Project Report On

Attendance Mate

Submitted in the Partial Fulfilment of The Requirement for The
Award Of Degree of Bachelor of Technology
In
Computer Science & Engineering
Batch:2019-23

Supervised By:
Dr. Satinder Kaur
Assistant Professor
Department of Computer Engg. & Technology
Guru Nanak Dev University, Amritsar.

Submitted By:
Apardeep Singh Soni
17031974303
B.Tech 8th Sem

DEPARTMENT OF COMPUTER ENGINEERING AND TECHNOLOGY

GURU NANAK DEV UNIVERSITY, AMRITSAR, INDIA

Certificate

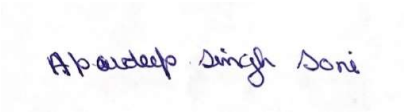
This is to certify that Apardeep Singh Soni, student of B.Tech CSE-8th semester, Guru Nanak Dev University, Amritsar has done project work on topic "Attendance Mate" under my guidance, towards the fulfilment of the award of B.Tech CSE-8th semester and have completed it successfully.

Signature of Guide

Dr. Satinder Kaur

Declaration

I hereby declare that the project work entitled “Attendance Mate” submitted to the Guru Nanak Dev University, Amritsar is a record of an original work done by me under the guidance of Dr. Satinder Kaur, Assistant Professor, Dept. of Computer Engineering and Technology, Guru Nanak Dev University, Amritsar and this project is submitted in the partial fulfilment of requirements for the award of the degree of Bachelor of Technology in computer science and engineering. The results embodied in this report have not been submitted to any other university or Institute for the award of any other degree or diploma.

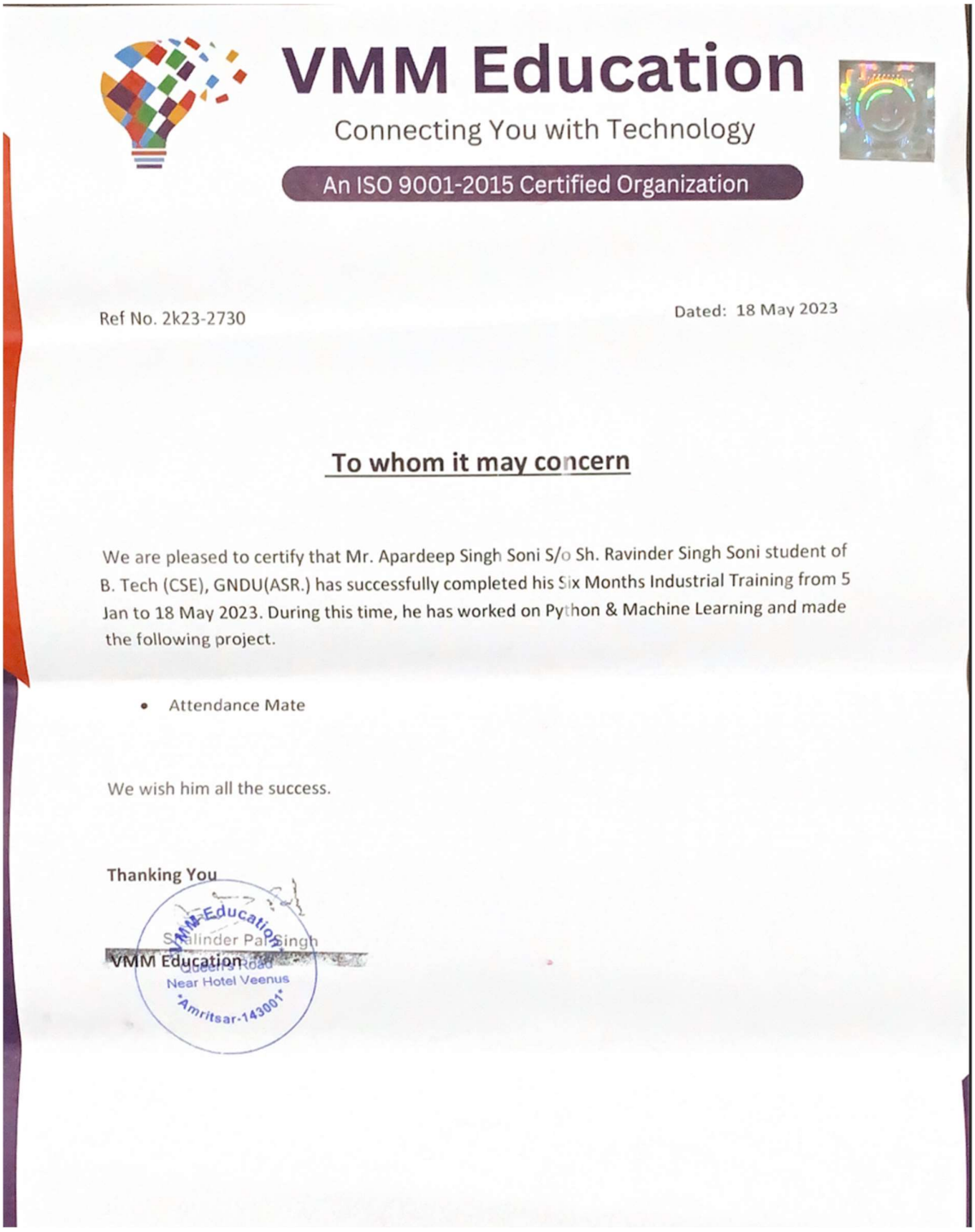


Apardeep Singh Soni
B.Tech (CSE)

Acknowledgement

I would like to express my gratitude to my teacher Dr. Satinder Kaur for providing support and guidance. I got to learn a lot more about this project (How to detect face and recognize from database) which will be very helpful for me.
I would also like to show my gratitude towards my peers for motivating me to work on this project. In the end, I would like to thank my parents. Without them, I would not have been able to complete this project.

Certificate of Company



VMM Education

Connecting You with Technology

An ISO 9001-2015 Certified Organization

Ref No. 2k23-2730

Dated: 18 May 2023

To whom it may concern

We are pleased to certify that Mr. Apardeep Singh Soni S/o Sh. Ravinder Singh Soni student of B. Tech (CSE), GNDU(ASR.) has successfully completed his Six Months Industrial Training from 5 Jan to 18 May 2023. During this time, he has worked on Python & Machine Learning and made the following project.

- Attendance Mate

We wish him all the success.

