**บทที่ 3**

**วิธีการดำเนินงาน**

**3.1 แผนการดำเนินงาน**

มีการวางแผนและดำเนินงานตามแผนที่ได้วางไว้ เพื่อให้การจัดทำโครงการดำเนินการไปอย่างมีประสิทธิภาพ โดยเริ่มจากการศึกษาข้อมูล จากนั้นรวบรวมความต้องการของระบบและกำหนดขอบเขตของโครงการ เพื่อนำไปออกแบบโครงสร้างระบบ ออกแบบ User Interface (UI) พัฒนาเว็บไซต์ ทดสอบและแก้ไขปัญหา และจัดทำเอกสารและปริญญานิพนธ์ ซี่งจะแบ่งเป็นกิจกรรมย่อยตั้งแต่เริ่มจัดทำโครงการจนสิ้นสุดโครงการ

**ตารางที่ 3.1** แผนการดำเนินงานของโครงการ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| กิจกรรม | | ปี 2568 | ปี 2569 | | |
| ธ.ค. | ม.ค. | ก.พ. | มี.ค. |
| 1 | ศึกษาข้อมูลที่เกี่ยวข้องกับโครงการ |  |  |  |  |
| 2 | รวบรวมความต้องการของระบบ |  |  |  |  |
| 3 | กำหนดขอบเขตโครงการ |  |  |  |  |
| 4 | ออกแบบโครงสร้างระบบ |  |  |  |  |
| 5 | ออกแบบ UI |  |  |  |  |
| 6 | พัฒนาเว็บไซต์ |  |  |  |  |
| 7 | ทดสอบและแก้ไขปัญหา |  |  |  |  |
| 8 | จัดทำเอกสารและเล่มปริญญานิพนธ์ |  |  |  |  |

แสดงแผนการดำเนินงาน

แสดงการดำเนินงานจริง

**3.2 ภาพรวมของระบบและเครื่องมือที่ใช้**

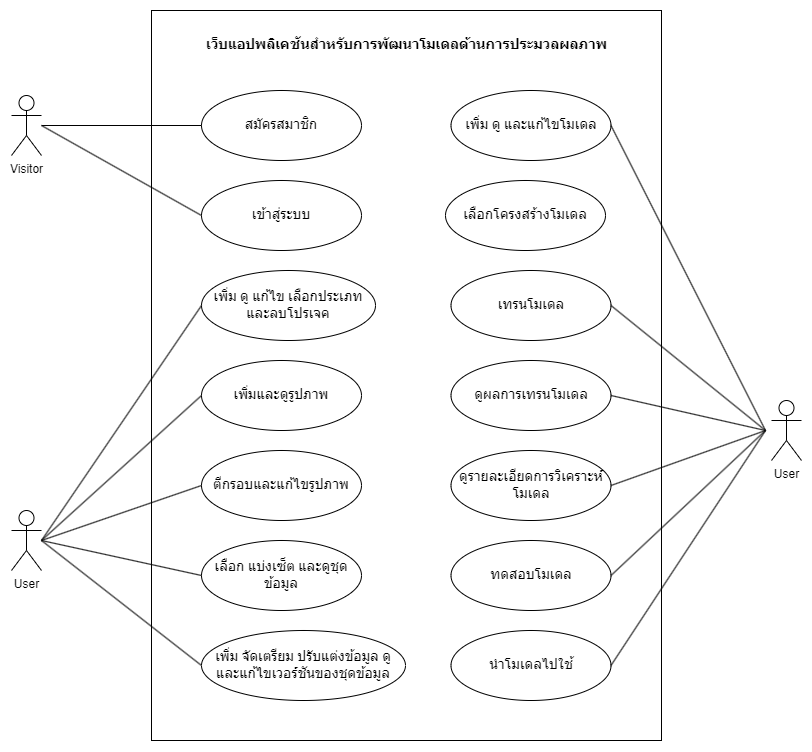
3.2.1 ภาพรวมของระบบ

เว็บแอปพลิเคชันสำหรับการฝึกโมเดลประมวลผลภาพ เป็นเว็บที่ทำการจัดเตรียมชุดข้อมูล และนำชุดข้อมูลเหล่านั้นมาฝึกฝน เพื่อสร้างโมเดลสำหรับนำไปใช้งานในด้านการตรวจจับวัตถุในรูปภาพ และการวิเคราะห์ภาพ โดยผู้ใช้งานต้องลงชื่อเข้าสู่ระบบด้วยอีเมล (email) และรหัสผ่าน (Password) ก่อนทุกครั้งเพื่อเข้าใช้งาน โดยมีรายละเอียดดังนี้

1) ผู้ใช้งาน

* สามารถอัปโหลดข้อมูลรูปภาพ
* สามารถตีกรอบวัตถุที่สนใจในภาพ
* สามารถปรับแต่งรูปภาพ
* สามารถเลือกประเภทการทำงาน
* สามารถเลือกโมเดลที่ต้องการใช้ได้
* สามารถดูผลวิเคราะห์การฝึกฝน ทดสอบ และดาวน์โหลดโมเดล

ซึ่งความสามารถทั้งหมดนี้สรุปเป็นแผนภาพ Use Case ได้ดังรูปที่ 3.1



**รูปที่ 3.1** แผนภาพ Use Case

**3.3 เครื่องมือที่ใช้**

เครื่องมือที่ใช้ในโครงการประกอบด้วยเครื่องมือที่ใช้ในการพัฒนาเว็บแอปพลิเคชันของแต่ละส่วนการทำงาน โดยที่การทำงานหน้าบ้านและหลังบ้านเชื่อมต่อกันผ่าน API และการทำงานหลังบ้านและเซิร์ฟเวอร์ติดต่อสื่อสารกันผ่าน Docker ซึ่งมีรายละเอียดดังนี้

3.3.1 ส่วนของการทำงานหน้าบ้าน (Front-End)

1) การออกแบบ UX/UI

* Figma

2) ภาษาที่ใช้ในการพัฒนา

* JavaScript
* React.js
* Node.js

3.3.2 ส่วนของการทำงานหลังบ้าน (Back-End)

1) ภาษาที่ใช้ในการพัฒนา

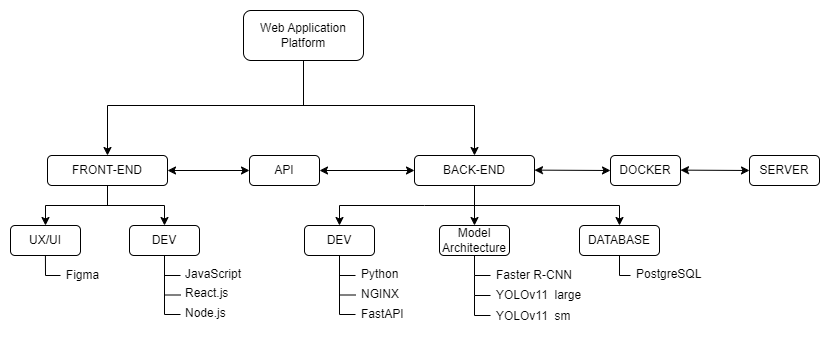
* Python
* NGINX
* FastAPI

2) โครงสร้างของโมเดล

* Faster R-CNN
* YOLOv11-large
* YOLOv11-sm (Small and Medium)

3) ฐานข้อมูล

* PostgreSQL

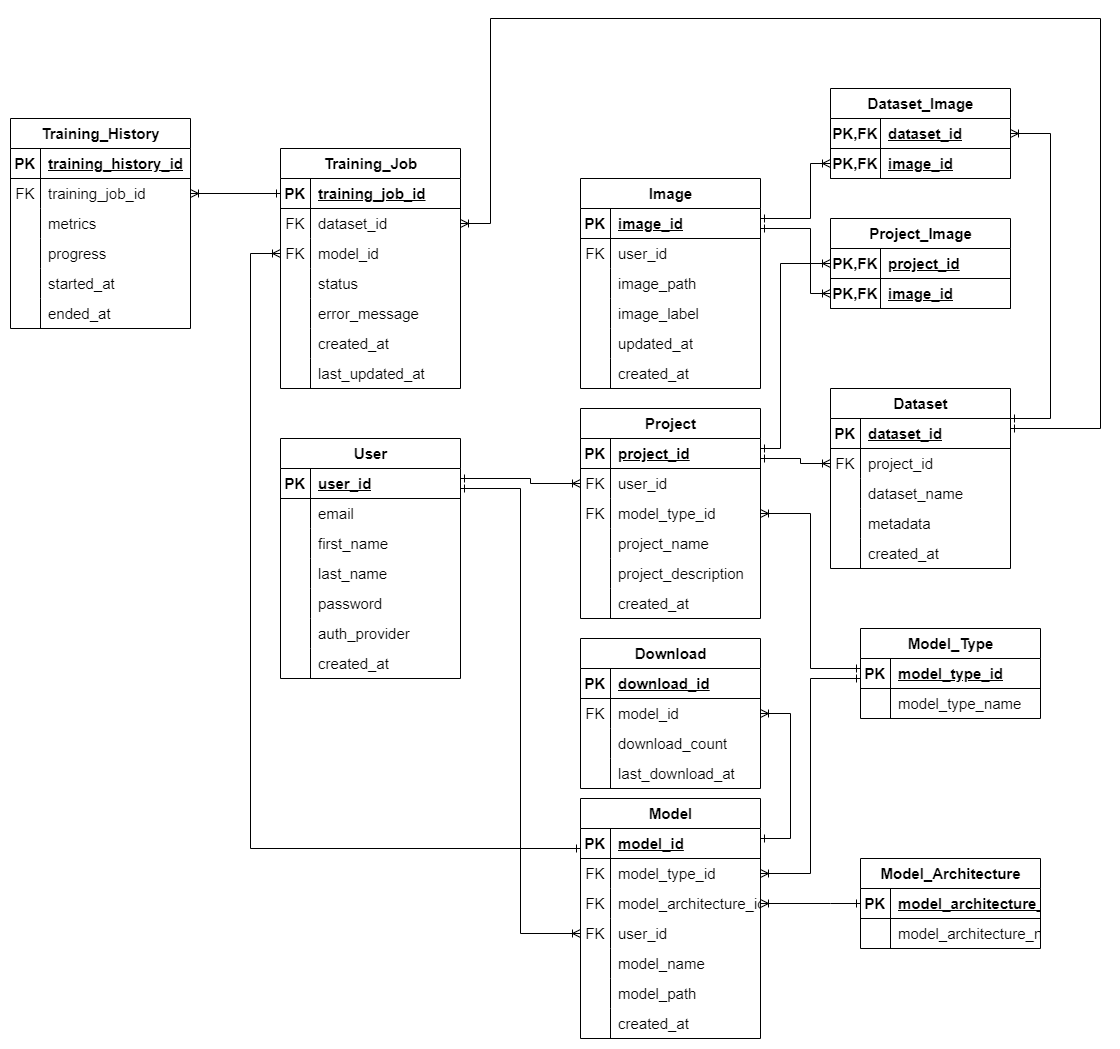


**รูปที่ 3.2** เครื่องมือที่ใช้

**3.4 การออกแบบ**

3.4.1 การออกแบบฐานข้อมูล

ข้อมูลของระบบถูกจัดเก็บในฐานข้อมูลที่จัดการโดยโปรแกรม PostgreSQL ซึ่งเป็นระบบจัดการฐานข้อมูล หรือ Database Management System (DBMS) แบบข้อมูลเชิงสัมพันธ์ (Relational DBMS) ประกอบด้วยตารางเก็บข้อมูล 12 ตารางที่มีความสัมพันธ์กันดังรูปที่ 3.3



**รูปที่ 3.3** ความสัมพันธ์ของข้อมูลในระบบ

1) ตาราง User

ใช้เก็บข้อมูลผู้ใช้งานของระบบ ประกอบด้วยแอทริบิวต์ user\_id ใช้เก็บรหัสผู้ใช้งานระบบเป็นคีย์หลัก (Primary Key, PK) และแอทริบิวต์ email ที่เป็นค่าห้ามซ้ำ (UNIQUE) ใช้เก็บอีเมลของผู้ใช้งาน ดังตารางที่ 3.2

**ตารางที่ 3.2** ตาราง User

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Description** | **Type** | **Domain** | **Details** |
| user\_id | รหัสผู้ใช้งาน | UUID |  | PK |
| email | อีเมลของผู้ใช้งาน | VARCHAR(255) |  | UNIQUE, NOT NULL |
| first\_name | ชื่อจริงของผู้ใช้งาน | VARCHAR(255) |  | NOT NULL |
| last\_name | นามสกุลของผู้ใช้งาน | VARCHAR(255) |  | NOT NULL |
| password | รหัสผ่านที่ถูกเข้ารหัส (เฉพาะกับการเข้าสู่ระบบผ่านอีเมล) | VARCHAR(255) |  | NULL |
| auth\_provider | ผู้ให้บริการการยืนยันตัวตน | ENUM | email, google | NOT NULL |
| created\_at | เวลาที่บัญชีผู้ใช้ถูกสร้างขึ้น | BIGINT |  | NOT NULL |

2) ตาราง Project

ใช้เก็บข้อมูลโปรเจกต์ที่ผู้ใช้งานสร้างขึ้นในระบบ ประกอบด้วยแอทริบิวต์ project\_id ใช้เก็บรหัสของโปรเจกต์เป็นคีย์หลัก (PK) มีแอทริบิวต์ user\_id เป็นคีย์อ้างอิง (Foreign Key, FK) มาจากตาราง User เพื่อระบุว่าโปรเจกต์นี้เป็นของผู้ใช้งานคนใด และ model\_type\_id เป็นคีย์อ้างอิงจากตาราง Model\_Type เพื่อระบุประเภทของโมเดล AI ที่ใช้ในโปรเจกต์ ดังตารางที่ 3.3

**ตารางที่ 3.3** ตาราง Project

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Description** | **Type** | **Domain** | **Details** |
| project\_id | รหัสโปรเจคต์ | UUID |  | PK |
| user\_id | รหัสผู้ใช้ที่เป็นเจ้าของโปรเจคต์ | UUID |  | FK, NOT NULL |
| model\_type\_id | รหัสประเภทโมเดลที่ใช้ในโปรเจคต์ | UUID |  | FK, NOT NULL |
| project\_name | ชื่อของโปรเจคต์ | VARCHAR(255) |  | NOT NULL |
| project\_description | คำอธิบายโปรเจคต์ | VARCHAR(255) |  | NOT NULL |
| created\_at | เวลาที่โปรเจคต์ถูกสร้างขึ้น | BIGINT |  | NOT NULL |

3) ตาราง Dataset

ใช้เก็บข้อมูลชุดข้อมูลที่ใช้สำหรับการฝึกโมเดล AI ประกอบด้วยแอทริบิวต์ dataset\_id ใช้เก็บรหัสของชุดข้อมูลเป็นคีย์หลัก (PK) มีแอทริบิวต์ project\_id เป็นคีย์อ้างอิง (FK) มาจากตาราง Project เพื่อระบุว่าชุดข้อมูลนี้เป็นของโปรเจกต์ใด ดังตารางที่ 3.4

**ตารางที่ 3.4** ตาราง Dataset

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Description** | **Type** | **Domain** | **Details** |
| dataset\_id | รหัสชุดข้อมูล | UUID |  | PK |
| project\_id | รหัสโปรเจคต์ที่เป็นเจ้าของชุดข้อมูล | UUID |  | FK, NOT NULL |
| dataset\_name | ชื่อของชุดข้อมูล | VARCHAR(255) |  | NOT NULL |
| metadata | ข้อมูลเมตาในรูปแบบ JSON (partition, augmentation, preprocessing, ฯลฯ) | JSONB |  | NULL |
| created\_at | เวลาที่ชุดข้อมูลถูกสร้างขึ้น | BIGINT |  | NOT NULL |

4) ตาราง Image

ใช้เก็บข้อมูลของรูปภาพที่อัปโหลดเข้ามาในระบบ ประกอบด้วยแอทริบิวต์ image\_id ใช้เก็บรหัสของรูปภาพเป็นคีย์หลัก (PK) มีแอทริบิวต์ user\_id เป็นคีย์อ้างอิง (FK) มาจากตาราง User เพื่อระบุว่าผู้ใช้งานคนใดเป็นผู้อัปโหลดรูปภาพ ดังตารางที่ 3.5

