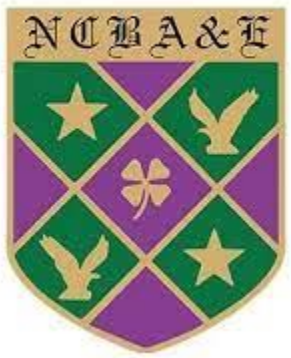
FACE-RECOGNITION ATTENDANCE SYSTEM

Final Year Project

Session 2019-2023

A project submitted in partial fulfillment of the degree of

BS in Computer Science



Department of Computer Science

NCBA&E West Canal Campus

Submitted By:

Muhammad Suliman Riaz BSCS-FALL-19-34

Project Supervisor: Professor Muhammad Attaullah

DECLARATION

We state that the grounds of this project report are authentic work, exceptions are for quotations and citations, which have been properly recognized. We also state that it has not been formerly or at this time submitted for any other award or degree in the Department of Computer Science at NCBA&E WCC or in any other institutions.

Project Group Members			
Submission date	Student Name	Email ID	*Signature
	M Suliman Riaz	Sulimangorsi623@gmail.com	

Final Year Project Approval Letter

It is to certify that this project report entitled "**Face-recognition attendance system**" prepared by **Muhammad Suliman Riaz.** He has fulfilled the necessary standards for presenting this project as a requirement for the award/degree of BSCS at the NCBA&E West Canal Campus, Lahore.

Supervisor's name:	**Supervisor's signature:**
Professor Muhammad Attaullah	

ACKNOWLEDGEMENT

No person is perfect or near perfect in things they do, they are limited in knowledge and thinking. It is only from the guidance of Almighty Allah that a person performs and completes anything. Almighty Allah gives us strength and power to overcome difficulties and perform our duties, through His guidance only we are able to complete our tasks. We cannot thank Allah enough for the opportunities He gives us. Alhamdulillah on the completion of this project.

I am special thankful to **Prof. Muhammad Attaullah** who helped me in this project, that might not be possible for me to finish this project without his help and who made me able to be at this position and as he appreciated me to do this.

And thanks to our parents and teachers as well, who support us at every difficulty of life, have faith in us and thanks to their belief in us for our success.

TABLE OF CONTENTS

CHAPTER 5

IMPLEMENTATION

CHAPTER 6

TESTING AND REVIIEW

CONCLUSION

REFERENCES

Chapter 1

Introduction

Chapter 1

INTRODUCTION

The Face Recognition Attendance System is a groundbreaking solution aimed at revolutionizing the conventional attendance tracking process. In an era where automation and efficiency are paramount, this system leverages state-of-the-art facial recognition technology to address the challenges associated with manual attendance marking. This chapter offers a comprehensive overview of the project's significance and the context in which it operates. This System is a modern and efficient solution designed to streamline the attendance management process. This provides an overview of the project, its objectives, and its relevance in today's educational and corporate environments, scope, problem statement, problem discussion and tools and techniques.

1.1 Problem Statement

Manual attendance tracking methods are riddled with inefficiencies and limitations. Traditional processes involve the time-consuming and error-prone manual entry of attendance data. These methods often result in discrepancies, incomplete records, and wasted instructional time. Furthermore, the requirement for physical proximity during attendance marking has become a significant concern in the wake of global health crises, such as the COVID-19 pandemic.

1.2 Problem Discussion

Inefficiency and Error-Prone Processes: The manual calling of names, checking off lists, or scanning paper attendance sheets is slow and prone to human error. This can lead to inaccurate attendance records, which are critical for assessing student participation and compliance.

Resource Wastage: In educational institutions, a significant amount of valuable instructional time is devoted to attendance-taking. This time could be better spent on actual teaching and learning activities.

Adaptation to New Challenges: Recent global events have underscored the need for contactless and socially distanced solutions. Traditional attendance methods have struggled to adapt to these changing circumstances.

Security and Data Privacy: Maintaining the confidentiality and security of attendance data is a challenge with traditional methods. Storing and managing physical attendance sheets can expose sensitive information to unauthorized access.

Scalability and Record Keeping: As institutions grow or diversify, the manual handling and storage of attendance records become increasingly complex and resource-intensive.

The Face Recognition Attendance System seeks to address these problems by automating attendance tracking, enhancing accuracy, and providing a convenient and secure solution. By utilizing facial recognition technology, this system aligns with the demands of modern education and corporate environments, where efficiency, data security, and contactless operations are of paramount importance.

This project's introduction now includes a problem statement and a problem discussion, providing context for the system's development and highlighting the issues it aims to resolve. Subsequent sections of the documentation will explore the design, implementation, and testing of the system.

1.3 Objective

At its core, the primary objective of this project is to create a cutting-edge, automated attendance management system that removes the need for labor-intensive manual processes. Instead, it harnesses the capabilities of facial recognition technology to precisely and rapidly record individuals' attendance. The system is intended to provide a solution that is not only accurate but also convenient, making it an ideal fit for a wide range of educational and corporate environments.

1.4 Background

Traditional methods of attendance tracking often rely on time-consuming, error-prone, and manual data entry. These methods may involve calling out names, checking off lists, or scanning physical attendance sheets. The Face Recognition Attendance

System was conceptualized to overcome these shortcomings by capitalizing on advanced facial recognition algorithms.

1.5 Why Facial Recognition?

Facial recognition technology has seen significant advancements in recent years, making it a powerful tool for various applications. It operates by capturing and analyzing distinctive facial features to identify and verify individuals. The technology's accuracy and speed have improved, making it an ideal choice for automating tasks like attendance management. The Face Recognition Attendance System is designed to harness these technological advancements to streamline attendance recording and bring it into the digital age.

1.6 Relevance in Education and Industry

The Face Recognition Attendance System is highly relevant in educational institutions, where maintaining accurate attendance records is critical for assessing student participation and compliance. Additionally, the system finds application in corporate settings, simplifying the process of employee attendance tracking. By automating this task, it not only saves time but also minimizes errors, leading to more efficient and effective management of resources.

This project's introduction highlights the transformational potential of the Face Recognition Attendance System, shedding light on its objectives, background, and the ever-increasing relevance of facial recognition technology in today's educational and corporate landscapes. The subsequent sections of the documentation will delve into the specific details, design, and implementation of the system.

1.7 Scope of the Project

1.7.1 In-scope Items

The Face Recognition Attendance System encompasses a set of features and functionalities aimed at automating the attendance management process, enhancing

data accuracy, and improving efficiency. The following elements are within the project's scope:

Real-time Face Detection and Recognition:

The system will employ advanced facial recognition technology to identify and verify individuals in real time.
It will utilize computer vision algorithms to capture and analyze facial features and patterns.

Automatic Attendance Record Keeping:

The system will automatically record the attendance of individuals as they are recognized.
Timestamps will be applied to attendance entries, indicating the date and time of each recognition event.

Notification of Recognized Individuals:

The system will provide notifications when a recognized individual is detected.
Notifications can be in the form of text messages on the user interface or audio announcements, making it useful for a variety of environments, including classrooms and conference rooms.

Support for Multiple Users and Faces:

The system will be capable of managing multiple user profiles, each associated with a set of registered faces.
This allows institutions or organizations to use the system for various classes, events, or groups.

Integration with External Libraries:

The project will integrate with established libraries and frameworks for face detection and recognition, such as OpenCV and Face Recognition.
This integration will leverage existing open-source tools to enhance the system's capabilities.

1.7.2 Out-of-scope Items

While the project focuses on automating attendance tracking using facial recognition, it also has certain limitations and exclusions that define what is not included in the scope:

Attendance Data Storage and Analysis:

The project does not encompass long-term storage and in-depth analysis of attendance data.Data storage and analysis may require additional systems or databases, which are considered out of scope.

Integration with Other Systems or Databases:

The system will not directly integrate with external databases or student information systems.Such integrations may be required in the future but are considered external to the current scope.

Additional Security Features Beyond Face Recognition:

While the system provides facial recognition-based security, it does not include advanced security features like biometric authentication or access control beyond the recognition of individuals.The Face Recognition Attendance System's scope is carefully defined to ensure that the project addresses critical attendance management needs while remaining feasible and focused. This scope delineates the boundaries of the system's functionality, clarifying what the system will and will not deliver. It provides a foundation for the subsequent phases of analysis, design, and implementation in the project's development process.

1.8 Functional Requirements

Functional requirements outline the specific features and capabilities that the Face Recognition Attendance System must provide to meet its objectives. These requirements serve as a blueprint for the system's functionality and guide its development.

1.8.1 User Authentication

The system must allow authorized users to log in with unique credentials.
Users should have predefined roles, such as administrators and instructors, each with specific privileges and access levels.

1.8.2 Face Detection and Recognition

The system must be capable of real-time face detection in the video feed.
It should accurately recognize faces of individuals registered in the system's database.
The recognition process should take place rapidly to maintain the real-time nature of attendance tracking.

1.8.2 Attendance Recording

The system must automatically record attendance for recognized individuals.
Each attendance entry should be timestamped with the date and time of the recognition event.A log of attendance records should be maintained and made available for review and export.

1.8.4 Notification System

The system must provide notifications when a recognized individual is detected.
Notifications can take the form of text messages displayed on the user interface or audio announcements.
Notifications must be clear and easy to understand, ensuring that instructors or administrators can take appropriate action.

1.8.5 User Management

Administrators should have the capability to add, modify, or delete user profiles.
Users should be able to update their own profiles, including their registered faces.
The system should manage user data securely, including personal information and facial recognition data.

1.9 Non-Functional Requirements

Non-functional requirements focus on the system's qualities and constraints that affect its performance, usability, and overall user experience.

1.9.1 Performance

The system should respond to face recognition requests in real-time to maintain the efficiency of attendance tracking.
It should be capable of handling a certain number of concurrent users without significant degradation in performance.

1.9.2 Security

Face recognition data, user profiles, and authentication data must be securely stored and protected.User authentication and data transmission should be encrypted to prevent unauthorized access and data breaches.
The system should have mechanisms in place to prevent unauthorized attempts to manipulate attendance data.

1.9.3 Usability

The user interface should be intuitive and user-friendly, ensuring that users can navigate and interact with the system with minimal training or guidance.
The system should provide clear feedback and instructions to users during the face recognition process.

1.9.4 Reliability

The system should have a high recognition accuracy rate to minimize false positives and false negatives.

It should be resilient to variations in lighting conditions, facial expressions, and minor changes in appearance, such as facial hair or eyewear.

