CHAPTER THREE

LIFE CYCLE DESIGN OF PROPOSED SYSTEM

3.1 Introduction

The methodology is a key part in any key project, where we decide the ways and means, and then the technologies with which we want to build our projects. This chapter defines the system Architecture design process, development techniques, the functional and non-functional requirements, and a valid and vivid visual representation of the system and its use cases in the form of all sorts of diagrams. Kothari (2004) Emphasizes that methodology is "the systematic way to solve a problem," including the logic behind methods. The methodology of this research paper aims to systematically outline the adopted approach in the development of the attendance system tailored for the Kantanka Financial Co-operative Society. In this context, the chosen methodology is based mainly on the iterative approach to software development. This methodology proved suitable due to its strong emphasis on user-centered design, and the provided ability for stakeholders to collaborate with the development team ensuring flexibility to accommodate and execute evolving system requirements and feedback gathered during development cycles.

The iterative development approach promotes a process that is adaptive and dynamic by incorporating continuous cycles of planning, design, testing and implementation. Rather than following a rigid linear path, these iterations build upon the previous one hence allowing for enhancements that are informed by user interactions, system performance evaluations, and evolving organizational needs. This process ensures that the final system aligns with the operational requirements of the Kantaka Financial Co-operative Society, ultimately delivering a solution that is functional, user friendly, and reliable.

By providing a detailed evaluation of each phase, beginning with the initial planning and requirement analysis, followed by the user interface and system architecture design, the facial recognition deep learning model integration, and extensive testing, this methodology section aims to deliver a detailed account of the development process. It highlights how the application of an iterative model facilitates the creation of a robust, scalable, and efficient facial recognition attendance system capable of operating seamlessly in real-time within the organizational context of the Kantanka Financial Co-Operative Society.

3.2 CRYSTALIZATION OF THE PROBLEM

In the context of our project, the core problem centers around the persistent inefficiencies and limitations associated with traditional attendance management systems, particularly within professional environments such as the Kantanka Financial Co-Operative Society. Manual attendance taking methods, whether through paper-based logs or digital check-ins, are often prone to human error, and susceptible to manipulation. These challenges not only reduce productivity but also tend to compromise the security and integrity of attendance records, which are critical for effective management of the workforce. Delays caused by long queues during sign-in and sign-out procedures further contribute to employee dissatisfaction and general slowdown of operations.
Given the increasing demand for seamless and reliable attendance solutions, a pressing need for innovative systems that can automate the process while maintaining high levels of accuracy and security presents itself. To address this need, our project proposes the development of a real-time facial recognition attendance system powered by deep learning algorithms. This solution leverages the capabilities of computer vision and artificial intelligence to authenticate identities quickly and accurately, thereby eliminating the bottlenecks of manual attendance systems. By integration this technology into the organizational workflow of Kantanka Financial Co-Operative Society, the system aims to streamline the taking of attendance, enhance operational efficiency, and ensure integrity of attendance data in a secure and time-efficient manner.

3.3 ANALYSIS AND DESIGN OF THE SYSTEM

The analysis and design phase plays a vital role in ensuring that the proposed real-time facial recognition attendance system aligns with the objectives and effectively addresses the attendance challenges faced by the Kantanka Financial Co-Operative Society (KFCS). This phase hence involves a structured flow of activities aimed at translating user needs into a concrete, functional system blueprint. The steps involved in the analysis and design of our proposed system are outlined below.

1.Requirements Gathering:
This initial phase involved engaging with key stakeholders at KFCS, including administrative personnel and end-users, to understand their current attendance procedures, existing pain points or points of struggle, and expectations from an automated process. In addition to stakeholder consultations, research was conducted on existing biometric and facial recognition systems to identify the best practices and the potential limitations. These findings later informed the articulation of both functional requirements,

such as real time face detection, integration of a database system and user role management, and non-functional requirements like system responsiveness, security and ease of use.

2.System Architecture Design:

Based on the identified requirements, the overall architecture of the system was structured to support modularity, scalability and real-time processing. Key components include a facial recognition engine powered by deep learning models, a secure back end for data management, and a front-end interface for interaction and administration. The system adopts a client-server model where facial recognition is performed at the client end, and the results are sent to a centralized server for validation, storage and report generation.