**ตารางที่ 3.5** ตาราง Image

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Description** | **Type** | **Domain** | **Details** |
| image\_id | รหัสรูปภาพ | UUID |  | PK |
| user\_id | รหัสผู้ใช้ที่อัปโหลดรูปภาพ | UUID |  | FK, NOT NULL |
| file\_path | ที่อยู่จัดเก็บไฟล์รูปภาพ | VARCHAR(255) |  | NOT NULL |
| labels | ข้อมูลคำอธิบายในรูปแบบ JSON (bounding boxes, labeling) | JSONB |  | NULL |
| updated\_at | เวลาที่ข้อมูลของรูปภาพถูกอัปเดตล่าสุด | BIGINT |  | NOT NULL |
| created\_at | เวลาที่รูปภาพถูกอัปโหลด | BIGINT |  | NOT NULL |

5) ตาราง Project\_Image

ใช้เชื่อมความสัมพันธ์ระหว่างโปรเจกต์และรูปภาพ โดยมี project\_id และ image\_id เป็นคีย์หลักร่วม (Composite PK) และเป็นคีย์อ้างอิง (FK) จากตาราง Project และ Image ดังตารางที่ 3.6

**ตารางที่ 3.6** ตาราง Project\_Image

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Description** | **Type** | **Domain** | **Details** |
| project\_id | รหัสชุดข้อมูลที่เกี่ยวข้องกับรูปภาพ | UUID |  | PK, FK |
| image\_id | รหัสรูปภาพที่ใช้ในโปรเจคต์ | UUID |  | PK, FK |

6) ตาราง Dataset\_Image

ใช้เชื่อมความสัมพันธ์ระหว่างชุดข้อมูลและรูปภาพ โดยมี dataset\_id และ image\_id เป็นคีย์หลักร่วม (Composite PK) และเป็นคีย์อ้างอิง (FK) จากตาราง Dataset และ Image ดังตารางที่ 3.7

**ตารางที่ 3.7** ตาราง Dataset\_Image

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Description** | **Type** | **Domain** | **Details** |
| dataset\_id | รหัสชุดข้อมูลที่เกี่ยวข้องกับรูปภาพ | UUID |  | PK, FK |
| image\_id | รหัสรูปภาพที่ใช้ในชุดข้อมูล | UUID |  | PK, FK |

7) ตาราง Training\_Job

ใช้เก็บข้อมูลของกระบวนการฝึกโมเดล AI ประกอบด้วยแอทริบิวต์ training\_job\_id ใช้เก็บรหัสของงานฝึกเป็นคีย์หลัก (PK) มีแอทริบิวต์ user\_id เป็นคีย์อ้างอิง (FK) มาจากตาราง User เพื่อระบุว่าผู้ใช้งานคนใดเป็นผู้เริ่มการฝึก มีแอทริบิวต์ dataset\_id เป็นคีย์อ้างอิง (FK) มาจากตาราง dataset ใข้เก็บรหัสโมเดลที่ถูกเทรน ดังตารางที่ 3.8

**ตารางที่ 3.8** ตาราง Training\_Job

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Description** | **Type** | **Domain** | **Details** |
| training\_job\_id | รหัสงานที่ต้องเทรน | UUID |  | PK |
| user\_id | รหัสผู้ใช้ที่เริ่มงานเทรน | UUID |  | FK, NOT NULL |
| dataset\_id | รหัสชุดข้อมูลที่ใช้ในการฝึกสอน | UUID |  | FK, NOT NULL |
| model\_id | รหัสโมเดลที่ถูกเทรน | UUID |  | FK, NOT NULL |
| status | สถานะปัจจุบัน | ENUM |  | NOT NULL |
| last\_updated\_at | เวลาล่าสุดที่มีการอัปเดตสถานะ | BIGINT |  | NOT NULL |
| error\_message | ข้อความแสดงข้อผิดพลาดที่เกิดขึ้นระหว่างเทรน | VARCHAR(255) |  | NULL |
| created\_at | เวลาที่งานเทรนถูกสร้างขึ้น | BIGINT |  | NOT NULL |

8) ตาราง Training\_History

ใช้เก็บข้อมูลประวัติการฝึกของโมเดล AI แต่ละครั้ง ประกอบด้วยแอทริบิวต์ training\_history\_id ใช้เก็บรหัสของประวัติการฝึกเป็นคีย์หลัก (PK) มีแอทริบิวต์ training\_job\_id เป็นคีย์อ้างอิง (FK) มาจากตาราง Training\_Job เพื่อเชื่อมโยงข้อมูลกับงานฝึก ดังตารางที่ 3.9

**ตารางที่ 3.9** ตาราง Training\_History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Description** | **Type** | **Domain** | **Details** |
| training\_history\_id | รหัสประวัติการเทรน | UUID |  | PK |
| training\_job\_id | รหัสงานฝึกสอน | UUID |  | FK, NOT NULL |
| metrics | ข้อมูลเมทริกในรูปแบบ JSON (precision, mAP, loss) | JSONB |  | NOT NULL |
| progress | ข้อมูลการอัปเดตความคืบหน้าต่อ epoch ในรูปแบบ JSON (loss และ accuracy) | JSONB |  | NOT NULL |
| started\_at | เวลาที่เริ่มต้นการเทรน | BIGINT |  | NOT NULL |
| ended\_at | เวลาที่สิ้นสุดการเทรน | BIGINT |  | NULL |

9) ตาราง Download

ใช้เก็บข้อมูลการดาวน์โหลดโมเดลของผู้ใช้งาน ประกอบด้วยแอทริบิวต์ download\_id ใช้เก็บรหัสของการดาวน์โหลดเป็นคีย์หลัก (PK) มีแอทริบิวต์ user\_id เป็นคีย์อ้างอิง (FK) มาจากตาราง User เพื่อระบุว่าผู้ใช้งานคนใดเป็นผู้ดาวน์โหลดโมเดล ดังตารางที่ 3.10

**ตารางที่ 3.10** ตาราง Download

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Description** | **Type** | **Domain** | **Details** |
| download\_id | รหัสดาวน์โหลด | UUID |  | PK |
| user\_id | รหัสผู้ใช้ที่ดาวน์โหลด | UUID |  | FK, NOT NULL |
| model\_id | รหัสโมเดลที่ถูกดาวน์โหลด | UUID |  | FK, NOT NULL |
| download\_count | จำนวนครั้งที่โมเดลถูกดาวน์โหลด | INT |  | NOT NULL |
| last\_download\_at | เวลาการดาวน์โหลดโมเดลล่าสุด | BIGINT |  | NOT NULL |

10) ตาราง Model\_Architecture

ใช้เก็บข้อมูลเกี่ยวกับสถาปัตยกรรมของโมเดล AI ประกอบด้วยแอทริบิวต์ model\_architecture\_id ใช้เก็บรหัสของสถาปัตยกรรมโมเดลเป็นคีย์หลัก (PK) และ model\_architecture\_name ใช้เก็บชื่อของสถาปัตยกรรมโมเดล ดังตารางที่ 3.11

**ตารางที่ 3.11** ตาราง Model\_Architecture

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Description** | **Type** | **Domain** | **Details** |
| model\_architecture\_id | รหัสสถาปัตยกรรมโมเดล | UUID |  | PK |
| model\_architecture\_name | ชื่อของสถาปัตยกรรมโมเดล | VARCHAR(255) |  | NOT NULL |

11) ตาราง Model\_Type

ใช้เก็บข้อมูลประเภทของโมเดล AI ประกอบด้วยแอทริบิวต์ model\_type\_id ใช้เก็บรหัสของประเภทโมเดลเป็นคีย์หลัก (PK) และ model\_type\_name ใช้เก็บชื่อของประเภทโมเดล เช่น Object Detection, Image Classification ดังตารางที่ 3.12

**ตารางที่ 3.12** ตาราง Model\_Type

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Description** | **Type** | **Domain** | **Details** |
| model\_type\_id | รหัสประเภทโมเดล | UUID |  | PK |
| model\_type\_name | ชื่อของประเภทโมเดล | VARCHAR(255) |  | NOT NULL |

12) ตาราง Model

ใช้เก็บข้อมูลของโมเดล AI ที่ถูกฝึกเสร็จแล้ว ประกอบด้วยแอทริบิวต์ model\_id ใช้เก็บรหัสของโมเดลเป็นคีย์หลัก (PK) มีแอทริบิวต์ model\_type\_id เป็นคีย์อ้างอิง (FK) มาจากตาราง Model\_Type เพื่อระบุประเภทของโมเดล มีแอทริบิวต์ model\_architecture\_id เป็นคีย์อ้างอิง (FK) มาจากตาราง Model\_Architecture เพื่อระบุสถาปัตยกรรมของโมเดล มีแอทริบิวต์ user\_id เป็นคีย์อ้างอิง (FK) มาจากตาราง user ใช้เก็บผู้ใช้งานโมเดล ดังตารางที่ 3.13

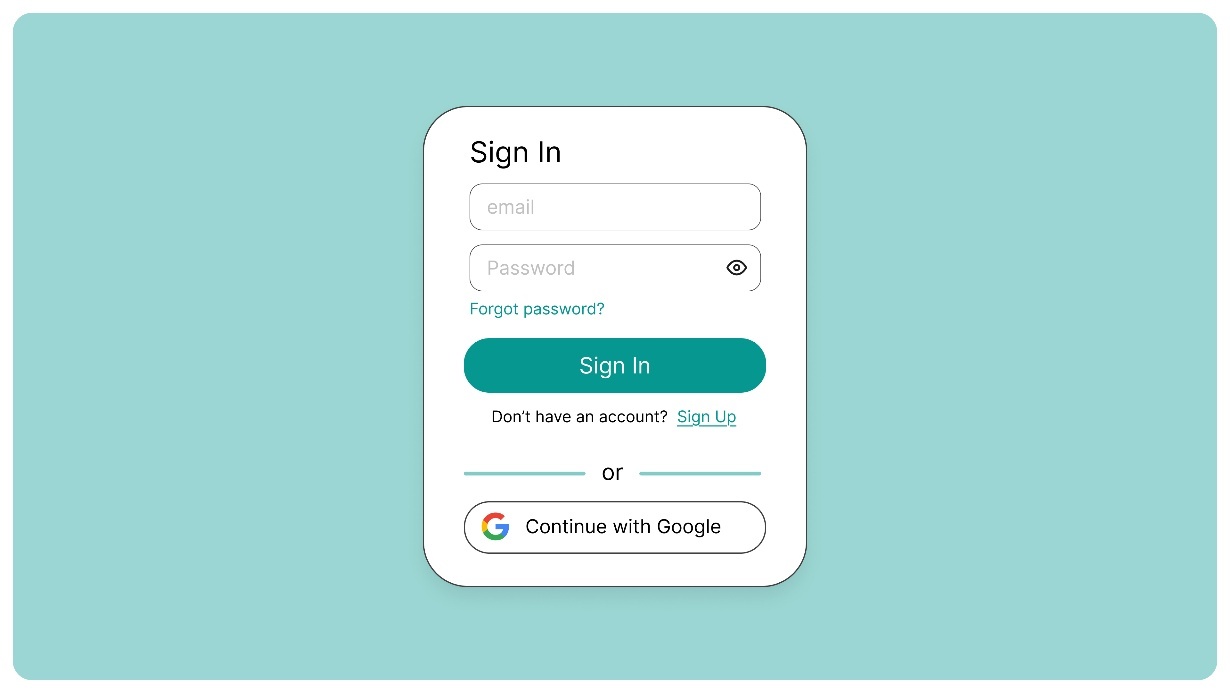
**ตารางที่ 3.13** ตาราง Model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Description** | **Type** | **Domain** | **Details** |
| model\_id | รหัสโมเดล | UUID |  | PK |
| model\_type\_id | รหัสประเภทโมเดล | UUID |  | FK, NOT NULL |
| model\_architecture\_id | รหัสสถาปัตยกรรมของโมเดล | UUID |  | FK, NOT NULL |
| user\_id | รหัสผู้ใช้ที่เป็นเจ้าของโมเดล | UUID |  | FK, NOT NULL |
| model\_name | ชื่อของโมเดล | VARCHAR(255) |  | NOT NULL |
| model\_path | ที่อยู่จัดเก็บไฟล์โมเดล | VARCHAR(255) |  | NOT NULL |
| created\_at | เวลาที่โมเดลถูกบันทึกลงในระบบ | BIGINT |  | NOT NULL |

3.4.2 การออกแบบ UI ของเว็บแอปพลิเคชัน

1) หน้าเข้าสู่ระบบ

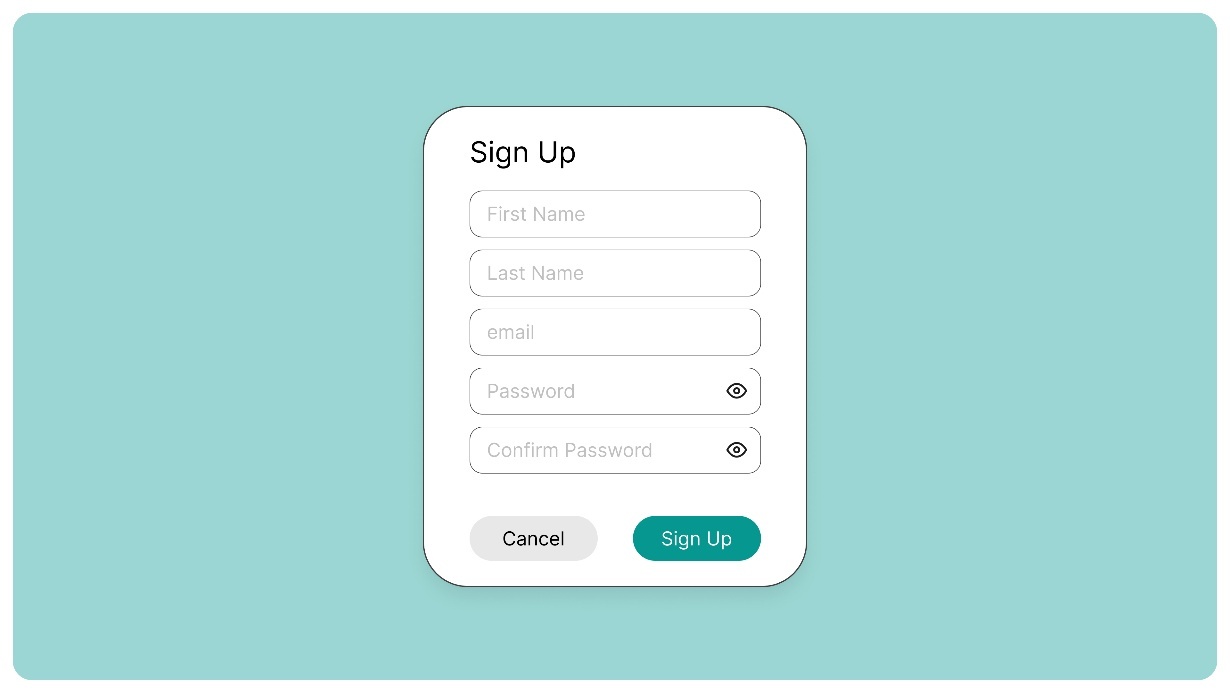
เข้าสู่ระบบโดยการป้อนอีเมลและรหัสผ่านจากการสมัครของผู้ใช้งาน จะมีปุ่มเข้าสู่ระบบการใช้งาน หรือเข้าสู่ระบบโดยผ่าน Google (Gmail) ดังรูปที่ 3.4