Thanking You

VMM Education

VMM Education
Near Hotel Veenus
Amritsar-143001

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction

The traditional attendance system that uses manual registers or ID cards has several drawbacks such as the possibility of human error, fraudulent practices, and time-consuming tasks. Therefore, there is a need to develop a more efficient and accurate attendance system that uses face recognition technology to streamline the process. A Face Attendance System is a modern technological solution that is used to record the attendance of employees, students, or any other individual by capturing their facial features. It is an advanced version of the traditional attendance system that was previously used in organizations and educational institutions. The Face Attendance System uses facial recognition technology to identify an individual and mark their attendance. It involves capturing an image of a person's face and using algorithms to detect and match the facial features against a database. Once the system has successfully identified an individual, it automatically records their attendance.a

Chapter 2: Problem Statement

Attendance tracking is a crucial aspect of managing any organization, but the use of traditional attendance systems has several limitations that can lead to errors and inefficiencies in tracking attendance data. Let's take a look at some of the problems associated with these systems:

1. Traditional attendance systems, such as paper registers or ID card swipes, are often inaccurate and prone to errors, leading to incorrect attendance tracking and loss of productivity.
2. Manual attendance tracking can be time-consuming and labour-intensive, requiring administrative staff to input and manage data manually.
3. Traditional attendance systems can be easily manipulated, allowing employees to manipulate attendance records or engage in buddy punching, where one employee clocks in for another.
4. The use of ID cards can also lead to loss or theft, leading to a security risk and potential for attendance fraud.
5. Traditional attendance systems are also limited in terms of data analysis, as they provide only basic information on attendance and do not allow for more in-depth analysis.
6. There is a lack of real-time data with traditional attendance systems, making it difficult for managers to make informed decisions about staffing and workload allocation.
7. With the ongoing COVID-19 pandemic, traditional attendance systems can pose a risk of infection transmission through shared equipment such as fingerprint scanners or ID card readers.

2.1 Objectives

The objective of the "Attendance Mate" is to develop an automated attendance tracking and management system that uses face recognition technology to capture attendance accurately and efficiently in real-time. The project will aim to address the challenges faced by traditional attendance systems, such as human error, fraudulent practices, and time-consuming processes, and provide a more reliable and secure solution for attendance management.

To achieve this objective, the project will focus on the following objectives:

- Utilize face recognition technology to capture attendance in real-time.
- Develop a user-friendly and scalable system that can detect faces, match them with an existing database, and mark attendance automatically.
- Ensure the privacy and security of the attendance data by storing it securely and making it accessible only to authorized personnel.
- Handle large numbers of users while improving accuracy and reliability in attendance tracking.
- Reduce the time and effort required for attendance tracking.
- Provide an effective attendance management solution for academic institutions, businesses, and other organizations.

Chapter 3: Methodologies Used

3.1 Introduction to Data Analysis

Data analysis is the process of inspecting, transforming, and interpreting raw data to extract meaningful insights, draw conclusions, and support decision-making. It involves using various techniques and methods to uncover patterns, trends, and relationships within the data. The purpose of data analysis is to gain a deeper understanding of the information contained in the data and to derive actionable insights that can inform business strategies, scientific research, or problem-solving. It plays a crucial role in fields such as business, finance, marketing, healthcare, social sciences, and many others.

Types of data:

Every kind of data has a rare quality of describing things after assigning a specific value to it. For analysis, you need to organise these values, processed and presented in a given context, to make it useful. Data can be in different forms; here are the primary data types.

· Qualitative data**:** When the data presented has words and descriptions, then we call it qualitative data.

· Quantitative data**:** Any data expressed in numbers of numerical figures are called quantitative data. This type of data can be distinguished into categories, grouped, measured, calculated, or ranked.

· Categorical data**:** It is data presented in groups. However, an item included in the categorical data cannot belong to more than one group.

Qualitative Data analysis

Qualitative data analysis refers to the process of examining and interpreting non-numerical data to derive meaningful insights and understanding. It involves systematically organizing, categorizing, and analyzing textual, visual, or audio data to identify patterns, themes, and relationships.

Finding patterns in the qualitative data

Although there are several ways to find patterns in the textual information, a word-based method is the most relied and widely used global technique in which researchers read the available data and find repetitive or commonly used words. The keyword context is widely used word-based technique. Other techniques - Scrutiny-based technique, Compare and contrast, Variable Partitioning

Methods used for analysis in qualitative data

- **Content Analysis:** used to analyse the documented information from text, images, and sometimes from the physical items.
- **Narrative Analysis:** used to analyse content gathered from various sources such as personal interviews, field observation, and surveys.

- **Discourse Analysis:** considers the social context under which or within which the communication between the researcher and respondent takes place.
- **Grounded Theory:** When you want to explain why a particular phenomenon happened.

Quantitative Data analysis

Preparing data for analysis

The first stage in data analysis is to make it for the analysis so that the nominal data can be converted into something meaningful. Data preparation consists of the below phases.

Phase I: Data Validation—Used to understand if the collected data is unbiased or not. Divided into four different stages

- **Fraud:** To ensure an actual human being records each response to the survey or the questionnaire
- **Screening:** To make sure each participant or respondent is selected or chosen in compliance with the selection criteria
- **Procedure:** To ensure ethical standards were maintained while collecting the data sample
- **Completeness:** To ensure that the respondent has answered all the questions in an online survey.

Phase II: Data Editing —Data editing is a process wherein the developers must confirm that the provided data is free of any errors. They need to conduct necessary checks and outlier checks to edit the raw edit and make it ready for analysis.

Phase III: Data Coding—Out of all three, this phase is associated with grouping and assigning values to the survey responses. It makes data easier to analyse as we have to deal with small data buckets rather than massive data pile.

Considerations in data analysis:

- Analysts must have the necessary skills to analyse the data.
- Usually, data analysis methods vary across different fields; therefore, seeking statistical advice at the outset of analysis can assist in designing survey questionnaires, selecting data collection methods, and choosing samples.
- The primary aim of data analysis and analysis is to derive ultimate insights that are unbiased.
- The motive behind data analysis is to present accurate and reliable data. As far as possible, avoid statistical errors, and find a way to deal with everyday challenges like outliers, missing data, data altering, data mining, or developing graphical representation.

3.2 Process of Data Analysis

Data Analysis is a process of collecting, transforming, cleaning, and modelling data with the goal of discovering the required information. The results so obtained are communicated, suggesting conclusions, and supporting decision-making.

Data Analysis Process consists of the following phases that are iterative in nature

Data Requirements Specification—The data required for analysis is based on a question or an experiment. Based on the requirements of those directing the analysis, the data necessary as inputs to the analysis is identified. Data may be numerical or categorical.