3.Database Design:

To ensure reliable and efficient management of attendance data, our database was designed to include employee records like their names, the facial features and embeddings in general, and timestamps. The choice of a robust Database Management System (DBMS) ensures data integrity, facilitates quick retrieval of records, and supports scalability as the number of users grow. Security features such as encrypted storage and access and access controls were also considered during this phase to safeguard sensitive user data.

4.User Interface Design:

The system's user interface was designed with usability in mind, focusing on simplicity, clarity and functionality. Administrative users can manage records and generate attendance reports, monitor the system status through an exclusive interface while employees interact with a minimal interface that captures and processes their facial data swiftly. UI elements such as icons, color schemes, and responsive layouts were selected to enhance user experience and accessibility across different devices.

5.Technical Design:

This stage focused on defining the technical stack required for the development of the system. Technologies such as python (for backend logic and model implementation), OpenCV (for facial detection and preprocessing), TensorFlow and Keras(which were used in model training), and pyinstaller for packaging and serving the application were selected for their compatibility and community support. Quality assurance strategies, including unit

testing and code review practices, were also established to maintain system readability during development

6.Prototype Development

A functional prototype of the attendance sytem was developed to provide stakeholders with a visual and interactive representation of the final product. This prototype allowed for early-stage validation of the system's core functionalities, such as registration, the face detection function, and the database logging. User feedback collected during this stage was instrumental in identify key usability issued and refining both the interface and system performance before full scale implementation.

3.3.1 System Requirement

In our project, *"Development of a Real-Time Facial Recognition Attendance System for Kantanka Financial Co-operative Society using Deep Learning Algorithms",* sytem requirements play a pivotal role in ensuring that the final solution is robust, effecient and tailored to the organization's specific needs. These requirements form the backbone of the sytem design and serve as a benchmark for validating functionality and performance. System requirements are broadly categorized into two main types:

3.3.2 Functional Requirements

- **Real time facial detection and recognition for attendance logging**: The system must be capable of detecting and identifying faces in teal time, associating them with the registered staff members, and automatically marking their attendance without intervention.

- **Prevent multiple entries for the same individual within a given period**: The system should ensure that an individual cannot be logged in multiple times within a short time frame. A user should not be allowed more than one session

within short time spans. This functionality helps maintain accurate attendance records and prevents redundancy.

- **Differentiate between check-in and check-out times**:
  The system must be able to track both entry and exit times for each employee, which can provide a fair idea of their working hours and generate weekly, or monthly attendance reports.
- **User registration and face enrollment module:**
  The system must provide functionality to register new users and capture their facial data securely. This feature proves essential for onboarding new personnel and updating existing facial data if necessary.

- **Administrator Dashboard for attendance monitoring and reporting:**
  An administrative interface should be available for HR personnel to monitor real time attendace status, generate reports and manage user profiles.

3.3.2 Non-Functional Requirements

Aside from functionality, the system also needs to meet performance, usability, and scalability requirements to work well in real-world situations. These non-functional requirements determine how efficiently and effectively the system operates.

- **Security:**
  The attendance system must implement strong security protocols to protect biometric data and attendance logs. This includes facial data encryption during storage and transmission, multi-level access control to restrict unauthorized usage, and logging mechanisms for system auditing. These measures are essential to maintain data confidentiality, integrity, and prevent identity theft or manipulation of attendance records.
- **Scalability:**
  The system must be scalable to support a growing number of employees and facial data entries without degradation in performance. As Kantanka Financial Co-operative Society expands or integrates the system across multiple branches, the infrastructure should seamlessly adapt to higher loads in both data processing and storage.
- **Performance:**
  The system must provide high-speed performance with minimal lag, ensuring that attendance is logged within seconds of face detection. This is especially critical during peak periods (e.g., start and close of work) when many users are interacting

with the system simultaneously. Efficient background processing and hardware acceleration should be utilized to meet real-time demands.