**รูปที่ 3.4** การออกแบบหน้าเข้าสู่ระบบผู้ใช้งาน

2) หน้าสมัครสมาชิก

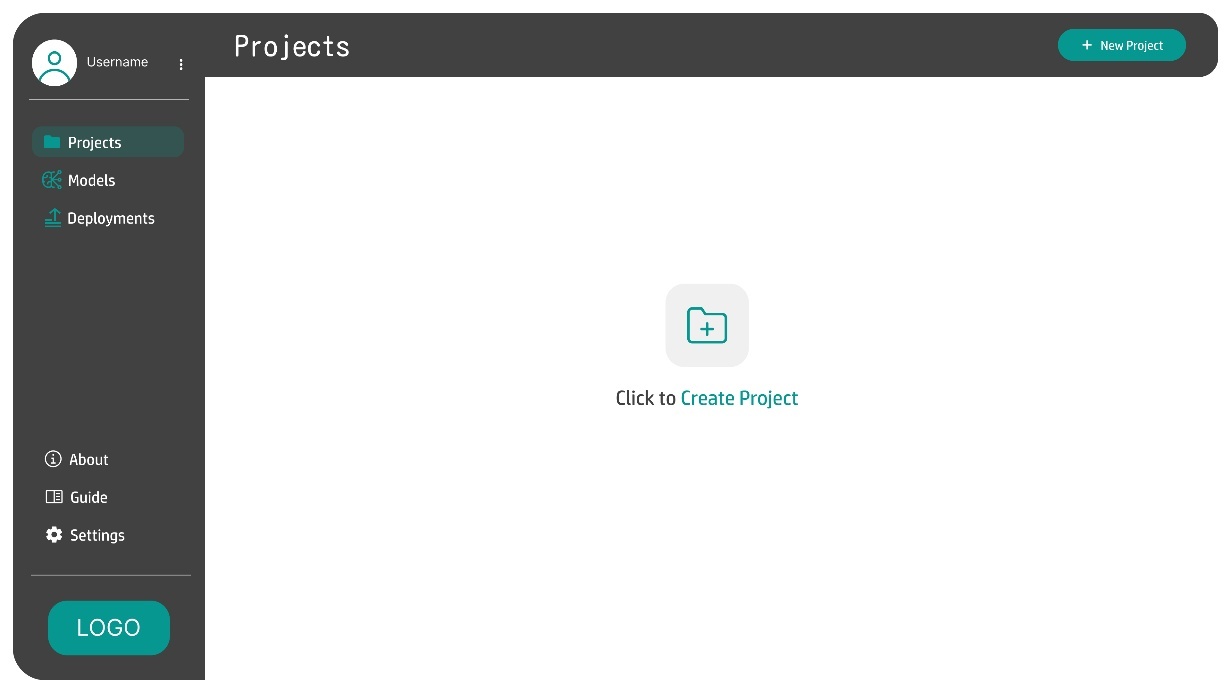
ต้องทำการสมัครสมาชิกโดยกรอกชื่อจริง นามสกุล อีเมล รหัสผ่าน และยืนยันรหัสผ่าน โดยจะมีปุ่มกดเพื่อยืนยันการสมัครสมาชิก ดังรูปที่ 3.5



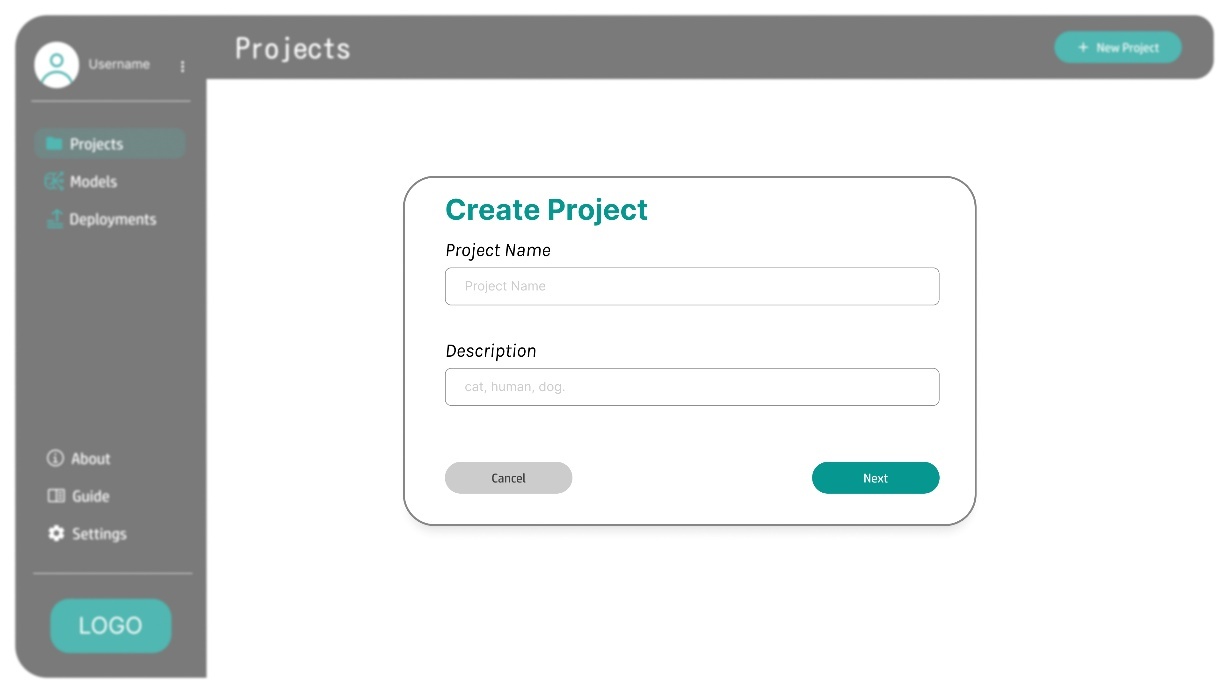
**รูปที่ 3.5** การออกแบบหน้าสมัครสมาชิกเพื่อเข้าสู่ระบบ

3) หน้าสร้างโปรเจค

หน้าสร้างโปรเจคเป็นหน้าแรกของเว็บแอปพลิเคชันหลังจากที่ผู้ใช้เข้าสู่ระบบมา ผู้ใช้สามารถสร้างโปรเจคได้โดยการกดปุ่ม New Project ที่มุมขวาบน หรือ Create Project ที่พื้นที่ตรงกลางของหน้าสร้างโปรเจค จากนั้นกรอกชื่อโปรเจคและคำอธิบายโปรเจค เลือกประเภทของโปรเจค (Object Detection และ Classification) และจะแสดงหน้าโปรเจคที่ถูกสร้างขึ้นมา ดังรูปที่ 3.6 – 3.9 ตามลำดับ



**รูปที่ 3.6** การออกแบบหน้าสร้างโปรเจค

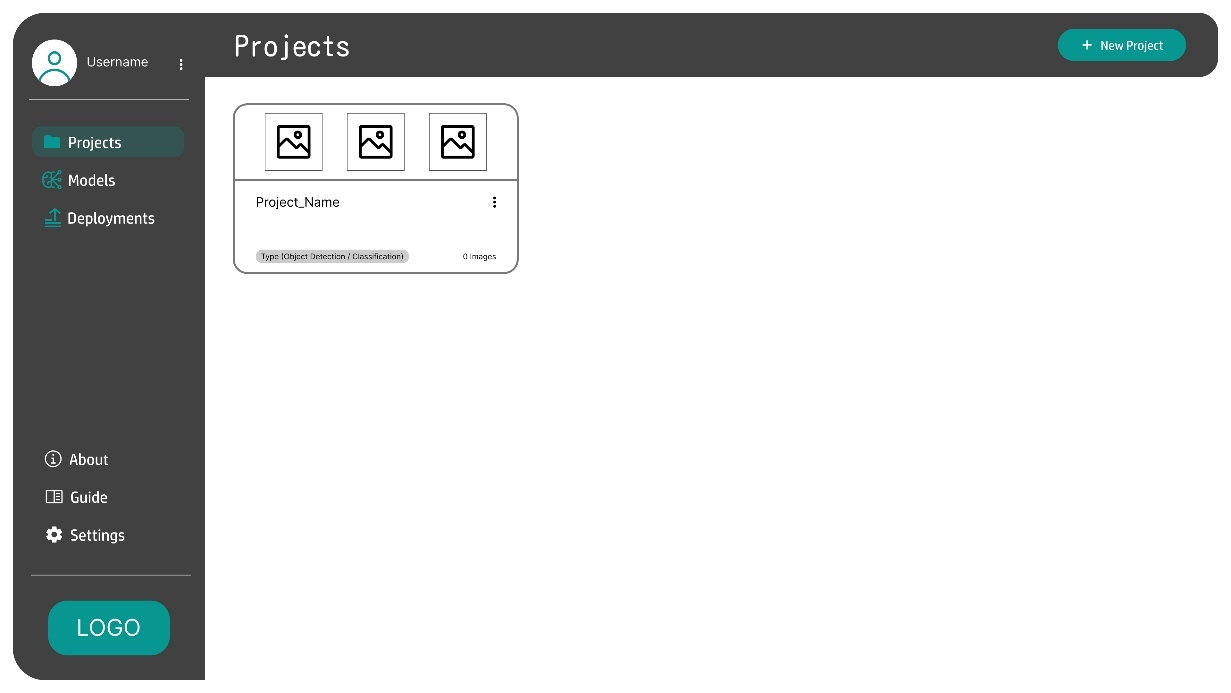


**รูปที่ 3.7** การออกแบบหน้าใส่รายละเอียดโปรเจค

A screenshot of a computer

AI-generated content may be incorrect.

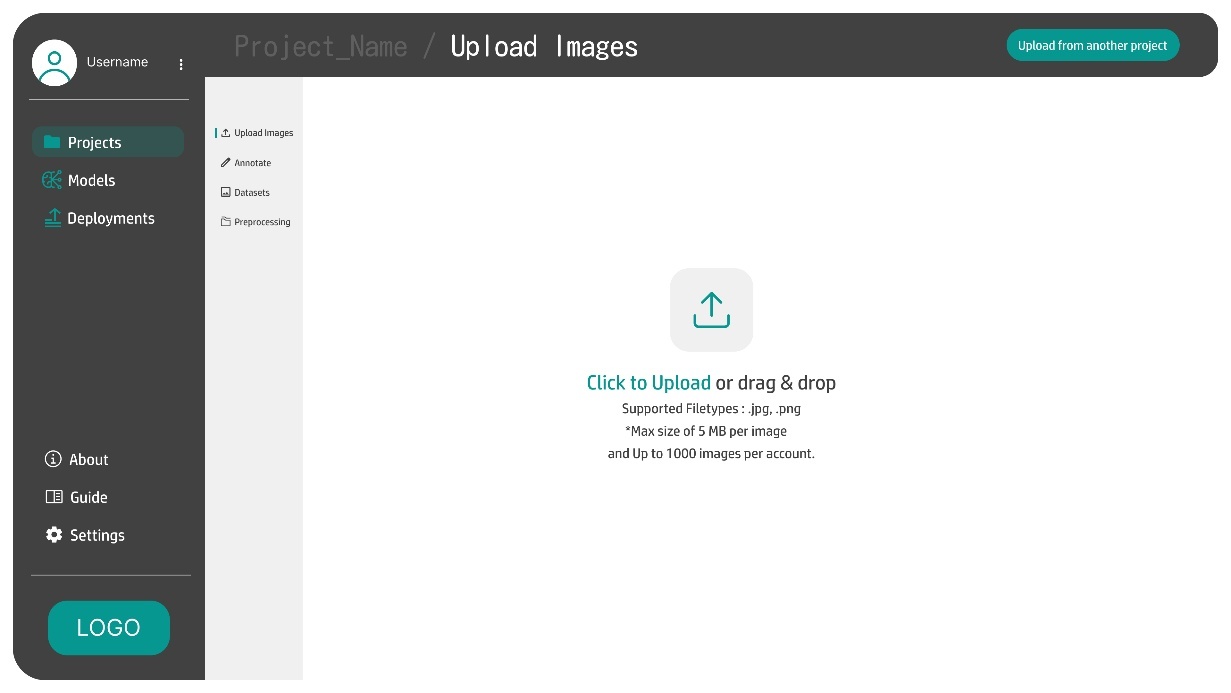
**รูปที่ 3.8** การออกแบบหน้าเลือกประเภทของโปรเจค



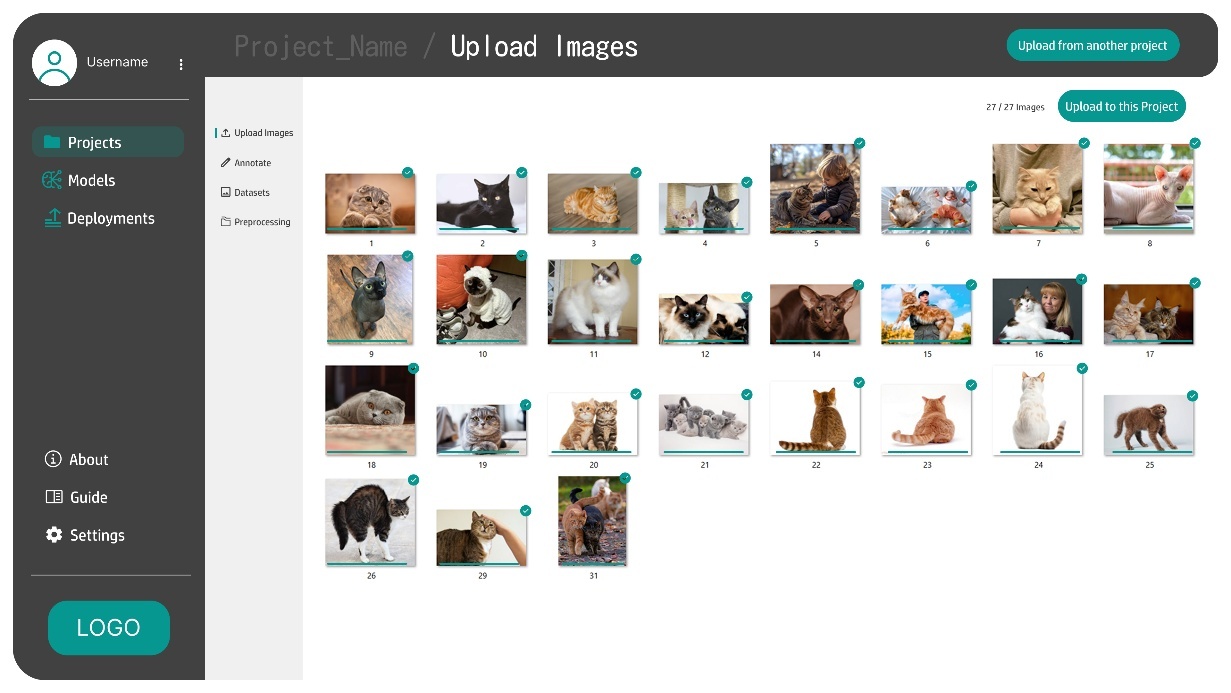
**รูปที่ 3.9** การออกแบบหน้าแสดงโปรเจค

4) หน้าอัปโหลดรูปภาพ

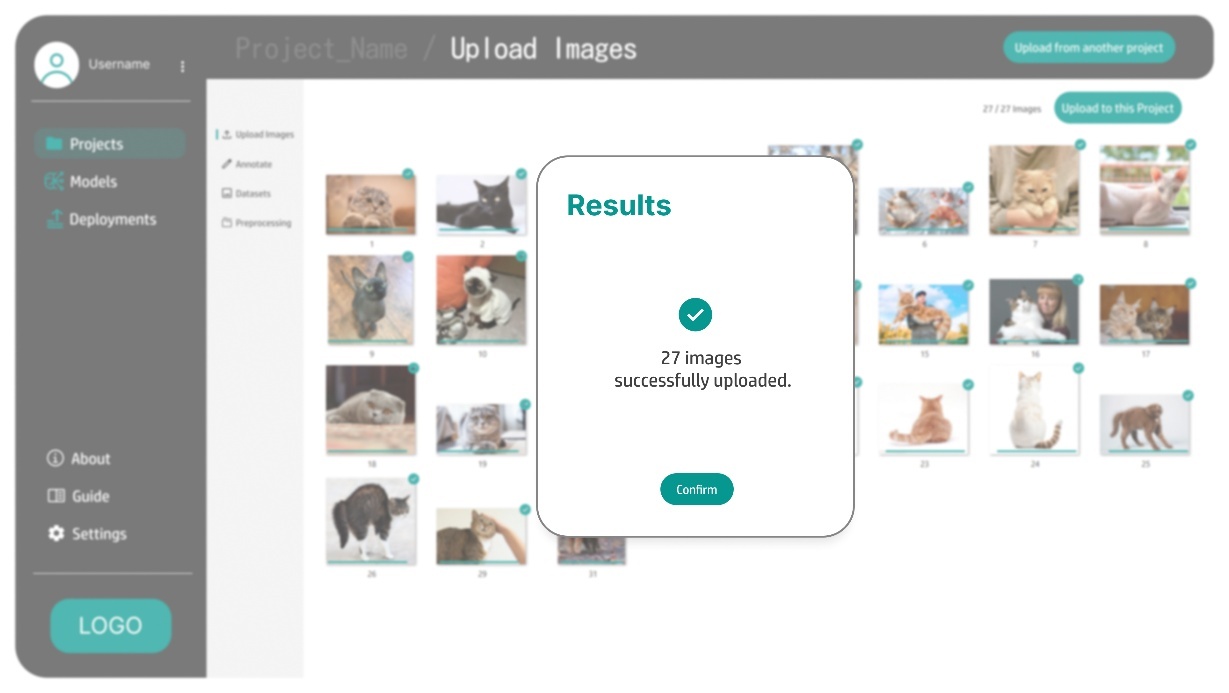
หน้าอัปโหลดรูปภาพจะแสดงหลังจากที่กดเข้ามาในโปรเจคที่ถูกสร้างขึ้น ผู้ใช้สามารถอัปโหลดรูปภาพได้โดยการกดปุ่ม Click to Upload ที่พื้นที่ตรงกลางของหน้าอัปโหลดรูปภาพ และสามารถอัปโหลดรูปภาพที่เคยถูกอัปโหลดไว้ในโปรเจคอื่นได้โดยการกดปุ่ม Upload from another project ที่มุมขวาบน เมื่อทำการอัปโหลดรูปภาพแล้ว จะมีปุ่ม Upload to this Project เพื่อยืนยันการอัปโหลดรูปภาพลงโปรเจค และจะมีหน้า Pop-Up แสดงผลลัพธ์การอัปโหลด ดังรูปที่ 3.10 – 3.12 ตามลำดับ