Data Collection— Data Collection is the process of gathering information on targeted variables. It ensures that data gathered is accurate such that the related decisions are valid. Data is collected from various sources ranging from organisational databases to the information in web pages. The data obtained, may not be structured, and may contain irrelevant information.

Data Processing—The data that is collected must be processed or organised for analysis. This includes structuring the data as required for the relevant Analysis Tools.

Data Cleaning—The processed and organised data may be incomplete, contain duplicates, or contain errors. Data Cleaning is the process of preventing and correcting these errors.

Data Analysis—Data that is processed, organised, and cleaned would be ready for the analysis. Various data analysis techniques are available to understand, interpret, and derive conclusions based on the requirements. Data Visualisation may also be used to examine the data in graphical format, to obtain additional insight regarding the messages within the data.

Communication—The results of the data analysis are to be reported in a format as required by the users to support their decisions and further action. The feedback from the users might result in additional analysis. The data analysts can choose data visualisation techniques, such as tables and charts, which help in communicating the message clearly and efficiently to the users.

Chapter 4: Hardware and Software Requirements

Hardware Requirements	Software Requirements
High-resolution webcam	Operating System(windows11, macOS)
Processor: 1.6 GHz Dual-Core intel Core i5	**Libraries:** tkinter, deepface, Open CV, numpy, Pandas
Installed Ram: 8GB	**Software:** IDE- Visual Studio Code, Python, Visual Studio Code, MySQL Workbench

Table 4.1 Hardware and Software Requirements

4.1 Introduction to Python

What is Python? Executive Summary

Python is an interpreter, object-oriented, high-level programming language with dynamic semantics. Its high-level built-in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, and can be used as scripting or glue language to connect existing components together. Python supports modules and packages, which encourages program modularity and code reuse.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation

fault. Instead, when the interpreter discovers an error, it raises an exception. When the program does not catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

Applications for Python

Python is used in many application domains. Here's a sampling.

- The Python Package Indexlists thousands of third-party modules for Python.

Web and Internet Development

Python offers many choices for web development:

- Frameworks such as Django and Pyramid.
- Micro-frameworks such as Flask and Bottle.
- Advanced content management systems such as Plone and Django CMS.

Python's standard library supports many Internet protocols:

- HTML and XML
- JSON
- E-mail processing.
- Support for FTP, IMAP, and other Internet protocol.
- Easy-to-use socket interface.

And the Package Index has yet more libraries:

- Requests, a powerful HTTP client library.
- Beautiful Soup, an HTML parser that can handle all sorts of oddball HTML.
- Twisted Python, a framework for asynchronous network programming.

Desktop GUIs

The Tk GUI library is included with most binary distributions of Python.

- Some toolkits that are usable on several platforms are available separately: wxWidgets , Kivy, for writing multitouch applications , Qt via pyqt or pyside

Platform-specific toolkits are also available: GTK+,Microsoft Foundation Classes through win32 extensions.

Software Development

Python is often used as a support language for software developers, for build control and management, testing, and in many other ways.

- SCons for build control.
- Buildbot and Apache Gump for automated continuous compilation and testing.
- Roundup or Trac for bug tracking and project management.

4.2 Visual Studio Code

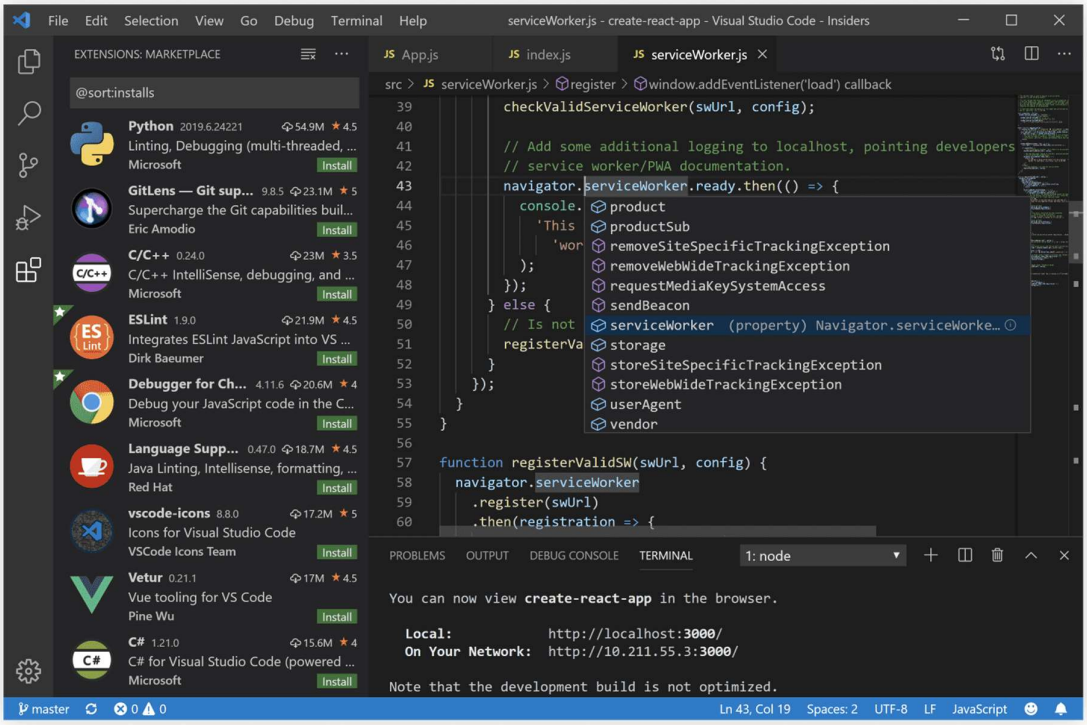
Introduction

Visual Studio Code (VS Code) is a free source-code editor developed by Microsoft. It is known for its lightweight and customizable interface, making it popular among developers. It offers features such as code editing with syntax highlighting and intelligent code suggestions, built-in Git integration, an extensive marketplace for extensions, and an integrated terminal for running commands. With its debugging capabilities, task automation, collaboration features, and support for various programming languages, VS Code provides a productive coding environment for developers across different platforms.

Extensibility and Marketplace: VS Code has a vast extension ecosystem that allows users to enhance its functionality for various programming languages, frameworks, and tools. The VS Code Marketplace offers a wide range of extensions, themes, and other resources contributed by the community.

