**Тема на проекта:**

***Система за разпознаване и съхранение на обекти в изображения***

**Цел:**

Да се разработи уеб-базирана платформа, където потребителите могат да качват изображения, за да идентифицират обекти и да проверяват дали тези обекти вече са били разпознати. Системата ще използва база данни за съхранение на информация за обектите и метаданни на изображенията.

**Преглед на технологичния стек:**

**Back-end Frameworks**: Django and Flask (за API и взаимодействие с базата данни)

**Front-end Framework**: Streamlit (за потребителски интерфейс и качване на изображения)

**Database**: SQL (например PostgreSQL или SQLite) или NoSQL (например MongoDB) за съхранение на информация за обектите и метаданни на изображенията

**Image Recognition Library**: OpenCV или TensorFlow (за откриване и разпознаване на обекти)

**Основни компоненти:**

1. **Интерфейс за качване на изображения (Streamlit)**:

* Създаване на лесен потребителски интерфейс с Streamlit за качване на изображения.
* Изпращане на качените изображения към бекенда за обработка.

2. **Модул за разпознаване на обекти**:

* Използване на предварително обучен модел за разпознаване на обекти (например TensorFlow или OpenCV), за да се открият и идентифицират обекти в качените изображения.
* Генериране на уникален идентификатор или клас за всеки разпознат обект.

3. **Структура на базата данни**:

* Проектиране на схема на базата данни с таблици за съхранение на разпознати обекти и изображения.
* Примерни таблици:
  + ***Objects***: Съхранява ID на обекти, имена и метаданни.
  + ***Images***: Съхранява метаданни за изображенията (име на файла, дата на качване) и свързани ID на обекти.
* Използване на SQL или NoSQL в зависимост от изискванията на проекта (SQL за структурирани, релационни данни; NoSQL ако се използва MongoDB).

4. **Back-end (Django и Flask)**:

* Използване на Django за управление на резултатите от разпознаването на обекти и взаимодействие с базата данни.
* Flask може да обслужва API заявки за обработка на качени изображения и стартиране на разпознаване.

5. **Поток на разпознаване на обекти**:

* Потребителят качва изображение чрез интерфейса на Streamlit.
* Изображението се изпраща към API на Flask за обработка и разпознаване на обекти.
* Разпознатите обекти се проверяват в базата данни. Ако обектът е разпознат, се извличат съществуващите данни; ако е нов, информацията се записва.

**6. Front-end (Streamlit)**:

* Показване на резултатите в организиран и удобен за потребителя начин.
* Показване на детайли за всеки разпознат обект, с връзки към предишни изображения, съдържащи същия обект (ако има такива).

### Следващи стъпки:

1. Разработване и тестване на отделните компоненти.
2. Интегриране на компонентите в една работна система.
3. Финализиране и внедряване на системата.