**รูปที่ 3.10** การออกแบบหน้าอัปโหลดรูปภาพ



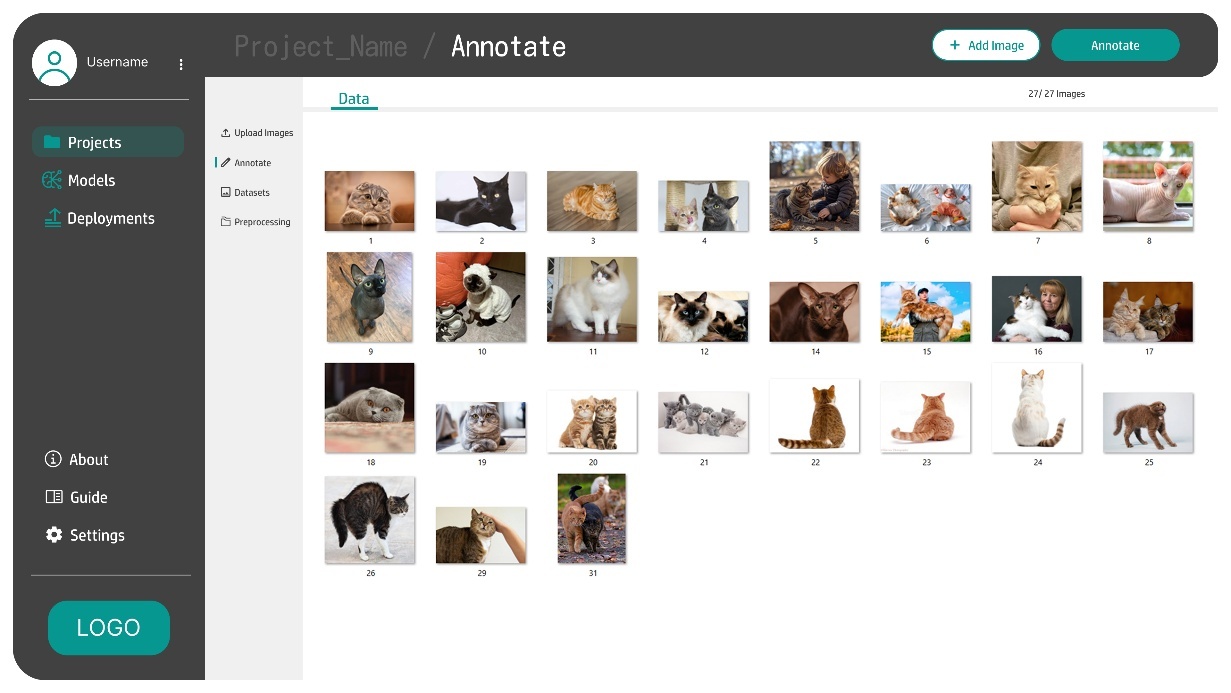
**รูปที่ 3.11** การออกแบบหน้าแสดงรูปภาพที่ถูกอัปโหลด



**รูปที่ 3.12** การออกแบบหน้า Pop-Up แสดงผลลัพธ์การอัปโหลด

5) หน้าแสดงรูปภาพในโปรเจค

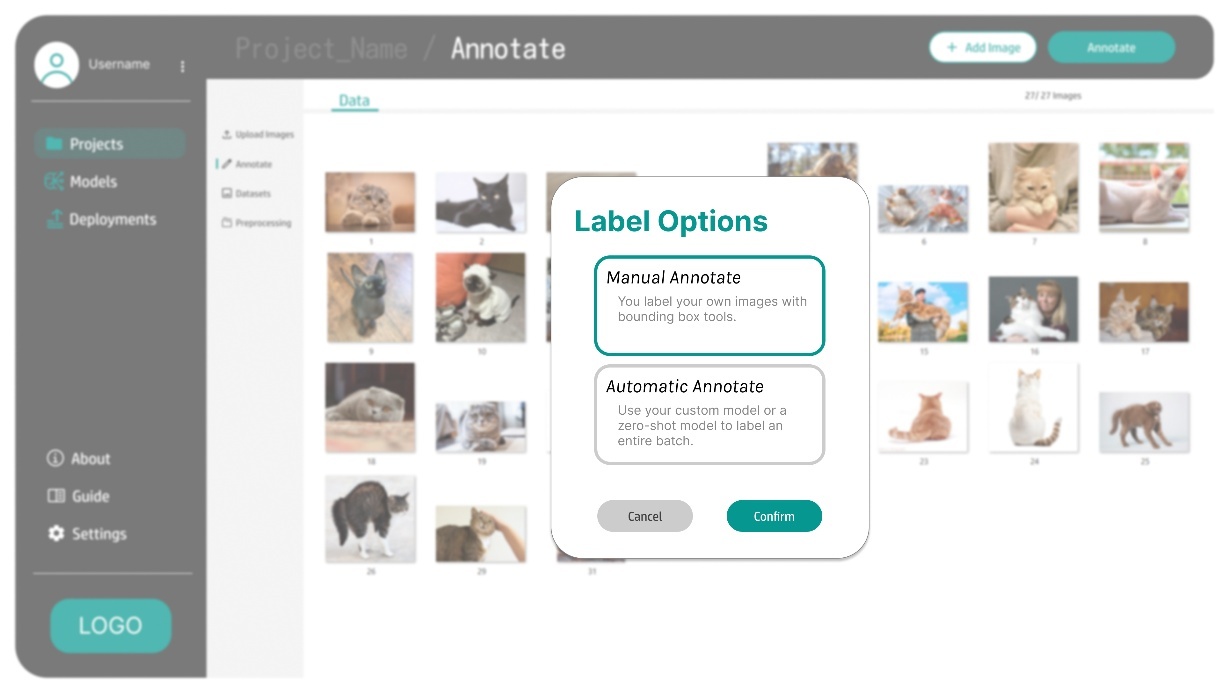
หน้าแสดงรูปภาพในโปรเจคจะแสดงหลังจากการอัปโหลดรูปภาพลงโปรเจค ผู้ใช้สามารถเพิ่มรูปภาพเพิ่มลงโปรเจคนี้ได้โดยการปุ่ม Add Image และสามารถเริ่มทำการตีกรอบรูปภาพได้โดยการกดปุ่ม Annotate ที่มุมขวาบน ดังรูปที่ 3.13



**รูปที่ 3.13** การออกแบบหน้าแสดงรูปภาพในโปรเจค

6) หน้าตัวเลือกการตีกรอบวัตถุในภาพ

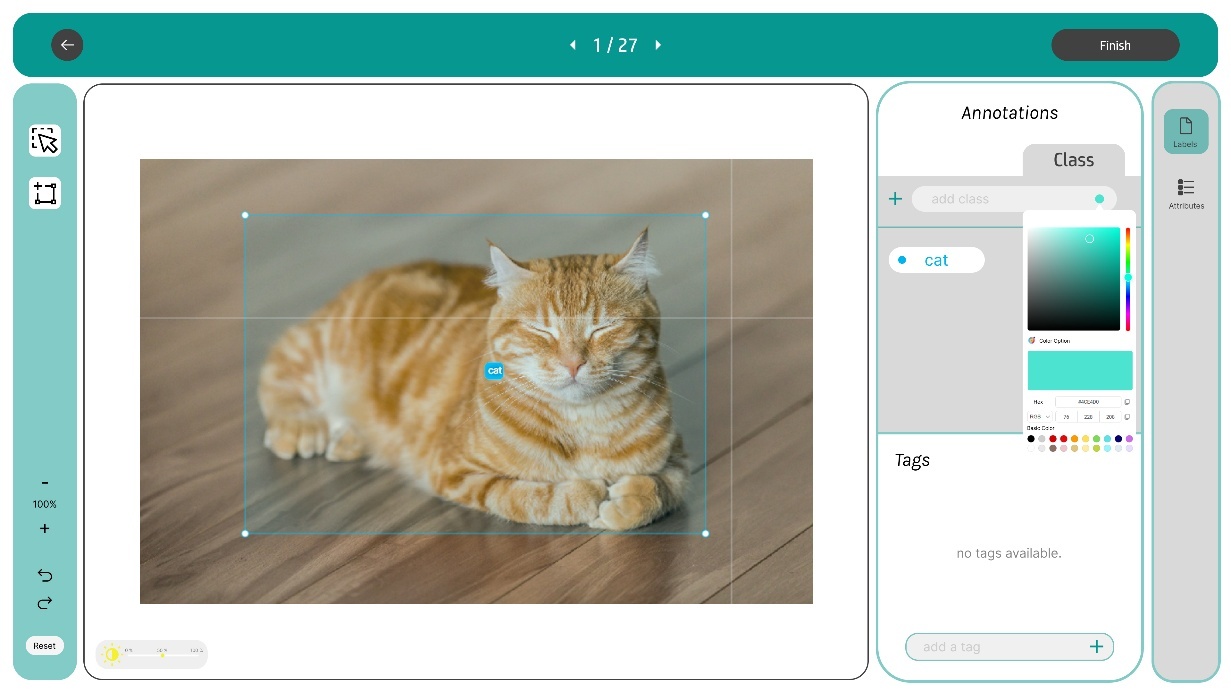
หน้า Pop-Up ตัวเลือกการตีกรอบในภาพนี้จะแสดงหลังจากที่กดปุ่ม Annotate โดยผู้ใช้สามารถเลือกได้ว่าจะทำการตีกรอบวัตถุเอง (Manual Annotate) หรือจะเลือกทำการตีกรอบแบบอัตโนมัติ (Automatic Annotate) และจะมีปุ่มกดเพื่อยืนยันการเลือกของผู้ใช้ ดังรูปที่ 3.14



**รูปที่ 3.14** การออกแบบหน้าตัวเลือกการตีกรอบวัตถุในภาพ

7) หน้าการตีกรอบวัตถุในรูปภาพแบบ Manual

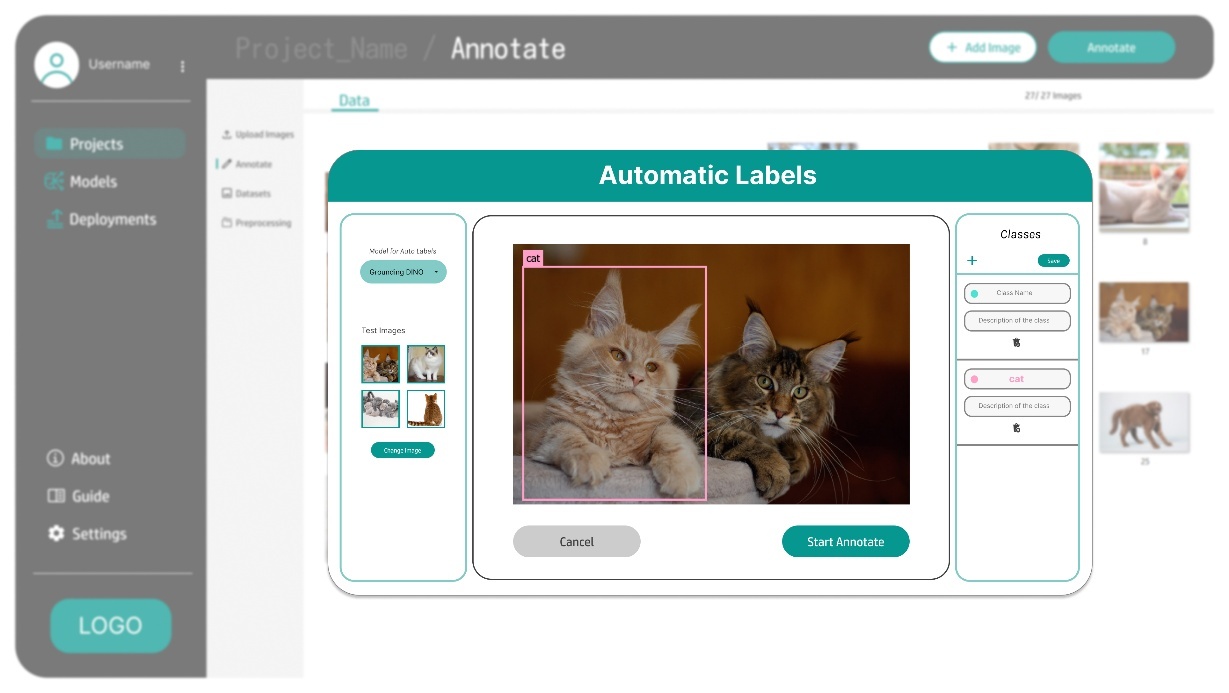
หน้าการตีกรอบวัตถุในรูปภาพแบบ Manual จะมีเครื่องมือที่แถบด้านซ้ายคือเครื่องมือจับเคลื่อนย้ายภาพ (Drag Tool) และเครื่องมือตีกรอบวัตถุในรูปภาพ (Bounding Box Tool) โดยสามารถย้อนกลับ (Undo) ทำซ้ำ (Redo) และรีเซ็ตการตีกรอบได้ และที่แถบด้านขวาเป็นการเพิ่มคลาสโดยกดปุ่มบวก กรอกชื่อคลาส และเลือกสีของคลาสได้ที่จุดสีข้างหลัง สามารถติดแท็กบอกข้อมูลของภาพได้ และในขณะที่ตีกรอบวัตถุในภาพ ผู้ใช้สามารถหยุดทำได้โดยกดปุ่มย้อนกลับ และสามารถกลับมาทำการตีกรอบรูปภาพต่อได้ ดังรูปที่ 3.15



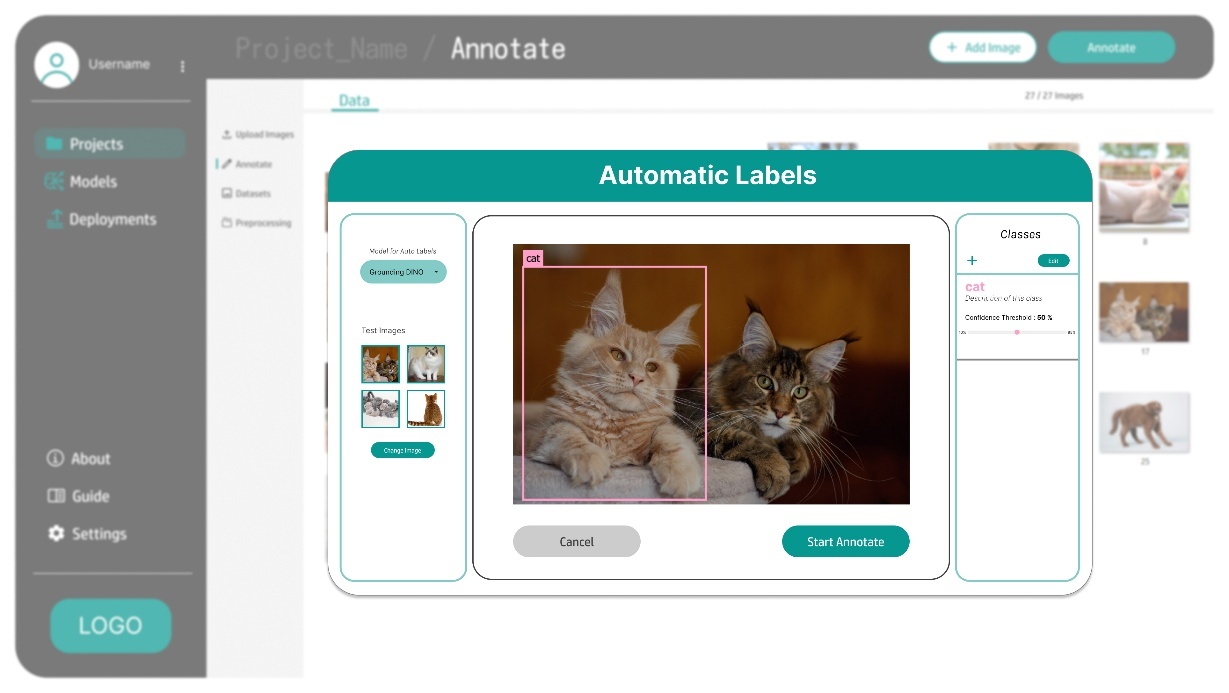
**รูปที่ 3.15** การออกแบบหน้าการตีกรอบวัตถุในภาพแบบ Manual