Visual Studio Code User Interface

The user interface of Visual Studio Code (VS Code) is designed to be intuitive, customizable, and efficient for developers. Here are some key aspects of its user interface:



Sidebar and Activity Bar: The sidebar provides easy access to various functionalities and resources. It includes navigation menus for files, extensions, search, and source control (Git). The activity bar, located on the side, provides quick access to commonly used features such as Explorer, Source Control, Run and Debug, and Extensions.

Editor Window: The editor window is where developers write and edit their code. It provides syntax highlighting, line numbers, and code folding to enhance code readability. IntelliSense offers intelligent code suggestions and completion based on the selected programming language or installed extensions.

Command Palette: The Command Palette is a powerful feature that allows users to execute commands, access settings, and install extensions. It provides a quick way to search and execute various actions within VS Code.

Status Bar: The status bar is located at the bottom of the window and displays useful information such as the current file's encoding, line and column number, Git branch status, and other indicators. It also provides access to additional features like changing the language mode or selecting the code formatter.

Chapter 5: Software Design

Waterfall Model

The waterfall model is the basic software development life cycle model. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially. The waterfall Model illustrates the software development process in a linear sequential flow. This means that any phase in the development process begins only if the previous phase is complete. In this waterfall model, the phases do not overlap.

The following illustration is a representation of the different phases of the Waterfall Model.

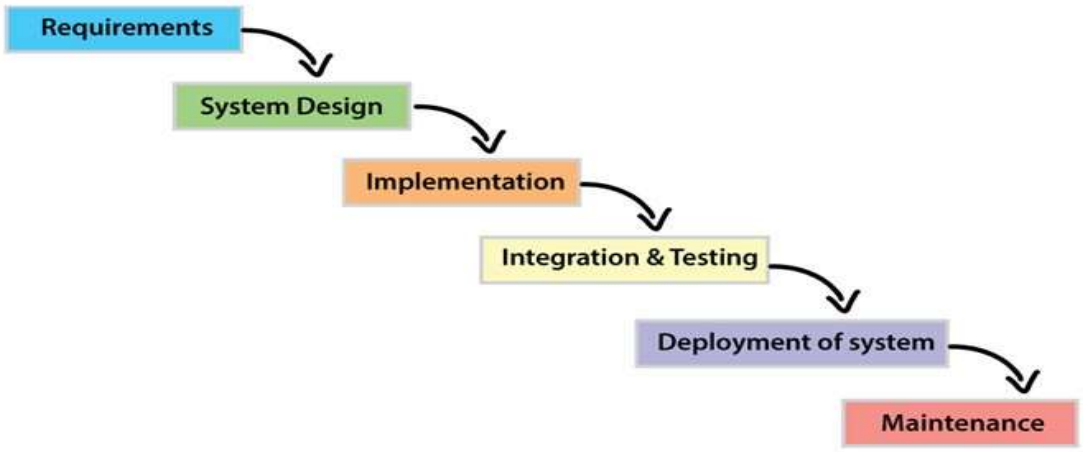


Figure 5.1 Waterfall Model

The sequential phases in Waterfall model are:

- **Requirement Gathering and analysis:** All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- **System Design**: The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation:** With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- **Integration and Testing:** All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- **Deployment of system**: Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- **Maintenance**: There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

5.1 Architectural Design

Admin Module: -

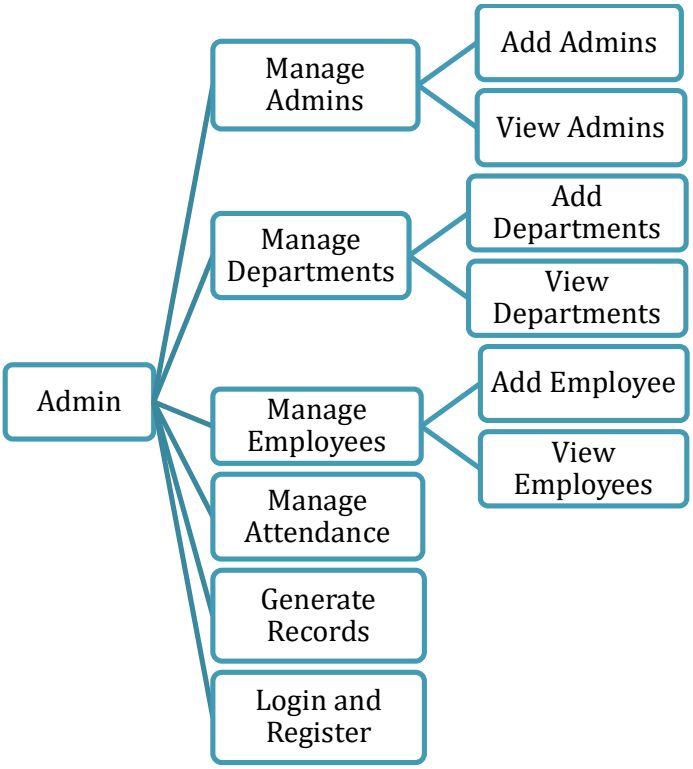


Figure 5.2 Admin Modules Hierarchy Structure

1. *Manage Admins*: The Manage Admins would enable the system's administrators to manage the users who have access to the system. This module would include features such as adding new admins, modifying their access permissions, and removing admins who no longer require access to the system. It has two sub-modules:
    - *Add Admin:* The Add Admin module is a feature that allows authorized personnel to add new administrators or managers to the system.
    - *View Admins:* The View Admin module is a feature that allows administrators to view a list of all the other administrators who have access to the system. This module typically displays details such as the administrator's name, contact information, etc. . The View Admin module also allow administrators to perform actions such as editing or deleting other administrator accounts, or modifying their access levels.
2. *Manage Departments*: The Manage Departments would enable the system's administrators to manage the departments or divisions within the organization. This module would include features such as creating new departments, editing existing departments' information, and removing departments that are no longer in use. It has two sub-modules:

- *Add Department:* The Add Department module is a feature that allows authorized personnel to add new Departments to the system.
- *View Departments:* The View Departments module is a feature that allows administrators to view a list of all the departments in the system. This module typically displays details such as the Department's name, contact information, etc. . The View Departments module also allow administrators to perform actions such as editing or deleting departments.