1.9.5 Scalability

The system should be designed to accommodate additional users and faces in the future.
Scalability considerations should be factored into the architecture to support the growth of the system's user base.

1.9.6 Compatibility

The system should be compatible with common operating systems and web browsers to ensure accessibility for a broad range of users.
Compatibility should also extend to external libraries and tools used for face recognition and video capture.

These functional and non-functional requirements provide a comprehensive set of guidelines that shape the development and performance expectations of the Face Recognition Attendance System. They ensure that the system not only meets the essential attendance tracking needs but also delivers a secure, user-friendly, and scalable solution for educational and corporate environments.

1.10 Tools and Techniques

The Face Recognition Attendance System relies on a range of tools and techniques to accomplish its objectives. These tools and techniques are crucial for the development and operation of the system. Here are some of the key tools and techniques involved:

1.10.1 Tools

OpenCV (Open Source Computer Vision Library): OpenCV is a widely used open-source computer vision library that provides various tools and functions for image and video processing. It is instrumental in face detection and image manipulation within the system.

Face Recognition Library: This library is specifically designed for face recognition tasks. It allows the system to identify and compare faces using deep learning techniques.

Python: Python serves as the primary programming language for the development of the system. It offers a rich ecosystem of libraries and frameworks for image processing and machine learning.

Streamlit: Streamlit is a popular Python library for building web applications with minimal code. It is used to create the user interface for the Face Recognition Attendance System, making it accessible via a web browser.

CSV (Comma-Separated Values) Files: The system utilizes CSV files for logging attendance data. These files are easily readable and exportable, making it convenient for administrators and instructors to access and analyze attendance records.

Database Management System (optional): While not explicitly mentioned in the project's scope, a database management system may be used for long-term storage and retrieval of attendance data. This could provide additional features, such as historical data analysis and reporting.

1.10.2 Techniques

Facial Recognition Algorithms: The system employs advanced facial recognition algorithms that extract facial features, create face embeddings, and compare them to recognize individuals.

Real-Time Video Processing: Video frames from the webcam feed are continuously processed in real-time to detect and recognize faces. This involves techniques like frame resizing, color conversion, and face recognition on each frame.

Machine Learning: Machine learning techniques are used for training the model to recognize known faces and classify unknown faces. The system learns to distinguish between different individuals based on historical data.

User Authentication: Techniques for user authentication ensure that only authorized individuals can access and interact with the system. This typically involves username-password authentication.

Notification Mechanisms: Techniques for sending notifications to instructors or administrators upon recognizing individuals. These mechanisms may include sending text notifications or using text-to-speech libraries for audio announcements.

User Interface Design: The design of the system's user interface is crucial to make it user-friendly and intuitive. Design principles and techniques, such as responsive design, are applied to create a clear and effective interface.

Security Measures: Techniques for ensuring data security, including encryption of data at rest and in transit, secure storage of facial recognition data, and measures to prevent unauthorized access or tampering with the system.

These tools and techniques collectively enable the Face Recognition Attendance System to fulfill its functions, provide a secure and efficient user experience, and ensure accurate attendance tracking in real-time.

Chapter 2

ANALYSIS

Chapter 2

ANALYSIS

In this chapter, we delve into the analysis phase of the Face Recognition Attendance System project. The analysis phase plays a crucial role in defining the system's functionality, serving as a bridge between project requirements and the actual development phase.

2.1 Use Case:

The use case diagram provides a visual representation of the system's interactions with actors (users and external systems) and the primary functions it performs. It outlines the key use cases that the system supports.

2.1.1 Use Case Diagram

This section provides detailed descriptions of each use case, specifying the interactions, flow of events, and the role of each actor in the system. Use cases cover actions like user authentication, face detection, attendance recording, and more.

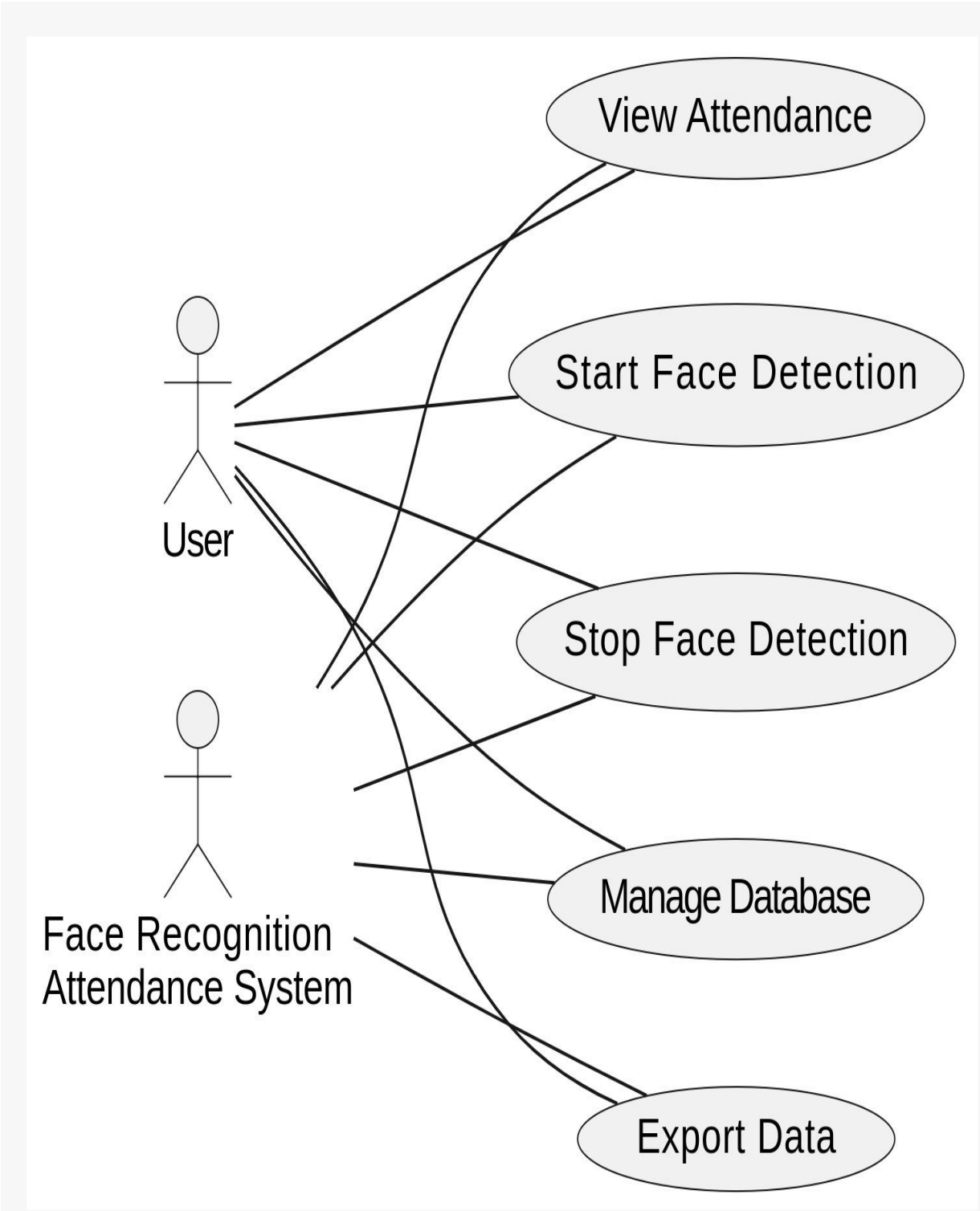

View Attendance
Start Face Detection
Stop Face Detection
Manage Database
Export Data
User
Face Recognition
Attendance System

Figure 2.1

2.1.2 Use Case Table

The use case table is a tabular representation of use cases, their descriptions, preconditions, postconditions, and other relevant information. It serves as a reference for developers and stakeholders, offering a concise overview of the system's functionality.

Actor	**Use Case**	**Description**
User, System	Start Face Detection	User initiates the face detection process
User, System	Stop Face Detection	User stops the face detection process by typing 'stop.'
User, System	Manage Database	User manages the database of known faces.
User, System	View Attendance	User views the attendance records.
User, System	Export Data	User exports attendance data.

Table 2.1

2.2 Prototype

There are some different methods and models, which are used by the people according to their work and planning. The brief introduction of various methodologies is discussed below, and then we will be able to decide that which methodology suits our project.

2.2.1 Iterative Development

Iterative development involves breaking the project into small cycles or iterations. In each iteration, a subset of the system's features is developed, tested, and reviewed. The software is continuously refined in successive iterations until it meets the desired quality and functionality.

2.2.2 Incremental Development

Incremental development involves adding and building upon features in a linear or incremental fashion. In each increment, new features are added to the existing system, expanding its capabilities. Each increment enhances the software's functionality.

2.2.3 Reason to choose the Incremental Development and Iterative Development

Adaptability: Software development can be unpredictable, with evolving requirements, changing user needs, and unforeseen issues. Iterative and incremental approaches allow developers to adapt to these changes more effectively.

Early Delivery: Both approaches allow for the early delivery of partial functionality, which can be beneficial in situations where stakeholders require or can benefit from intermediate releases.

Risk Management: These approaches mitigate the risk of late-stage failures or unexpected issues by addressing problems and uncertainties in smaller, more manageable portions.

User Involvement: Users and stakeholders can provide feedback and make decisions at various stages of development, leading to a system that better meets their expectations.

Continuous Improvement: Both iterative and incremental approaches foster a culture of continuous improvement, ensuring that the software evolves to better meet user needs and quality standards.

Complex Projects: In large and complex projects, breaking the development process into manageable parts simplifies project management and helps ensure that the final product is a result of well-thought-out increments.

In summary, iterative and incremental development approaches are used to create software that is adaptable, user-oriented, and quality-driven. They are particularly valuable in dynamic and complex development environments where change and improvement are essential for success.

2.2.4 How this prototype align

Here's an explanation of why this prototype aligns with such an approach:

Real-Time Face Detection and Recognition: The prototype is designed to continuously detect and recognize faces in real time. It leverages a webcam feed to process video frames, allowing for ongoing detection and tracking of individuals. This real-time aspect aligns with an iterative approach where the system continually processes data in cycles.

Continuous Improvement: The system can be fine-tuned and enhanced incrementally over time. Additional features, such as data storage, reporting, or integration with external systems, can be integrated into the system in subsequent iterations.

User Interface Refinement: The user interface is presented using Streamlit, a tool that allows for easy and iterative interface development. Adjustments and improvements to the user interface can be made iteratively as user feedback is gathered.

