Text string	Property ของ answer ใช้ตั้งค่ากรณีคำตอบเป็นข้อความว่าง ("") สามารถกำหนดให้ลบทิ้งได้

3.6.2.2 โครงสร้างข้อมูลสำหรับจัดเก็บคำตอบ

โครงสร้างข้อมูลสำหรับจัดเก็บคำตอบสำหรับหมวดหมู่พืชในท้องถิ่น ซึ่งอยู่ในรูปแบบของ JSON [19] โดยจัดเก็บลงใน Google Firestore [2] มีลักษณะดังต่อไปนี้

ตารางที่ 13 โครงสร้างข้อมูลคำตอบ หมวดหมู่พืชในท้องถิ่น ซึ่งจัดเก็บใน Google Firestore

<pre>{ "recordId": "QDVLCT-789806", "questionCategory": "--public--local-plant", "questionVersion": "2.0", "answers": { "plantName": ["ข้าว"], "address": { "subDistrict": "สีสุก", "district": "แก่งสนามนาง", </pre>

```

        "province": "นครราชสีมา",
        "street": null,
        "village": null,
        "postalCode": null,
        "country": null
    },
    "location": {
        "latitude": 15.6724,
        "longitude": 102.1587
    },
    "application": [
        {
            "value": "#NAME?",
            "isOther": true
        }
    ],
    "plantCount": [],
    "beginAvailableMonth": [
        6
    ],
    "endAvailableMonth": [
        11
    ],
    "haveOwner": {
        "value": "มี",
        "isOther": false
    },
    "owner": [
        "#NAME"
    ],
    "informant": [
        "#NAME"
    ],
    "overviewPhoto": [
        "gs://bio-c-assets/answers/--public--local-plant/1.0/overviewPhoto/plant01.jpeg"
    ],
    "leafPhoto": [
        "gs://bio-c-assets/answers/--public--local-plant/1.0/leafPhoto/plant02.jpeg"
    ],
    "trunkPhoto": [
        "gs://bio-c-assets/answers/--public--local-plant/1.0/trunkPhoto/plant03.jpeg"
    ],
    "flowerPhoto": [
        "gs://bio-c-assets/answers/--public--local-plant/1.0/flowerPhoto/plant04.jpeg"
    ],
    "fruitPhoto": [
        "gs://bio-c-assets/answers/--public--local-plant/1.0/fruitPhoto/plant04.jpeg"
    ]
    },
    "createdAt": 1620537665576,
    "updatedAt": 1661824066725
}

```

จากตารางที่ 13 แสดงลักษณะการเก็บข้อมูลคำตอบของหมวดหมู่พืชในท้องถิ่น ซึ่งจัดเก็บอยู่ใน Google Firestore [2] ซึ่งโครงสร้างของคำตอบได้ถูกกำหนดรูปแบบไว้ ในส่วนของโครงสร้างข้อมูลคำถาม โดยเมื่อมีการบันทึกข้อมูลคำตอบเข้ามาจะปรากฏตัวอย่างการเก็บข้อมูลในรูปแบบ JSON [19] ดังตารางที่ 13 นี้ ซึ่งสามารถสรุปเป็นตารางอย่างง่ายได้ดังตารางที่ 14

ตารางที่ 14 คำอธิบายโครงสร้างข้อมูลคำตอบซึ่งจัดเก็บใน Google Firestore

ชื่อข้อมูล	ชนิดข้อมูล	คำอธิบาย
recordId	Text string	รหัสเอกสารหรือรายการของคำตอบ
questionCategory	Text string	ระบุหมวดหมู่ของคำถาม
questionVersion	Text string	ระบุเวอร์ชันของคำถาม
answers	Map	ข้อมูลคำตอบของรายการ
plantName	Array	ชื่อชนิดของพืช โดยเก็บได้หลายชื่อ
address	Map	ข้อมูลสถานที่พบชนิดของพืช
location	Geo point	ข้อมูลพิกัดตำแหน่งที่พบ
application	Array	ข้อมูลการใช้ประโยชน์จากพืช
plantCount	Array	ข้อมูลจำนวนที่พบ
beginAvailableMonth	Array	เดือนเริ่มต้นที่พบ
endAvailableMonth	Array	เดือนสิ้นสุดที่พบ
haveOwner	Map	มีเจ้าของหรือไม่
owner	Array	กรณีมีเจ้าของ ให้ระบุชื่อเพิ่มเติม
informant	Array	กรณีมีชื่อผู้ให้ข้อมูล สามารถระบุเพิ่มเติมได้
overviewPhoto	Array	ข้อมูลที่อยู่ของไฟล์ภาพรวมของพืช
leafPhoto	Array	ข้อมูลที่อยู่ของไฟล์ภาพใบ
trunkPhoto	Array	ข้อมูลที่อยู่ของไฟล์ภาพลำต้น
flowerPhoto	Array	ข้อมูลที่อยู่ของไฟล์ภาพดอก
fruitPhoto	Array	ข้อมูลที่อยู่ของไฟล์ภาพผล
createdAt	Date and time	วันและเวลาที่สร้างเอกสารหรือสร้างรายการ
updatedAt	Date and time	วันและเวลาที่ปรับปรุงรายการ

จากโครงสร้างข้อมูลหมวดหมู่พืชในท้องถิ่นดังตารางที่ 11 และตารางที่ 13 ซึ่งมีการจัดเก็บลงในฐานข้อมูล Google Firestore [2] ซึ่งเป็นฐานข้อมูลชนิด NoSQL [9] รูปแบบ

Document Database นั้น แสดงให้เห็นว่าสามารถจัดเก็บรูปแบบข้อมูลได้หลากหลายชนิดตามโครงสร้างการออกแบบในหัวข้อ 0 ข้อ 7 โดยโครงสร้างในแต่ละเอกสารไม่ว่าจะเป็นส่วนโครงสร้างการเก็บคำถามหรือส่วนโครงสร้างการเก็บคำตอบ ไม่จำเป็นต้องมีรูปแบบที่เหมือนกัน ทำให้สามารถเก็บข้อมูลได้อย่างยืดหยุ่น

3.6.3 โครงสร้างข้อมูลรูปแบบ Dynamic Mapping

โครงสร้างข้อมูลรูปแบบ Dynamic Mapping เป็นโครงสร้างข้อมูลที่ Elasticsearch [4] จัดเตรียมให้แบบอัตโนมัติ ซึ่งผู้จัดการฐานข้อมูลไม่จำเป็นต้องกำหนด Mappings [54] โดยขอ ยกตัวอย่างหน้าตาข้อมูลจากหมวดหมู่พืชในท้องถิ่น เพื่อเปรียบเทียบให้เห็นความแตกต่างระหว่างโครงสร้างข้อมูลที่ผ่านการแปลงโดย Data Collection Schema จากหัวข้อ 0 และโครงสร้างข้อมูลรูปแบบ Dynamic Mapping ซึ่งมีลักษณะหน้าตาดังต่อไปนี้

ตารางที่ 15 โครงสร้างข้อมูลแบบ Dynamic Mapping ซึ่งถูกสร้างอัตโนมัติโดย Elasticsearch

```
{
  "test-local-plant": {
    "aliases": {},
    "mappings": {
      "properties": {
        "answers": {
          "properties": {
            "address": {
              "properties": {
                "country": {
                  "type": "text",
                  "fields": {
                    "keyword": {
                      "type": "keyword",
                      "ignore_above": 256
                    }
                  }
                }
              }
            },
            "district": {
              "type": "text",
              "fields": {
                "keyword": {
                  "type": "keyword",
                  "ignore_above": 256
                }
              }
            },
            "postalCode": {
              "type": "text",
              "fields": {
                "keyword": {
                  "type": "keyword",
                  "ignore_above": 256
                }
              }
            }
          }
        }
      }
    }
  }
}
```

```

    },
    "province": {
      "type": "text",
      "fields": {
        "keyword": {
          "type": "keyword",
          "ignore_above": 256
        }
      }
    },
    "subDistrict": {
      "type": "text",
      "fields": {
        "keyword": {
          "type": "keyword",
          "ignore_above": 256
        }
      }
    }
  },
  "application": {
    "properties": {
      "isOther": {
        "type": "boolean"
      },
      "value": {
        "type": "text",
        "fields": {
          "keyword": {
            "type": "keyword",
            "ignore_above": 256
          }
        }
      }
    }
  },
  "beginAvailableMonth": {
    "type": "long"
  },
  "endAvailableMonth": {
    "type": "long"
  },
  "flowerPhoto": {
    "type": "text",
    "fields": {
      "keyword": {
        "type": "keyword",
        "ignore_above": 256
      }
    }
  },
  "fruitPhoto": {
    "type": "text",
    "fields": {
      "keyword": {
        "type": "keyword",
        "ignore_above": 256
      }
    }
  },
  "haveOwner": {
    "properties": {
      "isOther": {

```

```

        "type": "boolean"
      },
      "value": {
        "type": "text",
        "fields": {
          "keyword": {
            "type": "keyword",
            "ignore_above": 256
          }
        }
      }
    },
    "informant": {
      "type": "text",
      "fields": {
        "keyword": {
          "type": "keyword",
          "ignore_above": 256
        }
      }
    },
    "leafPhoto": {
      "type": "text",
      "fields": {
        "keyword": {
          "type": "keyword",
          "ignore_above": 256
        }
      }
    },
    "location": {
      "properties": {
        "lat": {
          "type": "float"
        },
        "lon": {
          "type": "float"
        }
      }
    },
    "overviewPhoto": {
      "type": "text",
      "fields": {
        "keyword": {
          "type": "keyword",
          "ignore_above": 256
        }
      }
    },
    "owner": {
      "type": "text",
      "fields": {
        "keyword": {
          "type": "keyword",
          "ignore_above": 256
        }
      }
    },
    "plantCount": {
      "type": "long"
    },
    "plantName": {
      "type": "text",

```

```

        "fields": {
            "keyword": {
                "type": "keyword",
                "Ignore_above": 256
            }
        },
        "trunkPhoto": {
            "type": "text",
            "fields": {
                "keyword": {
                    "type": "keyword",
                    "ignore_above": 256
                }
            }
        },
        "createdAt": {
            "type": "date"
        },
        "questionCategory": {
            "type": "text",
            "fields": {
                "keyword": {
                    "type": "keyword",
                    "ignore_above": 256
                }
            }
        },
        "questionVersion": {
            "type": "text",
            "fields": {
                "keyword": {
                    "type": "keyword",
                    "ignore_above": 256
                }
            }
        },
        "recordId": {
            "type": "text",
            "fields": {
                "keyword": {
                    "type": "keyword",
                    "ignore_above": 256
                }
            }
        },
        "updatedAt": {
            "type": "date"
        }
    }
}

```

จากตารางที่ 15 แสดงลักษณะโครงสร้างข้อมูลแบบ Dynamic Mapping ซึ่งถูกสร้างอัตโนมัติโดย Elasticsearch [4] เนื่องจากไม่มีการกำหนด Mappings [54] ขณะเริ่มต้นสร้าง Index [43] สำหรับเก็บข้อมูลหมวดหมู่พีชในท้องถิ่น Elasticsearch [4] จึงพิจารณาลักษณะของ

ข้อมูลที่น่าเข้ามา จากนั้นจึงกำหนด Mappings [54] ให้โดยอัตโนมัติ ซึ่งมีลักษณะหน้าตาของโครงสร้างข้อมูลดังตารางที่ 15 นี้ โดยสามารถสรุปเป็นตารางอย่างง่ายดังตารางที่ 16 ซึ่งชนิดข้อมูลของ Elasticsearch [4] ที่เกี่ยวข้องนั้นประกอบไปด้วย ข้อความ (text) คำ (keyword) ตัวเลข (long/float) วันที่ (date) ข้อมูล JSON object (object) พิกัดตำแหน่ง (geo_point) เป็นต้น

ตารางที่ 16 คำอธิบายโครงสร้างข้อมูลแบบ Dynamic Mapping

ชื่อข้อมูล	ชนิดข้อมูล	คำอธิบาย
answers	object	โครงสร้างการเก็บข้อมูลแต่ละรายการ
properties	object	โครงสร้างของแต่ละ Property
address	object	เป็น Property สำหรับเก็บข้อมูลที่อยู่สถานที่พบ
application	text	เป็น Property สำหรับเก็บข้อมูลการนำไปใช้ประโยชน์
beginAvailableMonth	long	เป็น Property สำหรับเก็บเดือนเริ่มต้นที่พบ
endAvailableMonth	long	เป็น Property สำหรับเก็บเดือนสิ้นสุดที่พบ
haveOwner	text	เป็น Property สำหรับเก็บข้อมูลเจ้าของของพืช
owner	text	เป็น Property สำหรับเก็บข้อมูลชื่อเจ้าของ
informant	text	เป็น Property สำหรับเก็บข้อมูลชื่อผู้ให้ข้อมูล
overviewPhoto	text	เป็น Property สำหรับเก็บที่อยู่ของไฟล์ภาพรวมของพืช
leafPhoto	text	เป็น Property สำหรับเก็บที่อยู่ของไฟล์ภาพใบ
trunkPhoto	text	เป็น Property สำหรับเก็บที่อยู่ของไฟล์ภาพลำต้น
flowerPhoto	text	เป็น Property สำหรับเก็บที่อยู่ของไฟล์ภาพดอก
fruitPhoto	text	เป็น Property สำหรับเก็บที่อยู่ของไฟล์ภาพผล
location	text	เป็น Property สำหรับเก็บพิกัดตำแหน่งที่พบ
plantCount	text	เป็น Property สำหรับเก็บจำนวนที่พบ
plantName	text	เป็น Property สำหรับเก็บชื่อชนิดของพืช
questionCategory	text	เป็น Property สำหรับเก็บชื่อหมวดหมู่ของพืช
questionVersion	text	เป็น Property สำหรับเก็บเวอร์ชันของหมวดหมู่พืช
recordId	text	เป็น Property สำหรับเก็บ
createdAt	date	เป็น Property สำหรับเก็บวันที่และเวลาที่สร้างรายการ
updatedAt	date	เป็น Property สำหรับเก็บวันที่และเวลาที่ปรับปรุงรายการ

จากตารางที่ 16 ข้อมูล (Property) ที่มีการกำหนดชนิดของคำตอบเป็นข้อมูล (text) จะมีการกำหนดคุณลักษณะของฟิลด์ (fields) ประกอบเป็นค่าเริ่มต้นเหมือนกันทุก Property โดยกำหนด keyword object ซึ่งภายในกำหนดชนิด (type) เป็น keyword และจำกัดความยาวของข้อมูล (ignore_above) เป็น 256 ตัวอักษร

3.6.4 โครงสร้างข้อมูลซึ่งใช้งาน Thai Tokenizer

โครงสร้างข้อมูลที่กำหนดวิธีการตัดคำไทยโดยใช้ Thai Tokenizer ซึ่งสามารถกำหนดวิธีการตัดคำนี้ได้ผ่านการระบุ mappings [51] ส่วนของ analyzer เป็น thai ให้กับชุดข้อมูลที่จะทำการสืบค้น ประกอบคำสั่งการสร้าง Index [43] ใหม่ โดย properties ที่ไม่ได้กำหนด mappings [54] หรือคุณสมบัติแบบเฉพาะเจาะจงไว้ Elasticsearch [4] จะสร้าง mappings [54] ให้โดยอัตโนมัติ โดยจะยกตัวอย่างหน้าตาข้อมูลในหมวดหมู่พืชในท้องถิ่น ลักษณะหน้าตาดังต่อไปนี้

ตารางที่ 17 โครงสร้างข้อมูลซึ่งกำหนดวิธีการตัดคำไทยโดย Thai Tokenizer

```
{
  "test-local-plant-thai-analyzer": {
    "aliases": {},
    "mappings": {
      "properties": {
        "answers": {
          "properties": {
            "address": {
              "properties": {
                "country": {
                  "type": "text",
                  "fields": {
                    "keyword": {
                      "type": "keyword",
                      "ignore_above": 256
                    }
                  }
                }
              }
            },
            "district": {
              "type": "text",
              "fields": {
                "keyword": {
                  "type": "keyword",
                  "ignore_above": 256
                }
              }
            },
            "postalCode": {
              "type": "text",
              "fields": {
                "keyword": {
                  "type": "keyword",
                  "ignore_above": 256
                }
              }
            }
          }
        }
      }
    }
  }
}
```

```

    },
    "province": {
      "type": "text",
      "fields": {
        "keyword": {
          "type": "keyword",
          "ignore_above": 256
        }
      }
    },
    "subDistrict": {
      "type": "text",
      "fields": {
        "keyword": {
          "type": "keyword",
          "ignore_above": 256
        }
      }
    }
  },
  "application": {
    "properties": {
      "isOther": {
        "type": "boolean"
      },
      "value": {
        "type": "text",
        "fields": {
          "keyword": {
            "type": "keyword",
            "ignore_above": 256
          }
        }
      }
    }
  },
  "beginAvailableMonth": {
    "type": "long"
  },
  "endAvailableMonth": {
    "type": "long"
  },
  "flowerPhoto": {
    "type": "text",
    "fields": {
      "keyword": {
        "type": "keyword",
        "ignore_above": 256
      }
    }
  },
  "fruitPhoto": {
    "type": "text",
    "fields": {
      "keyword": {
        "type": "keyword",
        "ignore_above": 256
      }
    }
  },
  "haveOwner": {
    "properties": {
      "isOther": {

```

```

        "type": "boolean"
      },
      "value": {
        "type": "text",
        "fields": {
          "keyword": {
            "type": "keyword",
            "ignore_above": 256
          }
        }
      }
    },
    "informant": {
      "type": "text",
      "fields": {
        "keyword": {
          "type": "keyword",
          "ignore_above": 256
        }
      }
    },
    "leafPhoto": {
      "type": "text",
      "fields": {
        "keyword": {
          "type": "keyword",
          "ignore_above": 256
        }
      }
    },
    "location": {
      "properties": {
        "lat": {
          "type": "float"
        },
        "lon": {
          "type": "float"
        }
      }
    },
    "overviewPhoto": {
      "type": "text",
      "fields": {
        "keyword": {
          "type": "keyword",
          "ignore_above": 256
        }
      }
    },
    "owner": {
      "type": "text",
      "fields": {
        "keyword": {
          "type": "keyword",
          "ignore_above": 256
        }
      }
    },
    "plantCount": {
      "type": "long"
    },
    "plantName": {
      "type": "text",

```

```

        "fields": {
          "keyword": {
            "type": "keyword",
            "ignore_above": 256
          }
        },
        "analyzer": "thai"
      },
      "trunkPhoto": {
        "type": "text",
        "fields": {
          "keyword": {
            "type": "keyword",
            "ignore_above": 256
          }
        }
      }
    },
    "createdAt": {
      "type": "date"
    },
    "questionCategory": {
      "type": "text",
      "fields": {
        "keyword": {
          "type": "keyword",
          "ignore_above": 256
        }
      }
    },
    "questionVersion": {
      "type": "text",
      "fields": {
        "keyword": {
          "type": "keyword",
          "ignore_above": 256
        }
      }
    },
    "recordId": {
      "type": "text",
      "fields": {
        "keyword": {
          "type": "keyword",
          "ignore_above": 256
        }
      }
    },
    "updatedAt": {
      "type": "date"
    }
  }
}