3. *Manage Employees*: The Manage Employees would enable the system's administrators to manage the employee records within the system. This module would include features such as creating new employee records, editing existing employee information, and removing employee records from the system. This module further has two sub-modules:
    - *Add Employee:* The Add Employee module is a feature that allows authorized personnel to add new employees to the system.
    - *View Employees:* The View Employees module is a feature that allows administrators to view a list of all employees in the system. This module typically displays details such as the employee's name, contact information, etc. . The View employees module also allow administrators to perform actions such as editing or deleting employee accounts.
4. *Manage Attendance*: This will allow administrators to view and manage attendance records for all employees. This module would provide administrators with the tools they need to ensure accurate attendance tracking and make data-driven decisions about attendance management.
5. *Generate Records*: This will enable administrators to generate attendance reports based on various criteria such as department or date range. This module would provide administrators with insights into attendance trends and patterns and help them make informed decisions about attendance management.
6. *Login and Register:* This will provide administrators with secure access to the system by allowing them to create an account or login with existing credentials. This will ensure data privacy and security for administrators and allow them to access the system's functionality.

Employee Module: -

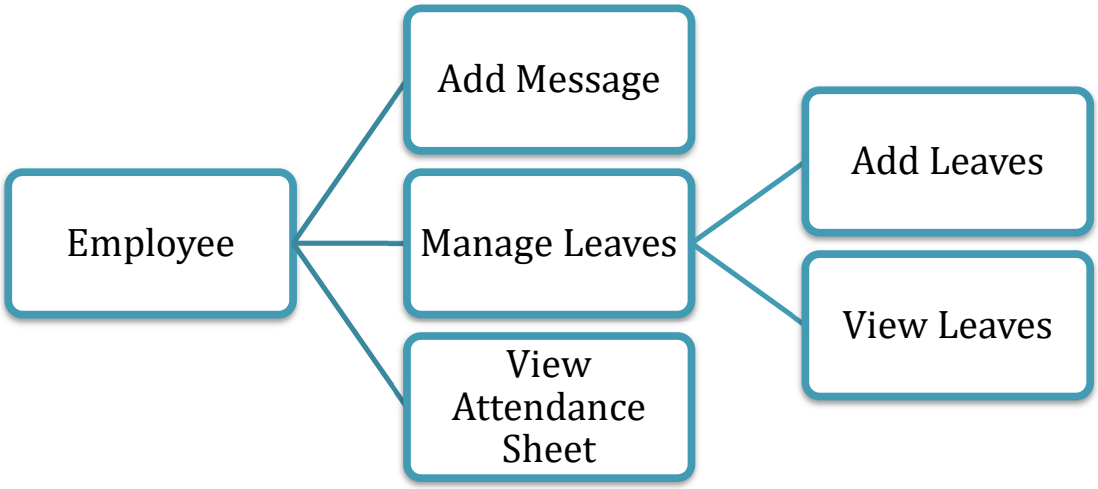


Figure 5.3 Employee Modules Hierarchy Structure

1. *Add Message:* The "Add Message" module in your face attendance system allows employees to send messages to the admin. This module provides a platform for employees to communicate important information, concerns, or requests directly to the system administrator.
2. *View Attendance sheet:* This will allow employees to view their own attendance data in the form of a spreadsheet or report. This module would provide employees with visibility into their attendance records and enable them to monitor their own attendance performance.
3. *Manage Leaves:* The Manage Leaves module is a feature that enables employees to request and manage their leaves or time off from work. It has two sub-modules:
    - *Add leave:* This module typically allows employees to submit leave requests.
    - *View leaves:* This module allows employees to view their leave balances and histories, and track the status of their requests. This module allows employees to cancel or modify their leave requests.

5.2 Data Flow Diagrams(DFDs)

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It can be manual, automated, or a combination of both .It shows how data enters and leaves the system, what changes the information, and where data is stored. The objective of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communication tool between a system analyst and any person who plays a part in the order that

acts as a starting point for redesigning a system. The DFD is also called as a data flow graph or bubble chart. The DFD may be used to perform a system or software at any level of abstraction. In fact , DFDs may be partitioned into levels that represent increasing information flow and functional detail. Levels in DFD are numbered 0, 1, 2 or beyond. Here, we will see primarily three levels in the data flow diagram, which are: 0-level DFD, 1-level DFD, and 2-level DFD.

0-level DFD

It is also known as fundamental system model, or context diagram represents the entire software requirement as a single bubble with input and output data denoted by incoming and outgoing arrows. Then the system is decomposed and described as a DFD with multiple bubbles. Parts of the system represented by each of these bubbles are then decomposed and documented as more and more detailed DFDs. This process may be repeated at as many levels as necessary until the program at hand is well understood. It is essential to preserve the number of inputs and outputs between levels, this concept is called leveling by DeMacro. Thus, if bubble "A" has two inputs x_1 and x_2 and one output y, then the expanded DFD, that represents "A" should have exactly two external inputs and one external output.

1-level DFD

In 1-level DFD, a context diagram is decomposed into multiple bubbles/processes. In this level, we highlight the main objectives of the system and breakdown the high-level process of 0-level DFD into subprocesses.