8) การตีกรอบวัตถุในรูปภาพแบบ Auto

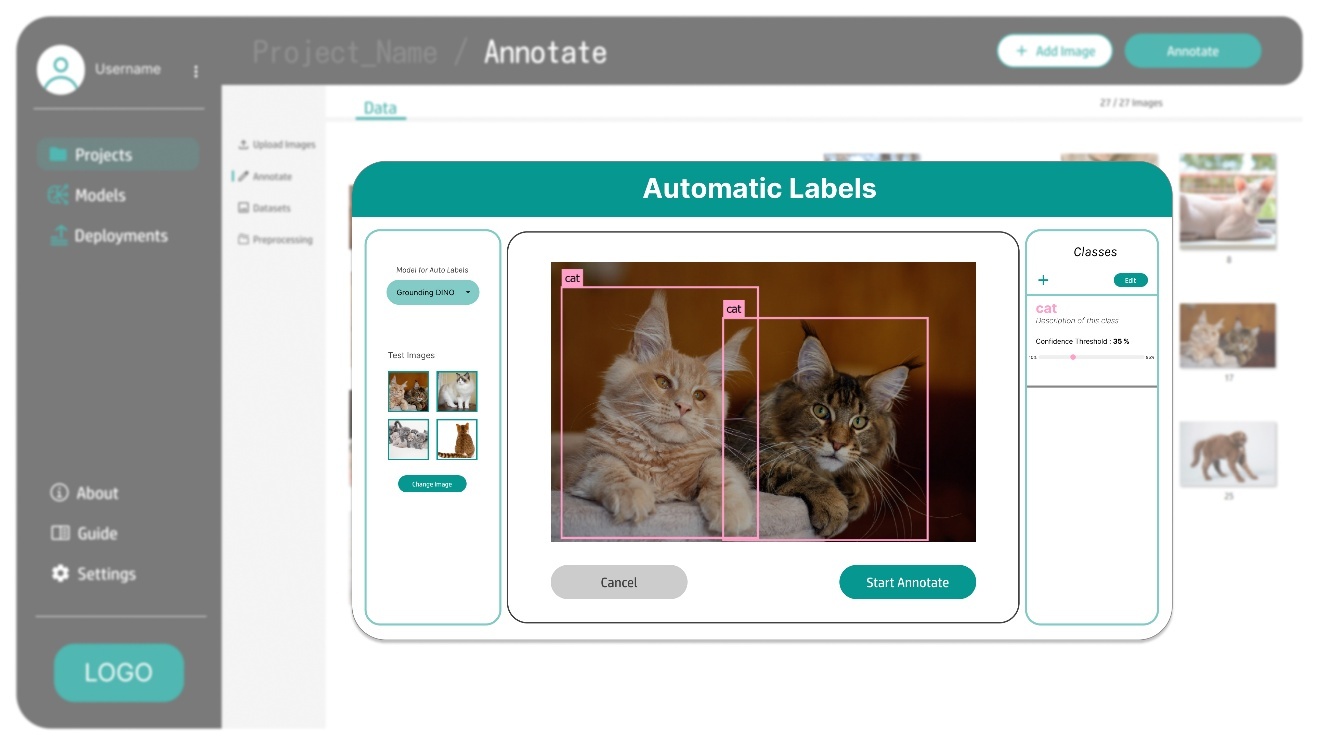
หน้าการตีกรอบวัตถุในรูปภาพแบบ Auto จะแสดงในรูปแบบของ Pop-Up โดยใช้โมเดล Grounding DINO เป็นตัวทำ Automatic Annotate ซึ่งจะแสดงรูปภาพบางส่วนในการทดสอบโมเดล สามารถเปลี่ยนเป็นรูปภาพอื่น ๆ ที่อยู่ภายในโปรเจคได้ สามารถเพิ่มคลาสโดยการกดปุ่มบวก กรอกชื่อคลาส คำอธิบายคลาส เมื่อกดปุ่ม Save จะมีค่าความมั่นใจของโมเดลให้ผู้ใช้ได้ลองปรับจนเจอวัตถุในรูปภาพ และผู้ใช้สามารถแก้ไขชื่อคลาสและคำอธิบายคลาสได้โดยกดปุ่ม Edit หากผู้ใช้พอใจในผลลัพธ์แล้วสามารถเริ่มการตีกรอบวัตถุในรูปภาพได้โดยการกดปุ่ม Start Annotate ดังรูปที่ 3.16 – 3.18 ตามลำดับ



**รูปที่ 3.16** การออกแบบหน้าการตีกรอบวัตถุในภาพแบบ Auto ในส่วนของการเพิ่มคลาส



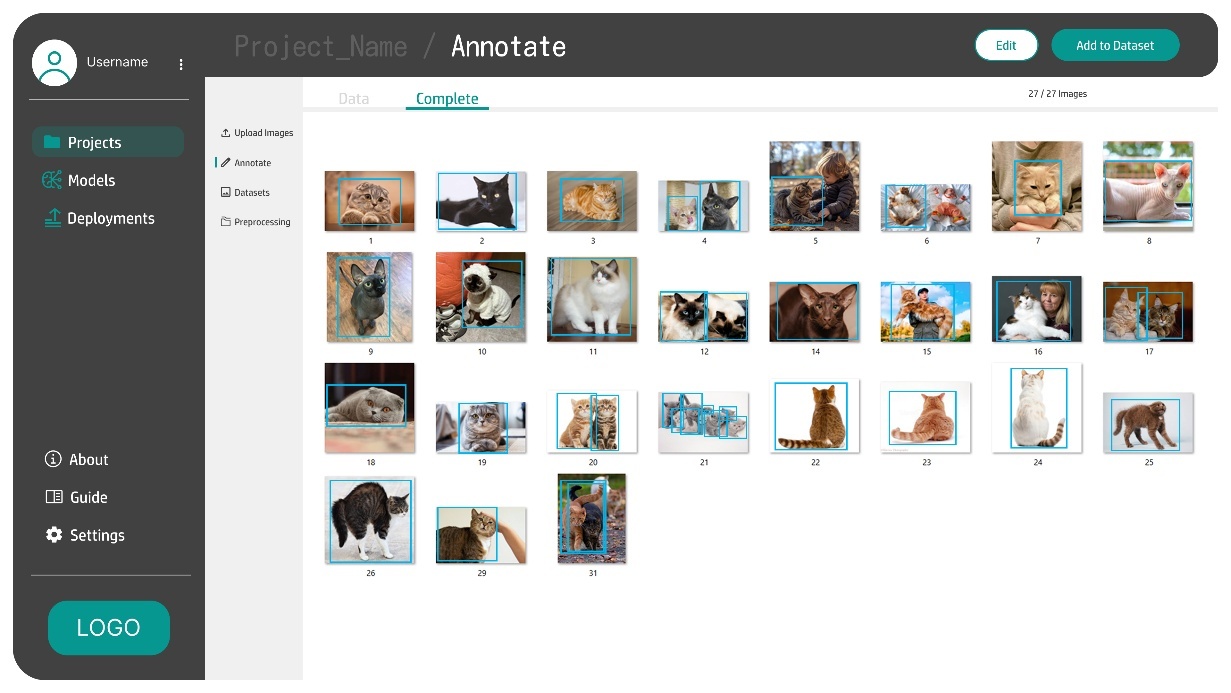
**รูปที่ 3.17** การออกแบบหน้าการตีกรอบวัตถุในภาพแบบ Auto ในส่วนของการ Save คลาส



**รูปที่ 3.18** การออกแบบหน้าการตีกรอบวัตถุในรูปภาพแบบ Auto การปรับค่าความมั่นใจของโมเดล

9) หน้าการแสดงผลรูปภาพที่ผ่านการตีกรอบวัตถุ

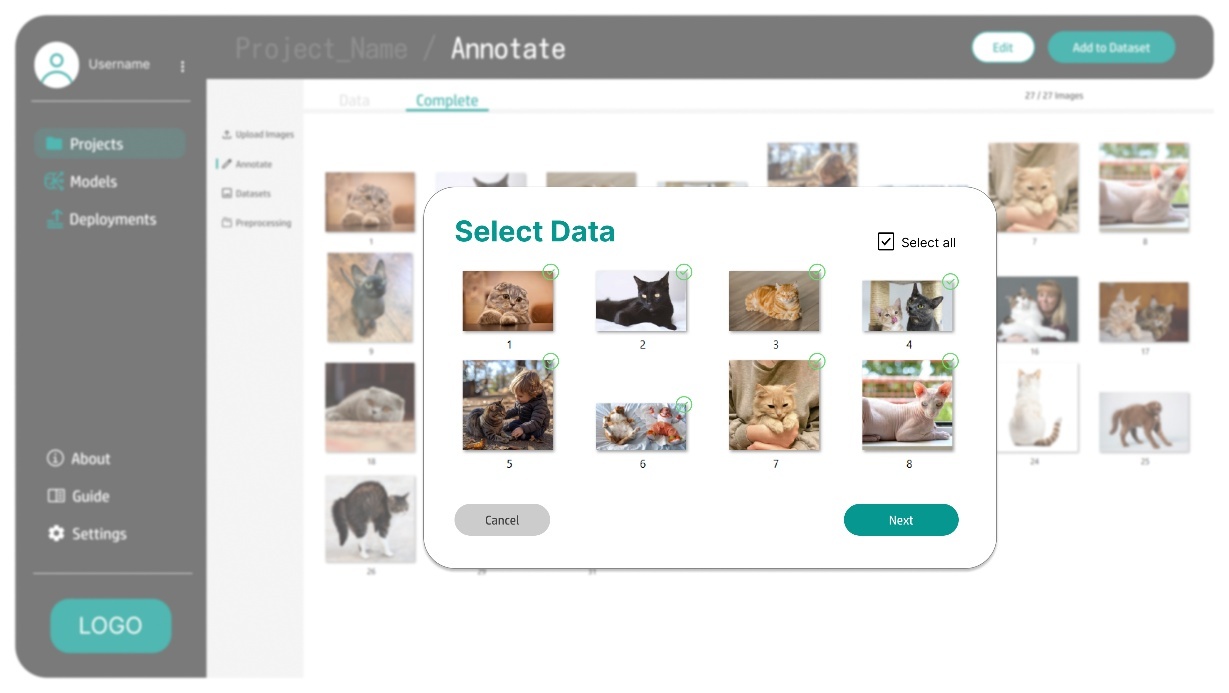
หน้าการแสดงผลรูปภาพที่ผ่านการตีกรอบวัตถุ ผู้ใช้สามารถแก้ไขการตีกรอบใหม่ได้โดยกดปุ่ม Edit และเพิ่มรูปภาพเข้า Dataset ได้โดยกดปุ่ม Add to Dataset ดังรูปที่ 3.19



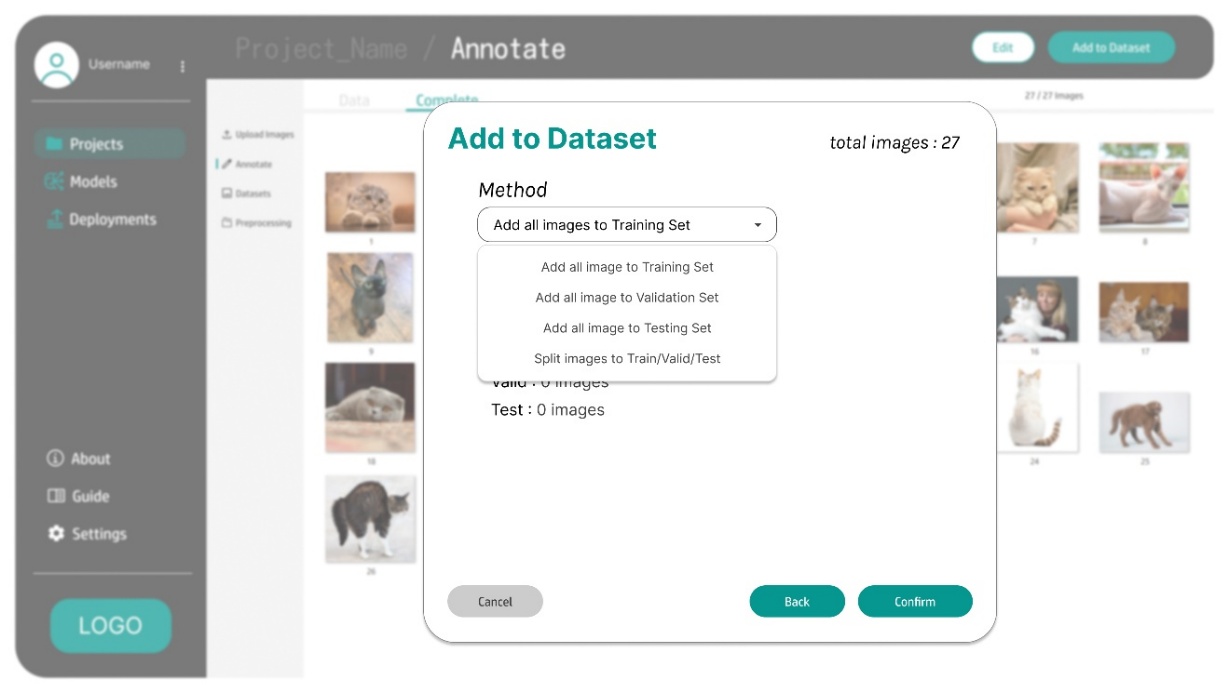
**รูปที่ 3.19** การออกแบบหน้าการแสดงผลรูปภาพที่ผ่านการตีกรอบวัตถุ

10) หน้าการเพิ่มรูปเข้า Dataset

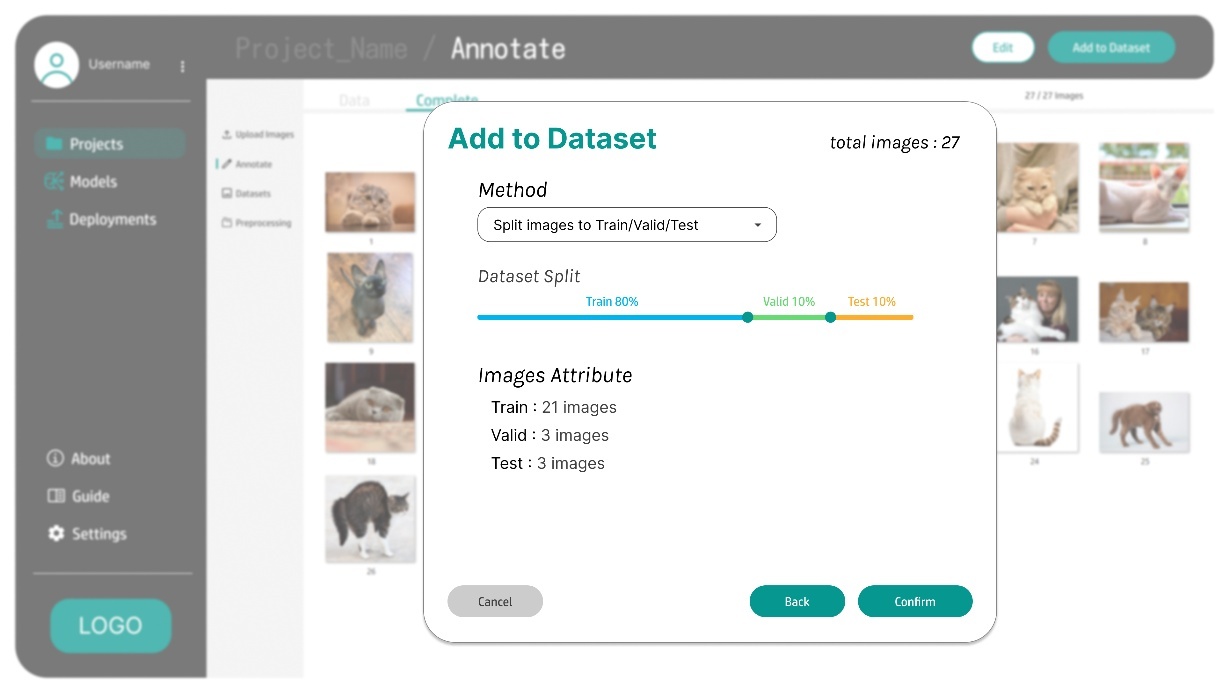
หน้าการเพิ่มรูปเข้า Dataset จะแสดงเป็น Pop-Up โดยเริ่มแรกให้ผู้ใช้เลือกรูปภาพที่ต้องการเพิ่มหรือสามารถทำเครื่องหมายถูกที่ Select all เพื่อเลือกรูปภาพทั้งหมด ถัดไปเป็นการเพิ่มรูปภาพลงเซ็ตที่เลือก หากผู้ใช้เลือกการแบ่งเซ็ตรูปภาพ ผู้ใช้จะสามารถเลือกเปอร์เซ็นต์ของแต่ละเซ็ต กดปุ่มเพื่อยืนยัน ดังรูปที่ 3.20 – 3.22 ตามลำดับ