```

จากตารางที่ 17 แสดงลักษณะโครงสร้างข้อมูลหมวดหมู่พืชในท้องถิ่น ซึ่งออกแบบเพื่อใช้งานกับ Elasticsearch [4] โดยกำหนดวิธีการตัดคำไทยโดยใช้ Thai Tokenizer ซึ่งมีลักษณะหน้าตาของโครงสร้างข้อมูลดังตารางที่ 17 นี้ โดยสามารถสรุปเป็นตารางอย่างง่ายดังตารางที่ 18

ตารางที่ 18 คำอธิบายโครงสร้างข้อมูลซึ่งกำหนดวิธีการตัดคำไทยโดย Thai Tokenizer

ชื่อข้อมูล	ชนิดข้อมูล	คำอธิบาย
answers	object	โครงสร้างการเก็บข้อมูลแต่ละรายการ
properties	object	โครงสร้างของแต่ละ Property
address	object	เป็น Property สำหรับเก็บข้อมูลที่อยู่สถานที่พบ
application	text	เป็น Property สำหรับเก็บข้อมูลการนำไปใช้ประโยชน์
beginAvailableMonth	long	เป็น Property สำหรับเก็บเดือนเริ่มต้นที่พบ
endAvailableMonth	long	เป็น Property สำหรับเก็บเดือนสิ้นสุดที่พบ
haveOwner	text	เป็น Property สำหรับเก็บข้อมูลเจ้าของของพืช
owner	text	เป็น Property สำหรับเก็บข้อมูลชื่อเจ้าของ
informant	text	เป็น Property สำหรับเก็บข้อมูลชื่อผู้ให้ข้อมูล
overviewPhoto	text	เป็น Property สำหรับเก็บที่อยู่ของไฟล์ภาพรวมของพืช
leafPhoto	text	เป็น Property สำหรับเก็บที่อยู่ของไฟล์ภาพใบ
trunkPhoto	text	เป็น Property สำหรับเก็บที่อยู่ของไฟล์ภาพลำต้น
flowerPhoto	text	เป็น Property สำหรับเก็บที่อยู่ของไฟล์ภาพดอก
fruitPhoto	text	เป็น Property สำหรับเก็บที่อยู่ของไฟล์ภาพผล
location	text	เป็น Property สำหรับเก็บพิกัดตำแหน่งที่พบ
plantCount	text	เป็น Property สำหรับเก็บจำนวนที่พบ
plantName	text	เป็น Property สำหรับเก็บชื่อชนิดของพืช เพิ่มการใช้งาน Thai Tokenizer โดยการระบุ Property ชื่อ analyzer เป็น thai
questionCategory	text	เป็น Property สำหรับเก็บชื่อหมวดหมู่ของพืช
questionVersion	text	เป็น Property สำหรับเก็บเวอร์ชันของหมวดหมู่พืช
recordId	text	เป็น Property สำหรับเก็บ
createdAt	date	เป็น Property สำหรับเก็บวันที่และเวลาที่สร้างรายการ
updatedAt	date	เป็น Property สำหรับเก็บวันที่และเวลาที่ปรับปรุงรายการ

จากตารางที่ 18 ข้อมูล (Property) ที่มีการกำหนดชนิดของคำตอบเป็นข้อมูล (text) จะมีการกำหนดคุณลักษณะของฟิลด์ (fields) ประกอบเป็นค่าเริ่มต้นเหมือนกันทุก Property โดยกำหนด keyword object ซึ่งภายในกำหนดชนิด (type) เป็น keyword และจำกัดความยาวของข้อมูล (ignore_above) เป็น 256 ตัวอักษร โดยได้กำหนดการใช้งาน Thai Tokenizer ให้กับ Property ชื่อ plantName ซึ่งจะใช้ในการสืบค้นในการทดลองโดยเฉพาะ โดยเพิ่มการระบุ Property ชื่อ analyzer เป็น thai ให้กับ plantName property ดังกล่าว เพื่อใช้คุณสมบัติการตัดคำสำหรับภาษาไทยซึ่ง Elasticsearch [4] เปิดให้บริการได้

3.6.5 โครงสร้างข้อมูลรูปแบบ Custom Mapping

โครงสร้างข้อมูลซึ่งถูกแปลงสำหรับสำหรับฐานข้อมูล Elasticsearch [4] ซึ่งเป็นโครงสร้างข้อมูล ซึ่งออกแบบสำหรับจัดเก็บข้อมูลเพื่อใช้ในการสืบค้นได้อย่างรวดเร็วโดยเฉพาะ (Custom Mapping) โดยผ่านการแปลงจากส่วน Data Collection Schema ซึ่งมีการกำหนดวิธีการตัดคำภาษาไทยโดยใช้ N-gram Tokenizer รูปแบบ N-gram2,3 โดยจะยกตัวอย่างหน้าตาข้อมูลในหมวดหมู่พืชในท้องถิ่น ลักษณะหน้าตาดังต่อไปนี้

ตารางที่ 19 โครงสร้างข้อมูลสำหรับการค้นหา (Custom Mapping)

```
{
  "bio_c_ans--public--local-plant_2.0_current": {
    "mappings": {
      "properties": {
        "answers": {
          "properties": {
            "address": {
              "properties": {
                "country": {
                  "type": "keyword",
                  "store": true,
                  "fields": {
                    "analyzed": {
                      "type": "text",
                      "analyzer": "ngram_2_3"
                    }
                  }
                },
                "district": {
                  "type": "keyword",
                  "store": true,
                  "fields": {
                    "analyzed": {
                      "type": "text",
                      "analyzer": "ngram_2_3"
                    }
                  }
                }
              }
            }
          }
        }
      }
    }
  },
}
```

```

        "postalCode": {
            "type": "keyword",
            "store": true,
            "fields": {
                "analyzed": {
                    "type": "text",
                    "analyzer": "postal_code"
                }
            }
        },
        "province": {
            "type": "keyword",
            "store": true,
            "fields": {
                "analyzed": {
                    "type": "text",
                    "analyzer": "ngram_2_3"
                }
            }
        },
        "street": {
            "type": "keyword",
            "store": true,
            "fields": {
                "analyzed": {
                    "type": "text",
                    "analyzer": "ngram_2_3"
                }
            }
        },
        "subDistrict": {
            "type": "keyword",
            "store": true,
            "fields": {
                "analyzed": {
                    "type": "text",
                    "analyzer": "ngram_2_3"
                }
            }
        },
        "village": {
            "type": "keyword",
            "store": true,
            "fields": {
                "analyzed": {
                    "type": "text",
                    "analyzer": "ngram_2_3"
                }
            }
        }
    },
    "application": {
        "type": "nested",
        "properties": {
            "isOther": {
                "type": "boolean",
                "store": true
            },
            "value": {
                "type": "keyword",
                "store": true,
                "fields": {
                    "analyzed": {

```

```

        "type": "text",
        "analyzer": "ngram_2_3"
    },
    },
    "copy_to": [
        "answers.application__value"
    ]
}
},
"application__value": {
    "type": "keyword",
    "store": true,
    "fields": {
        "analyzed": {
            "type": "text",
            "analyzer": "ngram_2_3"
        }
    }
},
"beginAvailableMonth": {
    "type": "double",
    "store": true
},
"endAvailableMonth": {
    "type": "double",
    "store": true
},
"haveOwner": {
    "properties": {
        "isOther": {
            "type": "boolean",
            "store": true
        },
        "value": {
            "type": "keyword",
            "store": true,
            "fields": {
                "analyzed": {
                    "type": "text",
                    "analyzer": "ngram_2_3"
                }
            }
        }
    }
},
"informant": {
    "type": "text",
    "fields": {
        "raw": {
            "type": "keyword"
        }
    },
    "analyzer": "ngram_2_3"
},
"location": {
    "type": "geo_point",
    "store": true
},
"owner": {
    "type": "text",
    "fields": {
        "raw": {
            "type": "keyword"
        }
    }
}

```

```

    },
    "analyzer": "ngram_2_3"
  },
  "plantCount": {
    "type": "double",
    "store": true
  },
  "plantName": {
    "type": "text",
    "fields": {
      "raw": {
        "type": "keyword"
      }
    }
  },
  "analyzer": "ngram_2_3"
}
},
"createdAt": {
  "type": "date",
  "store": true
},
"questionCategory": {
  "type": "keyword",
  "store": true
},
"questionVersion": {
  "type": "keyword",
  "store": true
},
"recordId": {
  "type": "keyword",
  "store": true
},
"updatedAt": {
  "type": "date",
  "store": true
}
}
}
}
```

จากตารางที่ 19 แสดงลักษณะโครงสร้างข้อมูลหมวดหมู่พีชในท้องถิ่น (Custom Mapping) ซึ่งถูกแปลงมาจาก Data Collection Schema ซึ่งได้ออกแบบไว้เพื่อใช้งานกับ Elasticsearch [4] ในการค้นหาหรือข้อความอย่างมีประสิทธิภาพมากยิ่งขึ้น ซึ่งมีลักษณะหน้าตาของโครงสร้างข้อมูลดังตารางที่ 19 นี้ โดยสามารถสรุปเป็นตารางอย่างง่ายดังตารางที่ 20

ตารางที่ 20 คำอธิบายโครงสร้างข้อมูลแบบ Custom Mapping

ชื่อข้อมูล	ชนิดข้อมูล	คำอธิบาย
answers	object	โครงสร้างการเก็บข้อมูลแต่ละรายการ

ชื่อข้อมูล	ชนิดข้อมูล	คำอธิบาย
properties	object	โครงสร้างของแต่ละ Property
address	object	เป็น Property สำหรับเก็บข้อมูลที่อยู่สถานที่พบ เช่น ถนน ตำบล อำเภอ จังหวัด เป็นต้น โดยแต่ละข้อมูลย่อยนี้ ได้เพิ่มฟิลด์ analyzer เป็น ngram_2_3 และกำหนด store เป็น true เพื่อการสืบค้นข้อมูลได้อย่างรวดเร็ว
application	nested	เป็น Property สำหรับระบุการนำไปใช้ประโยชน์ โดยเพิ่มฟิลด์ analyzer เป็น ngram_2_3 และกำหนด store เป็น true เพื่อการสืบค้นข้อมูลได้อย่างรวดเร็ว โดยการกำหนดชนิดข้อมูลเป็นรูปแบบ nested เพื่อการเชื่อมโยงความสัมพันธ์กับโครงสร้างด้านใน
application__value	text	เป็น Property สำหรับเก็บข้อมูลการนำไปใช้ประโยชน์ที่ได้ระบุไว้ โดยเพิ่มฟิลด์ analyzer เป็น ngram_2_3 c
beginAvailableMonth	double	เป็น Property สำหรับเก็บเดือนเริ่มต้นที่พบ
endAvailableMonth	double	เป็น Property สำหรับเก็บเดือนสิ้นสุดที่พบ
haveOwner	object	เป็น Property สำหรับเก็บข้อมูลเจ้าของของพืช
owner	text	เป็น Property สำหรับเก็บข้อมูลชื่อเจ้าของ โดยเพิ่มฟิลด์ analyzer เป็น ngram_2_3
informant	text	เป็น Property สำหรับเก็บข้อมูลชื่อผู้ให้ข้อมูล โดยเพิ่มฟิลด์ analyzer เป็น ngram_2_3
location	geo_point	เป็น Property สำหรับเก็บข้อมูลพิกัดตำแหน่งที่พบ
plantCount	double	เป็น Property สำหรับระบุจำนวนพืชที่พบ
plantName	text	เป็น Property สำหรับระบุชื่อชนิดของพืช โดยเพิ่มฟิลด์ analyzer เป็น ngram_2_3
questionCategory	keyword	เป็น Property สำหรับเก็บชื่อหมวดหมู่ของพืช โดยเพิ่มฟิลด์ analyzer เป็น ngram_2_3 และกำหนด store เป็น true เพื่อการสืบค้นข้อมูลได้อย่างรวดเร็ว
questionVersion	keyword	เป็น Property สำหรับเก็บเวอร์ชันของหมวดหมู่พืช และกำหนด store เป็น true เพื่อการสืบค้นข้อมูลได้อย่างรวดเร็ว
recordId	keyword	เป็น Property สำหรับเก็บรหัสของเอกสารหรือรายการ และกำหนด store เป็น true เพื่อการสืบค้นข้อมูลได้อย่างรวดเร็ว
createdAt	date	เป็น Property สำหรับเก็บวันที่และเวลาที่สร้างรายการ และกำหนด store เป็น true เพื่อการสืบค้นข้อมูลได้อย่างรวดเร็ว

ชื่อข้อมูล	ชนิดข้อมูล	คำอธิบาย
updatedAt	date	เป็น Property สำหรับเก็บวันที่และเวลาที่ปรับปรุงรายการ และกำหนด store เป็น true เพื่อการสืบค้นข้อมูลได้อย่างรวดเร็ว

จากตารางที่ 20 ข้อมูล (Property) ที่มีการกำหนดชนิดของคำตอบเป็นข้อมูล (text) ได้เพิ่มการกำหนดคุณลักษณะของวิเคราะห์ข้อความในรูปแบบ Ngram Tokenizer โดยกำหนด Property ชื่อ analyzer เป็น ngram_2_3 และเพิ่ม Property ชื่อ store เป็น true ให้กับ Property ที่ต้องการให้ค่าที่มีการจัดเรียงพร้อมใช้งานเมื่อมีการสืบค้น ซึ่งจะช่วยให้ค้นหาข้อมูลได้อย่างรวดเร็วมากยิ่งขึ้น แต่จะใช้พื้นที่เก็บข้อมูลมากขึ้นเช่นกัน

3.7 วิธีการทดสอบประสิทธิภาพในการค้นหาด้วยคำหรือข้อความ

หัวข้อวิทยานิพนธ์นี้ มุ่งเน้นไปที่หัวข้อการสืบค้นด้วยคำหรือข้อความ จึงได้ออกแบบการทดลองสำหรับเปรียบเทียบประสิทธิภาพการค้นหาของแต่ละรูปแบบโครงสร้างข้อมูล จึงแบ่งหัวข้อการทดสอบออกเป็นสามรูปแบบดังหัวข้อ 3.7.1 ถึง 3.7.3 ดังต่อไปนี้

3.7.1 การทดลองที่ 1 การทดสอบความถูกต้องจากการค้นหาด้วยคำหรือข้อความ

วัตถุประสงค์ของการทดลองที่ 1 เพื่อเปรียบเทียบความถูกต้องสอดคล้องกันระหว่างโครงสร้างข้อมูลแบบ Dynamic Mapping และโครงสร้างที่แปลงมาจาก Data Collection Schema (Custom Mapping) โดยพิจารณาจากผลลัพธ์จากการค้นหาด้วยคำหรือข้อความใน 1 ลำดับ 5 ลำดับ และ 10 ลำดับแรก โดยข้อมูลที่น่าสนใจทดลองมาจากหมวดหมู่พีชในท้องถิ่น 1,952,571 รายการ โดยหัวข้อการทดสอบจะเป็นแบ่งออกเป็น 3 ข้อย่อยดังต่อไปนี้

1. การทดสอบค้นหาด้วยคำหรือข้อความโดยใช้ Dynamic Mapping

วิธีการทดสอบค้นหาข้อความ โดยใช้ Dynamic Mapping ซึ่งถูกสร้างโดยระบบอัตโนมัติของ Elasticsearch [4] โดยใช้ข้อมูลจากหมวดหมู่พีชในท้องถิ่น ซึ่งมีลักษณะโครงสร้างข้อมูลดังหัวข้อ 3.6.3 โครงสร้างข้อมูลรูปแบบ Dynamic Mapping วิธีการทดสอบมีขั้นตอนดังต่อไปนี้

- (1) นำออกข้อมูลหมวดหมู่พีชในท้องถิ่น จากส่วนเก็บข้อมูลดิบซึ่งอยู่ใน Google Firestore [2] รูปแบบ JSON [19]

- (2) นำเข้าข้อมูลดังกล่าวเข้าสู่ Elasticsearch [4] แบบไม่กำหนด Mapping โดย Elasticsearch [4] จะพิจารณาจากข้อมูลที่นำเข้าและสร้าง Mappings [54] ให้อัตโนมัติ (Dynamic Mapping)
- (3) เมื่อนำเข้าข้อมูลสู่ Elasticsearch [4] เรียบร้อยแล้ว จึงเริ่มค้นหาคำหรือข้อความ ผ่าน Query DSL โดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “สัก”
- (4) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ต้นสัก”
- (5) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “สักทอง”
- (6) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ฟ้าทะลายโจร”
- (7) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ฟ้าทะลาย”
- (8) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ทะลายโจร”
- (9) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ถั่วฝักยาว”
- (10) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ถั่วฝัก”
- (11) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ฝักยาว”
- (12) บันทึกผลการค้นหาทั้งหมดลงในตารางบันทึกผล และจัดทำกราฟสำหรับเปรียบเทียบค่าความแม่นยำที่ได้ จากนั้นจึงสรุปผลการทดลอง

2. การทดสอบค้นหาคำหรือข้อความโดยใช้ Thai Tokenizer

วิธีทดสอบการค้นหาคำหรือข้อความผ่านโครงสร้างข้อมูลที่กำหนดวิธีการตัดคำไทย โดยใช้ Thai Tokenizer ซึ่งสามารถกำหนดวิธีการตัดคำนี้ได้ผ่านการระบุ mappings [54] ส่วนของ analyzer เป็น thai ให้กับชุดข้อมูลที่จะทำการสืบค้น ประกอบคำสั่งการสร้าง Index [43] ใหม่ โดย properties ที่ไม่ได้กำหนด mappings [54] หรือคุณสมบัติแบบเฉพาะเจาะจงไว้ Elasticsearch [4]

จะสร้าง mappings [54] ให้โดยอัตโนมัติ โดยใช้ข้อมูลจากหมวดหมู่พีชในท้องถิ่น ซึ่งมีลักษณะโครงสร้างข้อมูลดังหัวข้อ 3.6.4 โครงสร้างข้อมูลซึ่งใช้งาน Thai Tokenizer วิธีการทดสอบมีขั้นตอนดังต่อไปนี้