User Experience Testing: The system can be tested with actual users or stakeholders to gather feedback and make iterative improvements to usability, notifications, and overall user experience.

Agile-Friendly: The system's structure and modularity allow for agility in development. New functionalities can be added in sprints or development cycles, making it compatible with Agile methodologies.

While the prototype demonstrates essential real-time face recognition and attendance tracking capabilities, it also leaves room for future iterations and incremental improvements. This aligns with an iterative and incremental development approach, allowing the system to evolve and adapt to changing requirements and user feedback over time.

Chapter 3

DESIGN

Chapter 3

DESIGN

3.1 System Architecture

The system architecture defines the high-level structure and organization of the Face Recognition Attendance System. This section outlines the key components and their interactions, providing insights into how the system operates.

3.1.1 Components

Known Faces Initialization: This component loads pre-defined facial images and encodings for known individuals. It includes images of individuals such as Suliman, Usman, and Umair, and their corresponding face encodings. This component serves as the foundation for face recognition.

Webcam Capture: The system captures video frames from the webcam using OpenCV (cv2). This component is responsible for providing a continuous stream of video frames for processing.

Face Recognition Engine: The system utilizes the face_recognition library to perform facial recognition. It identifies face locations, extracts face encodings, and compares them with known face encodings to recognize individuals.

Attendance Recording: When a recognized face is detected, the system records attendance by matching it to a list of known students. If a match is found, the student's name and the current time are recorded in a CSV file.

User Interface (Streamlit): The user interface is created using Streamlit. It provides a user-friendly way to start and stop face detection and displays the video feed with recognized faces and attendance information.

3.1.2 Interactions

The system continuously captures video frames from the webcam and processes them through the face recognition engine.
The face recognition engine detects face locations and encodings and compares them with known face encodings.
When a match is found, the system records the student's attendance by appending their name and the current time to a CSV file.
The user interface, built using Streamlit, provides the means to start and stop face detection, displays the video feed, and shows recognized faces and attendance details.

3.2 Technology Stack

OpenCV (cv2) for video capture.
face_recognition library for facial recognition.
Streamlit for creating the user interface.
Python as the primary programming language.
This **high-level system architecture** serves as a foundation for the subsequent design and implementation phases. It outlines the key components and their roles in the system, providing a clear understanding of how the system operates.

3.3 Data flow diagram

A Data Flow Diagram (DFD) is a graphical representation of how data moves within a system or between a system and its external entities. It's a structured way to depict the flow of data, processes that act upon the data, data stores, and the various entities that interact with the system. DFDs are commonly used in software engineering and business analysis to model the data flow and processes in a system.

DFDs come in different levels of detail, with Level 0 being the highest level that provides an overview of the entire system, and subsequent levels (Level 1, Level 2, etc.) providing more detailed views of specific parts of the system. The purpose of a DFD is to provide a visual and structured way to understand how data flows through a system, how it is processed, and where it is stored.DFDs are valuable for various purposes, including system design, requirements analysis, and documentation. They help in understanding the relationships between different components of a system and are an essential tool in system analysis and design processes.

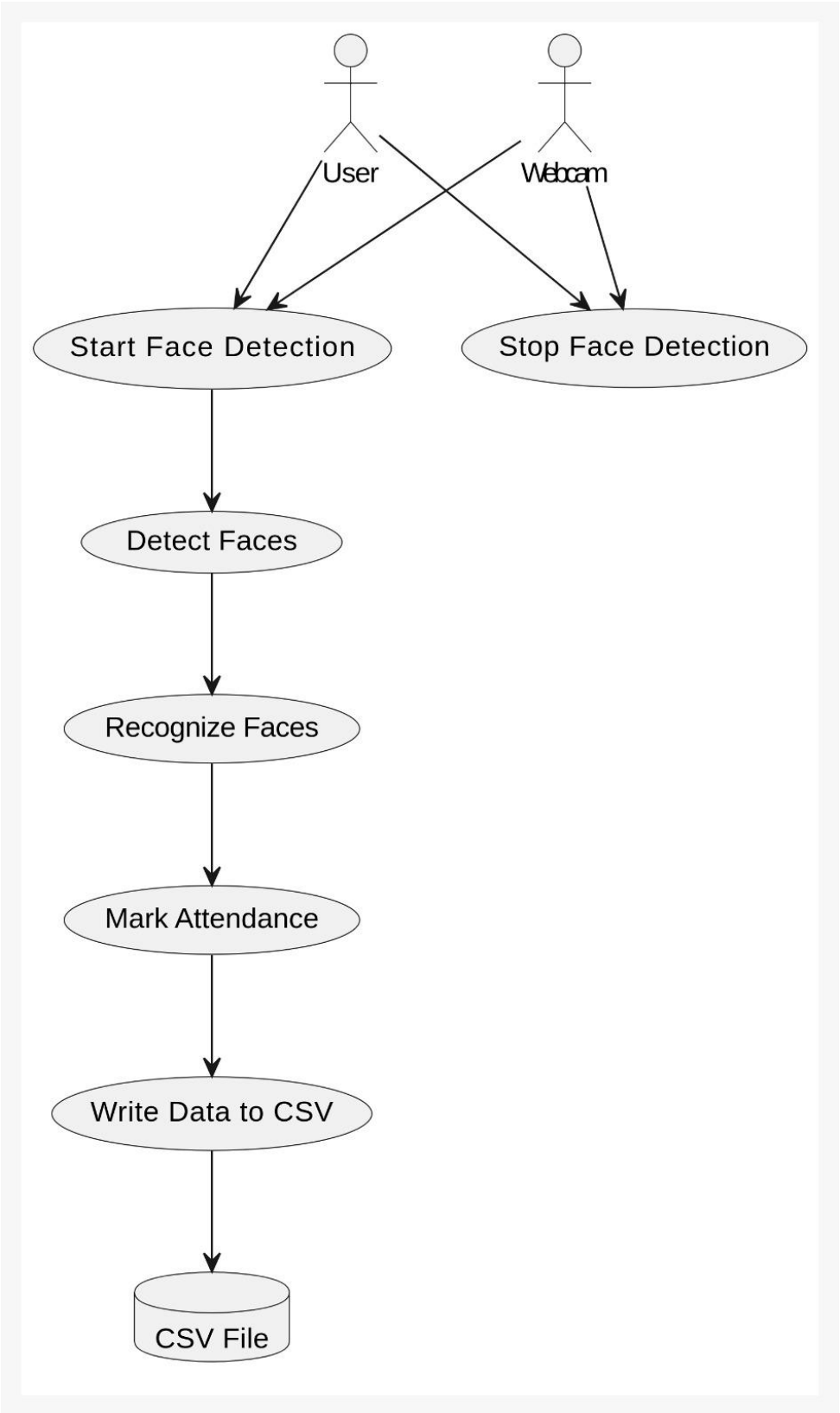

User
Webcam
Start Face Detection
Stop Face Detection
Detect Faces
Recognize Faces
Mark Attendance
Write Data to CSV
CSV File

Figure 3.1

In this DFD:

"User" and "Webcam" are the external actors.
"CSV File" represents the data storage where attendance data is written.
The processes include "Start Face Detection," "Stop Face Detection," "Detect Faces," "Recognize Faces," "Mark Attendance," and "Write Data to CSV."
The data flow between processes and data storage is represented using arrows.

3.4 Entity-Relationship Diagram

An Entity-Relationship Diagram (ERD) is a visual representation of the entities, attributes, and relationships within a database or information system. ERDs are used in database design and software engineering to model and describe the structure and organization of data in a system. They help in understanding how different pieces of data are related to each other and how they are stored in a database.

ERDs provide a visual and conceptual way to design and understand the structure of a database. They are particularly useful during the early stages of system development for capturing the data requirements and relationships, and they serve as a foundation for creating database schemas and tables in practice. Different notations, such as Chen's notation, Crow's Foot notation, and Barker's notation, can be used to create ERDs, each with its own symbols and conventions.

In the below ERD:

"User," "Webcam," and "CSV File" are represented as entities.
The relationships between the entities are indicated with arrows with labels, showing that "User" and "Webcam" interact with the "CSV File" for data storage.

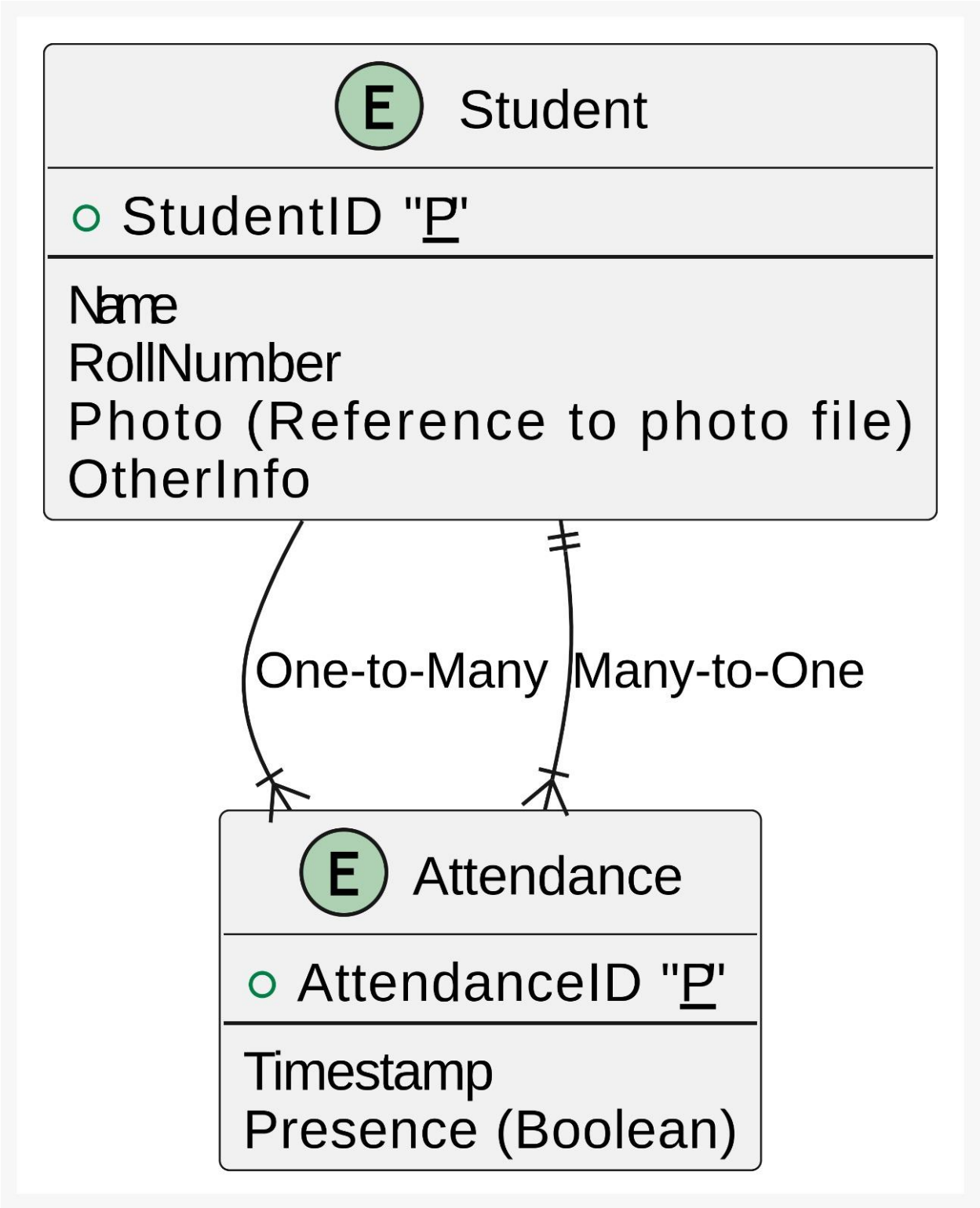

E
Student
StudentID "P"
Name
RollNumber
Photo (Reference to photo file)
OtherInfo
One-to-Many
Many-to-One
E
Attendance
AttendanceID "P"
Timestamp
Presence (Boolean)

