



Kyungdong University (KDU)
Bachelor of Smart Computing (BSC)

Problem Solving Concepts

Final Project

Real-Time Facial Recognition System Using Flask and React

Submitted by

Jumale Abdi Jillo

2024127

Submitted to

Professor Dr. Mohammed A.A

Assistant Professor

Department of Smart Computing

ABSTRACT

This project presents a real-time facial recognition system developed using a Flask backend and a React frontend. The system leverages modern web technologies, computer vision, and machine learning techniques to perform user registration and login through facial recognition. Users can register and log in by capturing images via a webcam, which are then processed to extract and store facial encodings in a MongoDB database. The system demonstrates a practical application of real-time data processing and seamless communication between backend and frontend components.

The architecture of the system is designed to ensure scalability and efficiency. The Flask backend handles image processing, facial encoding, and database interactions, while the React frontend provides a user-friendly interface for image capture. Real-time data transfer between the backend and frontend is facilitated by HTTP requests, ensuring a responsive user experience. This project highlights the integration of various technologies to create an effective and user-centric facial recognition system.

INTRODUCTION

Facial recognition technology has gained significant attention due to its wide range of applications in security, authentication, and personalized user experiences. Traditional username and password authentication methods, while widely used, have inherent vulnerabilities. Passwords can be forgotten, stolen, or compromised, leading to unauthorized access and potential security breaches. In contrast, a facial recognition system provides a more secure and convenient method of authentication by using unique biometric data that is difficult to replicate or steal.

In recent years, face recognition logins have become increasingly popular due to their enhanced security and ease of use. The advancements in this technology have significantly improved its accuracy and reliability, making it a preferred choice for many businesses and organizations. The fundamental concept of a facial recognition-based login system is straightforward: instead of entering a password or PIN, the system verifies a user's identity by capturing and analyzing their facial image. If the captured image matches the stored facial data, the system grants access.

This project aims to develop a real-time facial recognition system that allows users to register and log in using their facial images. The system employs a Flask-based backend for processing and storing facial encodings and a React-based frontend for capturing user images and displaying the results. By integrating computer vision libraries like OpenCV and face_recognition, the project provides a robust and efficient solution for facial recognition tasks. The use of facial recognition enhances security, reduces the reliance on passwords, and provides a seamless user experience.

How Does the Facial Login System Work

A facial login system uses artificial intelligence to match an image of a person's face with the face encodings stored in a database. The process involves the following steps:

1. Receives an input image of the user's face: The system captures the user's facial image via a webcam or another imaging device.
2. Analyses the image of the user's face: The captured image is processed to identify facial features.
3. Creates a facial encoding of the user's face: The system generates a unique facial encoding based on the identified features.
4. Compares the facial features to the face encoding stored in the database: The generated facial encoding is compared with the encodings already stored in the database.
5. Determines if the facial encoding matches or closely resembles any of the images in the database: If the encoding matches or is like any stored encoding, the system verifies the user's identity.

The accuracy and robustness of facial login systems depend on several factors, such as the quality of the image, the size and power of the application server, and the database. To train a powerful facial recognition system, a large dataset of images is required.

Application of Facial Recognition

Facial recognition technology has a wide range of applications across various domains:

1. **Access to Personal Devices:** Many Apple devices use facial recognition to grant access to smartphones with the Face ID feature. This allows users to log in without a PIN or password. A good example is Apple's Face ID.
2. **Accessibility:** Facial recognition can enhance accessibility for individuals with disabilities by enabling hands-free device control and simplifying interactions with technology.
3. **Social media:** Many social media platforms such as TikTok, Snapchat, and Instagram have filter features that let users apply filters to their images and videos.
4. **Finding Missing People:** In law enforcement, facial recognition systems can help find missing people. When paired with street surveillance cameras or traffic cameras, these systems can pick a single face out of crowds of thousands.
5. **Access to Transportation:** Transport authorities use facial recognition and face scans to match images of travelers to their identification documents at airports, streamlining the security process.