- **Usability:**
  The system interface should be designed with user experience in mind, offering a simple, clean, and intuitive UI for staff and administrators alike. It should guide users through tasks such as logging in, registering new users, and generating reports, while minimizing the need for technical know-how. Visual indicators (like success/error prompts) and tooltips should enhance interaction.
- **Compatibility:**
  The system should integrate easily with Kantanka's existing employee management systems and database infrastructure. Compatibility with current hardware (cameras, workstations) and software (databases, reporting tools) is crucial for smooth deployment and operation without requiring a complete overhaul of existing assets.
- **Maintainability:**
  The system should be easy to maintain and update. Modular code design, thorough documentation, and clearly separated system components will allow future developers or administrators to troubleshoot or enhance the system with minimal friction.
- **Availability and Reliability:**
  The system must be highly available during all working hours and designed to handle failures gracefully. Features such as automatic data backups, error logging, and fallback mechanisms (e.g., local storage if the network is down) should be incorporated to ensure reliable operation even during unexpected downtimes.

These requirements help create a **reliable, secure, and fair** facial recognition system. By balancing strong functional features with well-defined non-functional goals, the system can perform efficiently and be ready for real-world use.

3.3.4 Hardware Requirements

In this subsection, we outline the hardware requirements necessary for the effective deployment and smooth functioning of our proposed real-time facial recognition attendance system for Kantanka Financial Co-operative Society:

- Processing Power:
  The central processing unit (CPU) is critical to the system's ability to perform real-time facial recognition. Since the system relies on deep learning models and image processing tasks, it must run on machines equipped with high-performance CPUs that can handle multiple threads and parallel processing. This ensures low-latency detection, fast recognition speeds, and responsiveness during peak attendance times. For optimal results, the system should ideally run on processors with multiple cores.

- Memory (RAM):
  Random Access Memory (RAM) plays an important role in maintaining system fluidity during facial detection and recognition. To support real-time video input processing, loading of the machine learning model, and simultaneous attendance logging, the system should operate on devices with 8GB RAM. Higher memory capacities like 16GB RAM are recommended for an enhanced performance and experience, especially as the userbase grows.

- **Graphics Processing Unit (GPU):**
  While not mandatory for all deployments, a dedicated GPU significantly enhances the performance of deep learning inference tasks. For systems expected to handle high-resolution video streams and faster recognition, integration with GPUs such as NVIDIA GTX/RTX series is highly beneficial. This offloads complex computations from the CPU and accelerates model execution times.

- **Secondary Storage:**
  Secondary storage provides persistent data storage for user records, facial image data, attendance logs, model weights, and system backups. Solid State Drives (SSDs) are recommended over traditional Hard Disk Drives (HDDs) for faster read/write operations, which directly impact system boot time and real-time data access. A minimum of 256GB SSD is suggested, with higher capacities allocated based on the volume of registered users and historical data retention policies.
- **Camera Hardware:**
  A high-definition (HD) camera is essential for accurate facial detection and recognition. The camera should support at least 720p resolution, with preference for 1080p or higher for improved image clarity and accuracy, especially in varying lighting conditions. Infrared or night-vision capabilities may be considered for environments with low lighting.