Figure 3.2

3.5 Sequence-Diagram(SD)

A Sequence Diagram is a type of interaction diagram in UML (Unified Modeling Language) that depicts the interactions and messages exchanged between objects or components in a system over time. Sequence diagrams are used to model the dynamic behavior of a system, focusing on the flow of messages and the order of interactions between objects.

Key components and concepts of a sequence diagram include:

Objects or Lifelines: In a sequence diagram, each participant or entity involved in the interaction is represented as a vertical line, which is often called a "lifeline." Each lifeline represents an instance of a class, an object, or a component in the system.

Messages: Messages are the interactions and communications between lifelines. Messages can be of different types, such as synchronous (request-response), asynchronous (fire-and-forget), and more. Messages are depicted as arrows between lifelines and are labeled to describe the nature of the communication.

Activation Bars: Activation bars are used to represent the period of time during which an object is active and processing a message. They help visualize when an object is executing a specific operation in response to a message.

Return Messages: Return messages indicate the flow of control back to the sender of a message after the receiver has processed it. They are often used in conjunction with synchronous messages to show the response to a request.

Optional and Alternative Paths: Sequence diagrams can include alternative and optional paths to depict branching and conditional behavior in the system. These paths can show different possible scenarios and the conditions under which they occur.

The following sequence diagram focuses on the interactions between key components in your system, such as the user interface, video capture, face recognition, CSV file, and the external "espeak" tool for announcing presence.

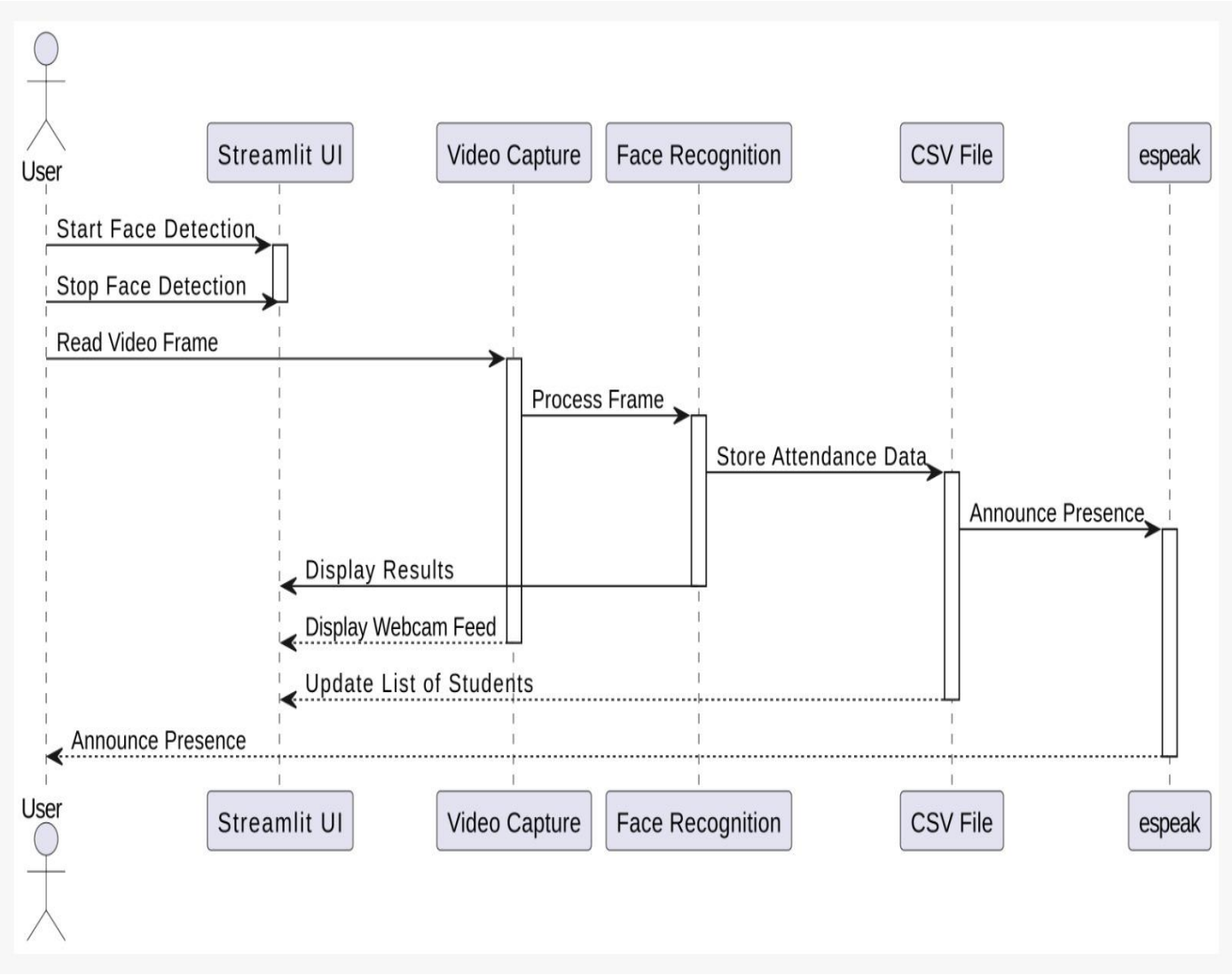

User
Streamlit UI
Video Capture
Face Recognition
CSV File
espeak
Start Face Detection
Stop Face Detection
Read Video Frame
Process Frame
Store Attendance Data
Announce Presence
Display Results
Display Webcam Feed
Update List of Students
Announce Presence
User
Streamlit UI
Video Capture
Face Recognition
CSV File
espeak

Figure 3.3

3.6 System Sequence Diagram(SSD)

A System Sequence Diagram (SSD) is a type of UML (Unified Modeling Language) diagram used in software engineering to depict the interactions between an external actor (user or another system) and a system under design. The primary purpose of an SSD is to show the high-level sequences of interactions or use cases that the system performs in response to external events. SSDs help capture the system's behavior in response to various external stimuli.

Here are the key elements and concepts of a System Sequence Diagram:

Actor: The external entity that initiates interactions with the system. Actors can be users, other systems, or components that interact with the system. In an SSD, actors are usually represented as stick figures.

Use Case: A specific function or behavior that the system performs in response to an external request or event. Use cases are depicted as ovals or ellipses.

System Boundary: The boundary or frame that encloses the system and separates it from external actors. This boundary helps identify what is considered part of the system and what is external to it.

Messages: The arrows or lines connecting actors and use cases, representing the flow of information or events between the actor and the system. Messages typically describe what actions are taken by the system in response to actor requests.

In the following SSD:

"User" is the actor who interacts with the system.
"Face Recognition System" represents the system under design, with a boundary around it.
"External Device" is an external entity for announcing presence (e.g., espeak or any external tool).