Advantages of Face Recognition Login System

Facial recognition login systems offer several benefits over traditional authentication methods:

1. **Enhanced Security:** Unlike passwords or PINs, facial recognition is difficult to forge. This reduces the risk of unauthorized access due to stolen or guessed credentials.
2. **Convenience:** Users can access their devices or accounts quickly and effortlessly without the need to remember complex passwords or carry additional authentication devices.
3. **User Experience:** The seamless and intuitive nature of facial recognition provides a smooth and efficient user experience, improving overall satisfaction.
4. **Hygienic:** Since facial recognition is contactless, it eliminates the need for physical interaction with devices, which is particularly advantageous in maintaining hygiene.
5. **Reduced Fraud:** Facial recognition adds an extra layer of verification that can help reduce instances of identity theft and fraudulent activities.
6. **Accessibility:** It can be particularly beneficial for individuals with disabilities who may find traditional input methods challenging.
7. **Integration:** Facial recognition systems can be easily integrated with various applications and devices, providing a versatile solution for different authentication needs.

Disadvantages of Face Recognition Login System

While facial recognition login systems offer numerous advantages, they also come with certain drawbacks:

1. **Privacy Concerns:** The collection and storage of facial data raise significant privacy issues. Users may be uncomfortable with the potential misuse of their biometric information.
2. **Accuracy Issues:** Although facial recognition technology has improved, it can still be less accurate under certain conditions, such as poor lighting, changes in appearance (e.g., facial hair, makeup), or wearing accessories like glasses or hats.
3. **Security Risks:** Despite being more secure than passwords, facial recognition systems are not foolproof. They can be vulnerable to spoofing attacks using high-quality photos or videos.
4. **High Costs:** Implementing facial recognition systems can be expensive, requiring sophisticated hardware and software, as well as ongoing maintenance and updates.
5. **Bias and Discrimination:** Facial recognition technology can exhibit biases, often performing less accurately for certain demographic groups. This can lead to discrimination and unequal treatment.
6. **Dependence on Technology:** Users must rely on the availability and functionality of facial recognition technology. Any technical issues or failures can prevent access and disrupt user activities.
7. **Ethical Concerns:** The use of facial recognition, particularly by governments and law enforcement, raises ethical concerns about surveillance and the potential for abuse of power.