**รูปที่ 3.20** การออกแบบหน้าเลือกรูปภาพ



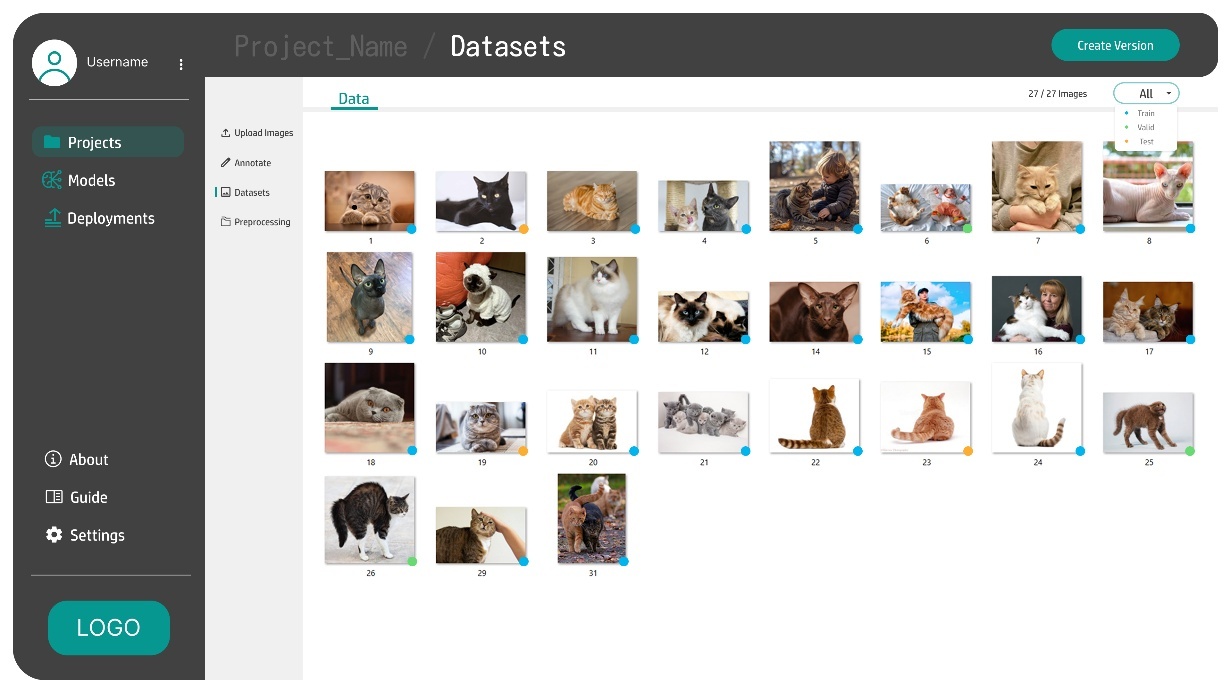
**รูปที่ 3.21** การออกแบบหน้าเพิ่มรูปภาพลงเซ็ตที่เลือก



**รูปที่ 3.22** การออกแบบหน้าแบ่งเซ็ตของรูปภาพ

11) หน้าการแสดงผลข้อมูลรูปภาพใน Dataset

หน้าการแสดงผลข้อมูลรูปภาพใน Dataset จะแสดงรูปภาพในแต่ละเซ็ตที่ถูกแทนด้วยสี ซึ่งสีฟ้าคือ Training Set สีเขียวคือ Validation Set และสีส้มคือ Testing Set และผู้ใช้สามารถเพิ่มเวอร์ชันของ Dataset ได้โดยการกดปุ่ม Create Version ดังรูปที่ 3.23

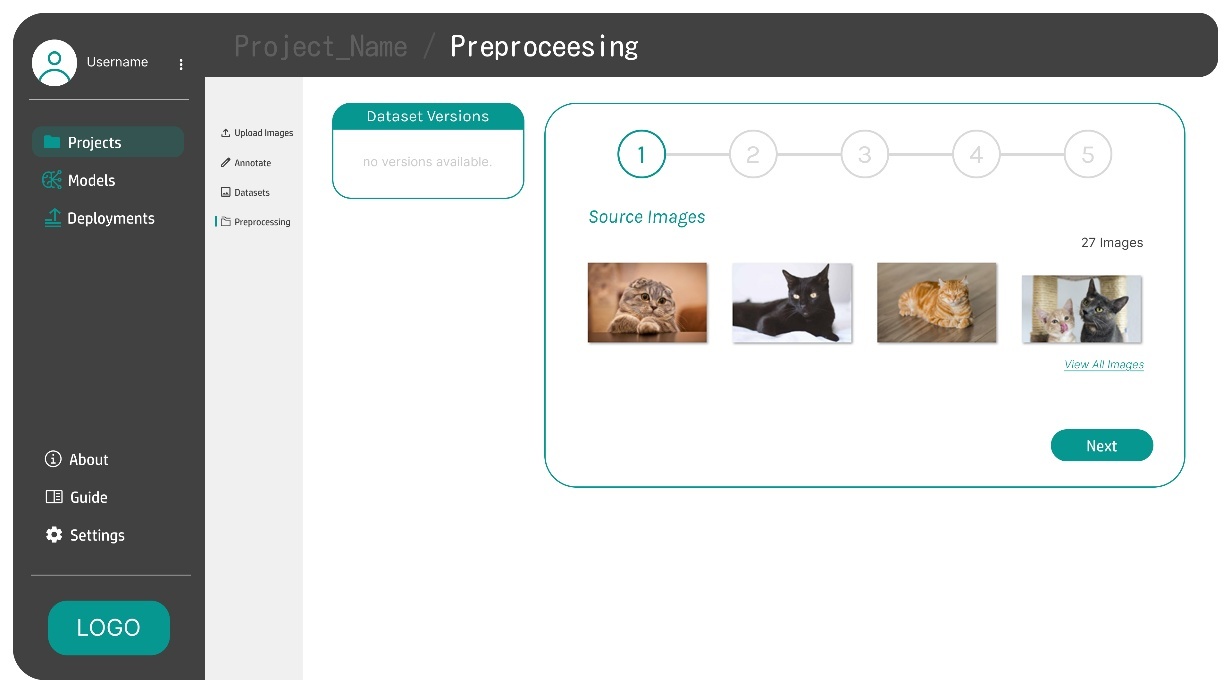


**รูปที่ 3.23** การออกแบบหน้าการแสดงผลข้อมูลรูปภาพใน Dataset

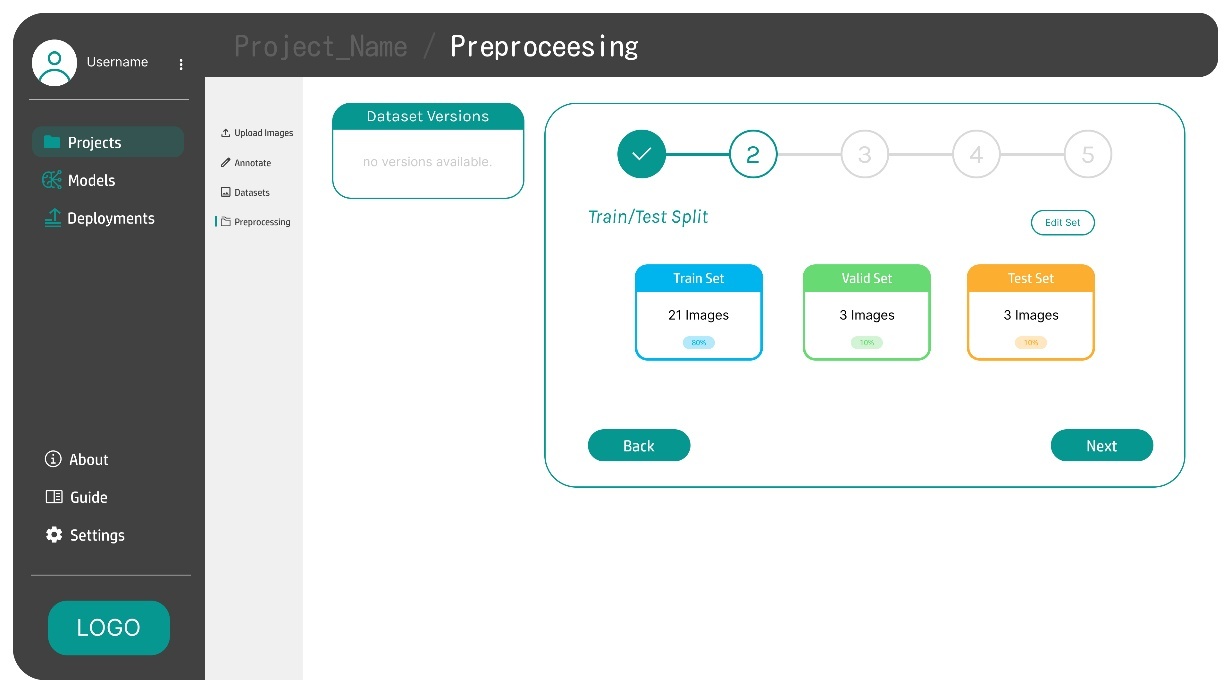
12) หน้าการเพิ่มเวอร์ชันของ Dataset

ในหน้าการเพิ่มเวอร์ชันของ Dataset จะทำงานไปเป็นขั้นตอน ดังรูปที่ 3.24 – 3.28 ตามลำดับ

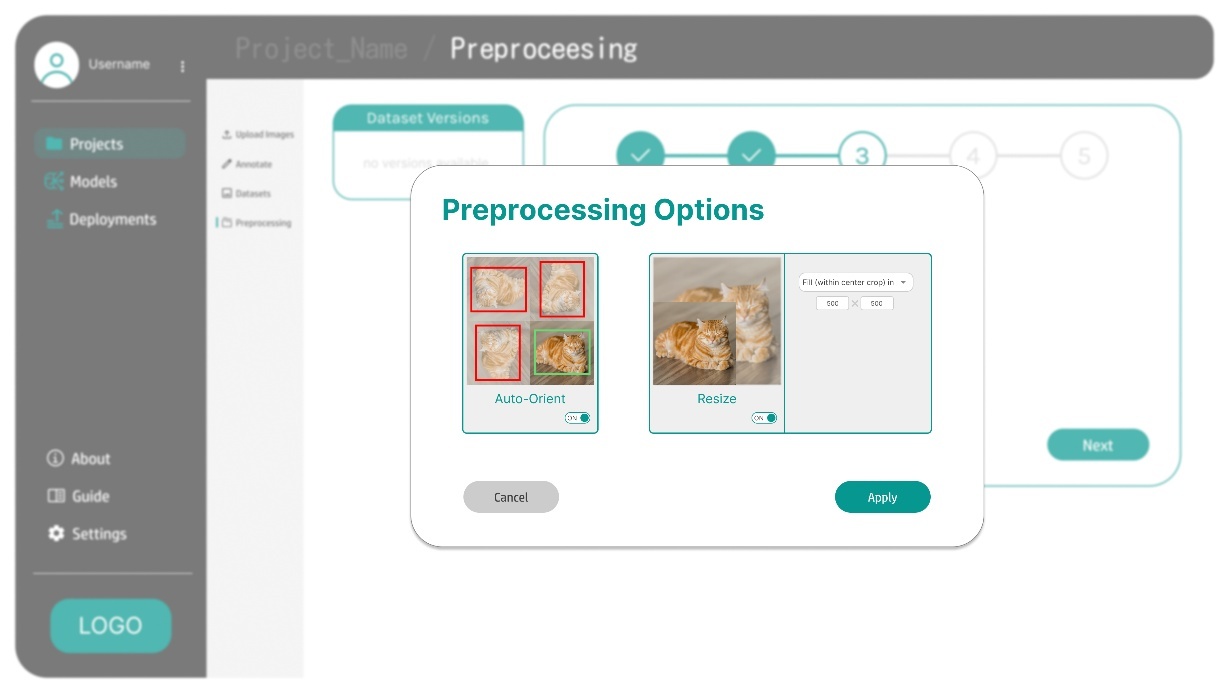
* ขั้นตอนแรกให้ผู้ใช้ตรวจสอบข้อมูลรูปภาพใน Dataset
* ขั้นตอนที่สองตรวจสอบเซ็ตของข้อมูลรูปภาพและสามารถแบ่งเซ็ตของข้อมูลใหม่ได้
* ขั้นตอนที่สามเป็นการเตรียมชุดข้อมูล (Preprocessing)
* ขั้นตอนที่สี่เป็นการปรับแต่งข้อมูล (Augmentation)
* ขั้นตอนสุดท้ายเป็นการกรอกชื่อ Dataset และเพิ่มจำนวนข้อมูลใน Dataset



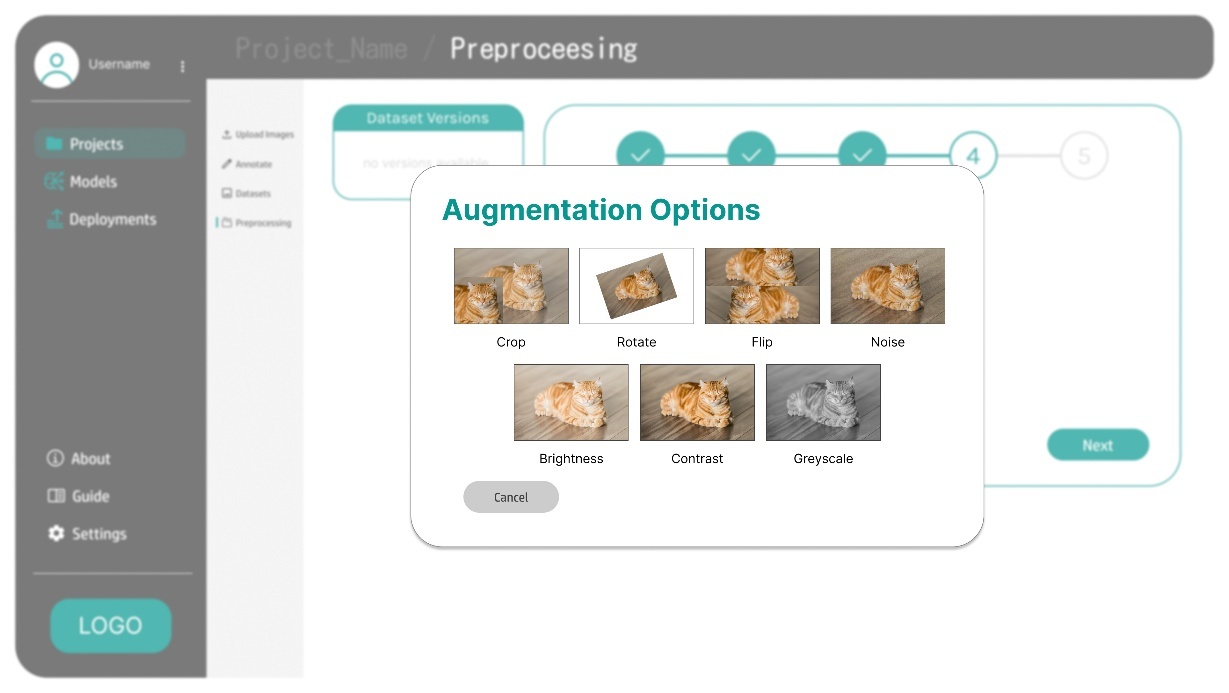
**รูปที่ 3.24** การออกแบบหน้าตรวจสอบข้อมูลรูปภาพใน Dataset



