Зміст

[1 Аналіз технологій, інструментів, програмних бібліотек та математичних моделей 1](#_Toc165814780)

[2 Структурна схема реалізації проекту 3](#_Toc165814781)

[3 Інтерфейс роботи програми 7](#_Toc165814782)

## Аналіз технологій, інструментів, програмних бібліотек та математичних моделей

##### **Мова** програмування

Мова програмування Python вибрана для цього проекту з кількох причин. По-перше, Python має простий і лаконічний синтаксис, що робить розробку швидшою та ефективнішою, особливо на етапі прототипування і розвитку програми. Вона також має велику кількість бібліотек і фреймворків, що спрощують роботу з обробкою зображень (для модуля розпізнавання обличчя), криптографією (для шифрування файлів), та роботою з базами даних.

Другою причиною є широке сприйняття Python у сфері машинного навчання та обробки зображень. Використання Python для модуля розпізнавання обличчя дозволяє легко інтегрувати сторонні бібліотеки та фреймворки, такі як OpenCV або TensorFlow, що полегшує реалізацію функціональності ідентифікації особистості.

Нарешті, Python є більш гнучкою мовою, що сприяє швидкому розвитку і змінам у проекті, що може бути важливим у випадку постійних оновлень та вдосконалень в системі без значного збоку часу і зусиль. У порівнянні з C++ або Java, Python зазвичай потребує менше коду для досягнення тих самих результатів, що полегшує процес розробки та збільшує продуктивність команди розробників.

##### Бібліотеки та інструменти

* OpenCV: Бібліотека OpenCV (Open Source Computer Vision Library) надає широкий набір інструментів для роботи з зображеннями та відео, що включає обробку зображень, відстеження облич та багато іншого.
* face\_recognition: Ця бібліотека реалізує простий інтерфейс для обробки зображень та визначення облич людей на них. Вона базується на бібліотеці dlib, яка використовує машинне навчання для розпізнавання облич.
* SQLAlchemy: Це потужний ORM (Object-Relational Mapping) для роботи з базами даних у Python. Ви можете використовувати SQLAlchemy для спрощення взаємодії з базою даних PostgreSQL.»;

##### Шифрування

- cryptography: Це бібліотека для криптографічних операцій у Python. Вона надає реалізації різних алгоритмів шифрування, включаючи AES, який може бути використаний для шифрування файлів у вашому проекті.

##### Графічний інтерфейс

- PyQt6: PyQt є набором Python-зв'язків для Qt, який надає можливості для розробки крос-платформених графічних інтерфейсів. Ви можете використовувати PyQt6 для створення зручного інтерфейсу користувача для вашого проекту.

* 1. **Сервер та база даних**

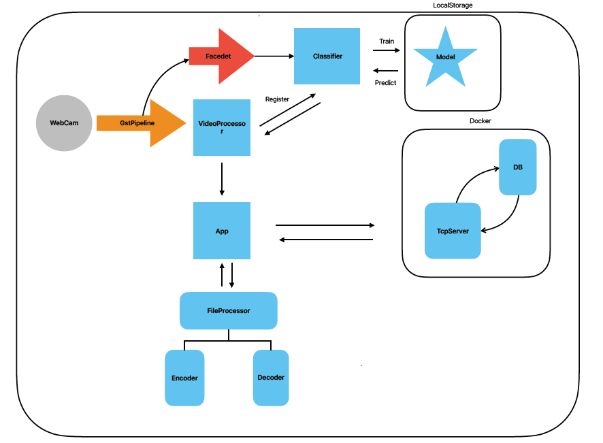
- Docker Compose: Використання Docker Compose для контейнеризації вашого додатку разом із сервером та базою даних дозволить легко розгорнути та керувати середовищем вашого проекту.

- PostgreSQL: Це потужна об'єктно-реляційна система керування базами даних, яка забезпечить надійне зберігання та управління даними вашого проекту.

## Структурна схема реалізації проекту

На данному етапі розробки програми розгляається дві структури проекту. Одна з використанням стандартного функціоналу OpevCV для заходлення відео та окремого потоку для детекторів. Або реалізація за допомогою сучасних методів буферизації та конвеєризації відео потоку за допомогою GStreamer Pipeline, що дозволить роздвоювати відео потік без втрати частоти зображень в секунду.

##### Розробка структурної схеми системи з використанням GStreamer Pipeline



*Рис. 2.1.1 Структурної схема системи з використанням GStreamerPipeline*.