- (1) นำออกข้อมูลหมวดหมู่พีชในท้องถิ่น จากส่วนเก็บข้อมูลดิบซึ่งอยู่ใน Google Firestore [2] รูปแบบ JSON [19]
- (2) นำเข้าข้อมูลดังกล่าวเข้าสู่ Elasticsearch [4] โดยกำหนดวิธีการตัดคำภาษาไทยซึ่งใช้ Thai Tokenizer โดยการระบุ mappings [54] ส่วนของ analyzer เป็น thai
- (3) เมื่อนำเข้าข้อมูลสู่ Elasticsearch [4] เรียบร้อยแล้ว จึงเริ่มค้นหาคำหรือข้อความ ผ่าน Query DSL โดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “สัก”
- (4) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ต้นสัก”
- (5) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “สักทอง”
- (6) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ฟ้าทะลายโจร”
- (7) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ฟ้าทะลาย”
- (8) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ทะลายโจร”
- (9) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ถั่วฝักยาว”
- (10) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ถั่วฝัก”
- (11) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ฝักยาว”
- (12) บันทึกผลการค้นหาทั้งหมดลงในตารางบันทึกผล และจัดทำกราฟสำหรับเปรียบเทียบค่าความแม่นยำที่ได้ จากนั้นจึงสรุปผลการทดลอง

3. การทดสอบค้นหาด้วยคำหรือข้อความโดยใช้ Custom Mapping

วิธีการทดสอบค้นหาข้อความ โดยใช้ Mapping ตามรูปแบบ Data Schema Collection (Custom Mapping) ซึ่งออกแบบให้แปลงโครงสร้างเพื่อการค้นหาได้อย่างรวดเร็วและให้ผลลัพธ์ที่สอดคล้องกับคำค้นหามากขึ้นกว่าการใช้ Mapping ซึ่งถูกสร้างโดยระบบอัตโนมัติของ Elasticsearch [4] (Dynamic Mapping) โดยใช้ข้อมูลจากหมวดหมู่พีชในท้องถิ่น ซึ่งมีลักษณะโครงสร้างข้อมูลดังหัวข้อ 3.6.5 โครงสร้างข้อมูลรูปแบบ Custom Mapping วิธีการทดสอบมีขั้นตอนดังต่อไปนี้

- (1) นำออกข้อมูลหมวดหมู่พีชในท้องถิ่น จากส่วนเก็บข้อมูลดิบซึ่งอยู่ใน Google Firestore [2] รูปแบบ JSON [19]
- (2) นำเข้าข้อมูลดังกล่าวเข้าสู่ Elasticsearch [4] โดยกำหนด Mapping ซึ่งแปลงมาจาก Data Collection Schema
- (3) เมื่อนำเข้าข้อมูลสู่ Elasticsearch [4] เรียบร้อยแล้ว จึงเริ่มค้นหาคำหรือข้อความ ผ่าน Query DSL โดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “สัก”
- (4) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ต้นสัก”
- (5) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “สักทอง”
- (6) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ฟ้าทะลายโจร”
- (7) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ฟ้าทะลาย”
- (8) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ทะลายโจร”
- (9) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ถั่วฝักยาว”
- (10) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ถั่วฝัก”

- (11) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ฝักยาว”
- (12) บันทึกผลการค้นหาทั้งหมดลงในตารางบันทึกผล และจัดทำกราฟสำหรับเปรียบเทียบค่าความแม่นยำที่ได้ จากนั้นจึงสรุปผลการทดลอง

3.7.2 การทดลองที่ 2 การทดสอบประสิทธิภาพด้านความเร็วในการค้นหาข้อมูล

วัตถุประสงค์ของการทดลองที่ 2 เพื่อเปรียบเทียบความเร็วในการค้นหาข้อมูลระหว่างโครงสร้างข้อมูลแบบ Dynamic Mapping และโครงสร้างที่แปลงมาจาก Data Collection Schema โดยข้อมูลที่นำมาใช้ทดลองมาจากข้อมูลจากหมวดหมู่พืชในท้องถิ่น จำนวน 1,952,571 รายการ โดยหัวข้อการทดสอบจะเป็นแบ่งออกเป็น 3 ช้อย่อยดังต่อไปนี้

1. การทดสอบความเร็วในการค้นหาคำหรือข้อความโดยใช้ Dynamic Mapping

วิธีการทดสอบความเร็วในการค้นหาคำหรือข้อความ โดยใช้ Dynamic Mapping ซึ่งถูกสร้างโดยระบบอัตโนมัติของ Elasticsearch [4] โดยใช้ข้อมูลจากหมวดหมู่พืชในท้องถิ่น ซึ่งมีลักษณะโครงสร้างข้อมูลดังหัวข้อ 3.6.3 โครงสร้างข้อมูลรูปแบบ Dynamic Mapping วิธีการทดสอบมีขั้นตอนดังต่อไปนี้

- (1) นำออกข้อมูลหมวดหมู่พืชในท้องถิ่น จากส่วนเก็บข้อมูลดิบซึ่งอยู่ใน Google Firestore [2] รูปแบบ JSON [19]
- (2) นำเข้าข้อมูลดังกล่าวเข้าสู่ Elasticsearch [4] แบบไม่กำหนด Mappings [54] โดย Elasticsearch [4] จะพิจารณาจากข้อมูลนำเข้าและสร้าง Mapping ให้อัตโนมัติ (Dynamic Mapping)
- (3) เมื่อนำเข้าข้อมูลสู่ Elasticsearch [4] เรียบร้อยแล้ว จึงเริ่มค้นหาคำหรือข้อความ ผ่าน Query DSL โดยระบุคำค้นหาคำที่ 1 เป็น “สับปะรด” จากนั้นเริ่มการค้นหาและบันทึกผลลัพธ์เวลาที่ใช้ในการค้นหา
- (4) ค้นหาคำหรือข้อความ ผ่านการ Query โดยคำสั่งของ Elasticsearch [4] โดยระบุคำค้นหาคำที่ 2 เป็น “กะเพรา” จากนั้นเริ่มการค้นหาและบันทึกผลลัพธ์เวลาที่ใช้ในการค้นหา

- (5) ค้นหาคำหรือข้อความ ผ่านการ Query โดยคำสั่งของ Elasticsearch [4] โดยระบุคำค้นหาครั้งที่ 3 เป็น “สะท้อน” จากนั้นเริ่มการค้นหาและบันทึกผลลัพธ์เวลาที่ใช้ในการค้นหา
- (6) บันทึกผลการค้นหาทั้งหมดลงในตารางบันทึกผล และจัดทำกราฟสำหรับเปรียบเทียบเวลาในการค้นหาที่ได้ จากนั้นจึงสรุปผลการทดลอง

2. การทดสอบความเร็วในการค้นหาด้วยคำหรือข้อความโดยใช้ Thai Tokenizer

วิธีการทดสอบความเร็วในการค้นหาคำหรือข้อความผ่านโครงสร้างข้อมูลที่กำหนดวิธีการตัดคำไทยโดยใช้ Thai Tokenizer ซึ่งสามารถกำหนดวิธีการตัดคำนี้ได้ผ่านการระบุ mappings [54] ส่วนของ analyzer เป็น thai ให้กับชุดข้อมูลที่จะทำการสืบค้น ประกอบคำสั่งการสร้าง Index [43] ใหม่ โดย properties ที่ไม่ได้กำหนด mappings [54] หรือคุณสมบัติแบบเฉพาะเจาะจงไว้ Elasticsearch [4] จะสร้าง mappings [54] ให้โดยอัตโนมัติ โดยใช้ข้อมูลจากหมวดหมู่พีชในท้องถิ่น ซึ่งมีลักษณะโครงสร้างข้อมูลดังหัวข้อ 3.6.4 โครงสร้างข้อมูลซึ่งใช้งาน Thai Tokenizer0 วิธีการทดสอบมีขั้นตอนดังต่อไปนี้

- (1) นำออกข้อมูลหมวดหมู่พีชในท้องถิ่น จากส่วนเก็บข้อมูลดิบซึ่งอยู่ใน Google Firestore [2] รูปแบบ JSON [19]
- (2) นำเข้าข้อมูลดังกล่าวเข้าสู่ Elasticsearch [4] โดยกำหนดวิธีการตัดคำภาษาไทยซึ่งใช้ Thai Tokenizer โดยการระบุ mappings [54] ส่วนของ analyzer เป็น thai
- (3) เมื่อนำเข้าข้อมูลสู่ Elasticsearch [4] เรียบร้อยแล้ว จึงเริ่มค้นหาคำหรือข้อความ ผ่าน Query DSL โดยระบุคำค้นหาครั้งที่ 1 เป็น “สับประรด” จากนั้นเริ่มการค้นหาและบันทึกผลลัพธ์เวลาที่ใช้ในการค้นหา
- (4) ค้นหาคำหรือข้อความ ผ่านการ Query โดยคำสั่งของ Elasticsearch [4] โดยระบุคำค้นหาครั้งที่ 2 เป็น “กะเพรา” จากนั้นเริ่มการค้นหาและบันทึกผลลัพธ์เวลาที่ใช้ในการค้นหา
- (5) ค้นหาคำหรือข้อความ ผ่านการ Query โดยคำสั่งของ Elasticsearch [4] โดยระบุคำค้นหาครั้งที่ 3 เป็น “สะท้อน” จากนั้นเริ่มการค้นหาและบันทึกผลลัพธ์เวลาที่ใช้ในการค้นหา
- (6) บันทึกผลการค้นหาทั้งหมดลงในตารางบันทึกผล และจัดทำกราฟสำหรับเปรียบเทียบเวลาในการค้นหาที่ได้ จากนั้นจึงสรุปผลการทดลอง

3. การทดสอบความเร็วในการค้นหาด้วยคำหรือข้อความโดยใช้ Custom Mapping

วิธีการทดสอบความเร็วในการค้นหาข้อความ โดยใช้ Mapping ตามรูปแบบ Data Schema Collection ซึ่งออกแบบให้แปลงโครงสร้างเพื่อการค้นหาได้อย่างรวดเร็วและให้ผลลัพธ์ที่สอดคล้องกับคำค้นหามากขึ้นกว่าการใช้ Mapping ซึ่งถูกสร้างโดยระบบอัตโนมัติของ Elasticsearch [4] โดยใช้ข้อมูลจากหมวดหมู่พีชในท้องถิ่น ซึ่งมีลักษณะโครงสร้างข้อมูลดังหัวข้อ 3.6.5 โครงสร้างข้อมูลรูปแบบ Custom Mapping วิธีการทดสอบมีขั้นตอนดังต่อไปนี้

- (1) นำออกข้อมูลหมวดหมู่พีชในท้องถิ่น จากส่วนเก็บข้อมูลดิบซึ่งอยู่ใน Google Firestore [2] รูปแบบ JSON [19]
- (2) นำเข้าข้อมูลดังกล่าวเข้าสู่ Elasticsearch [4] โดยกำหนด Mapping ซึ่งแปลงมาจาก Data Collection Schema
- (3) เมื่อนำเข้าข้อมูลสู่ Elasticsearch [4] เรียบร้อยแล้ว จึงเริ่มค้นหาคำหรือข้อความ ผ่าน Query DSL โดยระบุคำค้นหาที่ 1 เป็น “สับปะรด” จากนั้นเริ่มการค้นหาและบันทึกผลลัพธ์เวลาที่ใช้ในการค้นหา
- (4) ค้นหาคำหรือข้อความ ผ่าน Query DSL โดยระบุคำค้นหาที่ 2 เป็น “กะเพรา” จากนั้นเริ่มการค้นหาและบันทึกผลลัพธ์เวลาที่ใช้ในการค้นหา
- (5) ค้นหาคำหรือข้อความ ผ่าน Query DSL โดยระบุคำค้นหาที่ 3 เป็น “สะตอ” จากนั้นเริ่มการค้นหาและบันทึกผลลัพธ์เวลาที่ใช้ในการค้นหา
- (6) ค้นหาคำหรือข้อความ ผ่าน Query DSL โดยระบุคำค้นหาที่ 3 เป็น “ ” จากนั้นเริ่มการค้นหาและบันทึกผลลัพธ์เวลาที่ใช้ในการค้นหา
- (7) บันทึกผลการค้นหาทั้งหมดลงในตารางบันทึกผล และจัดทำกราฟสำหรับเปรียบเทียบเวลาในการค้นหาที่ได้ จากนั้นจึงสรุปผลการทดลอง

3.7.3 การทดลองที่ 3 การทดสอบความถูกต้องจากการใช้งาน Autocomplete

วัตถุประสงค์ของการทดลองที่ 3 เพื่อเปรียบเทียบประสิทธิภาพการใช้งานคุณสมบัติ Autocomplete ด้านความถูกต้องในการค้นหาผลลัพธ์ ระหว่างโครงสร้างข้อมูลแบบ Dynamic Mapping และโครงสร้างที่แปลงมาจาก Data Collection Schema โดยพิจารณาจากผลลัพธ์จากการค้นหาด้วยคำหรือข้อความใน 1 ลำดับ 5 ลำดับ และ 10 ลำดับแรก โดยข้อมูลที่นำมาใช้ทดลองมาจากหมวดหมู่พีชในท้องถิ่น 1,952,571 รายการ โดยหัวข้อการทดสอบจะเป็นแบ่งออกเป็น 3 ข้อย่อยดังต่อไปนี้

1. การทดสอบความถูกต้องจากการใช้งาน Autocomplete ผ่านการค้นหาจากโครงสร้างข้อมูล Dynamic Mapping

วิธีการทดสอบประสิทธิภาพของ Autocomplete ซึ่งมีการใช้งานผ่าน User App เพื่อการพิมพ์ค้นหาคำที่ต้องการ โดยใช้ Dynamic Mapping ซึ่งถูกสร้างโดยระบบอัตโนมัติของ Elasticsearch [4] โดยใช้ข้อมูลจากหมวดหมู่พืชในท้องถิ่น ซึ่งมีลักษณะโครงสร้างข้อมูลดังหัวข้อ 3.6.3 โครงสร้างข้อมูลรูปแบบ Dynamic Mapping วิธีการทดสอบมีขั้นตอนดังต่อไปนี้

- (1) นำออกข้อมูลหมวดหมู่พืชในท้องถิ่น จากส่วนเก็บข้อมูลดิบซึ่งอยู่ใน Google Firestore [2] รูปแบบ JSON [19]
- (2) นำเข้าข้อมูลดังกล่าวเข้าสู่ Elasticsearch [4] แบบไม่กำหนด Mappings [54] โดย Elasticsearch [4] จะพิจารณาจากข้อมูลที่น่าเข้าและสร้าง Mappings [54] ให้อัตโนมัติ (Dynamic Mapping)
- (3) เมื่อนำเข้าข้อมูลสู่ Elasticsearch [4] เรียบร้อยแล้ว จึงเริ่มค้นหาคำที่ต้องการผ่าน Query DSL โดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ไม”
- (4) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ไมย”
- (5) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ไมยร”
- (6) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ไมยรา”
- (7) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ไมยราบ”
- (8) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “มะ”
- (9) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “มะเ”
- (10) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “มะเข”

- (11) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “มะเขี”
- (12) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “มะเขือ”
- (13) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “กร”
- (14) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “กระ”
- (15) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “กระถ”
- (16) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “กระถิ”
- (17) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “กระถิน”
- (18) บันทึกผลการค้นหาทั้งหมดลงในตารางบันทึกผล และจัดทำกราฟสำหรับเปรียบเทียบเวลาในการค้นหาที่ได้ จากนั้นจึงสรุปผลการทดลอง

2. การทดสอบความถูกต้องจากการใช้งาน Autocomplete ผ่านการค้นหาจากโครงสร้างข้อมูลซึ่งใช้งาน Thai Tokenizer

วิธีการทดสอบประสิทธิภาพของ Autocomplete ซึ่งมีการใช้งานผ่าน User App เพื่อการพิมพ์ค้นหาคำที่ต้องการ ผ่านโครงสร้างข้อมูลที่กำหนดวิธีการตัดคำไทยโดยใช้ Thai Tokenizer ซึ่งสามารถกำหนดวิธีการตัดคำนี้ได้ผ่านการระบุ mappings [54] ส่วนของ analyzer เป็น thai ให้กับชุดข้อมูลที่จะทำการสืบค้น ประกอบคำสั่งการสร้าง Index [43] ใหม่ โดย properties ที่ไม่ได้กำหนด mappings [54] หรือคุณสมบัติแบบเฉพาะเจาะจงไว้ Elasticsearch [4] จะสร้าง mappings [54] ให้โดยอัตโนมัติ โดยใช้ข้อมูลจากหมวดหมู่พีชในท้องถิ่น ซึ่งมีลักษณะโครงสร้างข้อมูลดังหัวข้อ 3.6.4 โครงสร้างข้อมูลซึ่งใช้งาน Thai Tokenizer วิธีการทดสอบมีขั้นตอนดังต่อไปนี้

- (1) นำออกข้อมูลหมวดหมู่พีชในท้องถิ่น จากส่วนเก็บข้อมูลดิบซึ่งอยู่ใน Google Firestore [2] รูปแบบ JSON [19]

- (2) นำเข้าข้อมูลดังกล่าวเข้าสู่ Elasticsearch [4] โดยกำหนดวิธีการตัดคำภาษาไทยซึ่งใช้ Thai Tokenizer โดยการระบุ mappings [54] ส่วนของ analyzer เป็น thai
- (3) เมื่อนำเข้าข้อมูลสู่ Elasticsearch [4] เรียบร้อยแล้ว จึงเริ่มค้นหาคำที่ต้องการผ่าน Query DSL โดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ไม”
- (4) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ไมย”
- (5) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ไมร”
- (6) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ไมยรา”
- (7) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ไมยราบ”
- (8) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “มะ”
- (9) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “มะเ”
- (10) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “มะเข”
- (11) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “มะเขี”
- (12) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “มะเขือ”
- (13) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “กร”
- (14) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “กระ”

- (15) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “กระถ”
- (16) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “กระถิ”
- (17) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “กระถิน”
- (18) บันทึกผลการค้นหาทั้งหมดลงในตารางบันทึกผล และจัดทำกราฟสำหรับเปรียบเทียบเวลาในการค้นหาที่ได้ จากนั้นจึงสรุปผลการทดลอง

3. การทดสอบความถูกต้องจากการใช้งาน Autocomplete ผ่านการค้นหาจากโครงสร้างข้อมูล Custom Mapping

วิธีการทดสอบประสิทธิภาพของ Autocomplete ซึ่งมีการใช้งานผ่าน User App เพื่อการพิมพ์ค้นหาคำที่ต้องการ โดยใช้ Mapping ตามรูปแบบ Data Schema Collection (Custom Mapping) ซึ่งออกแบบให้แปลงโครงสร้างเพื่อการค้นหาได้อย่างรวดเร็วและให้ผลลัพธ์ที่สอดคล้องกับคำค้นหามากขึ้นกว่าการใช้ Mapping ซึ่งถูกสร้างโดยระบบอัตโนมัติของ Elasticsearch [4] โดยใช้ข้อมูลจากหมวดหมู่พืชในท้องถิ่น ซึ่งมีลักษณะโครงสร้างข้อมูลดังหัวข้อ 3.6.5 โครงสร้างข้อมูลรูปแบบ Custom Mapping วิธีการทดสอบมีขั้นตอนดังต่อไปนี้