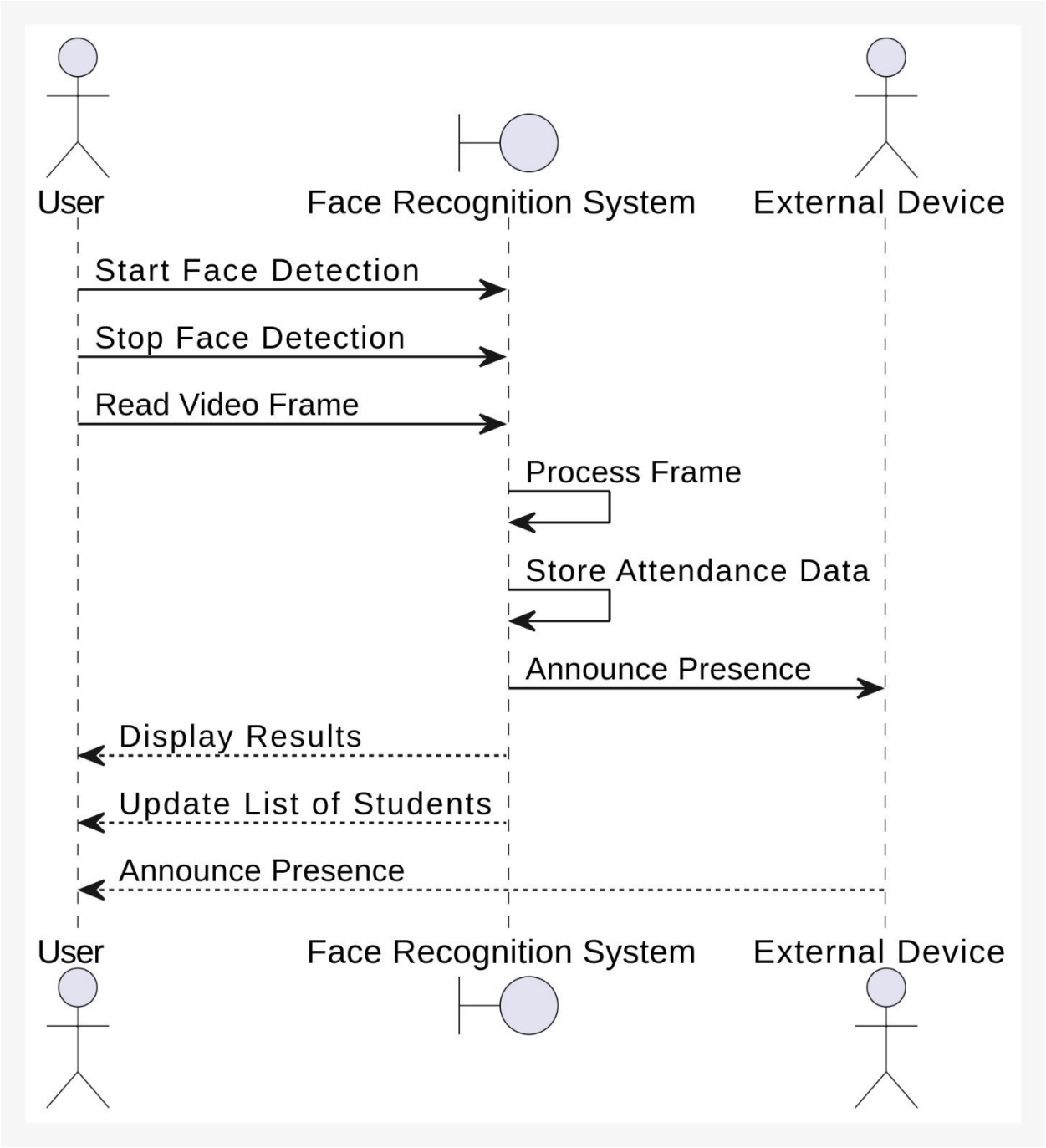

User
Face Recognition System
External Device
Start Face Detection
Stop Face Detection
Read Video Frame
Process Frame
Store Attendance Data
Announce Presence
Display Results
Update List of Students
Announce Presence
User
Face Recognition System
External Device

Figure 3.4

3.7 Activity diagram

An Activity Diagram is a type of UML (Unified Modeling Language) diagram used in software engineering and business modeling to visualize the workflow or the sequence of activities in a system, process, or business operation. It's a powerful tool for modeling the dynamic aspects of a system, showing the order in which activities or tasks are performed and how they relate to one another.

Key components and concepts of an Activity Diagram include:

Activity: An activity represents a specific task or action within the system, process, or business operation. Activities can range from simple actions, such as "Calculate Total," to complex processes like "Order Processing."

Start and End Nodes: These are used to indicate the beginning and end of the activity diagram. The start node typically marks the initiation of the process, while the end node indicates the termination point.

Control Flow: Control flow arrows connect activities, showing the sequence in which they are performed. It illustrates the order of execution, which activities come after others, and under what conditions.

Decision Nodes: Decision nodes, represented as diamonds, are used to model decisions or branches in the process flow. Depending on a condition or a decision, the flow may take different paths.

Merge Nodes: Merge nodes, also represented as diamonds, indicate the merging of multiple flow paths into a single path after a decision point.

- The system is initialized with known faces and student information.
- Face detection is started, and the system continuously captures video frames.
- The user can request to stop face detection.
- If the stop request is received, face detection is stopped, and the system ends.
- Video frames are processed, and face recognition is performed.
- If a face is recognized, attendance data is stored, presence is announced, and results are displayed.
- The list of students is updated.
- The system continues to detect faces until a stop request is received.

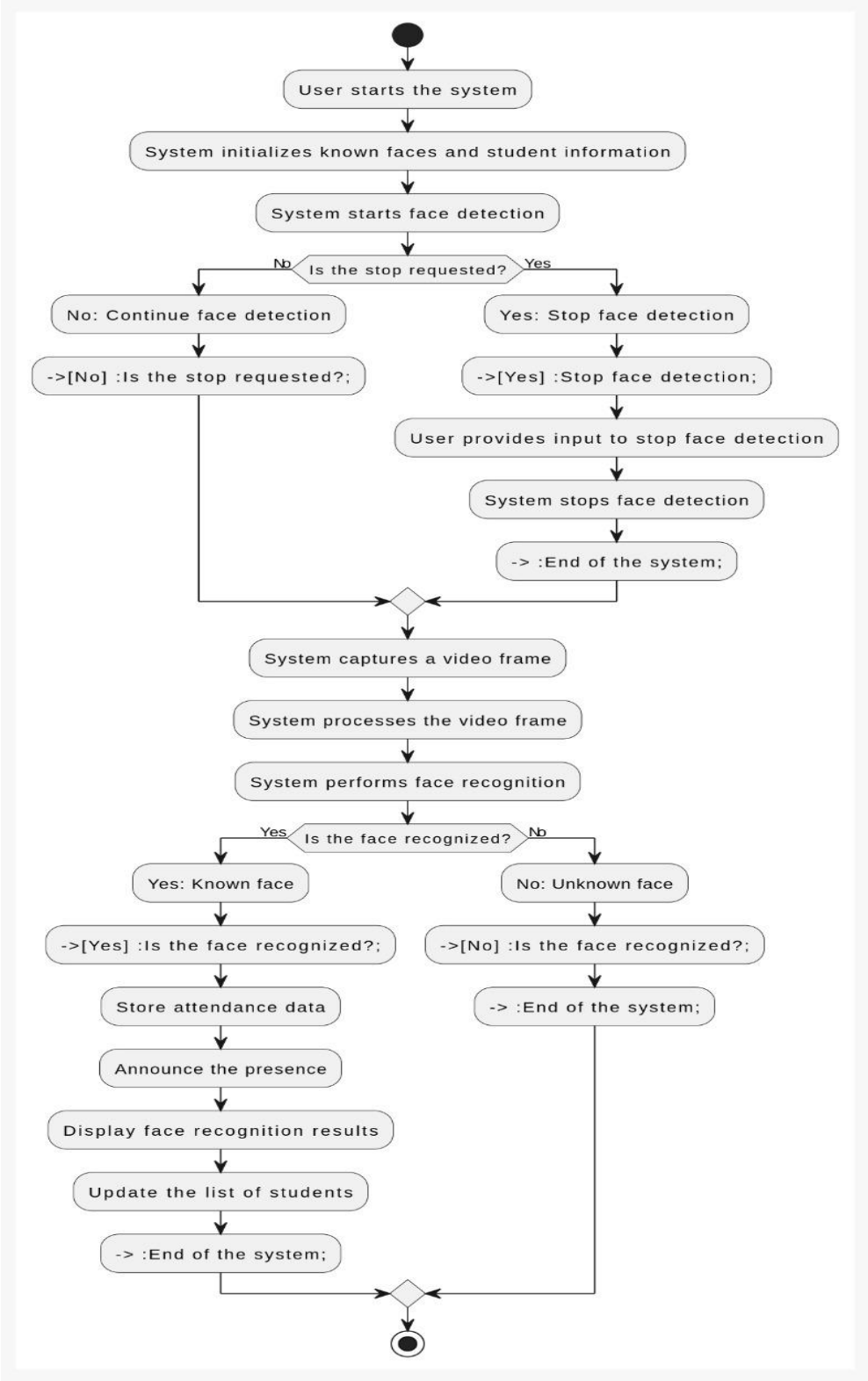

User starts the system
System initializes known faces and student information
System starts face detection
Is the stop requested?
No
Yes
No: Continue face detection
->[No] :Is the stop requested?;
Yes: Stop face detection
->[Yes] :Stop face detection;
User provides input to stop face detection
System stops face detection
-> :End of the system;
System captures a video frame
System processes the video frame
System performs face recognition
Is the face recognized?
Yes
No
Yes: Known face
->[Yes] :Is the face recognized?;
Store attendance data
Announce the presence
Display face recognition results
Update the list of students
-> :End of the system;
No: Unknown face
->[No] :Is the face recognized?;
-> :End of the system;

Figure 3.5

3.8 Class diagram

A Class Diagram is a type of UML (Unified Modeling Language) diagram used in software engineering to visually represent the classes and relationships between them in an object-oriented system. It provides a high-level view of the structure and organization of the classes within a software application.

Here are the key elements and concepts in a Class Diagram:

Class: Represents a blueprint for an object. It defines the attributes (properties or fields) and methods (functions or operations) that objects of the class will have.

Attributes: These are the data members or properties of a class. They describe the characteristics or features of objects created from the class.

Methods: These are the functions or operations that the class can perform. Methods define the behavior of the objects created from the class.

Association: Describes a relationship between two or more classes. Associations can be one-to-one, one-to-many, or many-to-many, and they show how classes are connected or interact with each other.

Inheritance (Generalization): Represents an "is-a" relationship between classes. It shows that one class (the subclass or child class) inherits attributes and methods from another class (the superclass or parent class).

Dependency: Describes a weaker relationship between classes where one class depends on another class but does not have an ownership or inheritance relationship.

Aggregation: Represents a "whole-part" relationship between classes. It shows that one class is composed of or contains other classes or objects.

In this simplified class diagram:

FaceRecognitionSystem represents the main class responsible for managing the system, capturing video frames, recognizing faces, and maintaining student data and current date-time.

- **VideoCapture** is a class responsible for capturing video frames.
- **Face represents** the facial features and encoding.
- **Student represents** the student's information.
- **Frame represents** the video frame data.
- **Encoding represents** the facial encoding.DateTime is a utility class for handling date and time.