На стурктурні схемі зображена схематична реалізація роботи проекту. Основиними блоками у цій схемі є App, VideoPocessor, FileProcessor та Docker що відповідають самій програмі та серверу для збереження інформації.

Блок App:

Блок App являє собою візуальний застосунок для використання функціоналу програми. Блок App комунікує з усіма іншими модулями системи та відповідає за розподіл процесів між ними. Наприклад зображення з відеокамери для візуалізації процесу розпізнавання обличчя Блок App отримує з черги в яку надсилає зображення VideoProcessor для того щоб не нагружати застосунок та не сповільнювати його роботу. Також блок App отримує поточне дерево каталогу з FileProcessor та може відобразити його для зручної навігації файловою системою.

Блок VideoProcessor:

VidoProcessor працює в окремому потоці та є не залежним від інших процесів. VideoProcessor також запускає ще один потік Classifer для відокремлення ресурсозатратних обрахунків виявленнь у інший потік. VideoProcessor підключається до вебкамери за допомогою GstreamerPipeline та отримує потік зображень які дублює без втрати FPS для блоку App та Classifer. Блок Detector ж в своєму темпі обробляє зображення перевіряє їх з даними з бази отриманої з серверу та лише дані про останнє виявлення у VideoProcessor.

Блок FileProcessor:

FileProcessor відповідає за отримання інформації з файлової системи для відображення її у блоці App, кодування/декодування файлів та стовренні приватних та публічних ключів для збереження в базу даних.

Блок Docker:

Блок Docker запускає декілька потоків для отримання та надсилання даних у інші модулі системи та зберіганню даних у базі даних. Блок TcpServer побудований так що спочатку необхідно надіслати типізований запит з інформацію про тип і кілкьість переданих даних після чого, він буде очікувати на обробку відповідного запиту.

##### Переваги реалізації за допомогою GStreame Pipeline

* Висока якість та стабільність потоку: Використання GStreamer Pipeline забезпечує високу якість та стабільність відеопотоку. Ця платформа оптимізована для роботи з мультимедійними данними, що дозволяє підтримувати високу швидкодію та якість відтворення.Також це сучасне рішення що використовується для транслювання/обробки/рендерингу відеопотоку у провідних компаніях світу.
* Можливітсь легко змінювати джерело потоку: GStreamer Pipeline дозволяє легко змінювати джерело вхідного відеопотоку. Це дозволяє підключати різні джерела відео (наприклад, веб-камери, відеофайли, IP-камери) без значних змін у коді.
* Легка можливість маштабування та заміни модулів для обробки відео: GStreamer Pipeline має гнучку архітектуру, яка дозволяє легко маштабувати систему та замінювати модулі для обробки відео в процесі розвитку проекту. Це дозволяє швидко внести зміни в систему та адаптувати її до нових потреб або вимог користувачів.
* Не потрібно робити запит в базу, не зберігається особиста інформація в базі:

У разі використання GSreamer ми зможемо відмовитись від викоирстання face\_recogniton та отримувати кропси з лице за допомогою GStreamer та в подальшому натренувати модель за допомогою OpenCV, що дозволить не загружати зображення в базу та одразу мати доступ до моделі.

##### Недоліки реалізації за допомогою GStreamer Pipeline

* Натренована модель зможе показувати лише число схожості, а не розпізнавання: Одним з недоліків використання GStreamer Pipeline є те, що натренована модель для розпізнавання облич може показувати лише число схожості між обличчям у відеопотоці та обличчями збереженими при тренуванні, а не саме ім'я або ідентифікатор особи. Це може вимагати додаткового коду для обробки результатів та визначення конкретних осіб на основі отриманих числових значень схожості.
* Час на тренування власної моделі: Ще одним недоліком є те, що для реалізації системи розпізнавання облич за допомогою GStreamer Pipeline може бути необхідний час на тренування власної моделі. Це може бути часомоємким процесом, особливо якщо потрібно натренувати модель на слабкому компютері.
* .Зберігання моделі: Недоліком є необхідність зберігання натренованої моделі для подальшого використання. Це може вимагати додаткового простору на диску та організації процесу зберігання та оновлення моделі. Також може виникати проблема з підтримкою та сумісністю версій моделі, що може ускладнити розвиток та підтримку системи в майбутньому.

## Інтерфейс роботи програми

##### Інтерфейс екрану для регістрації обличчя