- (1) นำออกข้อมูลหมวดหมู่พืชในท้องถิ่น จากส่วนเก็บข้อมูลดิบซึ่งอยู่ใน Google Firestore [2] รูปแบบ JSON [19]
- (2) นำเข้าข้อมูลดังกล่าวเข้าสู่ Elasticsearch [4] โดยกำหนด Mappings [54] ซึ่งแปลงมาจาก Data Collection Schema
- (3) เมื่อนำเข้าข้อมูลสู่ Elasticsearch [4] เรียบร้อยแล้ว จึงเริ่มค้นหาคำที่ต้องการผ่าน Query DSL โดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ไม”
- (4) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ไมย”
- (5) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ไมยร”
- (6) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ไมยรา”

- (7) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “ไมยราบ”
- (8) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “มะ”
- (9) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “มะเ”
- (10) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “มะเข”
- (11) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “มะเข็”
- (12) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “มะเขือ”
- (13) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “กร”
- (14) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “กระ”
- (15) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “กระถ”
- (16) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “กระถิ”
- (17) ค้นหาโดยกำหนดจำนวนผลลัพธ์ 10 5 และ 1 ตามลำดับ โดยการระบุตัวอักษรเป็น “กระถิน”
- (18) บันทึกผลการค้นหาทั้งหมดลงในตารางบันทึกผล และจัดทำกราฟสำหรับเปรียบเทียบเวลาในการค้นหาที่ได้ จากนั้นจึงสรุปผลการทดลอง

การพิจารณาเลือกหมวดหมู่ข้อมูลสำหรับการทดสอบนั้น คัดเลือกจากข้อมูลคำหรือข้อความที่ใช้เรียกกันในห้องเรียนซึ่งมีความหลากหลายและมีข้อมูลปริมาณมาก เมื่อพิจารณาจากตารางที่ 21 จึงได้เลือกหมวดหมู่พืชในห้องเรียนเป็นกลุ่มข้อมูลสำหรับดำเนินการทดสอบทั้งสามรูปแบบดังหัวข้อ 3.7.1 ถึง 3.7.3 ที่กล่าวไปข้างต้น

ตารางที่ 21 เปรียบเทียบจำนวนข้อมูลในแต่ละหมวดหมู่ ตั้งแต่ปี 2564 ถึง 2566

หมวดหมู่ข้อมูล	จำนวนข้อมูล ปี 2564	จำนวนข้อมูล ปี 2565	จำนวนข้อมูล ปี 2566	จำนวนข้อมูล ทั้งหมด
ที่พักหรือโรงแรม	24,529	31,493	1,449	57,471
เกษตรกรในท้องถิ่น	303,572	514,421	7,299	825,292
ผู้ย้ายกลับบ้านเนื่องจาก สถานการณ์โควิด	72,034	95,401	2,884	170,319
อาหารที่น่าสนใจประจำถิ่น	81,172	377,732	23,298	482,202
สัตว์ในท้องถิ่น	88,533	278,472	10,141	377,146
พืชในท้องถิ่น	307,209	1,554,585	90,778	1,952,571
ร้านอาหารในท้องถิ่น	73,461	154,742	6,204	234,407
แหล่งน้ำในท้องถิ่น	29,991	63,163	2,872	96,026
ภูมิปัญญาท้องถิ่น	33,191	67,933	2,329	103,453
แหล่งท่องเที่ยว	33,437	40,876	2,601	76,914
จำนวนข้อมูลทั้งหมด	1,047,128	3,178,818	149,855	4,375,801

จากตารางที่ 21 แสดงให้เห็นว่าหมวดหมู่ข้อมูลพืชในท้องถิ่นมีจำนวนข้อมูลมากที่สุด และหากพิจารณาจากข้อมูลชื่อพืชในท้องถิ่นเพิ่มเติมพบว่า มีความหลากหลายของชื่อเรียกพืชแต่ละชนิด โดยพืชชนิดเดียวกันสามารถมีชื่อเรียกได้มากกว่าหนึ่งชื่อ เช่น สัก ต้นสัก สักทอง เป็นต้น

บทที่ 4 ผลการศึกษา/ผลการวิจัย

ในบทนี้จะอธิบายถึงการวิเคราะห์ข้อมูลและผลลัพธ์ที่ได้ ซึ่งเกี่ยวข้องกับวิทยานิพนธ์ฉบับนี้ ซึ่งประกอบด้วย ผลการทดสอบความเร็วในการประมวลผล สถิติการใช้งานระบบเก็บข้อมูลหลากหลาย จำนวนผู้ใช้งานและการรองรับของระบบ การนำข้อมูลของชุมชนมาวิเคราะห์ร่วมกับแหล่งข้อมูลหลากหลาย

4.1 ผลการวิเคราะห์จากการทดลอง

4.1.1 ผลการวิเคราะห์จากทดลองที่ 1

จากการทดลองหัวข้อ 0 ซึ่งเป็นการทดสอบความถูกต้องจากการค้นหาด้วยคำหรือข้อความจากโครงสร้างข้อมูล 3 รูปแบบ ประกอบด้วย รูปแบบ Dynamic Mapping รูปแบบ Thai Tokenizer [52] และ Custom Mapping ซึ่งใช้ Tokenizer แบบ Ngram2,3 [53] โดยได้พิจารณาผลลัพธ์จากการค้นหาเป็น 1 ลำดับแรก (Top 1) 5 ลำดับแรก (Top 5) และ 10 ลำดับแรก (Top 10) ดังตารางที่ 22 โดยค่าความแม่นยำจะอยู่ระหว่าง 0 ถึง 1

เกณฑ์การคำนวณคะแนนความแม่นยำ กำหนดตามรายละเอียดดังนี้

- กรณีผลลัพธ์ตรงกับคำค้นหาทุกตัวอักษร จะได้คะแนนเท่ากับ 1.00
- กรณีพบคำซ้ำหรือคำเกินมาจากคำค้นหา และสะกดคำถูกต้อง โดยสามารถสื่อสารถึงชื่อชนิดพืชได้ตามคำค้นหา จะได้คะแนนเท่ากับ 0.90
- กรณีพบคำซ้ำหรือคำเกินมาจากคำค้นหา แต่สะกดคำผิด ซึ่งส่งผลให้สื่อสารถึงชื่อชนิดพืชผิดเพี้ยน จะได้คะแนนเท่ากับ 0.50
- กรณีผลลัพธ์ไม่สอดคล้องหรือไม่พบผลลัพธ์ จะได้คะแนนเท่ากับ 0

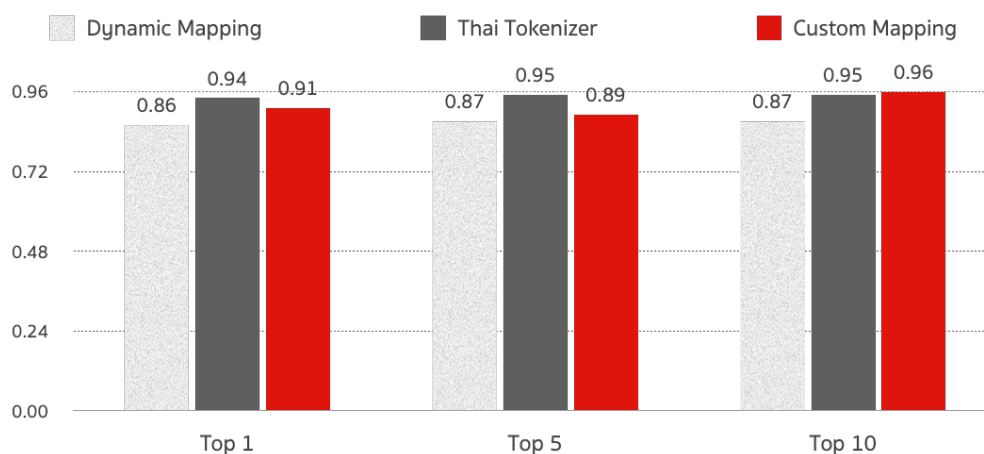
โดยนำผลคะแนนของแต่ละลำดับมาหาผลรวมและหารด้วยจำนวนผลลัพธ์ที่สนใจ ซึ่งสรุปออกมาได้ดังตารางที่ 22

ตารางที่ 22 ความแม่นยำในการค้นหาคำหรือข้อความ จากการทดลองที่ 1

คำค้นหา	Dynamic Mapping			Thai Tokenizer			Custom Mapping		
	Top 1	Top 5	Top 10	Top 1	Top 5	Top 10	Top 1	Top 5	Top 10
สัก	1.00	1.00	1.00	0.90	0.98	0.99	0.90	0.98	0.99
ต้นสัก	1.00	1.00	1.00	0.90	0.90	0.90	1.00	1.00	1.00
สักทอง	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
ฟ้าทะลายโจร	0.90	0.98	0.99	0.90	0.98	0.99	0.90	0.98	0.99
ฟ้าทะลาย	1.00	0.98	0.99	1.00	0.98	0.99	1.00	0.98	0.99
ทะลายโจร	0	0	0	0.90	0.90	0.90	0.50	0.29	0.79
ถั่วฝักยาว	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
ถั่วฝัก	1.00	1.00	1.00	1.00	0.94	0.92	1.00	0.94	0.98
ฝักยาว	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.90
ค่าเฉลี่ย	0.86	0.87	0.87	0.94	0.95	0.95	0.91	0.89	0.96

ตารางที่ 23 สรุปค่าความแม่นยำในการค้นหาคำเฉลี่ย จากผลการทดลองที่ 1

รูปแบบโครงสร้างข้อมูล	ค่าความแม่นยำในการค้นหาคำหรือข้อความเฉลี่ย		
	Top 1	Top 5	Top 10
Dynamic Mapping	0.86	0.87	0.87
Thai Tokenizer	0.94	0.95	0.95
Custom Mapping	0.91	0.89	0.96



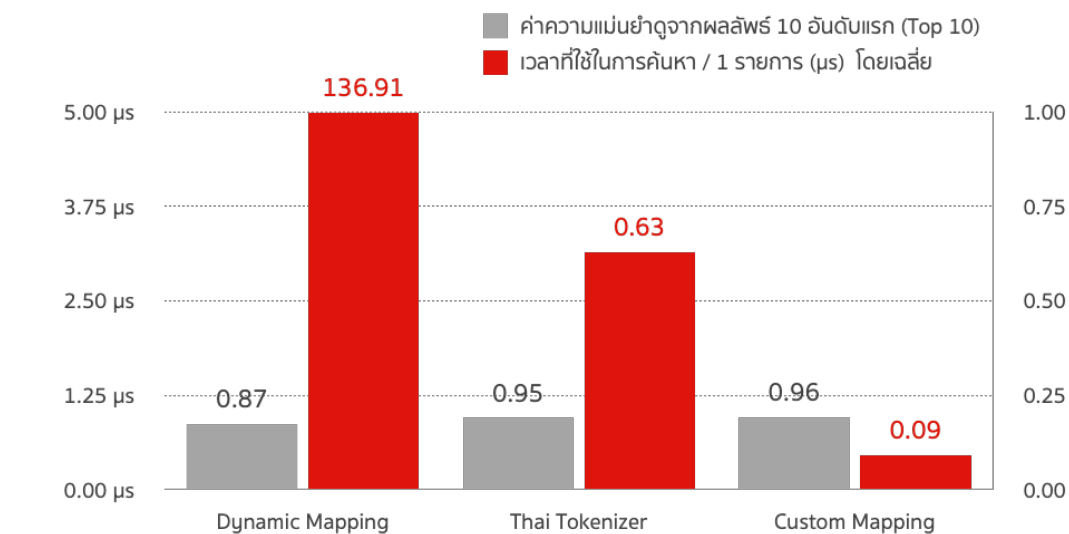
รูปที่ 23 กราฟเปรียบเทียบความแม่นยำในการค้นหาคำหรือข้อความ จากการทดลองที่ 1

ตารางที่ 24 จำนวนผลลัพธ์จากการค้นหาคำหรือข้อความ จากการทดลองที่ 1

คำค้นหา	จำนวนผลลัพธ์ทั้งหมดที่ค้นหาได้ (รายการ)		
	Dynamic Mapping	Thai Tokenizer	Custom Mapping
สัก	969	8,620	192,700
ต้นสัก	4,025	977,624	1,162,212
สักทอง	408	29,162	292,896
ฟ้าทะลายโจร	5,256	18,965	343,422
ฟ้าทะลาย	11	18,919	328,679
ทะลายโจร	0	9,601	147,294
ถั่วฝักยาว	4,569	6,920	453,400
ถั่วฝัก	2	9,734	267,326
ฝักยาว	2	270	372,256

ตารางที่ 25 เวลาที่ใช้ในการค้นหาต่อรายการ (μ s) จากการทดลองที่ 1

คำค้นหา	เวลาที่ใช้ในการค้นหา / 1 รายการ (μ s)		
	Dynamic Mapping	Thai Tokenizer	Custom Mapping
สัก	1.03	0.23	0.05
ต้นสัก	0.50	0.01	0.03
สักทอง	2.45	0.14	0.07
ฟ้าทะลายโจร	0.19	0.32	0.13
ฟ้าทะลาย	90.91	0.26	0.12
ทะลายโจร	ไม่พบผลลัพธ์	0.42	0.20
ถั่วฝักยาว	0.22	0.29	0.09
ถั่วฝัก	500.00	0.31	0.10
ฝักยาว	500.00	3.70	0.06
เฉลี่ย	136.91	0.63	0.09



รูปที่ 24 กราฟเปรียบเทียบค่าความแม่นยำจากผลลัพธ์ 10 อันดับแรก และ เวลาที่ใช้ในการค้นหา / 1 รายการ (μs) โดยเฉลี่ย จากการทดลองที่ 1

จากตารางที่ 22 ถึงตารางที่ 25 สรุปได้ว่าโครงสร้างข้อมูลแบบ Dynamic Mapping มีความแม่นยำต่ำที่สุด ส่วนโครงสร้างที่ใช้งาน Thai Tokenizer [52] และโครงสร้างแบบ Custom Mapping มีความแม่นยำใกล้เคียงกัน แต่เมื่อพิจารณาร่วมกันจำนวนผลลัพธ์ที่ค้นหาได้แล้ว โครงสร้างแบบ Custom Mapping สามารถสืบค้นข้อมูลได้ครอบคลุมมากกว่าอีกสองรูปแบบอย่างเห็นได้ชัดเจน โดยเมื่อพิจารณาร่วมกับรูปที่ 24 และรูปที่ 24 จะสังเกตเห็นได้ว่า Custom Mapping ใช้เวลาในการค้นหาได้เร็วที่สุดและมีความแม่นยำถูกต้องมากที่สุด

4.1.2 ผลการวิเคราะห์จากทดลองที่ 2

จากการทดลองหัวข้อ (3) ซึ่งเป็นการทดสอบความเร็วในการค้นหาคำหรือข้อความจากโครงสร้างข้อมูล 3 รูปแบบ ประกอบด้วย รูปแบบ Dynamic Mapping รูปแบบ Thai Tokenizer และ Custom Mapping ซึ่งใช้ Tokenizer แบบ Ngram2,3 [53] ผลการทดลองแสดงดังตารางที่ 26 และตารางที่ 27

ตารางที่ 26 ความเร็วในการค้นหาคำหรือข้อความ (ms) จากการทดลองที่ 2

คำค้นหา	เวลาที่ใช้ในการค้นหา (ms)		
	Dynamic Mapping	Thai Tokenizer	Custom Mapping
สับประคภูแล	1	3	20
กะเพรา	3	2	31

คำค้นหา	เวลาที่ใช้ในการค้นหา (ms)		
	Dynamic Mapping	Thai Tokenizer	Custom Mapping
สะท้อน	1	1	9
ผักเหวี่ยง	2	5	30
เห็ดเผาะ	11	3	11
บอระเพ็ด	2	1	22
ว่านหางจระเข้	2	5	93
ถั่วแระ	1	1	12
มะเขือเทศสีดา	2	7	55
ข้าวหอมมะลิ	1	5	36

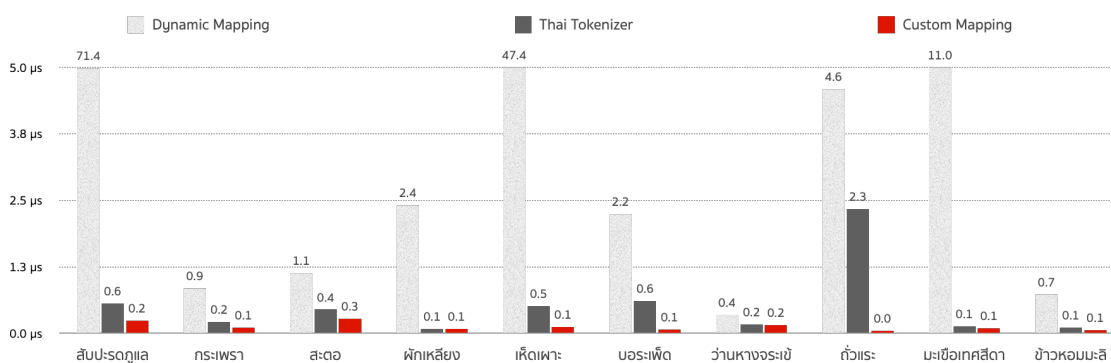
ตารางที่ 27 จำนวนผลลัพธ์จากการค้นหาคำหรือข้อความ จากการทดลองที่ 2

คำค้นหา	จำนวนผลลัพธ์ทั้งหมดที่ค้นหาได้ (รายการ)		
	Dynamic Mapping	Thai Tokenizer	Custom Mapping
สับปะรดภูเก็ต	14	5,323	85,733
กะเพรา	3,525	9,263	279,099
สะท้อน	883	2,239	32,571
ผักเหวี่ยง	831	62,312	365,397
เห็ดเผาะ	232	5,889	93,345
บอระเพ็ด	890	1,643	305,789
ว่านหางจระเข้	5,707	29,751	616,189
ถั่วแระ	218	429	265,675
มะเขือเทศสีดา	181	52,527	558,283
ข้าวหอมมะลิ	1,366	46,679	668,452

ตารางที่ 28 เวลาที่ใช้ในการค้นหาต่อรายการ (μ s) จากการทดลองที่ 2

คำค้นหา	เวลาที่ใช้ในการค้นหา / 1 รายการ (μ s)		
	Dynamic Mapping	Thai Tokenizer	Custom Mapping
สับปะรดภูเก็ต	71.43	0.56	0.23
กะเพรา	0.85	0.22	0.11
สะท้อน	1.13	0.45	0.28

คำค้นหา	เวลาที่ใช้ในการค้นหา / 1 รายการ (μ s)		
	Dynamic Mapping	Thai Tokenizer	Custom Mapping
ผักเหลียง	2.41	0.08	0.08
เห็ดเผาะ	47.41	0.51	0.12
บอระเพ็ด	2.25	0.61	0.07
ว่านหางจระเข้	0.35	0.17	0.15
ถั่วแระ	4.59	2.33	0.05
มะเขือเทศสีดา	11.05	0.13	0.10
ข้าวหอมมะลิ	0.73	0.11	0.05
เฉลี่ย	14.22	0.51	0.12