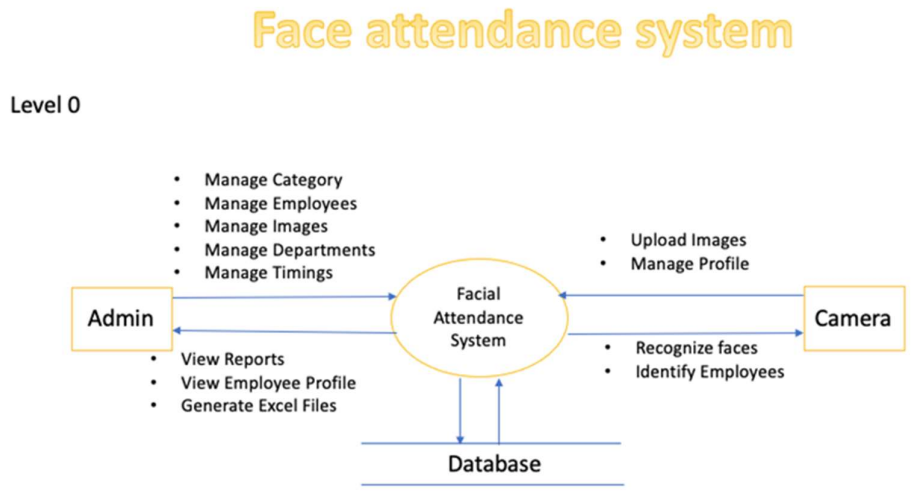


Figure 5.4 Face attendance system

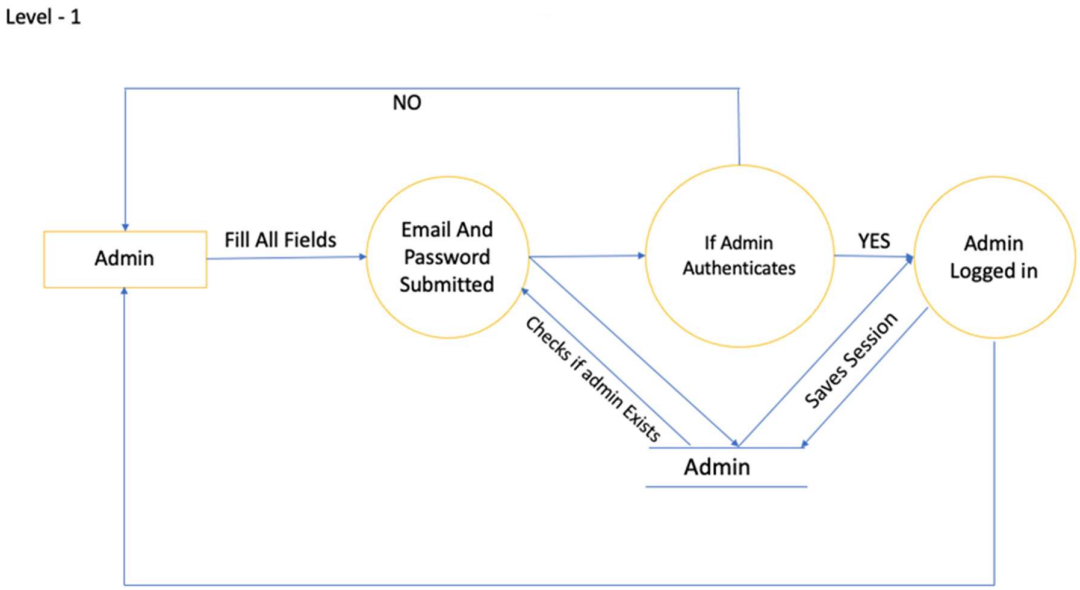


Figure 5.5 Admin Login

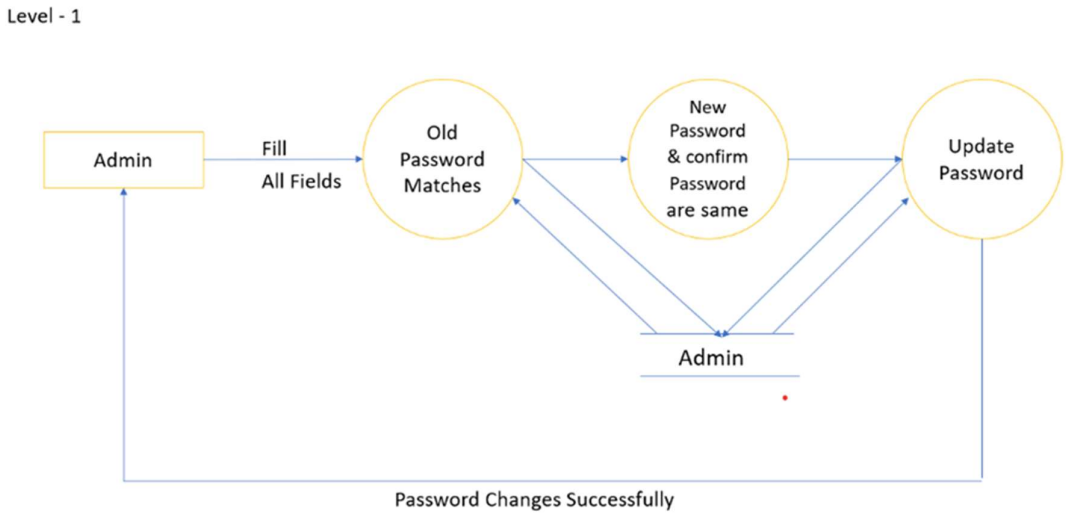


Figure 5.6 Change Password

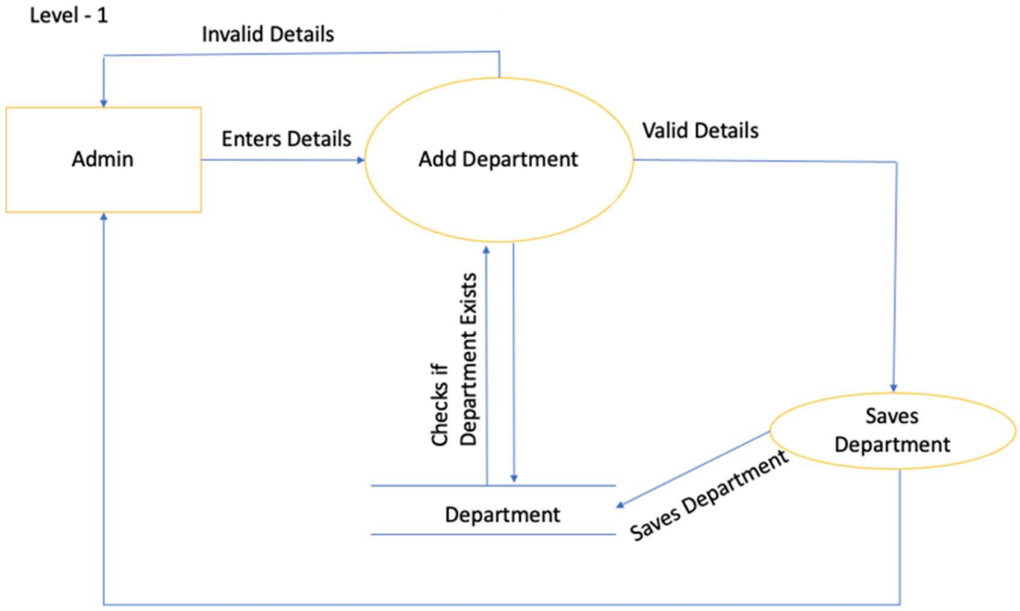


Figure 5.7 Add Department

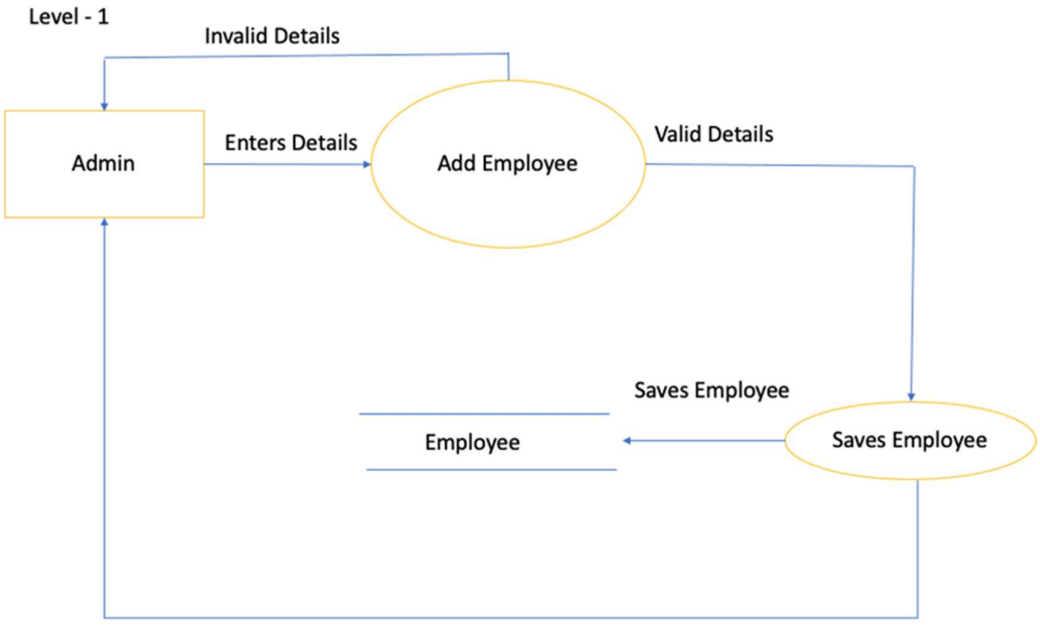


Figure 5.8 Add Employee

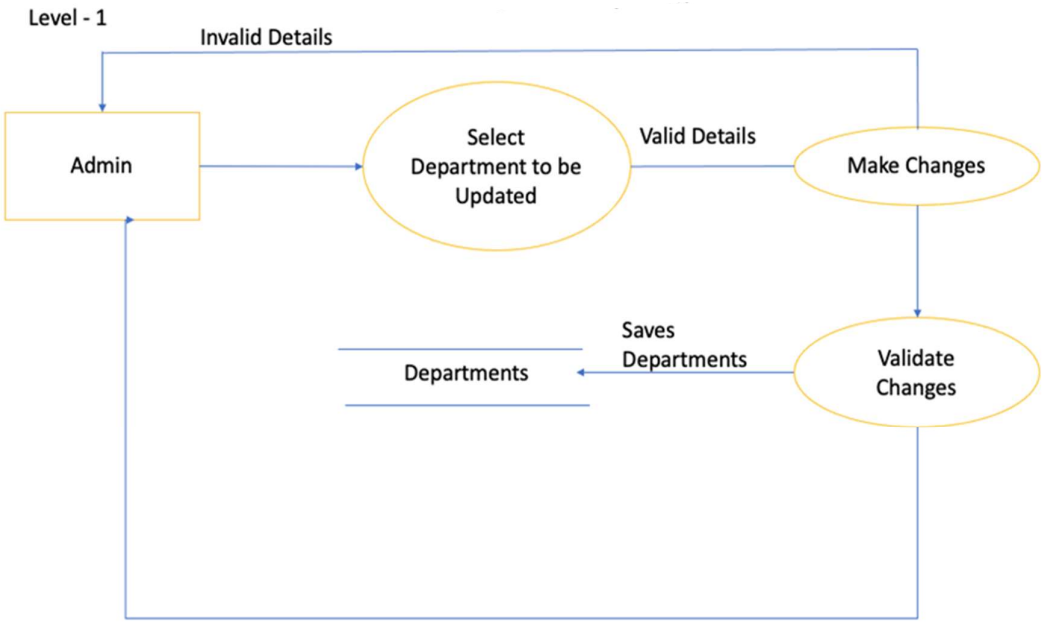


Figure 5.9 Update Department

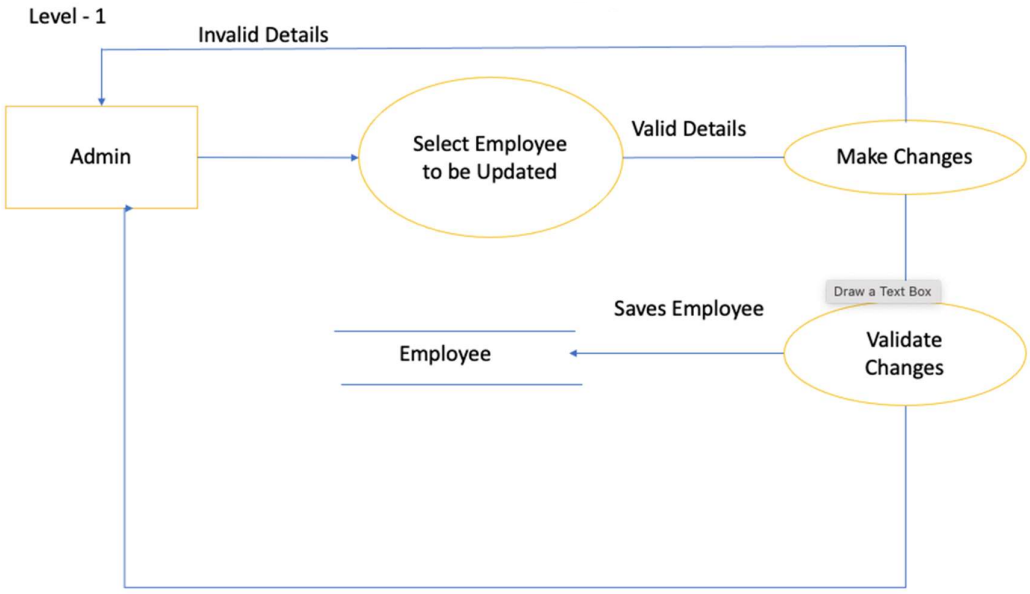


Figure 5.10 Update Employee

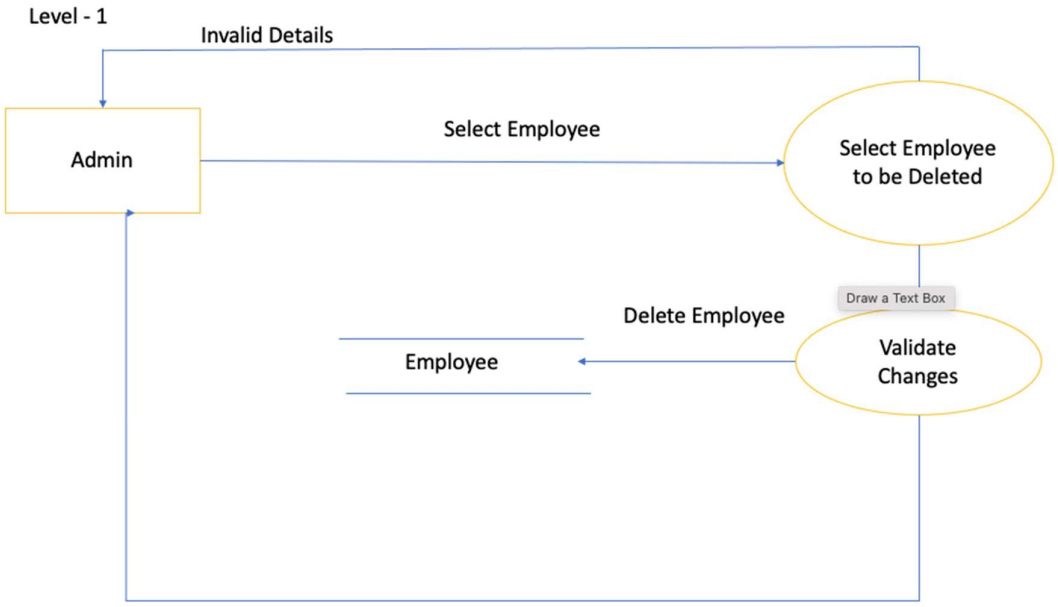


Figure 5.11 Delete Employee

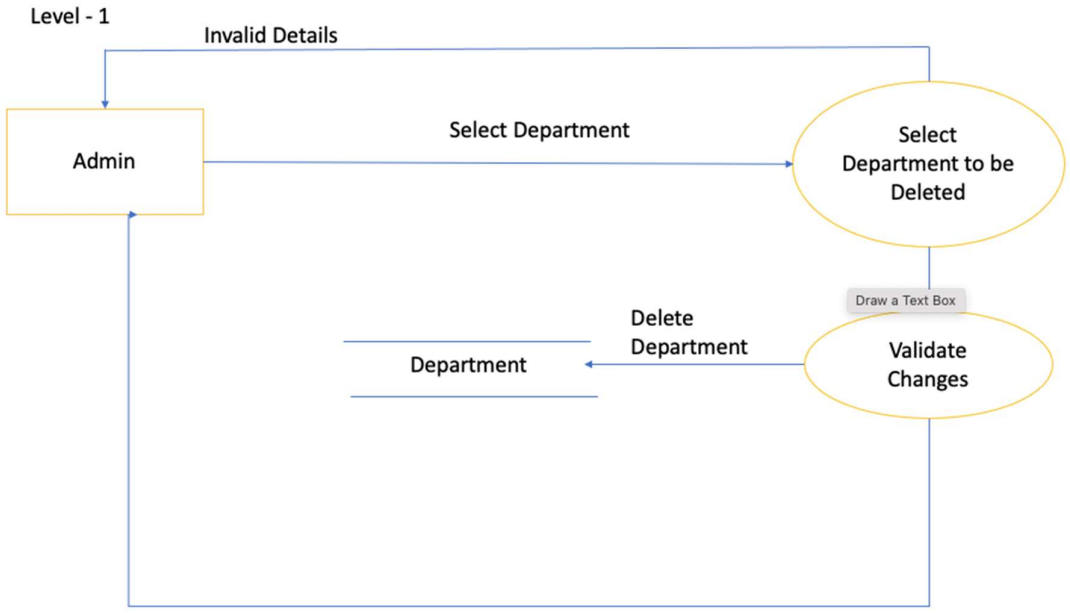


Figure 5.12 Delete Department

5.3 Data Design

5.3.1 Entities and Attributes

a) ***Admin****:* The "admin entity" is responsible for managing and overseeing the system. This includes tasks such as user management, system configuration, data management, and ensuring data security and compliance.

Attributes:

Column	Type	Null	Primary Key	Foreign Key
Id	INT	No	Yes	No
Name	Varchar(255)	No	No	No
Email	Varchar(255)	No	No	No
