

TÂN TẠO UNIVERSITY
SCHOOL OF INFORMATION TECHNOLOGY

ASSIGNMENT 2 REPORT

**Distributed Face Recognition
System
for Multi-Branch Coffeehouse**

Submitted by:

Huỳnh Văn Đông 2202086
Trần Phương Nam 22020867

Mục lục

1	INTRODUCTION	2
1.1	Background	2
1.2	Objectives	2
2	SYSTEM ARCHITECTURE	2
2.1	Overall Architecture	2
2.2	Component Interactions	2
3	DESIGN AND IMPLEMENTATION	3
3.1	Technology Stack	3
4	DATABASE DESIGN	3
4.1	MongoDB Collections	3
4.2	MinIO Storage	3
5	FACE RECOGNITION IMPLEMENTATION	3
5.1	Face Detection Algorithm	3
5.2	Processing Workflow	4
6	CONCLUSION	4

1 INTRODUCTION

1.1 Background

In the competitive coffeehouse industry, customer experience is paramount. Recognizing returning customers and knowing their preferences can significantly enhance service quality and customer loyalty. This project addresses this need by implementing an automated face recognition system that identifies customers and retrieves their order history.

1.2 Objectives

- Develop a distributed system with client-server architecture
- Implement automatic face recognition using open-source libraries
- Create an intuitive user interface for staff
- Enable real-time customer recognition

2 SYSTEM ARCHITECTURE

2.1 Overall Architecture

The system follows a distributed client-server architecture with the following components:

1. **Client Application:** GUI application deployed at each branch with automatic face capture capability
2. **Backend Server:** Processes face recognition requests and manages customer data
3. **Database Layer:**
 - MongoDB: Stores customer metadata and order information
 - MinIO: Stores customer face images

2.2 Component Interactions

The system workflow operates as follows:

1. Client captures customer face automatically using webcam (hidden from view)
2. Face image is sent to backend server for processing
3. Server extracts face encoding
4. System searches MongoDB for matching face encoding
5. If match found: Retrieve customer information and order history
6. If no match: Create new customer profile
7. Server sends response back to client
8. Client displays customer information and menu to staff

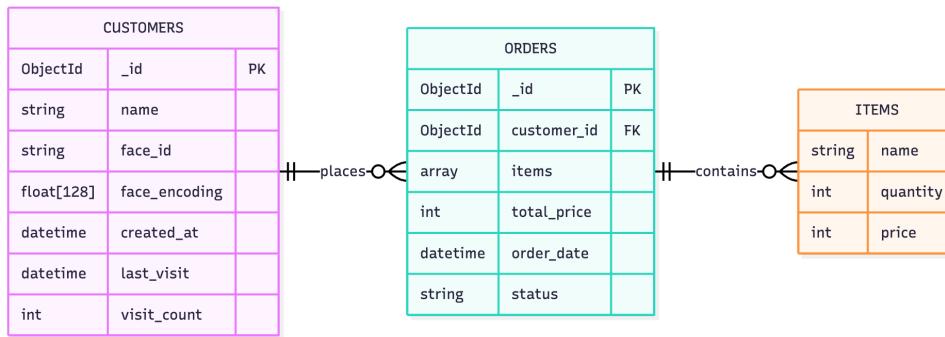
3 DESIGN AND IMPLEMENTATION

3.1 Technology Stack

- **Face Recognition:** face_recognition library (based on dlib)
- **Database:** MongoDB (document store)
- **Object Storage:** MinIO

4 DATABASE DESIGN

4.1 MongoDB Collections



4.2 MinIO Storage

Face images are stored in MinIO object storage:

- **Bucket:** customer-faces
- **Object naming:** {face_id}.jpg
- **Content-Type:** image/jpeg

This separation of metadata (MongoDB) and binary data (MinIO) provides scalability and efficient storage management.

5 FACE RECOGNITION IMPLEMENTATION

5.1 Face Detection Algorithm

The system uses the `face_recognition` library which provides:

1. **Face Detection:** Identifies faces in images using HOG or CNN
2. **Face Encoding:** Generates 128-dimensional vector representation
3. **Face Comparison:** Calculates Euclidean distance between encodings

Recognition tolerance is set to 0.6

5.2 Processing Workflow

1. Camera captures image automatically (every 3 seconds)
2. Image converted to RGB format
3. Face locations detected in image
4. Face encodings generated for detected faces
5. Encoding compared with all stored encodings in database
6. If match found ($\text{distance} \leq 0.6$): Customer recognized
7. If no match: New customer profile created
8. Result returned to client with customer data

6 CONCLUSION

The implementation showcases practical application of distributed systems concepts including client-server architecture, database design, and real-time data processing. The system enhances customer experience by enabling personalized service based on order history.

Future Enhancements:

- Multi-face detection for group orders
- Integration with payment systems