รูปที่ 25 กราฟเปรียบเทียบเวลาที่ใช้ในการค้นหาต่อรายการ (μ s) จากการทดลองที่ 2

จากตารางที่ 26 ถึงตารางที่ 28 และจากรูปที่ 25 นี้ เมื่อเปรียบเทียบเวลาที่ใช้ในการค้นหาและจำนวนผลลัพธ์ที่ได้ของโครงสร้างข้อมูลทั้งสามรูปแบบแล้ว สรุปได้ว่า โครงสร้างข้อมูลรูปแบบ Custom Mapping สามารถค้นหาข้อมูลค่าหรือข้อความได้รวดเร็วที่สุดและครอบคลุมมากที่สุดโดยใช้เวลาในการค้นหาเฉลี่ยต่อรายการเป็น 0.12 μ s ลำดับถัดมาคือโครงสร้างแบบ Thai Tokenizer โดยใช้เวลาในการค้นหาเฉลี่ยต่อรายการเป็น 0.51 μ s และโครงสร้างที่ค้นหาได้ช้ามากที่สุดคือโครงสร้างแบบ Dynamic Mapping โดยใช้เวลาในการค้นหาเฉลี่ยต่อรายการเป็น 14.22 μ s

หากพิจารณาค่าความแม่นยำหรือความถูกต้องสอดคล้องของผลลัพธ์ที่ได้จากการค้นหาคำต่าง ๆ จากโครงสร้างข้อมูล 3 รูปแบบ ประกอบด้วย รูปแบบ Dynamic Mapping รูปแบบ Thai Tokenizer [52] และ Custom Mapping ซึ่งใช้ Tokenizer แบบ Ngram2,3 [53] โดยได้พิจารณาผลลัพธ์จากการค้นหาเป็น 1 ลำดับแรก (Top 1) 5 ลำดับแรก (Top 5) และ 10 ลำดับแรก (Top 10) ดังตารางที่ 22 โดยค่าความแม่นยำจะอยู่ระหว่าง 0 ถึง 1

เกณฑ์การคำนวณคะแนนความแม่นยำ กำหนดตามรายละเอียดดังนี้

- กรณีผลลัพธ์ตรงกันกับคำค้นหาทุกตัวอักษร จะได้คะแนนเท่ากับ 1.00
- กรณีพบคำซ้ำหรือคำเกินมาจากคำค้นหา และสะกดคำถูกต้อง โดยสามารถสื่อสารถึงชื่อชนิดพืชได้ตามคำค้นหา จะได้คะแนนเท่ากับ 0.90
- กรณีพบคำซ้ำหรือคำเกินมาจากคำค้นหา แต่สะกดคำผิด ซึ่งส่งผลให้สื่อสารถึงชื่อชนิดพืชผิดเพี้ยน จะได้คะแนนเท่ากับ 0.50
- กรณีผลลัพธ์ไม่สอดคล้องหรือไม่พบผลลัพธ์ จะได้คะแนนเท่ากับ 0

โดยนำผลคะแนนของแต่ละลำดับมาหาผลรวมและหารด้วยจำนวนผลลัพธ์ที่สนใจ ซึ่งสรุปออกมาได้ดังตารางที่ 29

ตารางที่ 29 ความแม่นยำในการค้นหาคำหรือข้อความ จากการทดลองที่ 2

คำค้นหา	Dynamic Mapping			Thai Tokenizer			Custom Mapping		
	Top 1	Top 5	Top 10	Top 1	Top 5	Top 10	Top 1	Top 5	Top 10
สับปะรดภูเก็ต	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
กะเพรา	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
สะตอ	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
ผักเหลียง	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
เห็ดเผาะ	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
บอระเพ็ด	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
ว่านหางจระเข้	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
ถั่วแระ	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
มะเขือเทศสีดา	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
ข้าวหอมมะลิ	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
คำชะลิ้ว	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

จากตารางที่ 29 แสดงให้เห็นว่าการค้นหาด้วยรูปแบบคำที่สมบูรณ์ซึ่งสะกดถูกต้อง ด้วยรูปแบบ Dynamic Mapping รูปแบบ Thai Tokenizer และ Custom Mapping สามารถให้ผลลัพธ์การค้นหาที่ถูกต้องสอดคล้องกันทั้งหมด แต่หากพิจารณาภาพรวมจากการทดลองที่ 2 นี้ จะเห็นว่า Custom Mapping ใช้เวลาในการค้นหาได้เร็วที่สุดและได้ผลลัพธ์จากการค้นหาได้อย่างครอบคลุมมากที่สุด

4.1.3 ผลการวิเคราะห์จากทดลองที่ 3

จากการทดลองหัวข้อ 0 ซึ่งเป็นการทดสอบความถูกต้องในการค้นหาด้วยคุณสมบัติ Autocomplete ซึ่งเป็นรูปแบบการค้นหาทีละตัวอักษร โดยได้ทำการทดสอบจากโครงสร้างข้อมูล 3 รูปแบบ ประกอบด้วย รูปแบบ Dynamic Mapping รูปแบบ Thai Tokenizer [52] และ Custom Mapping ซึ่งใช้ Tokenizer แบบ Ngram2,3 [53] โดยได้พิจารณาผลลัพธ์จากการค้นหาเป็น 1 ลำดับแรก (Top 1) 5 ลำดับแรก (Top 5) และ 10 ลำดับแรก (Top 10) ดัง โดยค่าความแม่นยำจะอยู่ระหว่าง 0 ถึง 1

เกณฑ์การคำนวณคะแนนความแม่นยำ กำหนดตามรายละเอียดดังนี้

- กรณีผลลัพธ์ตรงกันกับคำค้นหาทุกตัวอักษร จะได้คะแนนเท่ากับ 1.00
- กรณีพบคำซ้ำหรือคำเกินมาจากคำค้นหา และสะกดคำถูกต้อง โดยสามารถสื่อสารถึงชื่อชนิดพืชได้ตามคำค้นหา จะได้คะแนนเท่ากับ 0.90
- กรณีพบคำซ้ำหรือคำเกินมาจากคำค้นหา แต่สะกดคำผิด ซึ่งส่งผลให้สื่อสารถึงชื่อชนิดพืชผิดเพี้ยน จะได้คะแนนเท่ากับ 0.50
- กรณีผลลัพธ์ไม่สอดคล้องหรือไม่พบผลลัพธ์ จะได้คะแนนเท่ากับ 0

โดยนำผลคะแนนของแต่ละลำดับมาหาผลรวมและหารด้วยจำนวนผลลัพธ์ที่สนใจ ซึ่งสรุปออกมาได้ดังตารางที่ 30 ถึงตารางที่ 32

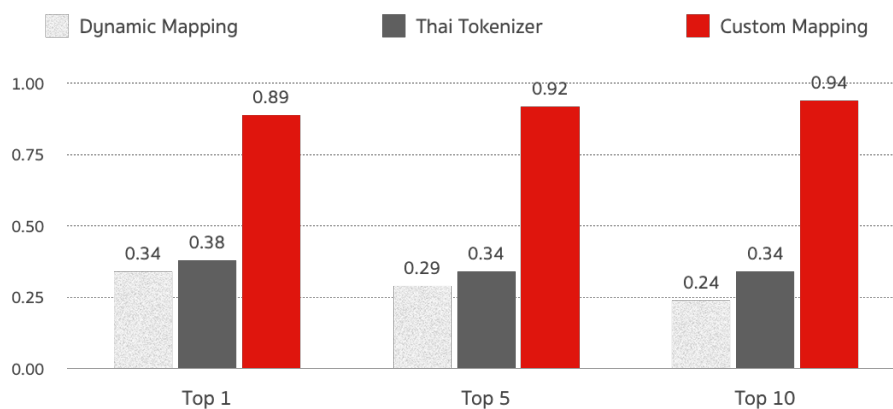
ตารางที่ 30 ความแม่นยำจากการทดสอบคุณสมบัติ Autocomplete จากการทดลองที่ 3

คำค้นหา	Dynamic Mapping			Thai Tokenizer			Custom Mapping		
	Top 1	Top 5	Top 10	Top 1	Top 5	Top 10	Top 1	Top 5	Top 10
ไม	0.50	0.10	0.10	0.50	0.10	0.10	0.50	0.82	0.86
ไมย	0	0	0	0	0	0	0.50	0.88	0.92
ไมยร	0	0	0	0	0	0	1.00	0.96	0.96
ไมยรา	0	0	0	0	0	0	1.00	0.96	0.96
ไมยราบ	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
มะ	0.90	0.90	0.45	0.50	0.50	0.50	0.90	0.76	0.88
มะเ	0	0	0	0	0	0	1.00	0.80	0.75
มะเข	0	0	0	0.50	0.50	0.50	0.90	0.98	0.99
มะเขี	0	0	0	0	0	0	0.90	0.98	0.99
มะเขือ	0.90	0.98	0.99	0.90	0.98	0.99	0.90	0.98	0.99

คำค้นหา	Dynamic Mapping			Thai Tokenizer			Custom Mapping		
	Top 1	Top 5	Top 10	Top 1	Top 5	Top 10	Top 1	Top 5	Top 10
กร	0	0	0	0.50	0.50	0.50	0.90	0.92	0.91
กระ	1.00	0.40	0.20	0.50	0.50	0.55	0.90	0.90	0.95
กระถ	0	0	0	0.50	0.05	0.10	1.00	1.00	1.00
กระถิ	0	0	0	0	0	0	1.00	1.00	1.00
กระถิน	0.90	0.98	0.99	0.90	0.98	0.99	1.00	1.00	1.00
ค่าเฉลี่ย	0.34	0.29	0.24	0.38	0.34	0.34	0.89	0.92	0.94

ตารางที่ 31 สรุปค่าความแม่นยำเฉลี่ย จากผลการทดลองที่ 3

รูปแบบโครงสร้างข้อมูล	ค่าความแม่นยำในการค้นหาคำหรือข้อความเฉลี่ย		
	Top 1	Top 5	Top 10
Dynamic Mapping	0.34	0.29	0.24
Thai Tokenizer	0.38	0.34	0.34
Custom Mapping	0.89	0.92	0.94



รูปที่ 26 กราฟเปรียบเทียบความแม่นยำในการค้นหาคำหรือข้อความ จากการทดลองที่ 3

ตารางที่ 32 จำนวนผลลัพธ์จากการทดสอบคุณสมบัติ Autocomplete จากการทดลองที่ 3

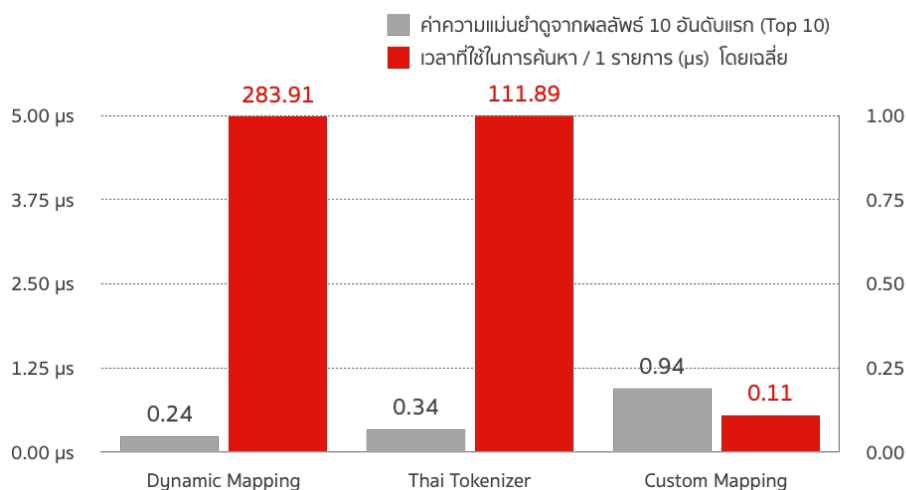
คำค้นหา	จำนวนผลลัพธ์ทั้งหมดที่ค้นหาได้ (รายการ)		
	Dynamic Mapping	Thai Tokenizer	Custom Mapping
ไม่	1	1	33,188
ไม่ย	0	0	34,472

คำค้นหา	จำนวนผลลัพธ์ทั้งหมดที่ค้นหาได้ (รายการ)		
	Dynamic Mapping	Thai Tokenizer	Custom Mapping
ไมยร	0	0	37,982
ไมยรา	0	0	110,914
ไมยราบ	731	2,451	130,904
มะ	5	11,478	319,361
มะเ	0	0	394,875
มะเข	0	11,605	425,610
มะเขี	0	0	425,639
มะเขือ	4,045	25,038	486,504
กร	0	179	148,417
กระ	2	16,676	200,307
กระถ	0	16,852	201,343
กระถิ	0	0	203,042
กระถิน	4,335	13,513	243,198

ตารางที่ 33 เวลาที่ใช้ในการค้นหาต่อรายการ (μ s) จากการทดลองที่ 3

คำค้นหา	เวลาที่ใช้ในการค้นหา / 1 รายการ (μ s)		
	Dynamic Mapping	Thai Tokenizer	Custom Mapping
ไม	1000.00	1000.00	0.09
ไมย	ไม่พบผลลัพธ์	ไม่พบผลลัพธ์	0.17
ไมยร	ไม่พบผลลัพธ์	ไม่พบผลลัพธ์	0.21
ไมยรา	ไม่พบผลลัพธ์	ไม่พบผลลัพธ์	0.08
ไมยราบ	2.74	0.41	0.09
มะ	200.00	0.17	0.01
มะเ	ไม่พบผลลัพธ์	ไม่พบผลลัพธ์	0.05
มะเข	ไม่พบผลลัพธ์	0.26	0.09
มะเขี	ไม่พบผลลัพธ์	ไม่พบผลลัพธ์	0.11
มะเขือ	0.25	0.08	0.13
กร	ไม่พบผลลัพธ์	5.59	0.01
กระ	500.00	0.12	0.18
กระถ	ไม่พบผลลัพธ์	0.24	0.10
กระถิ	ไม่พบผลลัพธ์	ไม่พบผลลัพธ์	0.12

คำค้นหา	เวลาที่ใช้ในการค้นหา / 1 รายการ (μ s)		
	Dynamic Mapping	Thai Tokenizer	Custom Mapping
กรณี	0.46	0.15	0.13
ค่าเฉลี่ย	283.91	111.89	0.11



รูปที่ 27 กราฟเปรียบเทียบค่าความแม่นยำจากผลลัพธ์ 10 อันดับแรก และ เวลาที่ใช้ในการค้นหา / 1 รายการ (μ s) โดยเฉลี่ย จากการทดลองที่ 3

จากตารางที่ 30 ถึงตารางที่ 32 และรูปที่ 26 สรุปได้ว่าโครงสร้างข้อมูลแบบ Custom Mapping สามารถค้นหาในลักษณะการใช้งาน Autocomplete ซึ่งจะมีการรับค่าเพิ่มสำหรับค้นหาเพิ่มไปทีละตัวอักษรนั้น ได้อย่างถูกต้องสอดคล้องและครบคลุมมากที่สุดมากที่สุด ส่วนโครงสร้างที่เหลืออีกสองรูปแบบยังไม่เหมาะสมสำหรับการค้นหาด้วย Autocomplete เนื่องจากไม่สามารถหาผลลัพธ์ได้จากการพิมพ์ทีละตัวอักษร ซึ่งจะยังไม่เป็นคำ จึงไม่เหมาะสมสำหรับการใช้งานแนะนำคำที่สอดคล้อง โดยเมื่อพิจารณาร่วมกับรูปที่ 27 จะสังเกตเห็นได้ว่า Custom Mapping ใช้เวลาในการค้นหาได้เร็วที่สุด

4.2 ผลการวิเคราะห์สถิติการใช้งานระบบเก็บข้อมูลหลากหลาย

ข้อมูลที่ถูกจัดเก็บในระบบจัดการข้อมูลหลากหลายนี้ สามารถนำมาวิเคราะห์ต่อยอดด้วยเครื่องมือ (Data Visualization Tools) ตามความถนัดของผู้วิเคราะห์ เช่น Kibana [51], Power BI [45] เป็นต้น วิทยานิพนธ์ฉบับนี้ได้นำข้อมูลไปวิเคราะห์และจัดเรียงเป็นกระดานสรุปผล

การติดตามข้อมูลสถิติหรือข้อมูลสรุปสำหรับผู้บริหารจัดการพื้นที่หรือชุมชนระดับต่าง ๆ ได้ในรูปแบบหลากหลาย เนื่องด้วยระบบจัดการข้อมูลความหลากหลายซึ่งได้ออกแบบมีการจัดเก็บข้อมูลสำหรับการสืบค้นลงใน Elasticsearch [4] ซึ่งสามารถเชื่อมโยงกับ Kibana [51] เพื่อ

แสดงผลข้อมูลสถิติในหลายมุมมอง สามารถเลือกกรองเฉพาะชุดข้อมูลที่สนใจ และระบุวันเวลาที่ต้องการค้นหาข้อมูลได้ จึงได้จัดทำ Dashboard ด้วย Kibana [51] เพื่อช่วยในการตัดสินใจวางแผนบริหารจัดการพัฒนาชุมชนต่อไปดังรูปที่ 28

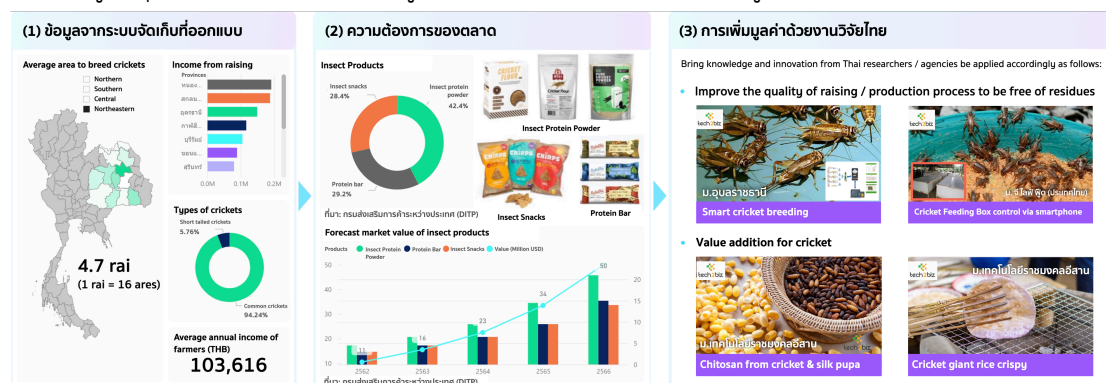


รูปที่ 28 Dashboard แสดงข้อมูลสถิติภาพรวมทั้งหมด

จากรูปที่ 28 แสดงข้อมูลสถิติภาพรวมซึ่งได้จากระบบจัดการข้อมูลความหลากหลายที่ออกแบบตั้งแต่ มีนาคม 2564 ถึง สิงหาคม 2565 ซึ่งจำนวนข้อมูลทั้งหมดที่ได้จากการสำรวจและบันทึกเข้าสู่ระบบ คือ 4,385,741 รายการ จาก 7,431 ตำบลในประเทศไทย ซึ่งคิดเป็น 99.95% ของตำบลทั้งหมดในประเทศไทยซึ่งมีจำนวน 7,435 ตำบล โดยมีการเก็บข้อมูลในหลากหลายด้านจำนวน 10 หมวดหมู่ข้อมูล โดยหมวดหมู่ที่มีการบันทึกข้อมูลมากที่สุด 3 ลำดับแรก ประกอบด้วย พืชในท้องถิ่น 43.71% (1,917,001 รายการ) เกษตรกรในท้องถิ่น 19.3% (846,619 รายการ) และอาหารที่น่าสนใจประจำถิ่น 10.92% (478,765 รายการ)