3.3.5 Software Requirements

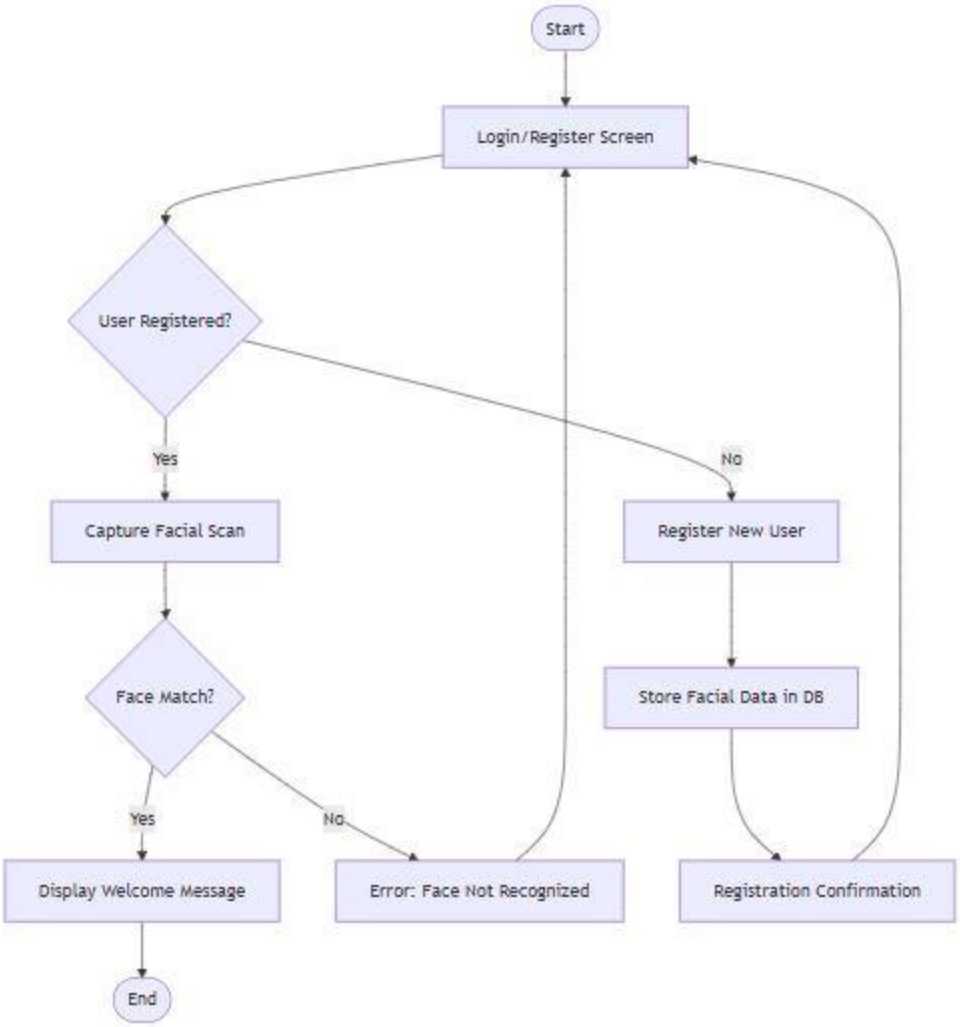
The successful deployment and operation of our **Real-Time Facial Recognition Attendance System** necessitate the installation of specific software components that support deep learning, image processing, and a smooth user interface. Below are the key software requirements tailored to our standalone packaged application:

- **Operating System:**
  The system will be developed to run primarily on the **Windows operating system (Windows 10 and above)**. However, provisions may be made for future cross-platform compatibility with **Linux-based** distributions, depending on the deployment environment. The system should run on a 64-bit OS to leverage maximum memory and processing capabilities.
- **Python Environment:**
  As the core development language for the system is **Python**, the host machine must have Python (version 3.7 or higher) installed. This environment supports the implementation of machine learning models, face recognition modules, and attendance logging functionalities.
- **Deep Learning Libraries:**
  The system will rely on industry-standard libraries such as:
  - **TensorFlow:** loading and executing deep learning models.
  - **OpenCV: real-time face detection and image processing tasks.**
  - **Dlib:** facial landmark detection and embedding generation.
  - **Face Recognition API:** built on Dlib, simplifies face encoding and matching processes.
-
- **Graphical User Interface (GUI) Toolkit:**
  The application will feature a lightweight desktop interface built using a GUI framework such as:
  - **Tkinter:** to provide intuitive interaction for administrators and users during attendance operations, model training, and logging.
- **Visual Studio C++ development package (Visual C++):**
  The system requires the visual C++ package to install dlib and cmake.