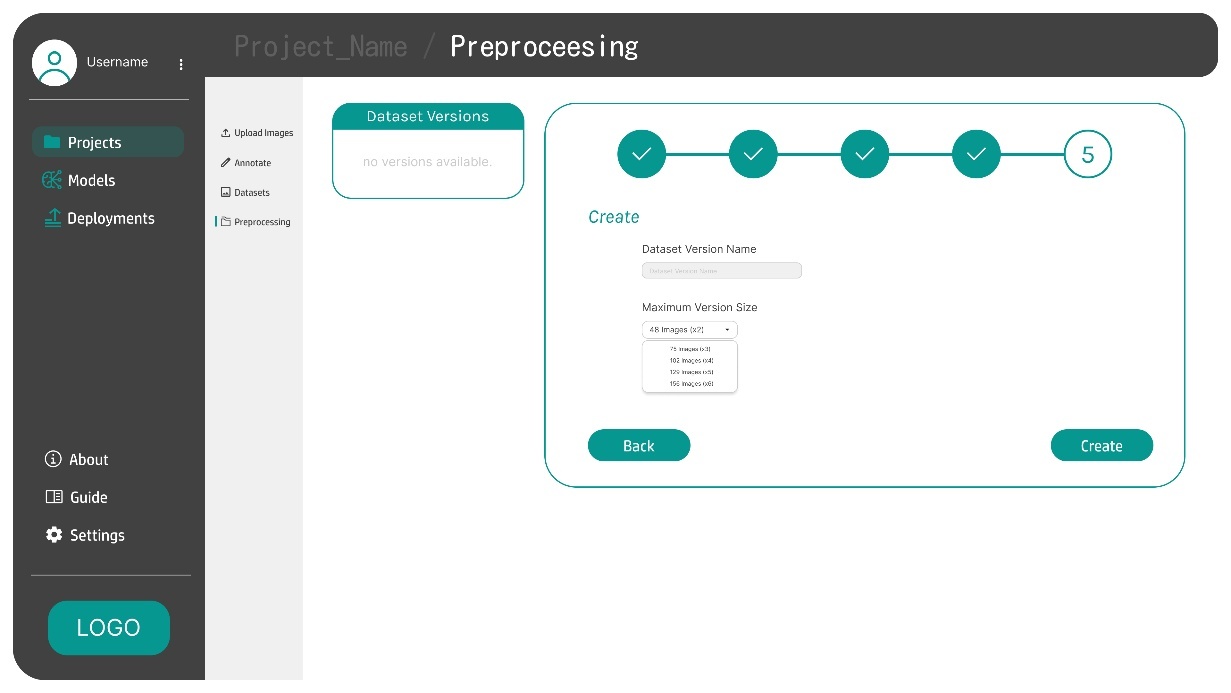
**รูปที่ 3.25** การออกแบบหน้าตรวจสอบเซ็ตของข้อมูลรูปภาพ



**รูปที่ 3.26** การออกแบบหน้าการเตรียมชุดข้อมูล (Preprocessing)



**รูปที่ 3.27** การออกแบบหน้าการปรับแต่งชุดข้อมูล (Augmentation)



**รูปที่ 3.28** การออกแบบหน้าการเพิ่มจำนวนข้อมูลใน Dataset

13) หน้าการแสดงผลการเพิ่มเวอร์ชันของ Dataset

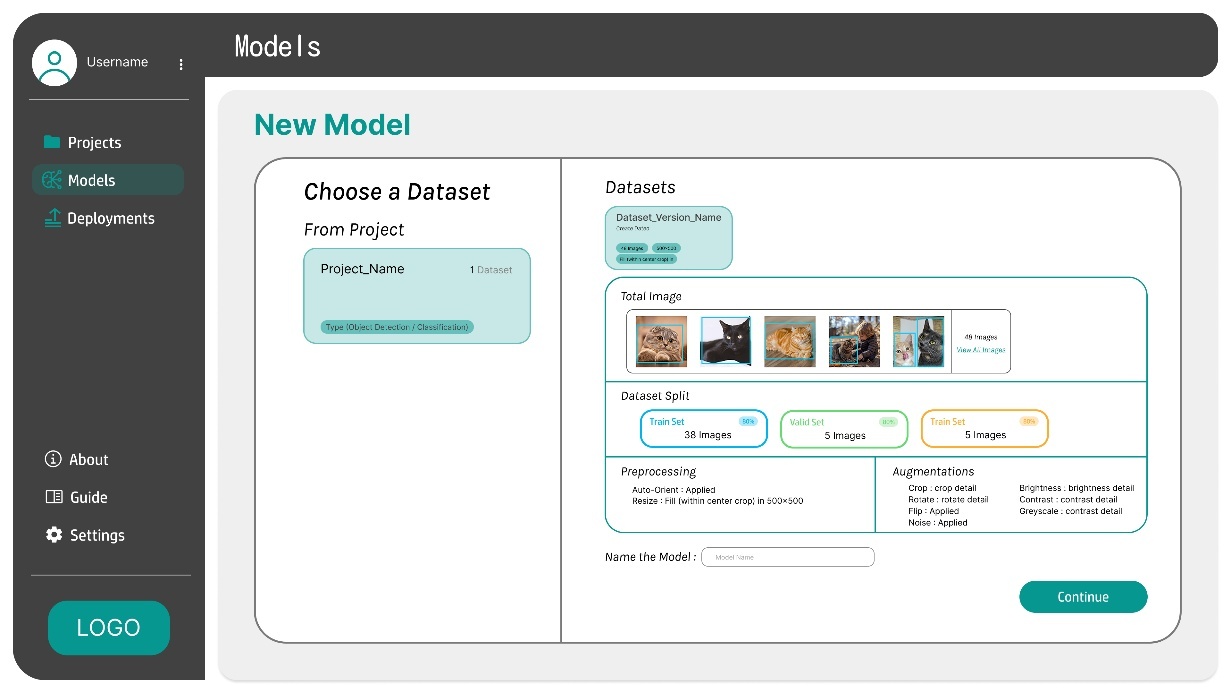
หน้าการแสดงผลการเพิ่มเวอร์ชันของ Dataset จะแสดงผลรายละเอียดข้อมูลต่าง ๆ ที่ได้ทำไปในแต่ละขั้นตอนของการเพิ่มเวอร์ชัน Dataset และผู้ใช้สามารถดาวน์โหลด Dataset ได้โดยการกดปุ่ม Download Dataset ดังรูปที่ 3.29



**รูปที่ 3.29** การออกแบบหน้าการแสดงผลของการเพิ่มเวอร์ชันของ Dataset

14) หน้าการเลือก Dataset เพื่อฝึกฝนโมเดล

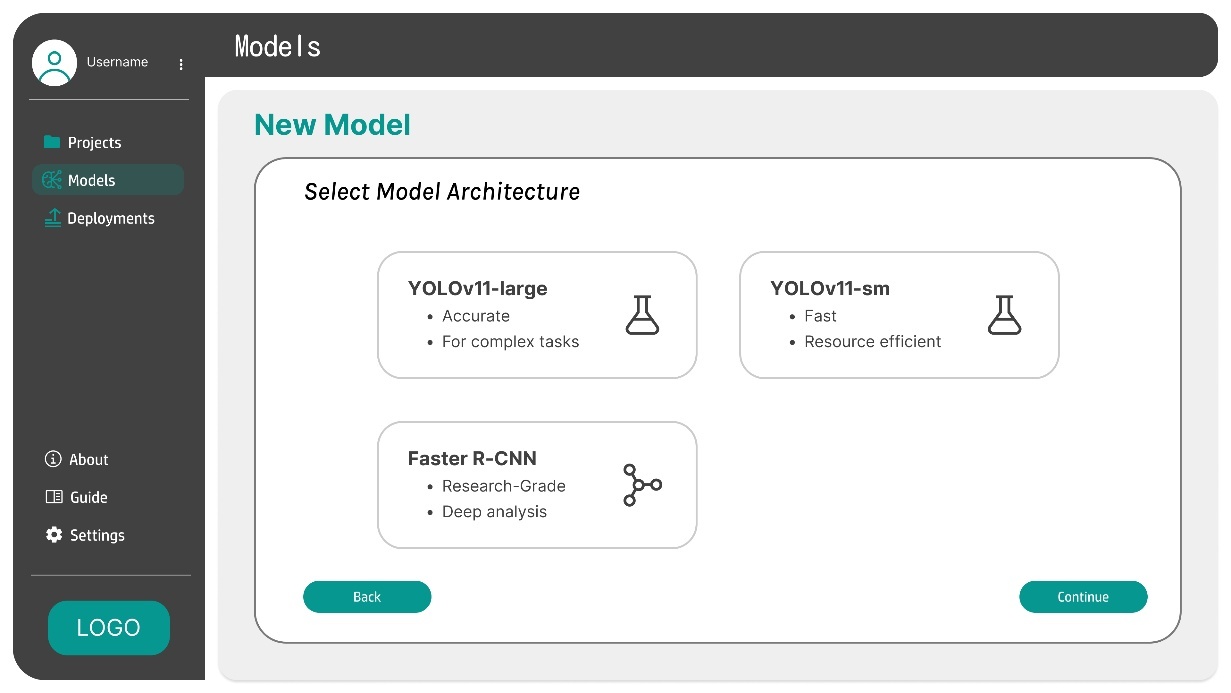
โดยผู้ใช้ต้องทำการเลือกโปรเจคก่อน จึงจะสามารถเลือก Dataset เวอร์ชันต่าง ๆ ที่อยู่ภายในโปรเจคนั้นได้ ผู้ใช้สามารถดูรายละเอียดและตรวจสอบข้อมูลภายใน Dataset นั้นได้ จากนั้นกรอกชื่อโมเดล และกดปุ่มเพื่อทำขั้นตอนถัดไป ดังรูปที่ 3.30



**รูปที่ 3.30** การออกแบบหน้าการเลือก Dataset เพื่อฝึกฝนโมเดล

15) หน้าการเลือกโครงสร้างเพื่อฝึกฝนโมเดล

โดยผู้ใช้สามารถเลือกโครงสร้างของโมเดลที่ต้องการและเหมาะสมกับงาน เพื่อนำไปฝึกฝนโมเดลได้ จากนั้นกดปุ่มเพื่อทำขั้นตอนถัดไป ดังรูปที่ 3.31



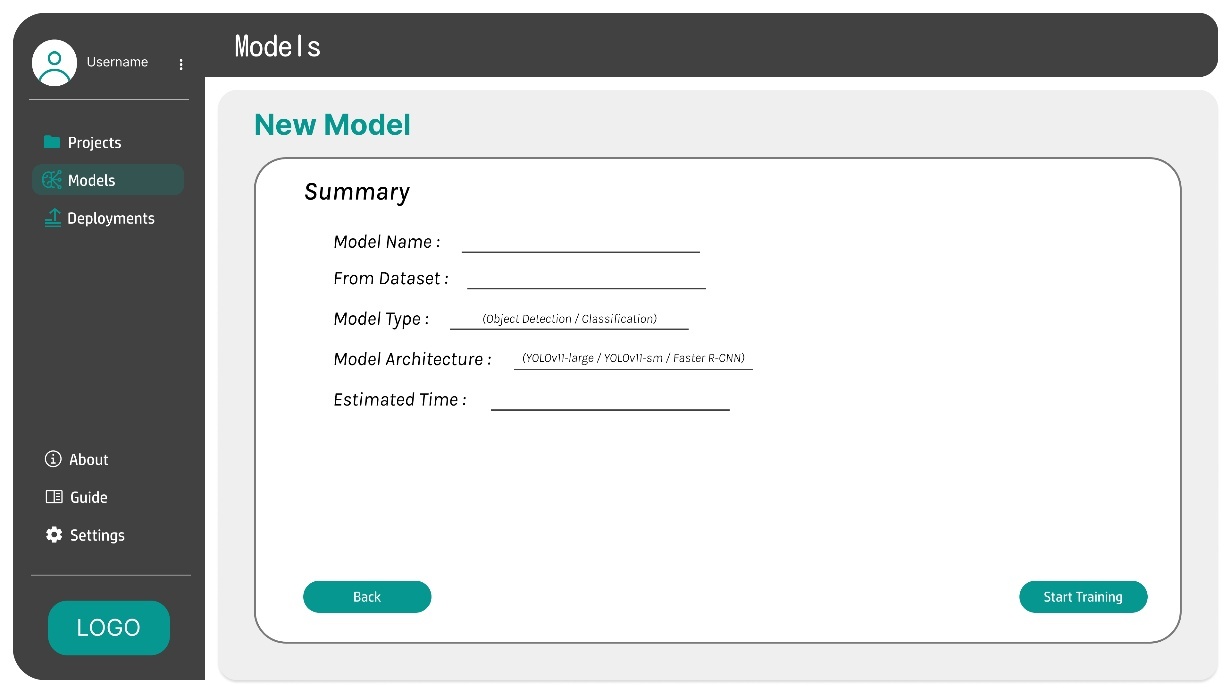
**รูปที่ 3.31** การออกแบบหน้าการเลือกโครงสร้างเพื่อฝึกฝนโมเดล

16) หน้าการสรุปรายละเอียดโมเดลก่อนนำไปฝึกฝน

หน้าการสรุปรายละเอียดโมเดลก่อนนำไปฝึกฝน โดยมีรายละเอียดดังนี้

* ชื่อโมเดล
* Dataset ที่เลือก
* ประเภทของโมเดล
* โครงสร้างของโมเดล
* เวลาที่ใช้ฝึกฝนโดยประมาณ

ผู้ใช้สามารถกดปุ่มกลับไปแก้ไขได้ และกดปุ่ม Start Training เพื่อเริ่มฝึกฝนโมเดล ดังรูปที่ 3.32



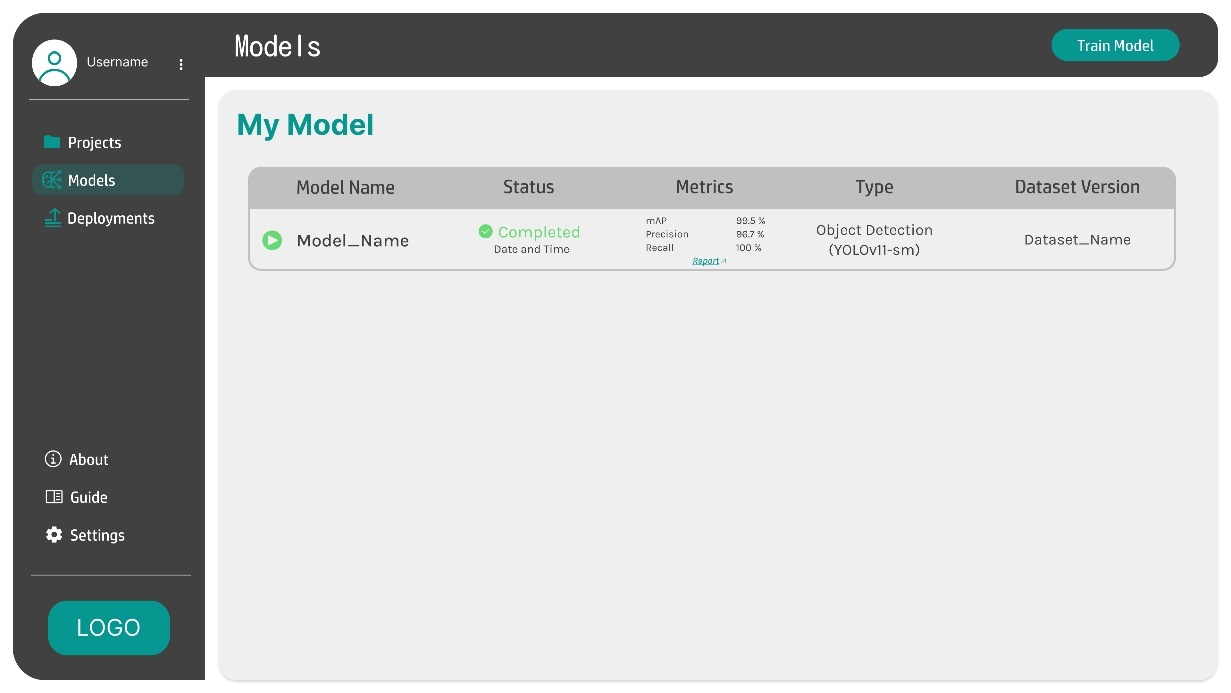
**รูปที่ 3.32** การออกแบบหน้าการสรุปรายละเอียดโมเดลก่อนนำไปฝึกฝน