ข้อมูลซึ่งถูกวิเคราะห์และแสดงผลในหลากหลายรูปแบบนี้ สามารถนำไปประกอบการตัดสินใจและใช้สำหรับการวางแผนบริหารจัดการชุมชนของผู้บริหารในระดับประเทศ เช่น การวิเคราะห์หาพื้นที่ใดมีสมุนไพรสำหรับรักษาโรคต่าง ๆ บ้าง แต่ละพื้นที่มีปริมาณของพืชสมุนไพรแต่ละชนิดเท่าไร เป็นต้น ซึ่งข้อมูลที่รวบรวมได้นี้ สามารถเลือกนำเอาชุดข้อมูลที่สนใจ ไปวิเคราะห์กับแหล่งข้อมูลอื่น ๆ เพื่อหาแนวทางการพัฒนาต่อยอดในระดับชุมชนและประเทศต่อไป เช่น การนำองค์ความรู้จากฐานข้อมูลด้านการวิจัยเกี่ยวกับสมุนไพรรักษาโรค มาวิเคราะห์ร่วมกัน เพื่อการวางแผนการนำพืชสมุนไพรท้องถิ่นแต่ละชนิดที่มีอยู่แล้วไปทดลองด้วยองค์ความรู้ที่คัดเลือกโดยผู้เชี่ยวชาญ การวางแผนการเพาะปลูกชนิดของพืชที่สอดคล้องกับความต้องการในปัจจุบัน เป็นต้น

นอกจากนั้น ยังสามารถนำข้อมูลเหล่านี้ไปต่อยอดเพื่อใช้งานในหลากหลายด้าน เช่น การพัฒนาโปรตีนทางเลือก การพัฒนาองค์ความรู้การเลี้ยงโคโคนม การวิเคราะห์สัดส่วนการผลิตเส้นใยธรรมชาติภายในประเทศเพื่อการต่อเป็นผลิตภัณฑ์ซึ่งช่วยเพิ่มมูลค่า เป็นต้น โดยตัวอย่าง ซึ่งได้จัดทำเป็นข้อมูลสรุป เพื่อการวิเคราะห์ข้อมูลได้สะดวกมากยิ่งขึ้น ดังแสดงรูปที่ 29



รูปที่ 29 การวิเคราะห์ข้อมูลของระบบที่ออกแบบ ร่วมกับแหล่งข้อมูลอื่น ๆ

จากรูปที่ 29 แสดงผลการวิเคราะห์ข้อมูลด้วย Power BI [45] ซึ่งสามารถนำเข้าชุดข้อมูลที่สนใจเข้ามาแสดงผลร่วมกับรูปภาพหรือสร้างรูปแบบเฉพาะสำหรับการแสดงผล โดยได้นำข้อมูลที่เกี่ยวข้องกับจังหวัดในประเทศไทย แนวโน้มความต้องการของตลาดจากสหรัฐอเมริกา และข้อมูลงานวิจัยของไทย มาวิเคราะห์ร่วมกัน โดยสรุปข้อมูลเป็น 3 ส่วนด้วยกันประกอบไปด้วย

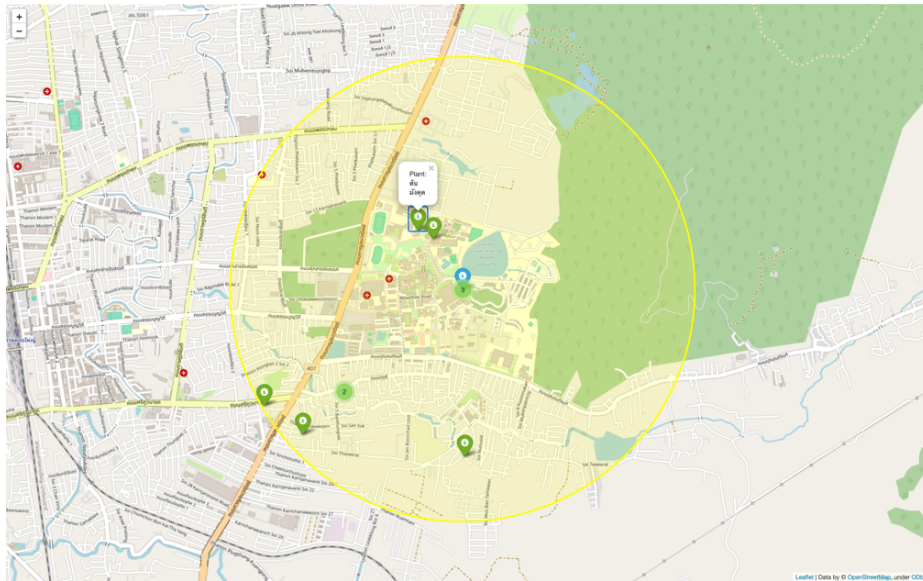
- (1) ข้อมูลการเลี้ยงจิ้งหรีดในประเทศไทย** ซึ่งมาจากการสำรวจและเก็บบันทึกลงในระบบจัดการข้อมูลความหลากหลายซึ่งได้ออกแบบนี้ พบว่ามีการเลี้ยงจิ้งหรีดเป็นจำนวนมากบริเวณภาคตะวันออกเฉียงเหนือ ซึ่งมีพื้นที่การเลี้ยงประมาณ 4.7 ไร่ โดยจังหวัดที่มีการเพาะเลี้ยงจิ้งหรีดมากที่สุดสามลำดับแรก คือ หนองคาย สกลนคร และอุดรธานี ซึ่งรายได้เฉลี่ยของเกษตรกรที่เพาะเลี้ยงจิ้งหรีดต่อปีอยู่ที่ 103,616 บาท
- (2) ข้อมูลความต้องการของตลาด** ซึ่งนำมาจากกรมส่งเสริมการค้าระหว่างประเทศ โดยได้นำข้อมูลแนวโน้มความต้องการสินค้าเกี่ยวกับแมลงจากสหรัฐอเมริกา ตั้งแต่ปี 2562 ถึง 2566 มาวิเคราะห์ร่วมด้วย พบว่าแนวโน้มสัดส่วนประเภทของผลิตภัณฑ์แปรรูปจากแมลงสามลำดับประกอบด้วย การนำมาทำขนมขบเคี้ยว (Snacks) ประมาณ 28.40% นำมาทำแท่งช็อกโกแลต (Protein Bar) 29.20% และนำมาทำแป้งจากแมลง (Insect Protein Powder) ประมาณ 42.40% โดยมูลค่าทางการตลาดมีแนวโน้มสูงขึ้นตามลำดับ คาดว่าใน

ปี 2565 มูลค่าผลิตภัณฑ์จากแมลงจะสูงถึง 34 ล้านเหรียญสหรัฐ และในปี 2566 จะสูงขึ้นไปอีกอยู่ที่ประมาณ 50 ล้านเหรียญสหรัฐ จะเห็นได้ว่าข้อมูลความต้องการของตลาดสามารถช่วยในการตัดสินใจวางแผนการสนับสนุนอุตสาหกรรมการเพิ่มมูลค่าและการแปรรูปแมลงของประเทศไทยเพื่อไปสู่ตลาดโลก

- (3) **การเพิ่มมูลค่าด้วยงานวิจัยไทย** จากข้อมูลเกี่ยวกับจังหวัดในประเทศไทย ข้อมูลความต้องการของตลาดของสหรัฐอเมริกา ช่วยให้เห็นแนวโน้มความต้องการต่ออุตสาหกรรมบริโภคแมลง จึงได้นำข้อมูลเกี่ยวกับงานวิจัยของประเทศไทยจากtech2biz [55] มาวิเคราะห์ร่วมกัน เพื่อนำองค์ความรู้หรือนวัตกรรมจากงานวิจัยทั้งจากหน่วยงานรัฐและเอกชนมาประยุกต์ เพื่อส่งเสริมการทำฟาร์มจังหวัดให้ทันสมัยและได้มาตรฐาน รวมถึงสนับสนุนการแปรรูปเป็นผลิตภัณฑ์ที่หลากหลาย เพื่อเข้าถึงผู้บริโภคได้อย่างกว้างขวาง ช่วยสร้างรายได้ให้กับเกษตรกร ชุมชน ซึ่งส่งผลต่อการพัฒนาเศรษฐกิจของประเทศ

จากการนำข้อมูลที่เก็บบันทึกได้ มาวิเคราะห์ร่วมกับข้อมูลหลากหลายแหล่ง ช่วยให้วิเคราะห์ข้อมูลได้อย่างครอบคลุมกับหัวข้อที่สนใจมากยิ่งขึ้น ซึ่งช่วยให้ผู้ใช้งานข้อมูลซึ่งมีความถนัดแตกต่างกันจากหลายสาขา วิเคราะห์ข้อมูลและวางแผนร่วมกันได้อย่างมีประสิทธิภาพมากยิ่งขึ้น

นอกจากนั้น ยังสามารถวิเคราะห์และจัดทำรูปแบบการแสดงผลได้อีกหลากหลาย เช่น การประมวลผลค้นหาตำแหน่งของวัตถุโดยรอบจุดศูนย์กลางที่สนใจ เป็นต้น โดยระบบ Search & Analytics Engine ที่เลือกใช้ในวิทยานิพนธ์นี้ สามารถรองรับการค้นหาด้วยพิกัด ตัวอย่างการประยุกต์ใช้ดังเช่น กรณีต้องการค้นหาพืชท้องถิ่นที่มีข้อมูลเก็บไว้ในระบบ และต้องการแสดงผลเฉพาะพืชที่ถูกบันทึกตำแหน่งไว้ในรัศมีระยะ 2 กิโลเมตรจากตำแหน่งศูนย์กลางที่ระบุ เมื่อระบุเงื่อนไขความต้องการและพิมพ์คำสั่งให้ระบบประมวลผลเรียบร้อยแล้ว จะได้ผลลัพธ์ลักษณะดังรูปที่ 30

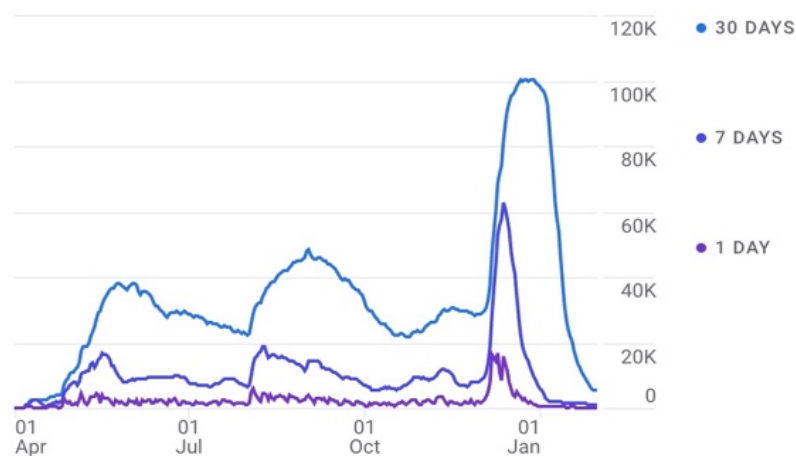


รูปที่ 30 ผลการค้นหาวัดอยู่ในรัศมีโดยรอบจากจุดศูนย์กลางที่กำหนด

จากรูปที่ 30 แสดงผลการค้นหาชนิดของพืชในท้องถิ่น ซึ่งอยู่ภายในรัศมี 2 กิโลเมตร จากจุดศูนย์กลางที่กำหนด โดยผู้ใช้งานสามารถย่อหรือขยายขนาด และคลิกดูชื่อของพืชบริเวณตำแหน่งที่สนใจได้

4.3 จำนวนผู้ใช้งานที่ระบบจัดการข้อมูลความหลากหลายรองรับ

เนื่องจากการบันทึกข้อมูลจากการสำรวจชุมชนทั่วประเทศ ซึ่งในบางช่วงเวลาจะพบการเข้าใช้งานระบบบันทึกข้อมูลพร้อมกัน ดังแสดงผลในรูปที่ 31



รูปที่ 31 จำนวนผู้ใช้งานที่ระบบจัดการข้อมูลความหลากหลายรองรับ

จากรูปที่ 31 แสดงจำนวนผู้ใช้งานที่บันทึกข้อมูลจากการสำรวจของแต่ละพื้นที่ในประเทศไทย ช่วงเวลาประมาณ เมษายน 2564 ถึง มกราคม 2565 พบว่ามีผู้ใช้งานสูงสุดมากกว่า 100,000 คนต่อเดือน โดยระบบจัดการข้อมูลความหลากหลาย สามารถให้บริการการจัดเก็บข้อมูล การแสดงผลข้อมูล การสืบค้นข้อมูลได้ตามปกติแม้จะเผชิญกับผู้ใช้งานจำนวนมากในเวลาเดียวกัน

บทที่ 5 สรุปผลการวิจัย/อภิปรายผล และข้อเสนอแนะ

ในบทนี้จะกล่าวถึงการสรุปผลการวิจัยและข้อเสนอแนะที่ได้จากการทำวิทยานิพนธ์ดังต่อไปนี้

5.1 สรุปผล

การรวบรวมกลุ่มข้อมูลซึ่งมีความหลากหลายของแต่ละชุมชนและจัดเก็บลงระบบดิจิทัลหรือระบบจัดการข้อมูลหลากหลาย ซึ่งรองรับการขยายตัวของข้อมูลซึ่งมีปริมาณเพิ่มขึ้นเรื่อย ๆ และสามารถช่วยประมวลผล ตรวจสอบความถูกต้องของข้อมูล และจัดเก็บข้อมูลอย่างถูกต้อง ส่งผลให้สามารถสืบค้นและนำข้อมูลกลับมาใช้งานได้อย่างครอบคลุม ซึ่งวิทยานิพนธ์ฉบับนี้ได้ออกแบบและพัฒนาระบบจัดการข้อมูลหลายหลาย เพื่อจัดเก็บข้อมูลการลงสำรวจชุมชนต่าง ๆ ในประเทศไทย โดยได้บริหารจัดการการจัดเก็บข้อมูลดิบ ข้อมูลชนิดไฟล์ ข้อมูลสำหรับการสืบค้นอย่างรวดเร็วด้วยรูปแบบ Custom Mapping ที่ได้ออกแบบขึ้นมาโดยเฉพาะ และได้พัฒนาระบบสำหรับอำนวยความสะดวกในการสร้างโครงสร้างการจัดเก็บข้อมูลตามหมวดหมู่ ระบบสำหรับอำนวยความสะดวกในการบันทึกข้อมูลจากการลงพื้นที่สำรวจ รวมถึงนำเครื่องมือวิเคราะห์ข้อมูลมาช่วยในการติดตามข้อมูลสถิติหรือข้อมูลสรุปสำหรับผู้บริหารจัดการพื้นที่หรือชุมชนในระดับต่าง ๆ ในรูปแบบหลากหลาย

ระบบจัดการข้อมูลความหลากหลายนี้ จึงสามารถช่วยให้ผู้ใช้งานข้อมูล เช่น นักวิจัย นักพัฒนาโปรแกรม นักวิทยาศาสตร์ข้อมูล วิศวกรข้อมูล ผู้บริหารจัดการเมือง สามารถเรียกใช้กลุ่มข้อมูลเหล่านี้ เพื่อนำไปใช้ประโยชน์และช่วยในการตัดสินใจในการแก้ไขปัญหาของเมืองหรือวางแผนพัฒนาทั้งในระดับชุมชน ตำบล อำเภอ จังหวัด และระดับประเทศ

5.2 ข้อเสนอแนะ

ควรทดลองกับกลุ่มข้อมูลชนิดอื่น ๆ เพิ่มเติม นอกเหนือจากชนิดข้อความ และควรทดลองกับกลุ่มข้อมูลขนาดใหญ่มากขึ้น เพื่อทดสอบประสิทธิภาพของสถาปัตยกรรมที่ออกแบบและพัฒนาในวิทยานิพนธ์ฉบับนี้ เพื่อปรับปรุงหรือต่อเติมให้สมบูรณ์ยิ่งขึ้นได้ในอนาคต

บรรณานุกรม

- [1] D. Jacobson, G. Brail, and D. Woods, *APIs: A Strategy Guide*. O'Reilly Media, Inc., 2012.
- [2] “Firestore,” Firebase. Accessed: Dec. 19, 2023. [Online]. Available: <https://firebase.google.com/docs/firestore>
- [3] “Cloud Storage documentation,” Google Cloud. Accessed: Dec. 19, 2023. [Online]. Available: <https://cloud.google.com/storage/docs>
- [4] “Elasticsearch Platform — Find real-time answers at scale | Elastic.” Accessed: Dec. 21, 2023. [Online]. Available: <https://www.elastic.co>
- [5] H. S. Munawar, S. Qayyum, F. Ullah, and S. Sepasgozar, “Big Data and Its Applications in Smart Real Estate and the Disaster Management Life Cycle: A Systematic Analysis,” *Big Data Cogn. Comput.*, vol. 4, no. 2, Art. no. 2, Jun. 2020, doi: 10.3390/bdcc4020004.
- [6] J. Vodák, D. Šulyová, and M. Kubina, “Advanced Technologies and Their Use in Smart City Management,” *Sustainability*, vol. 13, no. 10, Art. no. 10, Jan. 2021, doi: 10.3390/su13105746.
- [7] M. Ku and J. R. Gil-Garcia, “Ready for data analytics?: data collection and creation in local governments,” in *Proceedings of the 19th Annual International Conference on Digital Government Research: Governance in the Data Age*, Delft The Netherlands: ACM, May 2018, pp. 1–10. doi: 10.1145/3209281.3209381.
- [8] U. Aftab and G. F. Siddiqui, “Big Data Augmentation with Data Warehouse: A Survey,” in *2018 IEEE International Conference on Big Data (Big Data)*, Dec. 2018, pp. 2775–2784. doi: 10.1109/BigData.2018.8622182.
- [9] “NoSQL,” *Wikipedia*. Dec. 03, 2023. Accessed: Dec. 20, 2023. [Online]. Available: <https://en.wikipedia.org/w/index.php?title=NoSQL&oldid=1188167891>
- [10] V. M. Ngo, N.-A. Le-Khac, and M.-T. Kechadi, “Designing and Implementing Data Warehouse for Agricultural Big Data,” in *Big Data – BigData 2019*, K. Chen, S. Seshadri, and L.-J. Zhang, Eds., in Lecture Notes in Computer Science. Cham: Springer International Publishing, 2019, pp. 1–17. doi: 10.1007/978-3-030-23551-2_1.