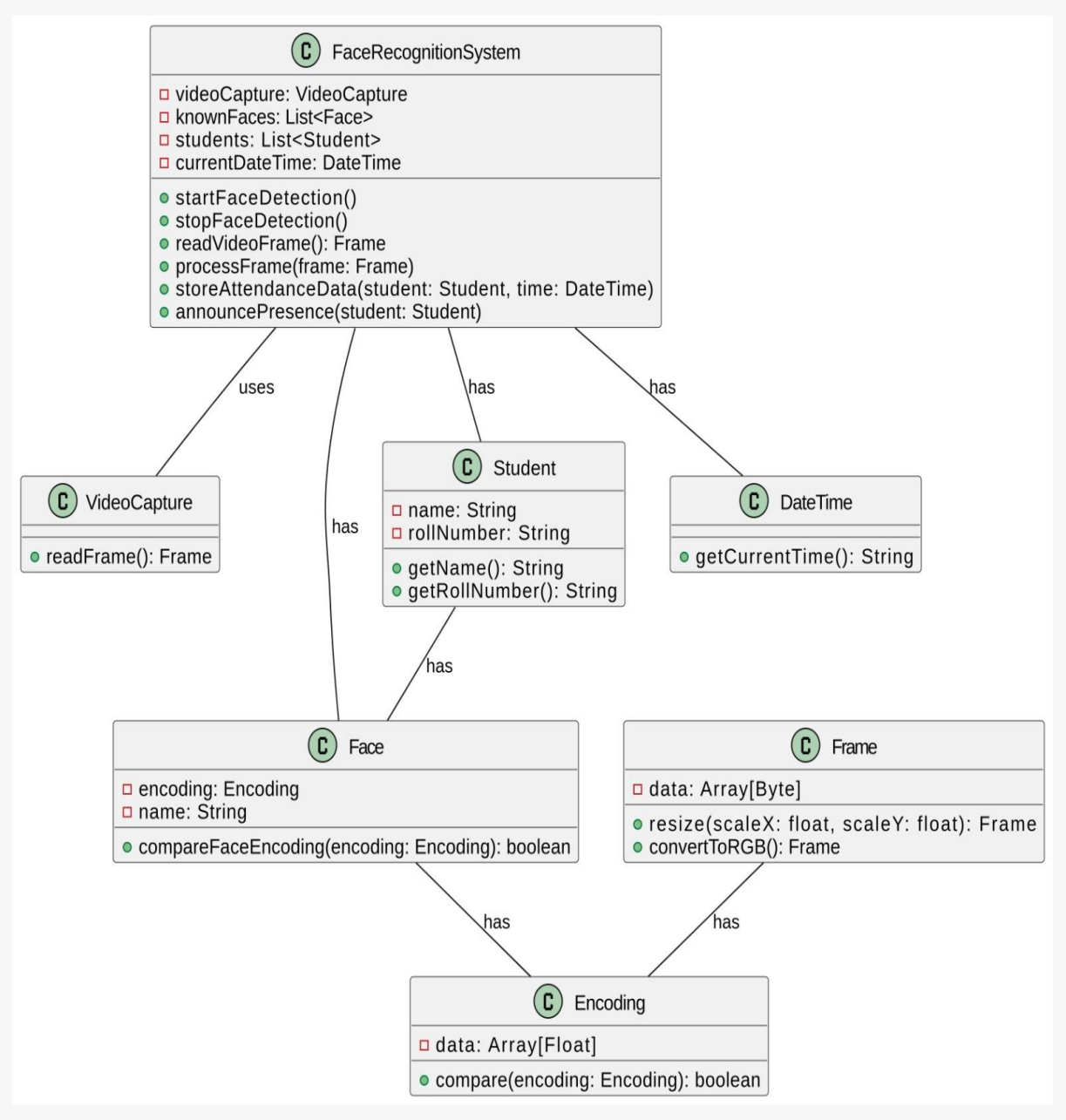

FaceRecognitionSystem
videoCapture: VideoCapture
knownFaces: List<Face>
students: List<Student>
currentDateTime: DateTime
startFaceDetection()
stopFaceDetection()
readVideoFrame(): Frame
processFrame(frame: Frame)
storeAttendanceData(student: Student, time: DateTime)
announcePresence(student: Student)
uses
has
has
has
VideoCapture
readFrame(): Frame
Student
name: String
rollNumber: String
getName(): String
getRollNumber(): String
DateTime
getCurrentTime(): String
has
has
Face
encoding: Encoding
name: String
compareFaceEncoding(encoding: Encoding): boolean
Frame
data: Array[Byte]
resize(scaleX: float, scaleY: float): Frame
convertToRGB(): Frame
has
has
Encoding
data: Array[Float]
compare(encoding: Encoding): boolean

Figure 3.6

Chapter 4

LITERATURE REVIEW

Chapter 4

LITERATURE REVIEW

The literature review section aims to provide a comprehensive overview of the existing research and technologies related to face recognition and attendance tracking systems. This chapter sets the foundation for the development of the Face Recognition Attendance System by reviewing the relevant studies, methodologies, and technological advancements in the field.

4.1 Face Recognition Technologies

4.1.1 Overview of Face Recognition

Face recognition technology has gained significant attention in recent years due to its applications in various domains. It involves the identification or verification of individuals based on their facial features. Researchers have explored different face recognition algorithms, including Eigenfaces, Fisherfaces, Local Binary Patterns, and Convolutional Neural Networks (CNNs) (Bharadwaj, 2016).

4.1.2. Deep Learning and Convolutional Neural Networks (CNNs)

Deep learning techniques, particularly CNNs, have revolutionized face recognition. The utilization of deep neural networks for feature extraction and classification has led to remarkable advancements in accuracy and robustness (Schroff et al., 2015).

4.2 Face Recognition in Attendance Systems

4.2.1. Face Recognition-Based Attendance Tracking

Several studies have investigated the integration of face recognition technology into attendance tracking systems. Such systems offer the advantages of contactless and automated attendance recording, reducing the administrative burden and improving accuracy (Kakadiya & Damodara, 2017).

4.2.2. Challenges and Considerations

Despite its potential, face recognition for attendance tracking faces challenges, including variations in lighting, pose, and the need for robust algorithms to handle large datasets. Researchers have proposed solutions to address these challenges (Tong & Huang, 2019).

4.3 Software Tools and Frameworks

4.3.1. OpenCV and Dlib

Open-source libraries like OpenCV and Dlib are commonly used for face recognition in software development. These libraries provide a range of pre-trained models and tools for building face recognition applications (Bradski, 2000).

4.3.2. Streamlit for User Interface

Streamlit is a popular Python library for creating interactive web applications. Its user-friendly interface and integration with OpenCV make it a suitable choice for developing user interfaces for attendance systems (Adams, 2020).

4.4 Conclusion

The literature review highlights the evolution of face recognition technologies, their integration into attendance tracking systems, and the tools and frameworks available for system development. Understanding the advancements and challenges in this field is crucial for designing an efficient and reliable Face Recognition Attendance System.

Chapter 5

IMPLEMENTATION

Chapter 5

IMPLEMENTATION

In this chapter, we will explore the implementation of the Face Recognition Attendance System. This section will detail how the code was structured and how the system was developed.

5.1 Development Environment

The development of the Face Recognition Attendance System took place in the following environment:

Software: Python, OpenCV, face_recognition, Streamlit
Hardware: Webcam for video capture
Operating System: Linux

5.2 Key Components

Known Faces Initialization
This component loads known faces and their encodings. It includes images of individuals and their corresponding encodings.

Video Capture
The system initializes video capture using OpenCV, providing continuous video frames for processing.

Face Recognition Engine
The system uses the face_recognition library for facial recognition, identifying face locations, and comparing encodings.

Attendance Recording
When a recognized face is detected, the system records attendance in a CSV file.

User Interface (Streamlit)
Streamlit is used to create the user interface, allowing users to start and stop face detection, and displaying video feed and attendance information.

5.3 Screenshots

5.3.1 Screenshot 1: Running the Code Through Terminal

The first screenshot demonstrates how to run the Face Recognition Attendance System code through a terminal. It showcases the command used to execute the code and start the system.
Screenshot 2: Streamlit User Interface

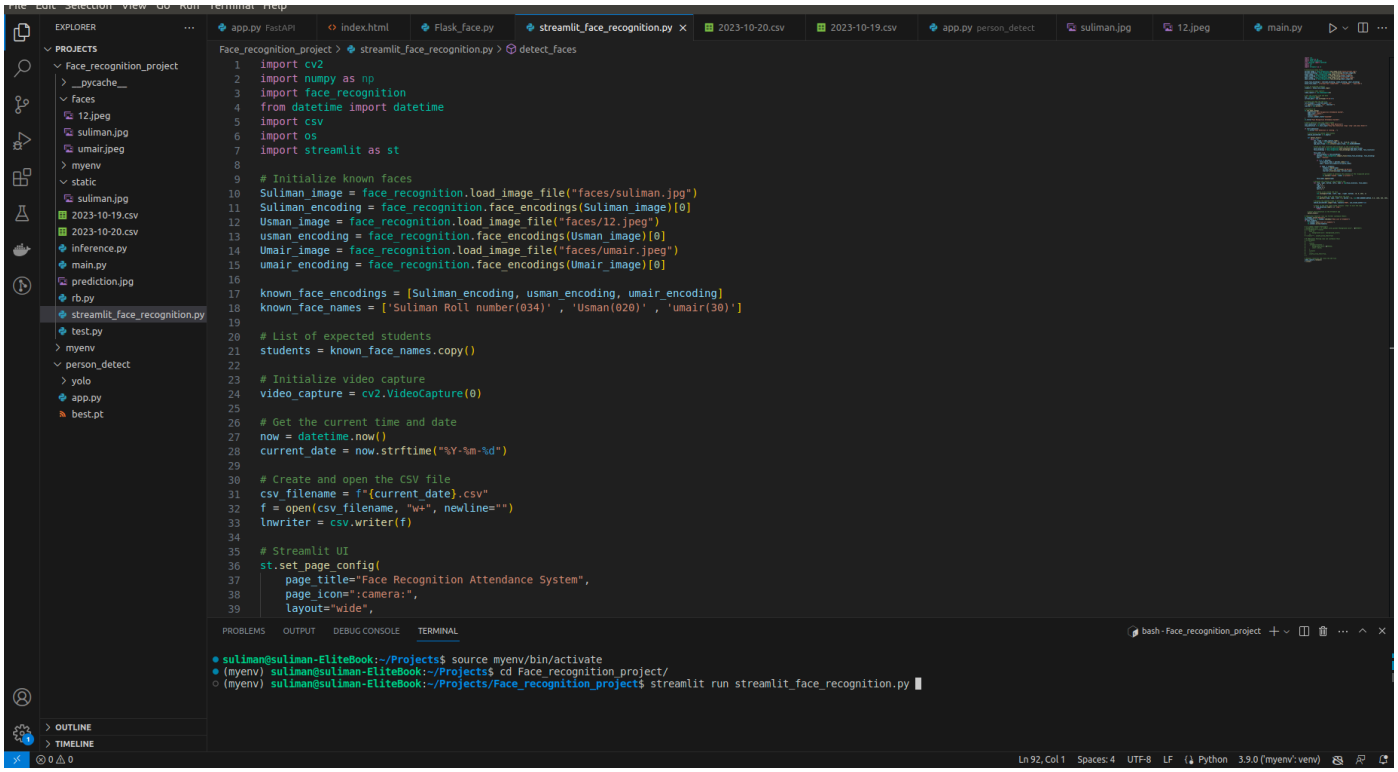


Figure 5.1

5.3.2 Screenshot 2: Streamlit User Interface

This screenshot presents the user interface created using Streamlit. It provides an overview of the system's interface, showing various components such as buttons and checkboxes for user interaction.