17) หน้าแสดงผลการเทรนโมเดล

หน้าแสดงผลการเทรนโมเดลจะแสดงรายละเอียดดังนี้

* ชื่อโมเดล
* สถานะการฝึกฝนของโมเดล
* ค่าความแม่นยำของโมเดล
* ประเภทของโมเดล พร้อมโครงสร้างของโมเดล
* ชื่อ Dataset

นอกจากนี้ผู้ใช้ยังสามารถทดสอบโมเดลได้โดยกดปุ่ม play และสามารถรายละเอียดการวิเคราะห์ความแม่นยำของโมเดลได้โดยการกด Report ดังรูปที่ 3.33

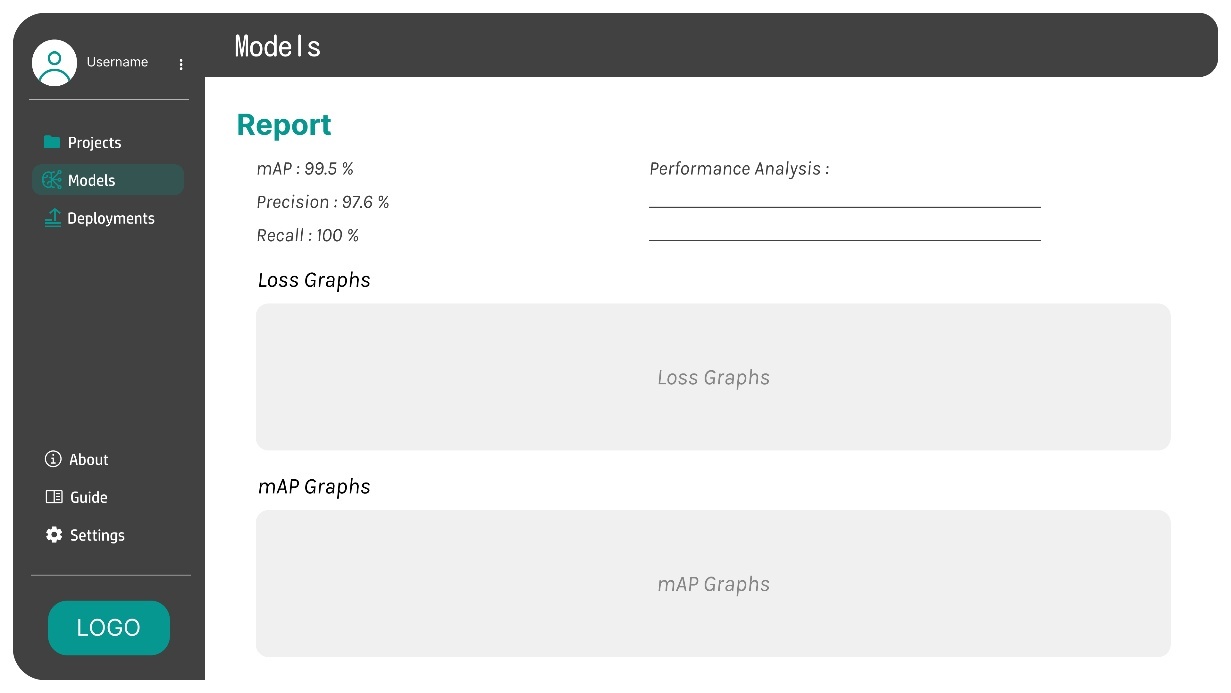


**รูปที่ 3.33** การออกแบบหน้าแสดงผลการเทรนโมเดล

18) หน้ารายละเอียดการวิเคราะห์ความแม่นยำของโมเดล

หน้ารายละเอียดการวิเคราะห์ความแม่นยำของโมเดล จะบอกสรุปการวิเคราะห์ประสิทธิภาพของโมเดล โดยมีรายละเอียดดังนี้ ดังรูปที่ 3.34

* ค่าความแม่นยำโมเดล
* วิเคราะห์ประสิทธิภาพของโมเดล
* Loss Graphs
* mAP Graphs

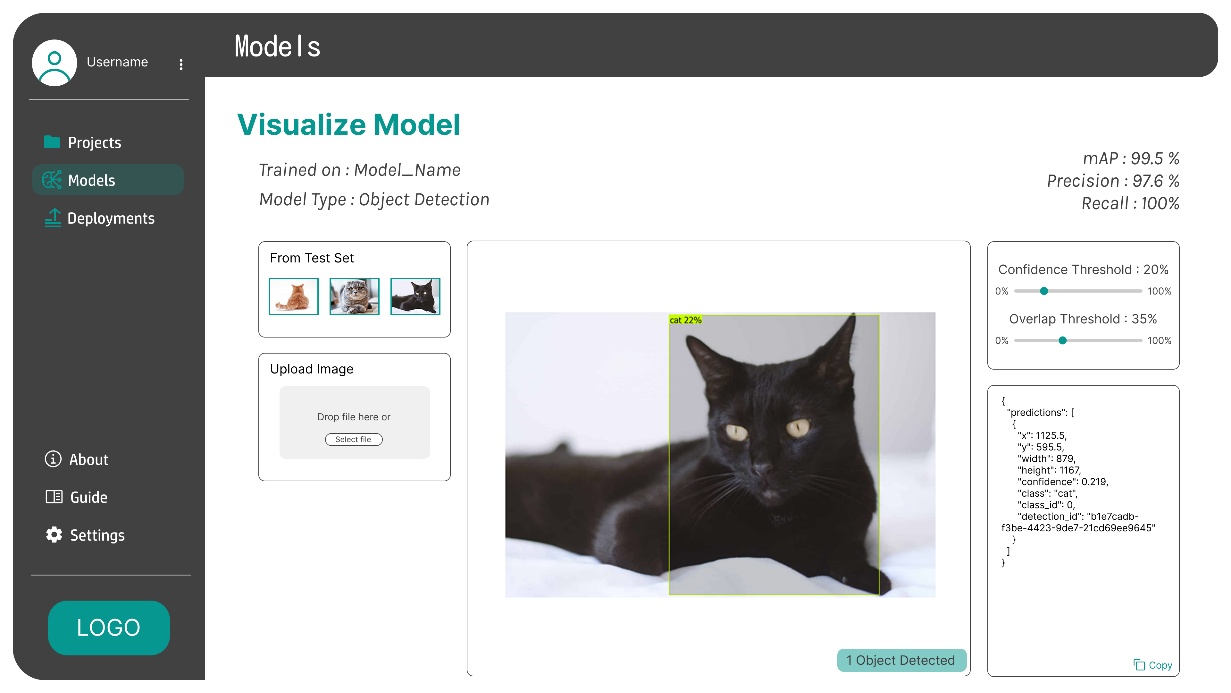


A screenshot of a computer

AI-generated content may be incorrect.**รูปที่ 3.34** การออกแบบหน้ารายละเอียดการวิเคราะห์ความแม่นยำของโมเดล

19) หน้าการทดสอบโมเดล

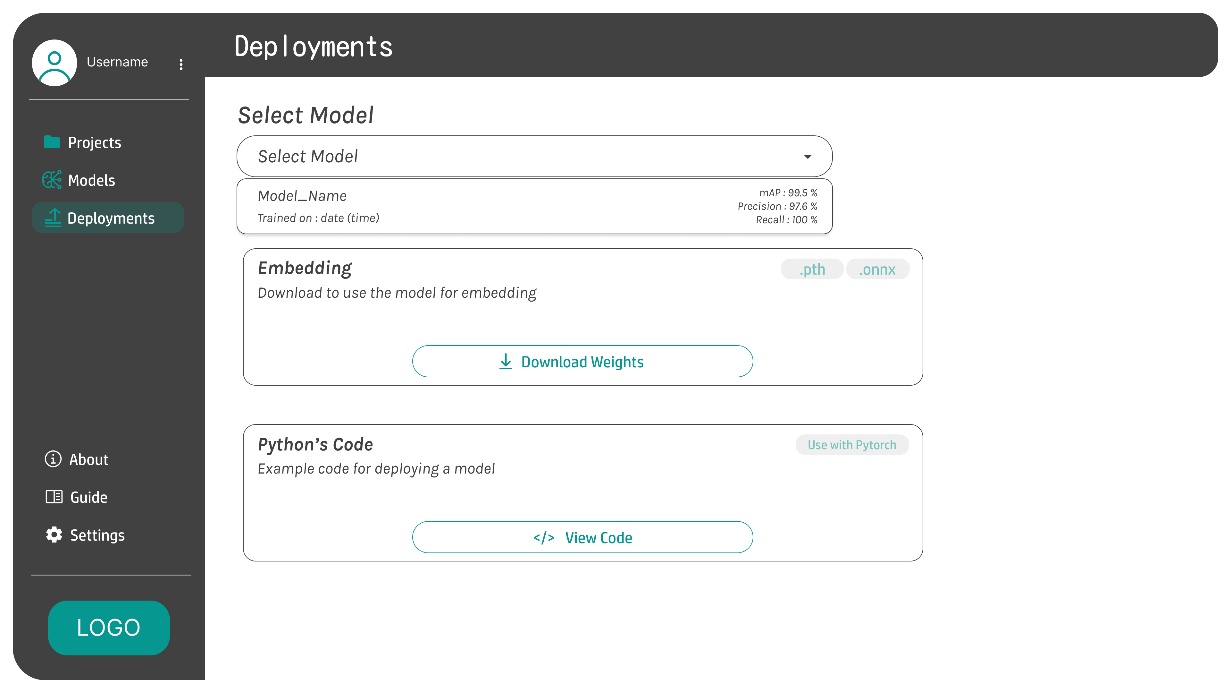
หน้าการทดสอบโมเดล ผู้ใช้สามารถนำรูปจาก Testing Set มาทดสอบโมเดลได้ หรือจะเลือกอัปโหลดรูปจากภายนอกมาทดสอบได้ และผู้ใช้สามารถปรับค่าความมั่นใจของโมเดลเพื่อสามารถดูได้ว่าโมเดลมีค่าความมั่นใจในการเจอวัตถุที่เท่าไหร่ และจะได้ตำแหน่งของวัตถุที่ตรวจจับได้ภายในภาพ ดังรูปที่ 3.35



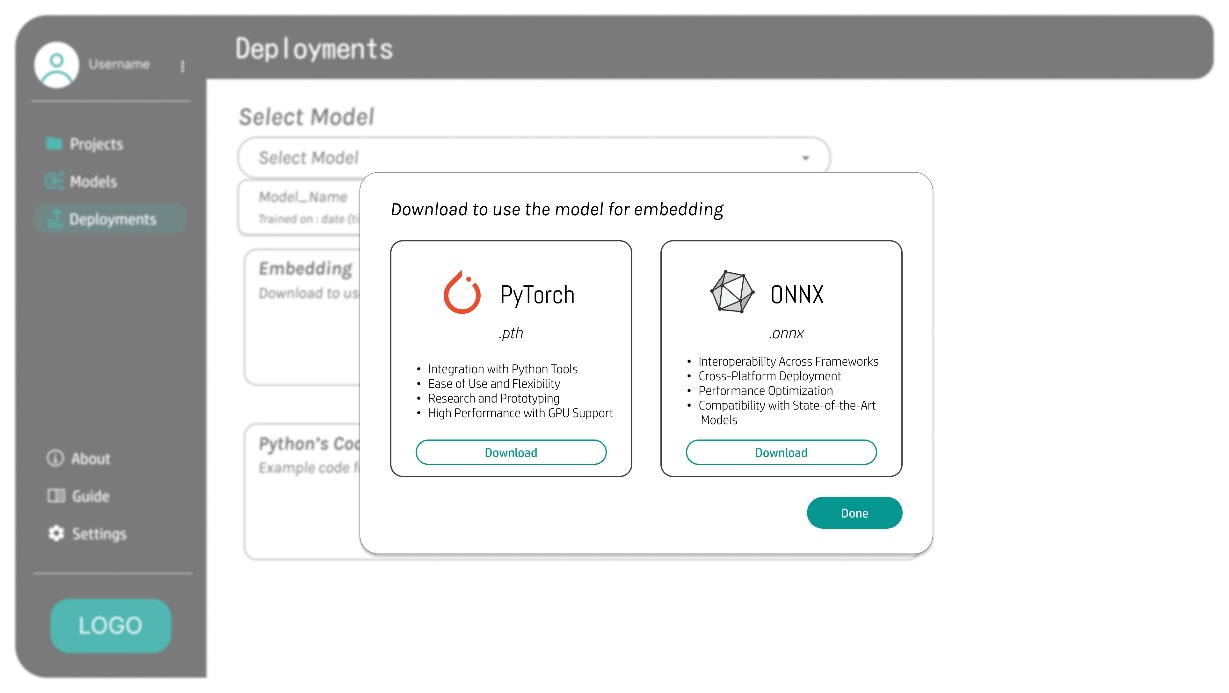
**รูปที่ 3.35** การออกแบบหน้าการทดสอบโมเดล

20) หน้าการนำโมเดลไปใช้

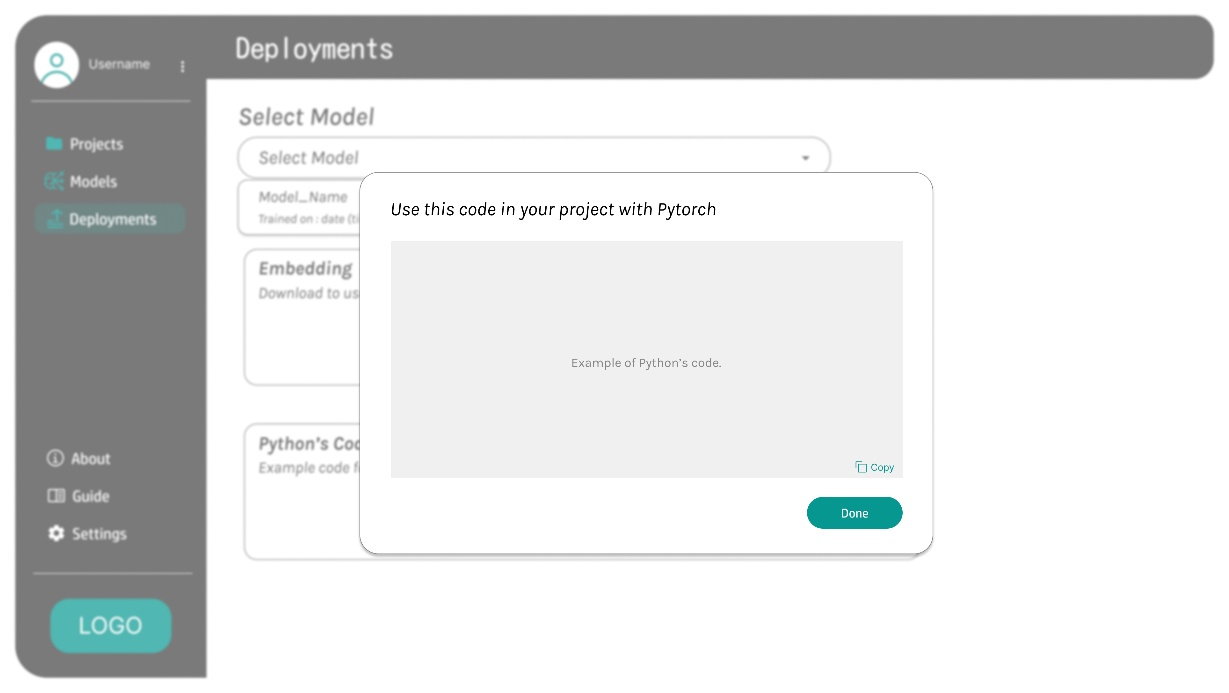
ผู้ใช้สามารถเลือกโมเดลที่ต้องการเพื่อดาวน์โหลดออกไปใช้งานภายนอกในงานต่าง ๆ ได้ และผู้ใช้สามารถดาวน์โหลดโค้ดไพทอนตัวอย่างของการนำไปใช้งานได้ ดังรูปที่ 3.36 – 3.38 ตามลำดับ



**รูปที่ 3.36** การออกแบบหน้าการนำโมเดลไปใช้



**รูปที่ 3.37** การออกแบบหน้าการดาวน์โหลดโมเดล



**รูปที่ 3.38** การออกแบบหน้าโค้ดไพทอนตัวอย่าง