- [11] H. S. Cha *et al.*, “The Korea Cancer Big Data Platform (K-CBP) for Cancer Research,” *Int. J. Environ. Res. Public. Health*, vol. 16, no. 13, Art. no. 13, Jan. 2019, doi: 10.3390/ijerph16132290.
- [12] “Relational database,” *Wikipedia*. Jan. 04, 2024. Accessed: Jan. 04, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Relational_database&oldid=1193552952
- [13] H. Zheng and X. Jiang, “Design and Implementation of College Consumption Analysis System Based on NoSQL Database,” in *2018 13th International Conference on Computer Science & Education (ICCSE)*, Aug. 2018, pp. 1–5. doi: 10.1109/ICCSE.2018.8468719.
- [14] H. Ganu and V. D. P., “Fast Query Expansion on an Accounting Corpus using Sub-Word Embeddings,” in *Proceedings of the Second Workshop on Subword/Character Level Models*, M. Faruqi, H. Schütze, I. Trancoso, Y. Tsvetkov, and Y. Yaghoobzadeh, Eds., New Orleans: Association for Computational Linguistics, Jun. 2018, pp. 61–65. doi: 10.18653/v1/W18-1208.
- [15] H. Hu, Y. Wen, T.-S. Chua, and X. Li, “Toward Scalable Systems for Big Data Analytics: A Technology Tutorial,” *IEEE Access*, vol. 2, pp. 652–687, 2014, doi: 10.1109/ACCESS.2014.2332453.
- [16] F. Ravat and Y. Zhao, “Data Lakes: Trends and Perspectives,” in *Database and Expert Systems Applications*, vol. 11706, S. Hartmann, J. Küng, S. Chakravarthy, G. Anderst-Kotsis, A. M. Tjoa, and I. Khalil, Eds., in *Lecture Notes in Computer Science*, vol. 11706, Cham: Springer International Publishing, 2019, pp. 304–313. doi: 10.1007/978-3-030-27615-7_23.
- [17] M. M. Rathore, H. Son, A. Ahmad, A. Paul, and G. Jeon, “Real-Time Big Data Stream Processing Using GPU with Spark Over Hadoop Ecosystem,” *Int. J. Parallel Program.*, vol. 46, no. 3, pp. 630–646, Jun. 2018, doi: 10.1007/s10766-017-0513-2.
- [18] S. Ahmad, P. Singh, and A. K. Sagar, “A Survey on Big Data Analytics,” in *2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN)*, Oct. 2018, pp. 256–260. doi: 10.1109/ICACCCN.2018.8748774.

- [19] N. Nurseitov, M. Paulson, R. Reynolds, and C. Izurieta, "Comparison of JSON and XML Data Interchange Formats: A Case Study".
- [20] "JSON Introduction." Accessed: Jan. 05, 2024. [Online]. Available: https://www.w3schools.com/js/js_json_intro.asp
- [21] "MySQL." Accessed: Jan. 05, 2024. [Online]. Available: <https://www.mysql.com/>
- [22] "Open-source software," *Wikipedia*. Mar. 01, 2024. Accessed: Mar. 03, 2024. [Online]. Available: [https://en.wikipedia.org/w/index.php?title=Open-source_software &oldid=1211147521](https://en.wikipedia.org/w/index.php?title=Open-source_software&oldid=1211147521)
- [23] P. G. D. Group, "PostgreSQL," PostgreSQL. Accessed: Jan. 05, 2024. [Online]. Available: <https://www.postgresql.org/>
- [24] "SQLite Home Page." Accessed: Jan. 05, 2024. [Online]. Available: <https://www.sqlite.org/index.html>
- [25] "Microsoft Data Platform | Microsoft." Accessed: Jan. 05, 2024. [Online]. Available: <https://www.microsoft.com/en-us/sql-server>
- [26] "Proprietary software," *Wikipedia*. Feb. 15, 2024. Accessed: Mar. 03, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Proprietary_software&oldid=1207708100
- [27] "Oracle | Cloud Applications and Cloud Platform." Accessed: Jan. 05, 2024. [Online]. Available: <https://www.oracle.com/>
- [28] "Managed SQL Database - Amazon Relational Database Service (RDS) - AWS," Amazon Web Services, Inc. Accessed: Jan. 05, 2024. [Online]. Available: <https://aws.amazon.com/rds/>
- [29] "Redis," Redis. Accessed: Jan. 05, 2024. [Online]. Available: <https://redis.io/>
- [30] "Random-access memory," *Wikipedia*. Feb. 24, 2024. Accessed: Mar. 03, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Random-access_memory&oldid=1209989742
- [31] "Amazon DynamoDB," Amazon Web Services, Inc. Accessed: Jan. 05, 2024. [Online]. Available: <https://aws.amazon.com/th/pm/dynamodb/>
- [32] "Apache Cassandra | Apache Cassandra Documentation." Accessed: Jan. 05, 2024. [Online]. Available: https://cassandra.apache.org/_/index.html

- [33] “MongoDB Atlas: Cloud Document Database,” MongoDB. Accessed: Jan. 05, 2024. [Online]. Available: <https://www.mongodb.com/cloud/atlas/lp/try4>
- [34] “Apache CouchDB.” Accessed: Jan. 05, 2024. [Online]. Available: <https://couchdb.apache.org/>
- [35] “Neo4j Graph Database & Analytics – The Leader in Graph Databases,” Graph Database & Analytics. Accessed: Jan. 05, 2024. [Online]. Available: <https://neo4j.com/>
- [36] “Cloud storage,” *Wikipedia*. Dec. 22, 2023. Accessed: Jan. 05, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Cloud_storage&oldid=1191338829
- [37] “Cloud Object Storage - Amazon S3 - AWS,” Amazon Web Services, Inc. Accessed: Jan. 05, 2024. [Online]. Available: <https://aws.amazon.com/s3/>
- [38] akashdubey-ms, “Introduction to Azure Storage - Cloud storage on Azure.” Accessed: Jan. 05, 2024. [Online]. Available: <https://learn.microsoft.com/en-us/azure/storage/common/storage-introduction>
- [39] “Full-text search,” *Wikipedia*. Oct. 29, 2023. Accessed: Jan. 05, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Full-text_search&oldid=1182535218
- [40] “Welcome to Apache Solr.” Accessed: Jan. 05, 2024. [Online]. Available: <https://solr.apache.org/index.html>
- [41] “HDFS Architecture Guide.” Accessed: Mar. 03, 2024. [Online]. Available: https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html
- [42] “MySQL :: MySQL 8.0 Reference Manual :: 12.9 Full-Text Search Functions.” Accessed: Jan. 05, 2024. [Online]. Available: <https://dev.mysql.com/doc/refman/8.0/en/fulltext-search.html>
- [43] “What is an Elasticsearch index?,” Elastic Blog. Accessed: Feb. 26, 2024. [Online]. Available: <https://www.elastic.co/blog/what-is-an-elasticsearch-index>
- [44] S. Bonthu and H. Bindu, *Review of Leading Data Analytics Tools*, vol. 7. 2018. doi: 10.14419/ijet.v7i3.31.18190.

- [45] “Power BI - Data Visualization | Microsoft Power Platform.” Accessed: Jan. 05, 2024. [Online]. Available: <https://www.microsoft.com/en-us/power-platform/products/power-bi>
- [46] “Microsoft,” *Wikipedia*. Mar. 01, 2024. Accessed: Mar. 03, 2024. [Online]. Available: <https://en.wikipedia.org/w/index.php?title=Microsoft&oldid=1211293861>
- [47] “On-premises software,” *Wikipedia*. Feb. 27, 2024. Accessed: Mar. 03, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=On-premises_software&oldid=1210659318
- [48] “Business Intelligence and Analytics Software | Tableau.” Accessed: Jan. 05, 2024. [Online]. Available: <https://www.tableau.com/>
- [49] “Grafana: The open observability platform | Grafana Labs.” Accessed: Jan. 05, 2024. [Online]. Available: <https://grafana.com/>
- [50] “Time series,” *Wikipedia*. Jan. 06, 2024. Accessed: Mar. 03, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Time_series&oldid=1193976963
- [51] “Kibana: Explore, Visualize, Discover Data,” Elastic. Accessed: Dec. 13, 2020. [Online]. Available: <https://www.elastic.co/kibana>
- [52] “Thai tokenizer | Elasticsearch Guide [8.11] | Elastic.” Accessed: Jan. 05, 2024. [Online]. Available: <https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-thai-tokenizer.html>
- [53] “N-gram tokenizer | Elasticsearch Guide [8.11] | Elastic.” Accessed: Jan. 05, 2024. [Online]. Available: <https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-ngram-tokenizer.html>
- [54] “Mapping | Elasticsearch Guide [8.12] | Elastic.” Accessed: Feb. 26, 2024. [Online]. Available: <https://www.elastic.co/guide/en/elasticsearch/reference/current/mapping.html>
- [55] “Tech2biz.” Accessed: May 11, 2022. [Online]. Available: <https://www.tech2biz.net/>

ภาคผนวก

ภาคผนวก ก ผลงานเผยแพร่และตีพิมพ์

11th International Conference on Data Science, Technology and Applications (DATA 2022). Held in Lisbon – Portugal, July 11 to 13, 2022.

Multitudinous Data Platform for Community Big Data

S.Junrat, J.Nopparat, M.Manopiroonporn, W.Suntiamorntut and S.Charoenpanyasak

Department of Computer Engineering, Prince of Songkla University, Songkhla, Thailand

sunisa.j@icloud.com, jirasak.r@me.com, mmongkon@gmail.com, wannarat.s@psu.ac.th, sakuna.ch@psu.ac.th

Keywords: big data, diversity data, data platform, data collection, data analytics

Abstract: A city's data is diversified. Many types of data are available, including time series data from various sensors, event data from human input (text, number, date, location, photos, and so on). Those data need a platform that can collect data of a city comprehensively and support querying to analyze data conveniently. We designed and developed a platform that support to automatically construct a variety of data structures by designing a collection of questions that can be used to collect various categories of data sets in each city flexibly. It can be applied to solve problems and develop a city pertinently.

1 INTRODUCTION

Each area of a country such as a city, a sub-district, a district, or a province. There are a variety of environments and local knowledge that may be different or similar to some areas. Many elements influence it, including geography, climate, water resources, way of life, culture, traditions, people's expertise, indigenous knowledge, and so on. Designing a method to collect all of those different types of data and analyze them in various dimensions together is challenging while different forms of each city's data are becoming increasingly massive. There is a greater variety of contexts of interest as well (Ku & Gil-Garcia, 2018).

Data can be many types such as text, number, date, location, and images (Dinh et al., 2020). Data can be collected and created via appropriate forms to serve various requirements pertinently in each area and stored each data type appropriately for later retrieval and analysis according to the time of interest (Halevy et al., 2016). Therefore, we designed a platform to collect a wide variety of data. The platform can create multiple question sets as required of each area. Explorers in each area can input to collect local information into the platform. City administrator, government, researcher, engineer, data scientist can access useful information to solve problems in each area such as disaster and risk management (Munawar et al., 2020), reflect the needs and expectations of its people in London via The Talk London Platform (Vodák et al., 2021).

The platform that we designed and developed can store data flexibly and cover diversity of data. Information that is properly stored can be used to analyze data to manage and plan the development of areas and countries.

2 DATA PLATFORM ARCHITECTURE

A platform that can gather a wide range of local data from many cities in a flexible and comprehensive. It must be developed begin with data structure designing to collect many data categories that is relevant to each area. The platform helps to validate and verify the accuracy of data before store to databases. It makes those data can be utilized in all parts. Therefore, we designed data platform architecture as shown in Figure 1.

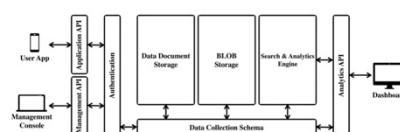


Figure 1: Data Platform Architecture

From Figure 1, data structures are managed through the Management Console which are communicated via Management API. It will be

verified through the Authentication part before accessing the authorized data. Then, the structure of each category is retrieved that can be managed through the Data Collection Schema, which is where the structure of each data group is stored. There will be an exchange of information between the Data Document Storage (collect raw data), the BLOB Storage (collect files) (Saif & Wazir, 2018), and the Search & Analytics Engine (collect data schema for searching especially). Then, the platform will send data that permitted users can access to display on the Management Console.

Recording data obtained from surveying the area via User App, which receives and sends data via the Application API and is verified through the Authentication. Before accessing the information in the platform. If logging in successfully then the platform will retrieve the data from the Data Collection Schema, which is the part that stores the structure of each data category. The platform will then send the user-accessible data back to the User App.

Searching for the required data to display on the dashboard is done through the Analytics API, which is interconnected with the Search & Analytics Engine (create data structure and collect data for search usage especially) and Data Collection Schema. When all the required data have been queried then send those data back to the Dashboard.

3 THE MAIN WORKING PROCESS OF THE PLATFORM

We have sequenced the steps of the important work processes. Each process has been made into a sequence diagram to visualize the workflow more easily. The process to be described here consists of category schema creation, data submission, data deletion, data modification, and data searching and retrieval.

3.1 Category Schema Creation

The structure of the data in each category is essentially defined or set by the experts in each field of knowledge. To obtain an appropriate structure for surveying and recording data in each area; where experts can create data categories through the category management system (Management Console), which we have designed and implemented steps to create data categories (Category Schema Creation) as follows:

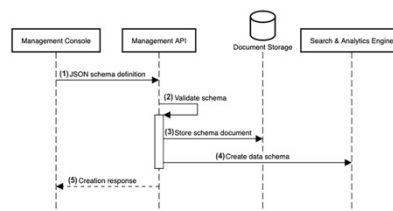


Figure 2: Category Schema Creation

The structure of data in each category is determined by the experts in each field of knowledge. In order to obtain an appropriate structure for surveying and recording data in each area; where experts can create data categories through the category management system (Management Console), which the research team has designed and Develop steps to create data categories (Category Schema Creation) as shown in Figure 2, which has the following steps:

- (1) User who is administrator or data structure creator start to create data structure via Management Console. The data structure is created in JSON format (JSON Schema Definition) (Bourhis et al., 2020). Then the data structure is sent to Management API. JSON Schema Definition will be explained further in the section.
- (2) When Management API has received the data structure, the next step is validated data schema.
- (3) If the data schema correctly, then store it to the Document Storage.
- (4) Then create data schema for searching particularly and send it to Search & Analytics Engine.
- (5) The next step, send the result to Management Console to confirm when save the data schemas successfully.

3.2 Data Submission

Once the data category has been created, user can add information of interested categories through the User App. The process of data submission is shown in Figure 3.

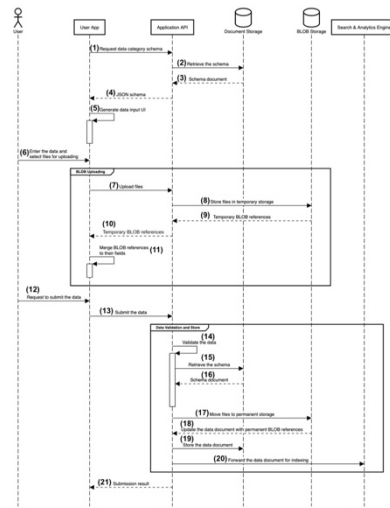


Figure 3 shown a diagram of the procedure for data submission with the following steps:

- (1) User who is recorder the surveyed data request desired data category schema via User App. Then the request is sent to Application API.
- (2) Then Application API get the data category schema from Document Storage.
- (3) When Document Storage found the data category schema, then send the schema document back to Application API.
- (4) Then Application API send the schema document in JSON format (JSON schema) back to User App.
- (5) The next step, User App generate data input UI (data form for filling out) from the JSON schema obtained.
- (6) User starts to input data and upload files.
- (7) If user uploaded files, then User App send the files to Application API.

- (8) The next step, Application API send the files to temporary storage which inside BLOB Storage part.
- (9) Then BLOB Storage send temporary BLOB references to Application API.
- (10) The next step, Application API send temporary BLOB references back to User App.
- (11) Then User App store the BLOB references temporality.
- (12) When the files is uploaded already and user fill the data form completely, then user send request to submit the data via User App.
- (13) Then User App send the data to Application API.
- (14) The next step, Application API validate data.
- (15) If the data correctly, then Application API retrieve the data schema from Document Storage.
- (16) The next step, Document Storage send the schema document to Application API.
- (17) Then Application API move the files to permanent storage.
- (18) The next step, BLOB Storage update the data document with permanent BLOB references to Application API.
- (19) Then Application API store the data document to Document Storage.
- (20) The next step, Application API forward the data document for indexing to Search & Analytics Engine.
- (21) When save the data completely, then Application API send the result back to User App.

3.3 Data Deletion

Once the information in the category of interest has been added to the User App successfully. If filling out mistake and want to delete that data. The User App has a process to delete (Data Deletion) as shown in Figure 4.

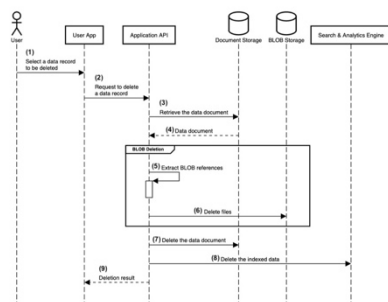


Figure 4: Data Deletion

From Figure 4 shown a diagram of the procedure for data deletion with the following steps:

- (1) User can select a data record to be deleted via User App.
- (2) Then User App send the request to delete a data record to Application API.
- (3) In the next step, Application API retrieve the data document from Document Storage.
- (4) Then Document Storage send data document back to Application API.
- (5) After that, Application API extract BLOB references.
- (6) Then Application API delete files from BLOB storage.
- (7) The next step, Application API delete the data document from document storage.
- (8) Then Application API delete the indexed data from Search & Analytics Engine.
- (9) After that, Application API send the result back to User App.

3.4 Data Modification

Once the information in the category of interest has been added to the User App successfully. If you want to modify some data, the User App has a process of modifying the desired data list (Data Modification), as shown in Figure 5.

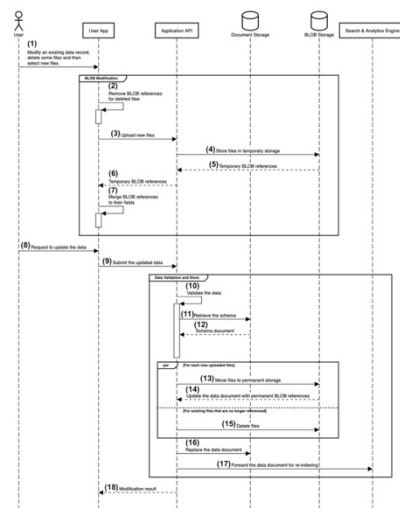


Figure 5: Data Modification

From Figure 5 shown a diagram of the data modification with the following steps:

- (1) User modify an existing data record including delete some files and select new files via User App.
- (2) Then User App remove BLOB references for deleted files.
- (3) After that, User App upload new files to Application API.
- (4) Then Application API store files in temporary storage that is a part of BLOB Storage.
- (5) In the next step, BLOB Storage send temporary BLOB references back to Application API.
- (6) Then Application API send the temporary BLOB references to User App.
- (7) After that, User App keep BLOB references temporality.
- (8) When the files are uploaded and fill out the data completely, user send update the data via User App.