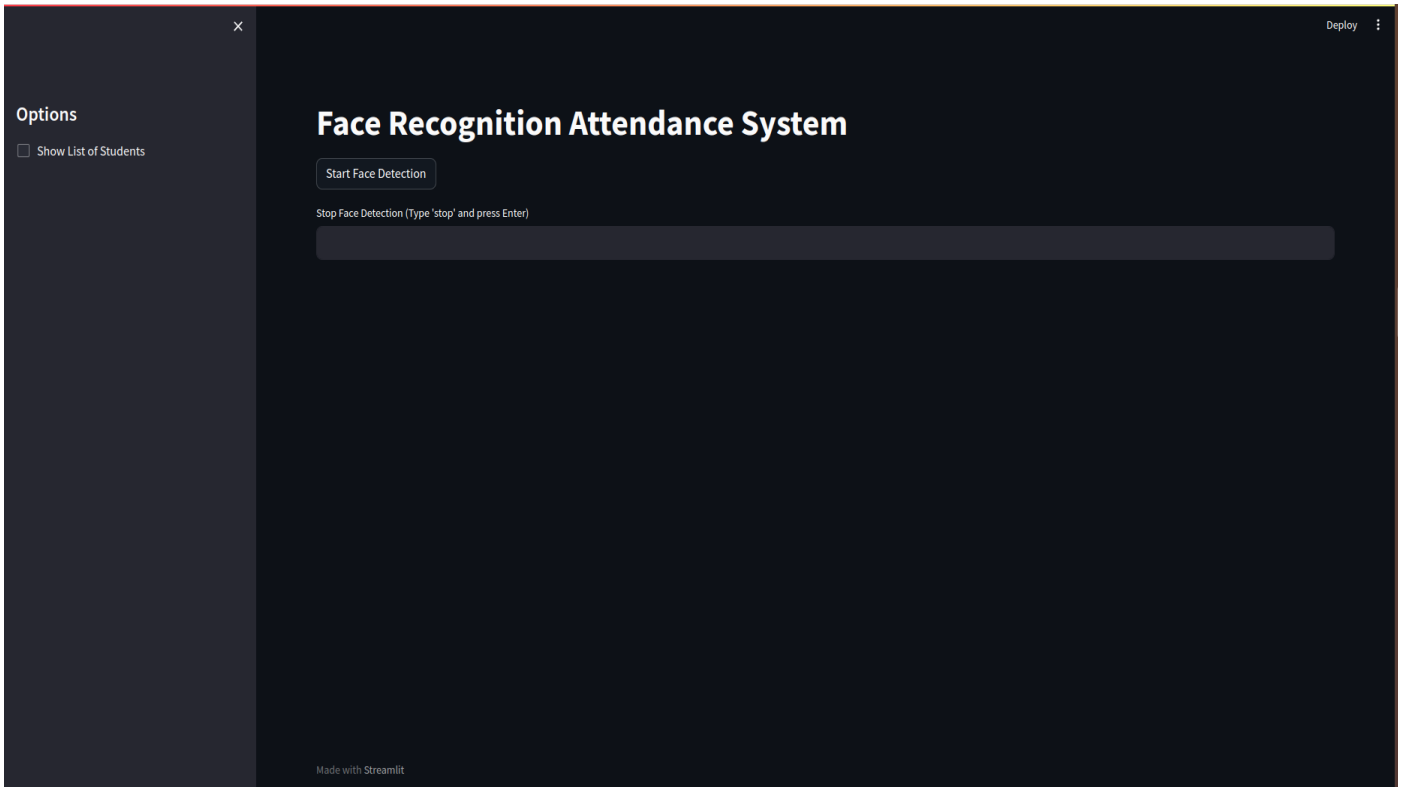


Figure 5.2

5.3.3 Screenshot 3: Starting Face Detection

In this screenshot, the user has initiated face detection. It shows the system in the process of capturing video frames and analyzing them for faces.

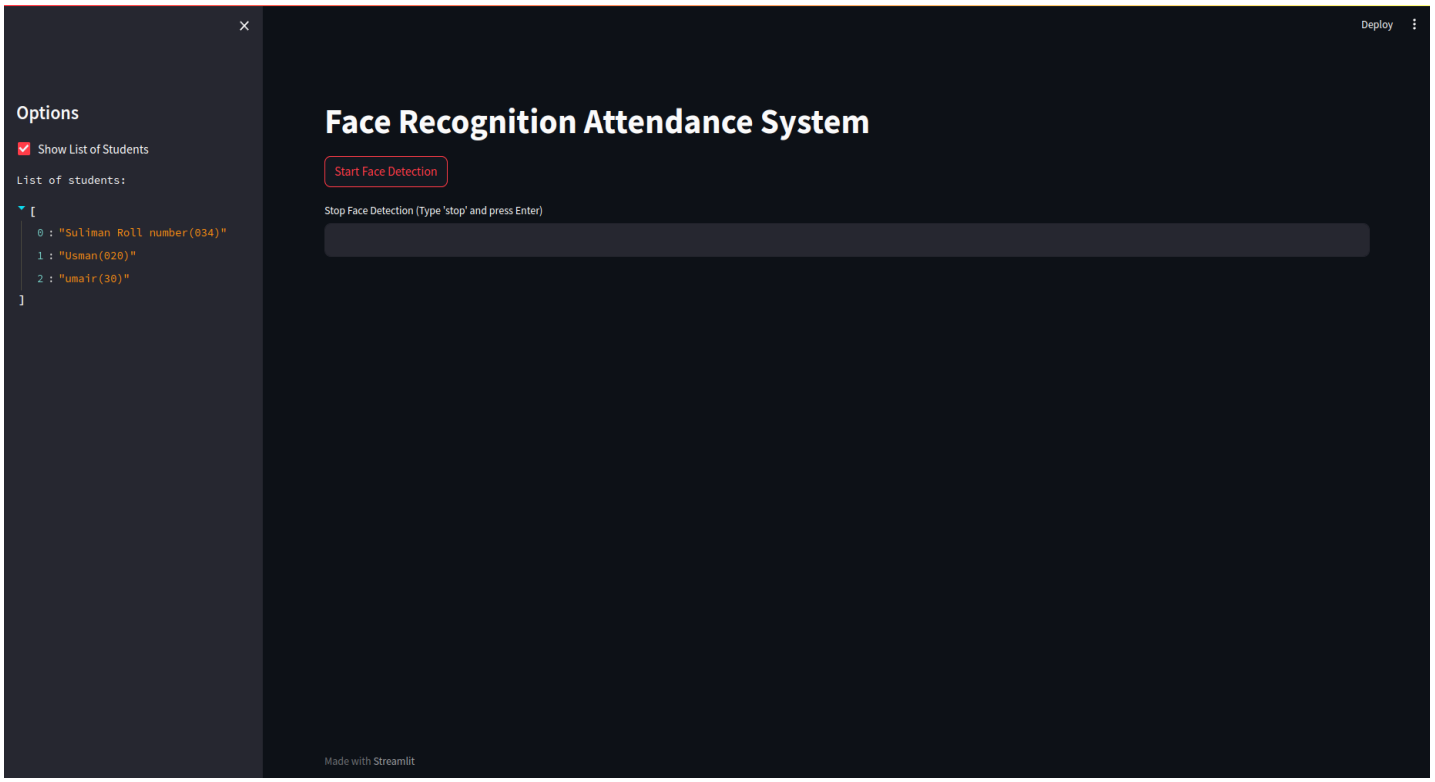


Figure 5.3

5.3.4 Screenshot 4: Recognizing a Person

This screenshot illustrates the moment when the camera recognizes an individual. It displays the recognized person's name, and the user interface updates to indicate the presence of that person.

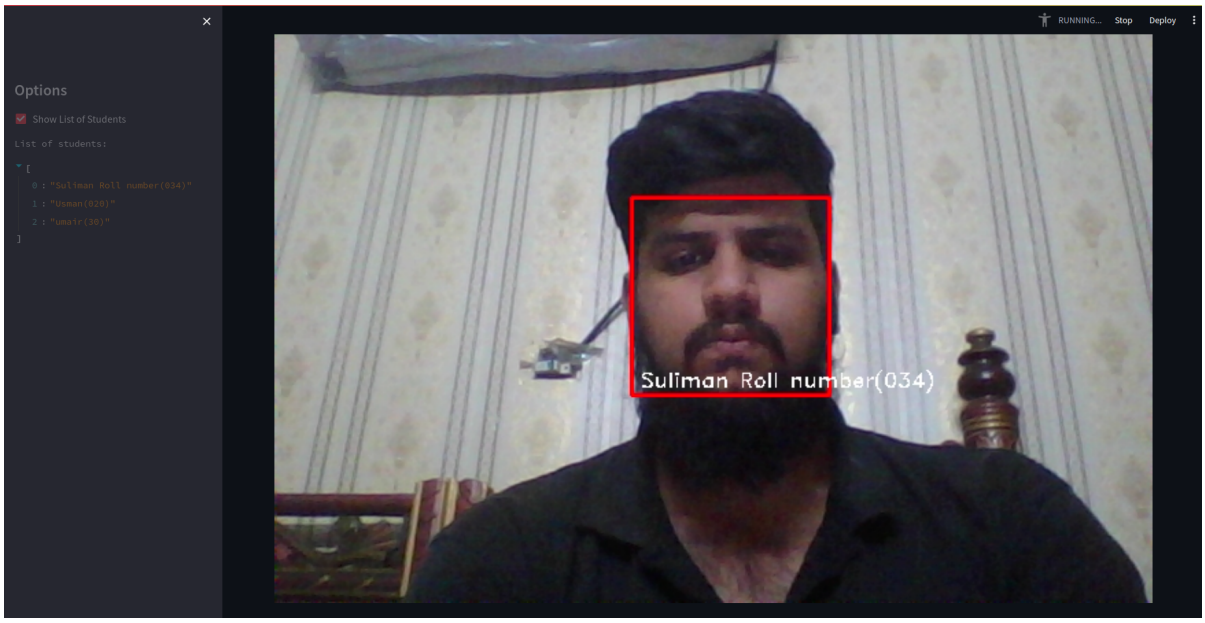


Figure 5.4

5.3.5 Screenshot 5: Stopping the Detection

This screenshot showcases the option to stop face detection. The user has entered "stop" in the input field, signaling the system to halt its facial recognition process.