SYSTEM ARCHITECTURE

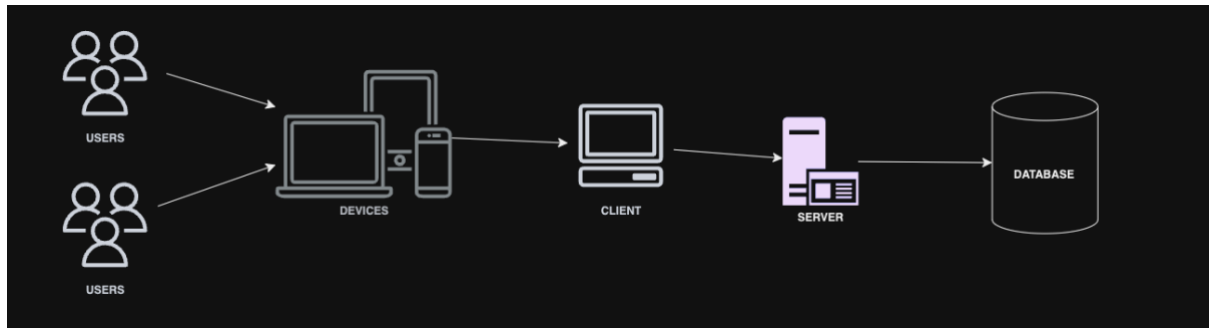
Technology used

Several technologies have been utilized in this project, here's a brief overview

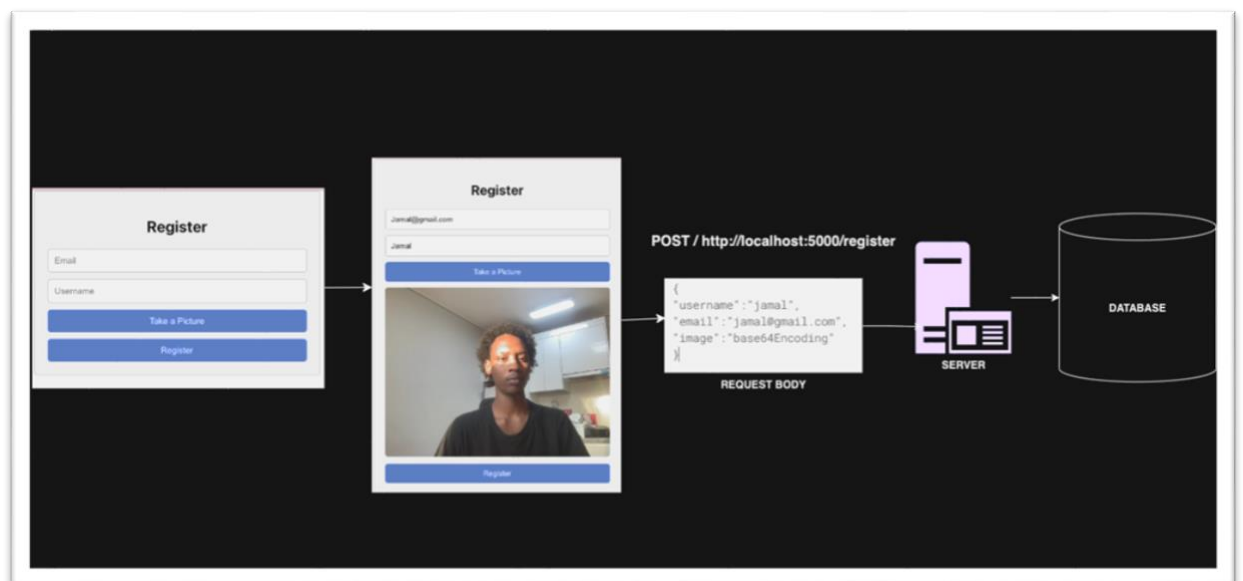
Technology	Purpose
Flask	Web framework for building the backend API
OpenCV	Image processing library for face recognition
NumPy	Library for numerical computing in python
Face_recognition	Library for face recognition
Flask-CORS	Extension for handling Cross-Origin Resource Sharing
PyMongo	MongoDB driver for Python
React	JavaScript library for building user interfaces
React-Router-Dom	Routing Library for React applications
Axios	Promise-based HTTP client for the browser
React-Webcam	React component for accessing the user's webcam
React-Toastify	React component for toast notifications