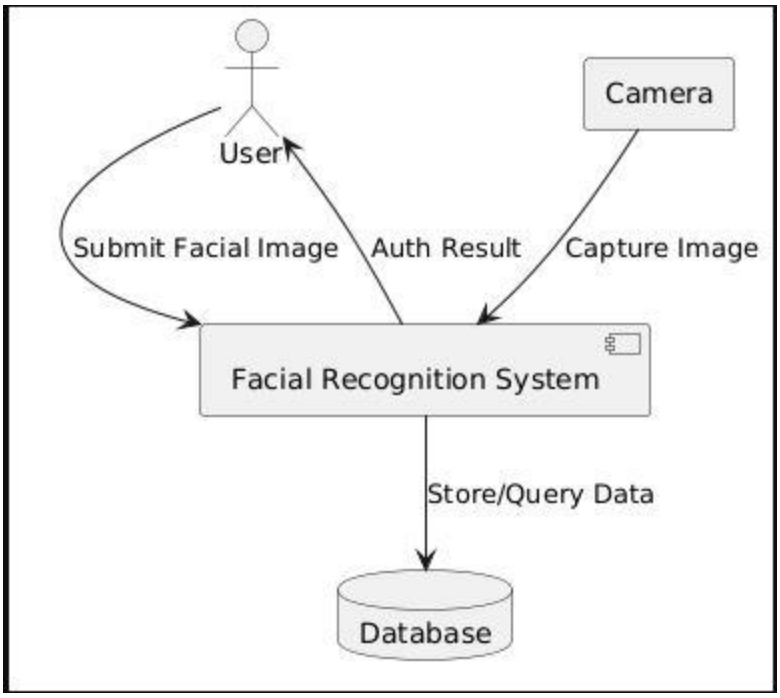
- **Packaging Tools:**
  For converting the Python-based application into a standalone executable file (.exe), the following tools will be used:
    - **PyInstaller** or **cx_Freeze** – to bundle all dependencies and ensure smooth deployment on target machines without requiring end users to manually install packages.

3.5.5 Flowchart

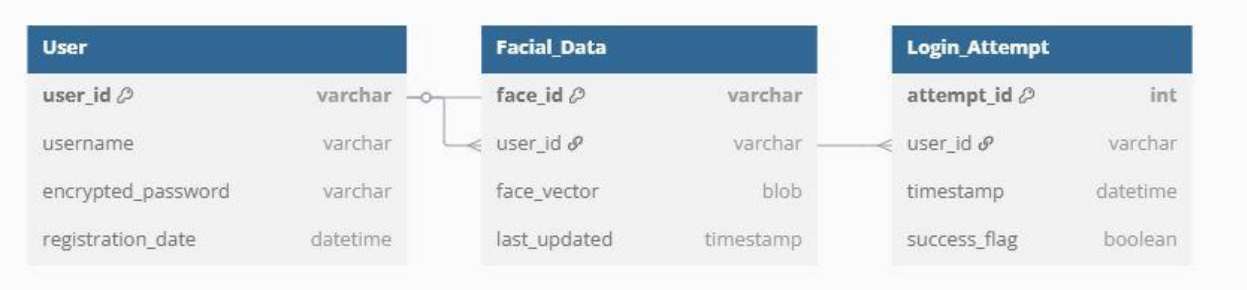
The Flowchart of the proposed System is shown below:



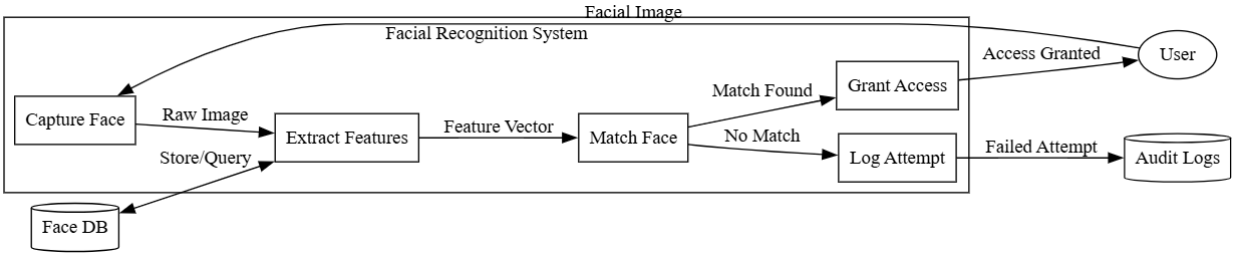
3.5.6 Context Diagram



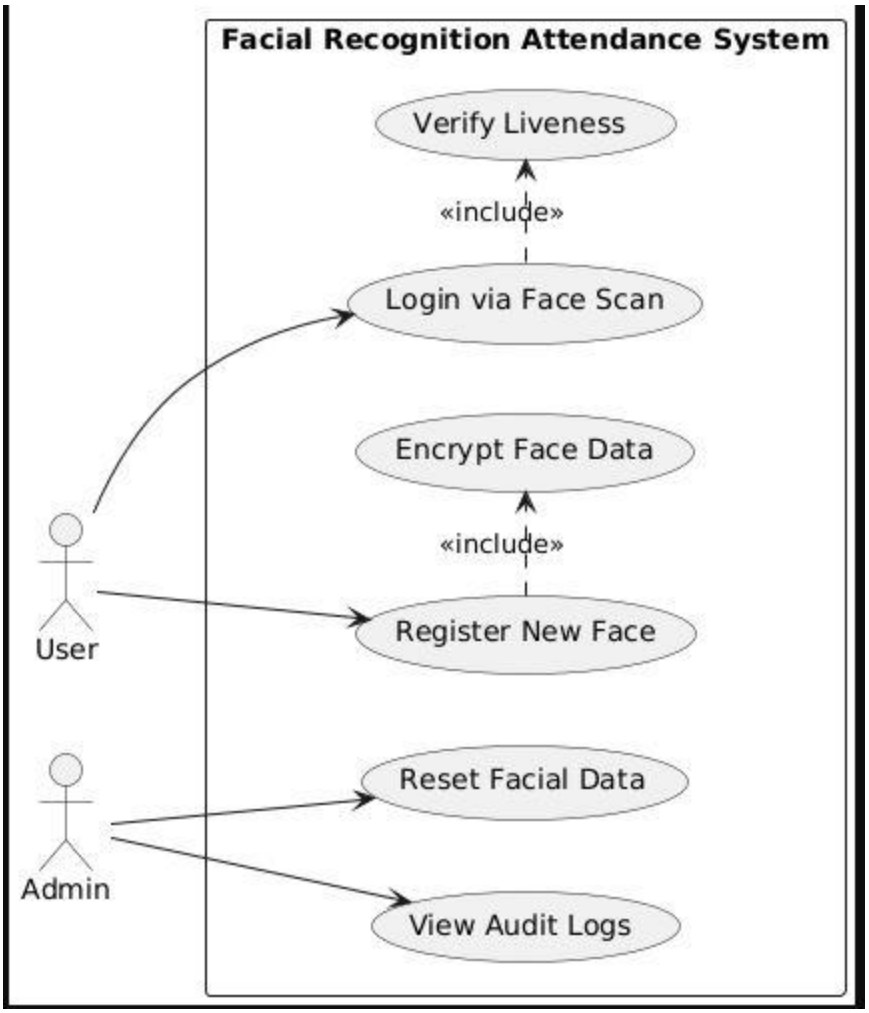
3.5.7 Entity Relationship Diagram.



3.5.8Data Flow



3..5.9 Use Case Diagram



3.5 TOOLS USED

In this subsection, we outline and briefly explain each of the tools that will be used to develop our proposed system as follows:

1. **Device Specifications**
   a) **Laptop Model:** HP EliteBook
   b) **Operating System:** Windows 10
   c) **Processor:** Intel(R) Core i3
   d) **RAM:** 8 GB
   e) **Graphics:** 2 GB Dedicated GPU
   f) **System Type:** 64-bit Operating System

Despite the modest specifications, the machine provided sufficient capability for model experimentation and local development of the packaged application.

2. **Python Programming Language**
   Python served as the core language for both the model logic and overall system functionality. Its extensive libraries like tkinter and pillow assisted in the overall system development

3. **Visual Studio Code (VS Code)**
   VS Code was the primary integrated development environment (IDE) used throughout the development process. Its lightweight nature and support for Python extensions made it ideal for scripting, debugging, and maintaining version control.
4. **Github (Version Control)**
   Github provided a platform for version control and managing different critical features through critical branches.