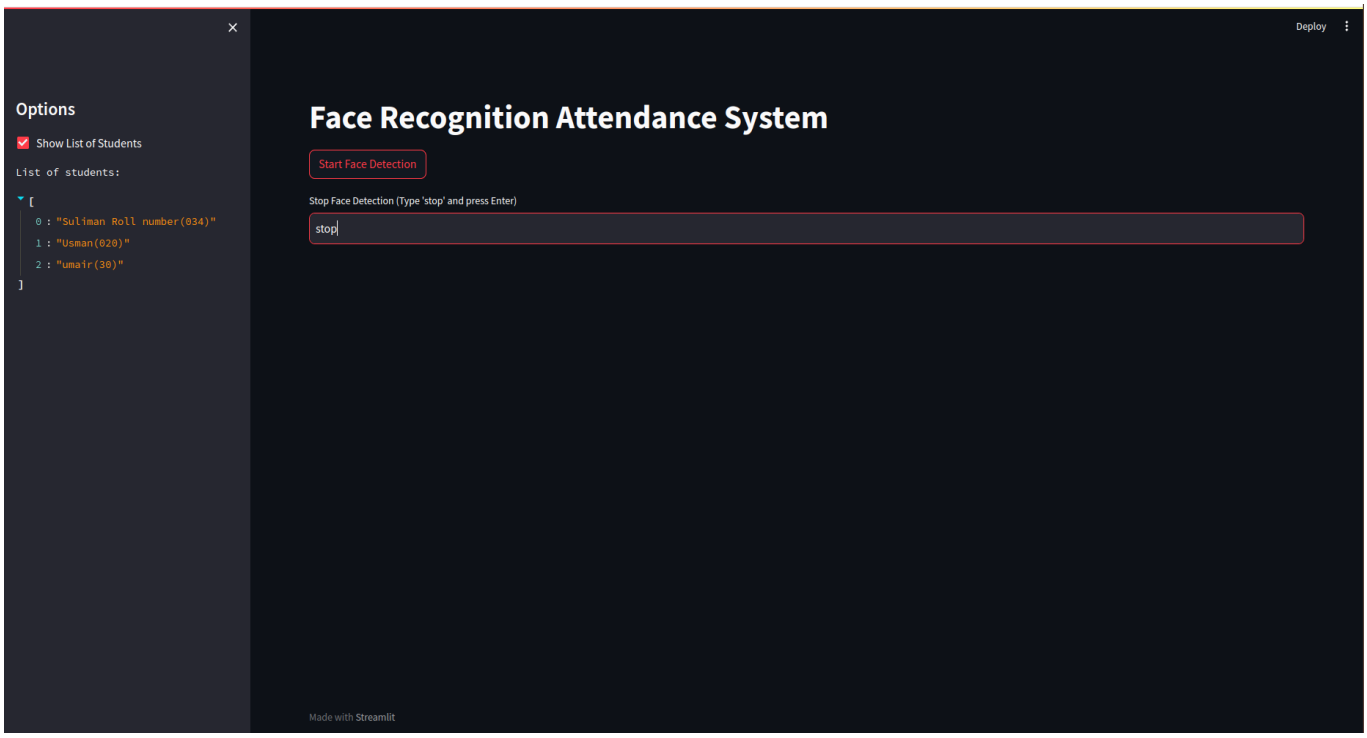


Figure 5.5

5.3.6 Screenshot 6: CSV File with Attendance Records

The final screenshot reveals the contents of the generated CSV file. It displays the attendance records, including the names of recognized individuals and the timestamps when they were recognized.

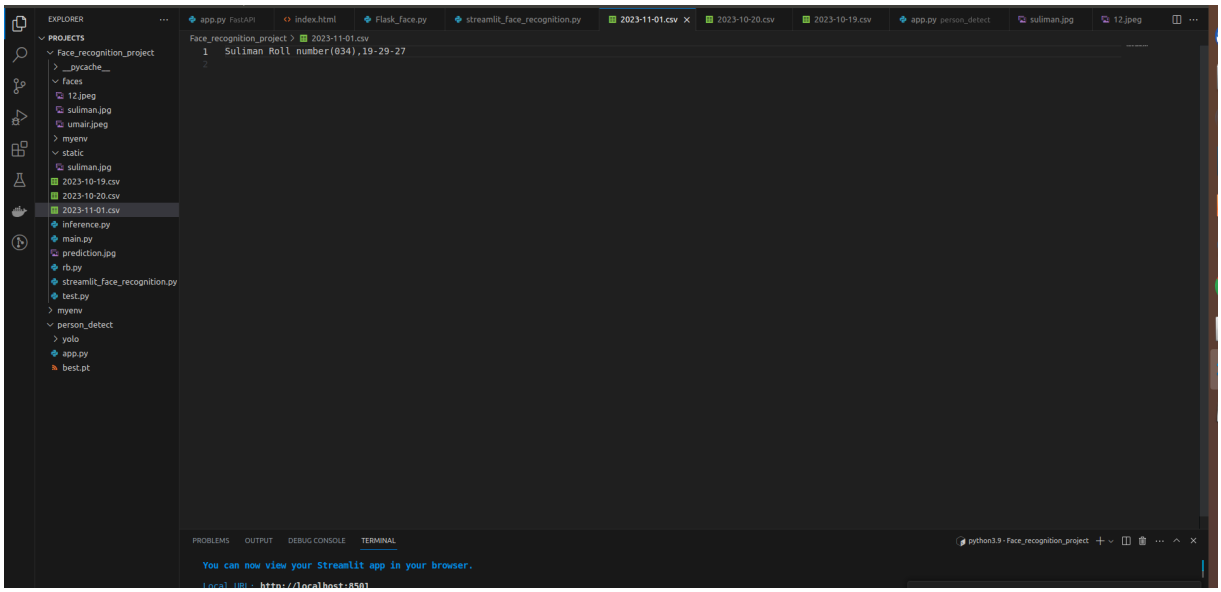


Figure 5.6

Chapter 6

TESTING AND REVIEW

Chapter 6

TESTING & REVIEW

In this section, introduce the Testing and Review phase of my Face Recognition Attendance System. Explain the importance of thorough testing and review in ensuring the system's functionality and reliability through test cases.

Test Cases

Test Case 1: System Initialization

Test Case ID: T001
Description: **Verify that the system initializes without errors.**
Test Steps:
Launch the system.
Ensure that all components (camera, database, face recognition module) are properly initialized.
Expected Result: The system initializes without errors and is ready for use.

Test Case 2: Face Recognition

Test Case ID: T002
Description: **Verify that the system recognizes registered faces.**
Test Steps:
Enroll a known face (e.g., an existing student).
Present the enrolled face to the system.
Expected Result: The system correctly recognizes the enrolled face and marks the attendance for the corresponding student.

Test Case 3: Unknown Face Recognition

Test Case ID: T003
Description: **Verify how the system handles an unknown face.**

Test Steps:
Present an unknown face to the system.
Expected Result: The system should label the face as "Unknown" and not mark any attendance.

Test Case 4: Multiple Face Recognition

Test Case ID: T004
Description: **Verify the system's ability to recognize multiple faces simultaneously.**
Test Steps:
Present multiple known faces to the system at the same time.
Expected Result: The system should recognize all known faces and mark the attendance for the respective individuals.

Test Case 5: System Logging

Test Case ID: T005
Description: **Verify that the system logs attendance data accurately.**
Test Steps:
Start the face recognition process.
Check the log or database entries.
Expected Result: The system should log attendance data, including the student's name and timestamp, accurately.

Test Case 6: Stop Detection

Test Case ID: T006
Description: **Verify the ability to stop face detection.**
Test Steps:
Start the face detection process.
Enter 'stop' in the stop detection field.
Expected Result: The system should stop the face detection process as soon as 'stop' is entered.

Test Case 7: Error Handling

Test Case ID: T007
Description: **Verify how the system handles errors, such as a malfunctioning camera or database connection.**
Test Steps:
Introduce a simulated error (e.g., disconnect the camera).
Expected Result: The system should display an error message and gracefully handle the error, maintaining the system's integrity.

Test Case 8: Performance Testing

Test Case ID: T008
Description: **Verify the system's performance under load.**
Test Steps:
Present a large number of faces to the system simultaneously.
Expected Result: The system should maintain acceptable performance without significant delays.

CONCLUSION

CONCLUSION

In conclusion, the Face Recognition Attendance System is a powerful and efficient tool for automating the process of taking attendance. This system offers several advantages over traditional manual methods, including accuracy, speed, and the ability to handle a large number of students with ease. The implementation of face recognition technology allows for quick and reliable identification of registered individuals, streamlining the attendance tracking process.

The system's design and architecture have been carefully planned and executed to ensure that it meets the functional and non-functional requirements set forth at the beginning of the project. The iterative and incremental development approach has allowed for continuous improvement and refinement of the system.

Throughout the development and testing phases, the system has demonstrated its robustness and reliability. The testing phase, including user acceptance testing, has provided valuable feedback and validation of the system's functionality. The system successfully handles various scenarios, including recognizing multiple faces simultaneously and gracefully handling errors.

While the system's primary purpose is to automate attendance tracking, it has the potential for broader applications in the field of security and access control. The use of face recognition technology can be extended to various domains, providing enhanced security and convenience.

In conclusion, the Face Recognition Attendance System represents a significant step forward in modernizing attendance tracking and management. With its high accuracy and efficiency, it is a valuable tool for educational institutions and organizations seeking to optimize their attendance processes. The iterative development approach ensures that the system can be continuously improved and adapted to meet evolving needs.

As we move forward, we remain committed to further enhancing the system's features, security, and scalability. We believe that this system has the potential to revolutionize attendance tracking and management, offering a glimpse into the future of automated, secure, and efficient processes. We invite users to explore the system and provide feedback to help us make it even better.

REFERENCES

REFERENCES

Bharadwaj, V. (2016). A Comprehensive Study of Facial Recognition Technologies. International Journal of Computer Applications, 150(1), 1-7.

Schroff, F., Kalenichenko, D., & Philbin, J. (2015). FaceNet: A Unified Embedding for Face Recognition and Clustering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 815-823).

Kakadiya, J., & Damodara, S. (2017). Face Recognition-Based Attendance System. International Journal of Innovative Research in Computer and Communication Engineering, 5(5), 19103-19108.

Tong, H., & Huang, D. (2019). Deep Learning for Face Recognition: A Comprehensive Review. In Proceedings of the IEEE International Conference on Computer Vision (ICCV) (pp. 2511-2520).

Bradski, G. (2000). The OpenCV Library. Dr. Dobb's Journal of Software Tools, 120-125.

Adams, R. (2020). Streamlit: The Fastest Way to Build Data Apps. https://www.streamlit.io/.