SYSTEM DESIGN

High level architecture diagram




User Registration



User Login

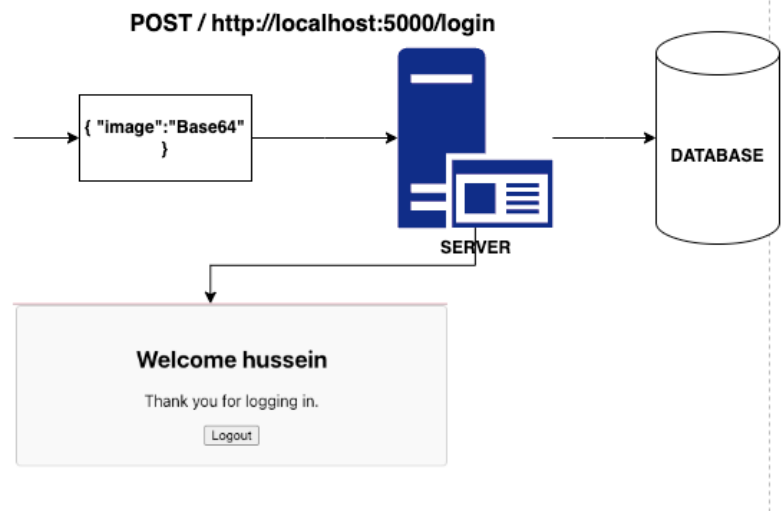
Login

Take a Picture

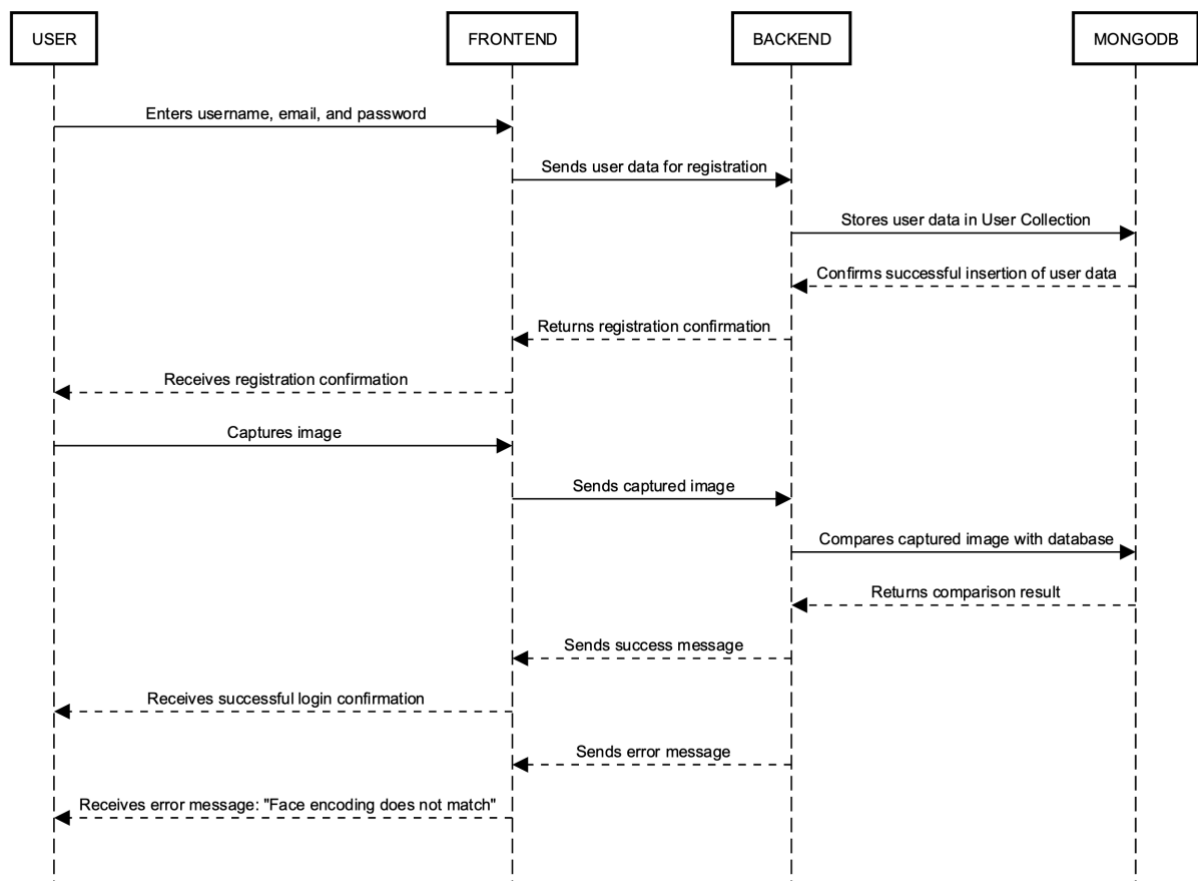


Login

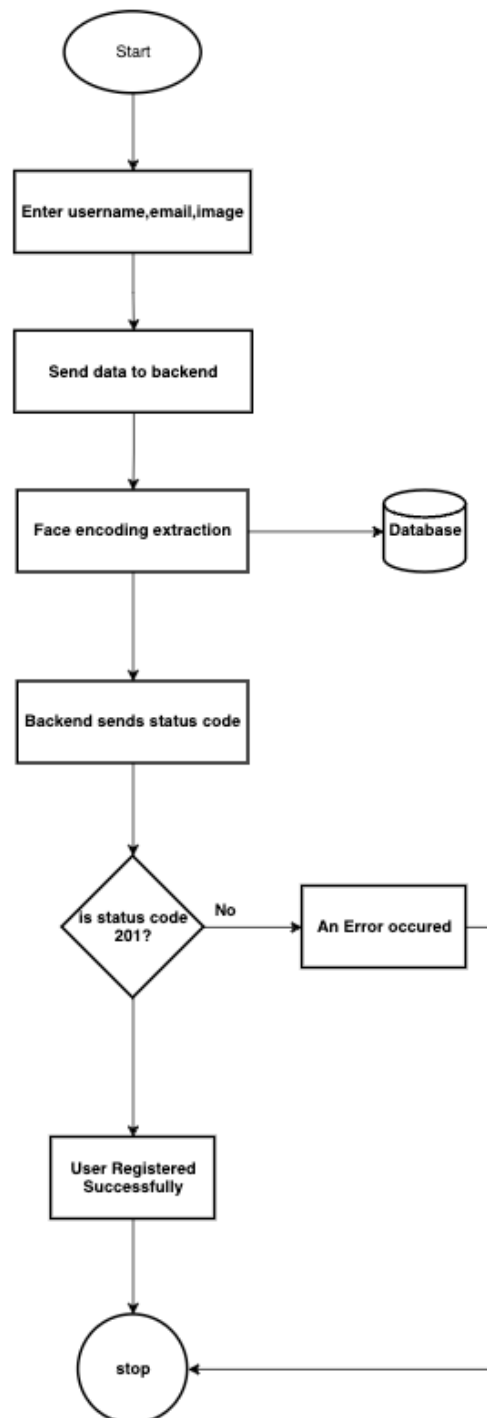
Toggle Camera



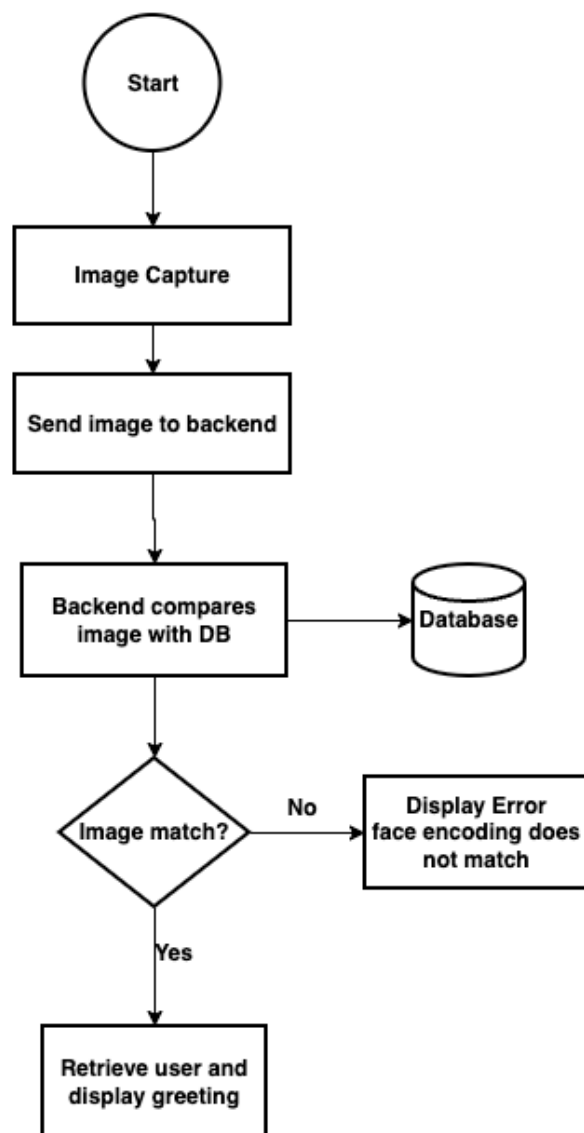
Sequence diagram



System architecture



Login flow



Database design

MongoDB was chosen as the database for this project due to its flexibility, scalability, and ease of use. MongoDB is a NoSQL database, meaning it stores data in a non-tabular format, allowing for more flexible data storage compared to traditional relational databases. In MongoDB, data is stored in collections, which are like tables in relational databases. Each collection contains documents, which are analogous to rows in a table. The "_id" field in each document serves as a primary key for uniquely identifying records within the collection.