- (9) Then User App send update request to Application API.
- (10) In the next step, Application API validate the data.
- (11) When Application API validate completely, then retrieve the schema from Document Storage.
- (12) Then Document Storage send the schema document back to Application API.
- (13) After that, Application API move files to permanent storage in BLOB Storage.
- (14) Then BLOB Storage send the data document with permanent BLOB references back to Application API.
- (15) In the next step, Application API delete files in BLOB Storage.
- (16) Then Application API replace the data document in Document Storage.
- (17) After that, Application API forward the data document for re-indexing in Search & Analytics Engine.
- (18) When update a data record completely, Application API send the modification result back to User App.

3.5 Data Searching and Retrieval

When data is stored in each category already. To browse history or search for the desired information. We have designed a data searching and retrieval process to query the results that meet the needs of the users. The below process is shown in Figure 6.

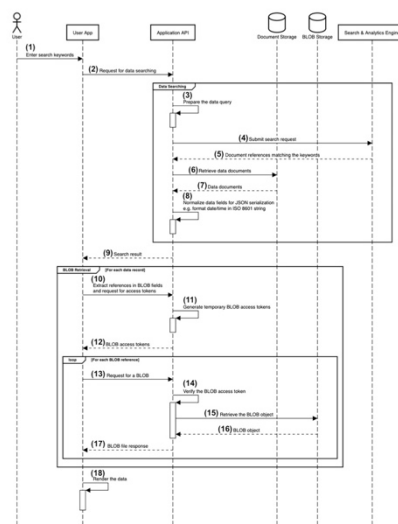


Figure 6: Data Searching and Retrieval

From Figure 6 shown a diagram of the data searching and retrieval with the following steps:

- (1) User enters search keywords via User App.
- (2) Then User App send the request to Application API.
- (3) After that, Application API prepare the data query.
- (4) Then Application API submit search request to Search & Analytics Engine.
- (5) In the next step, Search & Analytics Engine send document references that matching the keywords to Application API.
- (6) Then Application API retrieve data documents from Document Storage.
- (7) After that, Document Storage send data documents back to Application API.
- (8) Then Application API normalize data fields for JSON serialization e.g., format date/time in ISO 8601 string.
- (9) Then Application API send the search result to User App.

- (10) In the next step, User App extract references in BLOB fields and send request for BLOB access tokens to Application API.
- (11) Then Application API generate temporary BLOB access tokens.
- (12) After that, Application API send BLOB access tokens back to User App.
- (13) Then User App request for a BLOB to Application API.
- (14) After that, Application API verify the BLOB access tokens
- (15) Then Application API retrieve the BLOB object from BLOB Storage.
- (16) In the next step, the BLOB object is sent from BLOB Storage to Application API.
- (17) Then Application API send a BLOB file response to User App
- (18) Then User App render the data.

4 JSON SCHEMA DEFINITION

Data schema that is consistently optimized, as well as the ability to grow up in the future. It is critical and necessary. Because of the requirement to accommodate the increasing diversity of data that will be introduced in the future. To make the management, insertion, and alteration of this wide range of data easier. As a result, we created a data storage structure that includes the components depicted in Figure 7.



Figure 7: JSON Schema Definition

From Figure 7 shown data schema in JSON schema definition. There are main components in consist of CategorySchema, DataSection, DataItem, and

DataFieldSpec which we are going to explain detail in the next step.

4.1 CategorySchema

The CategorySchema is used for category creation for store new data category which has five fields below.

- **category** is used to store category id.
- **version** is used for specifying data category version.
- **createdAt** is used for specifying created datetime of data category.
- **updatedAt** is used for specifying updated datetime of data category.
- **sections** are used to collect list of DataSection.

4.2 DataSection

The DataSection has four fields below.

- **title** is used to store name of data section.
- **description** is used to store explanation concisely.
- **metadata** is used to collect addition information.
- **items** are used to collect list of DataItem.

4.3 DataItem

The DataItem has five fields below.

- **id** is used
- **title** is used to store name of data item.
- **description** is used to store explanation concisely.
- **metadata** is used to collect addition information.
- **data** is used to store DataFieldSpec.

4.4 DataFieldSpec

The DataFieldSpec is used to store type of data. In addition, it is also used to validate data that currently is designed to support eleven types. Types of data as follows.

- **Text field** is used to store text data.
- **Number field** is used to store numeric data.
- **Data field** is used to store date.
- **Time field** is used to store time.
- **Date and time field** is used to store date and time.
- **Geo point field** is used to store latitude and longitude.
- **Address field** is used to store address.

- **Drop down field** is used to store one selected data in the form of drop-down input.
- **Radio field** is used to store one selected data in the form of radio button.
- **Check boxes field** is used to select one or more from list of data.
- **File field** is used to store files such as image, pdf, and so on.

5 DEVELOPMENT TOOLS

From Figure 1 shown the data platform architecture which each component has a function as described in 2 data platform architecture section. The main components are the tools used in the development as follows.

5.1 Application API and Management API

In this part, we used serverless cloud architecture in order not to leave the server open all the time and can run when User App or Management Console requests it. Therefore, we have opted for Google Cloud Functions (Malawski et al., 2020) that is a service can charges on a per-user request basis and automatically supports resource scaling. In addition to the Google Cloud Functions, you can choose to use a similar service from Amazon Web Services (Saif & Wazir, 2018) by using a service called AWS Lambda (Malawski et al., 2020) or similar services from other providers.

5.2 Data Collection and Data Document Storage

In this part, we have designed for data storing in document based (NoSQL) (Diogo et al., 2019) format because we want to store data flexibility and want to use the features of serverless (Yu et al., 2020). It is no need to rent a server. Hence, we chose Google Firestore (Crisgar et al., 2021) which meets the requirements and charges according to usage and can supports concurrent connection up to million requests concurrently.

5.3 BLOB Storage

In this part, we have designed to store files flexibility and can expand the storage space automatically. It should be charged based on storage and actual data requests. We want to use the serverless features. Hence, we chose the Google Cloud Storage (Malawski et al., 2020) that meet the requirements.

Besides Google Cloud Storage, you can choose a similar service from Amazon Web Services called AWS S3 (Jonas et al., 2019).

5.4 Search and Analytics Engine

In this part, we have designed to store data in a format that suitable for data searching specifically. It must support horizontal expansion and full-text search which the features can help to fine the desired data quickly and efficiently. Therefore, we chose Elasticsearch (Zamfir et al., 2019) that meet the requirements.

5.5 Implementation Results

We apply visualization tools such as Kibana (Shah et al., 2022), Grafana (Chakraborty & Kundan, 2021), and Power BI (Bhargava et al., 2018) to clearly display useful facts from each city's data.

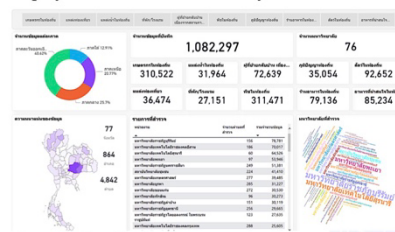


Figure 8: Visualization

From Figure 8 shown number of each data category that all explorers have collected more than one million records from various locations in Thailand. City administrators can access these data and analyze more in depth to see the issues that each community is experiencing. Included are strengths or identities that should be emphasized or encouraged.

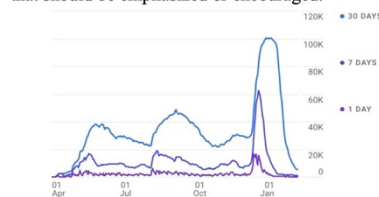


Figure 9: User activity over time

From Figure 9 shown the number of users who record survey data from around the assigned area over the nine months period. Maximum users more than 100K per month.

The platform can handle and scale when facing a lot of concurrent users

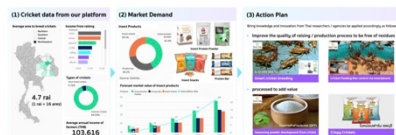


Figure 10: Data Analysis and Utilization

From Figure 10 is divided into three parts:

- (1) First part is cricket data from our platform which is collected by each city explorer. According to the data, the northeastern region has a large number of cricket raisings in Thailand. Nong Khai, Sakon Nakhon, and Udon Thani are the top three provinces for cricket farming. Cricket farming covers approximately 4.7 rai (1 rai = 16 acres). Cricket farmers make an annual average of 103, 616 bath.
- (2) Second part is market demand data from other sources for analysis together to make the information visible from a variety of perspectives. From the proportion of insect products in U.S. for the year 2019-2023 found that approximately 28.4% were made into snacks, 29.2% were made into protein bars, and 42.4% were made into protein powders. Forecast market value in the U.S. has consistently increased. The value of insect products is expected to reach 34 million dollars in 2022, and even more in 2023 which be cost roughly 50 million dollars (*Edible Insects*, n.d.). According to the information, market demand value can help city leaders to analyze the direction of supporting and promoting cricket cultivation and processing in order to meet worldwide demand.
- (3) Third part is action plan. According to the surveyed data combined with market demand data from different sources. When analyzed together, these produce guidelines for development and promotion by bringing knowledge, innovation, and research of Thailand (*Tech2biz*, n.d.) from both government and private agencies to promote modern and standardized cricket farming. To reach a diverse group of customers, it aids in the generation of income for farmers and communities, which results in better economic development of the country in the present and the future.

6 CONCLUSIONS

A variety of data of each area should be collected using the platform to validate the information's accuracy which support data expansion, processing and analyze information for decision-making. As a result, city administrators, researchers, data scientists, and entrepreneurs can gain access to community data and use it to address problems or build communities that represent each city's distinct personality. The platform can continuously provide data. It has the ability to build a high-quality society that is both present and forward-looking.

ACKNOWLEDGEMENT

This paper is funded by PSU FF funding, 2021 under the project title 'Quality data platform and Tambol-ERP'. The authors would like to thank you for these supports.

REFERENCES

- Bhargava, M. G., Kiran, K. T. P. S., & Rao, D. R. (2018). Analysis and Design of Visualization of Educational Institution database using Power BI Tool. *Global Journal of Computer Science and Technology*. <https://computerresearch.org/index.php/computer/article/view/1776>
- Bourhis, P., Reutter, J. L., & Vrgoč, D. (2020). JSON: Data model and query languages. *Information Systems*, 89, 101478. <https://doi.org/10.1016/j.is.2019.101478>
- Chakraborty, M., & Kundan, A. P. (2021). Grafana. In M. Chakraborty & A. P. Kundan (Eds.), *Monitoring Cloud-Native Applications: Lead Agile Operations Confidently Using Open Source Software* (pp. 187–240). Apress. https://doi.org/10.1007/978-1-4842-6888-9_6
- Crisgar, P. V., Wijaya, P. R., Pakpahan, M. D. F., Syamsuddin, E. Y., & Hasanuddin, M. O. (2021). GPS-Based Vehicle Tracking and Theft Detection Systems using Google Cloud IoT Core amp; Firebase. *2021 International Symposium on Electronics and Smart Devices (ISESD)*, 1–6. <https://doi.org/10.1109/ISESD53023.2021.9501928>
- Dinh, L. T. N., Karmakar, G., & Kamruzzaman, J. (2020). A survey on context awareness in big data analytics for business applications. *Knowledge and Information Systems*, 62(9), 3387–3415. <https://doi.org/10.1007/s10115-020-01462-3>
- Diogo, M., Cabral, B., & Bernardino, J. (2019). Consistency Models of NoSQL Databases. *Future Internet*, 11(2), 43. <https://doi.org/10.3390/fi11020043>

- Edible insects: Market value by category in the U.S. 2015-2023*. (n.d.). Statista. Retrieved May 10, 2022, from <https://www.statista.com/statistics/883474/edible-insects-market-value-by-category/>
- Halevy, A., Korn, F., Noy, N. F., Olston, C., Polyzotis, N., Roy, S., & Whang, S. E. (2016). Goods: Organizing Google's Datasets. *Proceedings of the 2016 International Conference on Management of Data*, 795–806. <https://doi.org/10.1145/2882903.2903730>
- Jonas, E., Schleier-Smith, J., Sreekanti, V., Tsai, C.-C., Khandelwal, A., Pu, Q., Shankar, V., Carreira, J., Krauth, K., Yadwadkar, N., Gonzalez, J. E., Popa, R. A., Stoica, I., & Patterson, D. A. (2019). Cloud Programming Simplified: A Berkeley View on Serverless Computing. *ArXiv:1902.03383 [Cs]*. <http://arxiv.org/abs/1902.03383>
- Ku, M., & Gil-Garcia, J. R. (2018). Ready for data analytics?: Data collection and creation in local governments. *Proceedings of the 19th Annual International Conference on Digital Government Research: Governance in the Data Age*, 1–10. <https://doi.org/10.1145/3209281.3209381>
- Malawski, M., Gajek, A., Zima, A., Balis, B., & Figiela, K. (2020). Serverless execution of scientific workflows: Experiments with HyperFlow, AWS Lambda and Google Cloud Functions. *Future Generation Computer Systems*, 110, 502–514. <https://doi.org/10.1016/j.future.2017.10.029>
- Munawar, H. S., Qayyum, S., Ullah, F., & Sepasgozar, S. (2020). Big Data and Its Applications in Smart Real Estate and the Disaster Management Life Cycle: A Systematic Analysis. *Big Data and Cognitive Computing*, 4(2), 4. <https://doi.org/10.3390/bdcc4020004>
- Saif, S., & Wazir, S. (2018). Performance Analysis of Big Data and Cloud Computing Techniques: A Survey. *Procedia Computer Science*, 132, 118–127. <https://doi.org/10.1016/j.procs.2018.05.172>
- Shah, N., Willick, D., & Mago, V. (2022). A framework for social media data analytics using Elasticsearch and Kibana. *Wireless Networks*, 28(3), 1179–1187. <https://doi.org/10.1007/s11276-018-01896-2>
- Tech2biz*. (n.d.). Retrieved May 11, 2022, from <https://www.tech2biz.net/>
- Vodák, J., Šulyová, D., & Kubina, M. (2021). Advanced Technologies and Their Use in Smart City Management. *Sustainability*, 13(10), 5746. <https://doi.org/10.3390/su13105746>
- Yu, T., Liu, Q., Du, D., Xia, Y., Zang, B., Lu, Z., Yang, P., Qin, C., & Chen, H. (2020). Characterizing serverless platforms with serverlessbench. *Proceedings of the 11th ACM Symposium on Cloud Computing*, 30–44. <https://doi.org/10.1145/3419111.3421280>
- Zamfir, V.-A., Carabas, M., Carabas, C., & Tapus, N. (2019). Systems Monitoring and Big Data Analysis Using the Elasticsearch System. *2019 22nd International Conference on Control Systems and Computer Science (CSCS)*, 188–193. <https://doi.org/10.1109/CSCS.2019.00039>

วารสารวิศวกรรมศาสตร์และเทคโนโลยีดิจิทัล : Journal of Engineering and Digital Technology (JEDT) ปีที่ 12 ฉบับที่ 2 (กรกฎาคม - ธันวาคม 2567)



สถาบันเทคโนโลยีไทย-ญี่ปุ่น
Thai-Nichi Institute of Technology
泰日工業大学

สำนักงานกวด ผลิตบัณฑิต สร้างนักประดิษฐ์ ผลิตนวัตกร

1771/1 ถนนพัฒนาการ แขวงสวนหลวง เขตสวนหลวง กรุงเทพฯ 10250
 โทรศัพท์: 0-2763-2600 โทรสาร: 0-2763-2700 Url: www.tni.ac.th

1771/1 Pattanakarn Road, Suanluang, Bangkok 10250 Thailand
 Telephone: 0-2763-2600 Facsimile: 0-2763-2700 E-mail: tniinfo@tni.ac.th

ที่ วน 1501/043

วันที่ 29 เมษายน 2567

เรื่อง ตอบรับบทความวิจัยเพื่อตีพิมพ์ในวารสารวิศวกรรมศาสตร์และเทคโนโลยีดิจิทัล : Journal of Engineering and Digital Technology (JEDT)

เรียน คุณสุวิสา จุลรัตน์ คุณจิรศักดิ์ นพรัตน์ คุณมงคล มโนพิรุฬห์พร คุณวรรณรัช สันติอมรทัต และ คุณสุกฤณา เจริญปัญญาศักดิ์

ตามที่ท่านได้นำส่งบทความวิจัยเรื่อง “การพัฒนาแพลตฟอร์มข้อมูลชุมชนเพื่อการวิเคราะห์และวางแผนจัดการ” และได้ผ่านการพิจารณาโดยผู้ทรงคุณวุฒิเพื่อตีพิมพ์ในวารสารวิศวกรรมศาสตร์และเทคโนโลยีดิจิทัลแล้วนั้น บัดนี้กองบรรณาธิการวารสารฯ พิจารณาแล้ว มีมติเห็นสมควรรับตีพิมพ์ในวารสารวิศวกรรมศาสตร์และเทคโนโลยีดิจิทัล ปีที่ 12 ฉบับที่ 2 (กรกฎาคม – ธันวาคม 2567)

จึงเรียนมาเพื่อโปรดทราบ

ขอแสดงความนับถือ

(รองศาสตราจารย์ ดร.รัตติกร วรากุลศิริพันธุ์)

บรรณาธิการวารสารวิศวกรรมศาสตร์และเทคโนโลยีดิจิทัล

วารสารวิศวกรรมศาสตร์และเทคโนโลยีดิจิทัล

โทร. 0-2763-2600 ต่อ 2704, 2752

อีเมล JEDT@tni.ac.th

ประวัติผู้เขียน

ชื่อ สกุล นางสาวสุนิสา จุรัตน์
รหัสประจำตัวนักศึกษา 6210120042

วุฒิการศึกษา

วุฒิ	ชื่อสถาบัน	ปีที่สำเร็จการศึกษา
วิศวกรรมศาสตรบัณฑิต (วิศวกรรมคอมพิวเตอร์)	มหาวิทยาลัยสงขลานครินทร์	2553

ผลงานตีพิมพ์เผยแพร่

- S. Junrat, J. Nopparat, M. Manopiroonporn, W. Suntiamorntut, and S. Charoenpanyasak, "Multitudinous Data Platform for Community Big Data," in *Proceedings of the 11th International Conference on Data Science, Technology and Applications*, 2022, pp. 328-336, ISBN: 978-989-758-583-8, ISSN: 2184-285X.
- S. Junrat, J. Nopparat, M. Manopiroonporn, W. Suntiamorntut, and S. Charoenpanyasak, "Development of a Community Data Platform for Analytics and Management Planning," *TNI Journal of Engineering and Technology*, vol. 12, no. 2, 2024, ISSN: 2672-9989. (ตอบรับแล้วรอตีพิมพ์)