In this design, the "users" collection is used to store user information such as email, username, and facial features encoding. The "email" and "username" fields store basic user details provided during registration, while the "face_encoding" field stores a Base64 encoded representation of the user's facial features. This design allows for efficient storage and retrieval of user data, enabling seamless user registration, authentication, and interaction within the application.

Collection user

Field	Type	Description
_id	ObjectId	Unique identifier for the user
Email	string	Email address of the user
Username	string	Username of the user
Face_encoding	string	Base64 encoded face encoding

Description:

- _id: Automatically generated unique identifier for each user.
- email: Email address of the user provided during registration.
- username: Username chosen by the user during registration.
- face_encoding: Base64 encoded representation of the facial features of the user for face recognition purposes.

API Endpoint Design

To facilitate effective communication between the frontend and backend sides of the application, a RESTful API was employed. A RESTful API adheres to a request-response cycle where a user initiates a request, the backend processes the request, and subsequently sends a response back to the user.

RESTful APIs utilize the HTTP protocol for communication, where requests are sent with a request body containing the payload provided by the user, which can be in the form of form data, binary data, or JSON format.

In this project, three API endpoints were designed for user registration, authentication, and interaction. Below is a detailed design of each API endpoint.

Endpoint	Method	Description	Request Body	Response
/register	POST	Register a new user	Email (String) Username(string) Image(string)	Message (String), user_id (string)
/login	POST	Authenticate a user by comparing their face	Image (String)	Message (string) Username (string) User_id (string)
/greeting	GET	Retrieve the username associated with a user ID	User_id (String)	Username (String)

Conclusion

The developed system effectively demonstrates the integration of facial recognition technology into a web-based authentication system. By leveraging Flask, MongoDB, OpenCV, face_recognition, and React, the application provides a modern and secure method for user authentication. This system can be expanded and adapted for various use cases where enhanced security is required, such as secure access systems, personalized user experiences, and more.

Results

The application provides a seamless experience for user registration and login using facial recognition. Key functionalities demonstrated include:

User Registration:

- Users enter their username, email, and capture their image.
- The image is processed to extract face encodings, which are then stored in MongoDB alongside the user data.
- Successful registration returns a confirmation message to the user.

User Login:

- Users capture their image.
- The captured image is processed, and the face encoding is compared with the stored encodings in the database.
- If a match is found, the user is logged in and greeted with a personalized message.
- If no match is found, an error message is displayed indicating the face encoding does not match.

References

1. PySeek. (2022). Face Recognition Login System using Python. Retrieved from <https://pyseek.com/2022/09/face-recognition-login-system-using-python/>
2. ComputerWeekly. (2022). Why diversity in AI remains a challenge and how to fix it. Retrieved from <https://www.computerweekly.com/opinion/Why-diversity-in-AI-remains-a-challenge-and-how-to-fix-it>
3. Ageitgey. (n.d.). face_recognition. Retrieved from https://github.com/ageitgey/face_recognition
4. Flask Documentation. (n.d.). Retrieved from <https://flask.palletsprojects.com/en/3.0.x/>
5. PyMongo Documentation. (n.d.). Retrieved from <https://pymongo.readthedocs.io/en/stable/>
6. React Documentation. (n.d.). Retrieved from <https://react.dev/>
7. NPM. (n.d.). react-webcam. Retrieved from <https://www.npmjs.com/package/react-webcam>
8. NPM. (n.d.). react-toastify. Retrieved from <https://www.npmjs.com/package/react-toastify>
9. Axios Documentation. (n.d.). Retrieved from <https://axios-http.com/docs/intro>