Mobile	Varchar(255)	No	No	No
Role	Varchar(255)	No	No	No
Password	Varchar(255)	No	No	No

Table 5.1 Admin Attributes

b) ***Attendance****:* The "Attendance entity represents the recorded data of individuals' presence or absence. It includes information such as the date, time, and identification of employees or individuals captured through facial recognition during the attendance process.

Attributes:

Column	Type	Null	Primary Key	Foreign Key
Id	Int	No	Yes	No
Emp_Id	Int	No	No	Yes
Date	Date	No	No	No
Time	Varchar(255)	No	No	No
Type	Varchar(255)	No	No	No

Table 5.2 Attendance Attributes

c) ***Employee****:* The "Employee entity" represents the individuals registered in the system for attendance tracking. It includes their unique ID, name, facial template, and attendance records.

Attributes:

Column	Type	Null	Primary Key	Foreign Key
Id	Int	No	Yes	No
Name	Varchar(255)	No	No	No
Father's Name	Varchar(255)	No	No	No
Email	Varchar(255)	No	No	No
Mobile	Varchar(255)	No	No	No
Address	Varchar(255)	No	No	No
Department	Varchar(255)	No	No	Yes
Category	Varchar(255)	No	No	Yes
Image	Varchar(255)	No	No	No
Password	Varchar(255)	No	No	No

Table 5.3 Employee Attributes

d) ***Department****:* The "Department entity" refers to the organizational units or divisions within an institution or company. It represents different departments or functional areas where employees or individuals belong.

Attributes:

Column	Type	Null	Primary Key	Foreign Key
Name	Varchar(255)	No	Yes	No
Mobile	Varchar(255)	No	No	N**o**
E**mail**	Varchar(255)	No	No	No
Head	Varchar(255)	No	No	No

Table 5.4 Department Attributes

e) ***Leaves**:* The "Leaves entity" refers to the recorded data related to employee or individual leaves or absences. The Leaves entity helps track and manage employee time off within the attendance system.

Attributes:

Column	Type	Null	Primary Key	Foreign Key
Id	Int	No	Yes	No
Emp_Id	Int	No	No	Yes
Date	Date	No	No	No
Remarks	Varchar(255)	No	No	No
Status	Varchar(45)	No	No	No

Table 5.5 Leaves Attributes

f) ***Messages**:* The "Messages entity" refers to the messages or communications sent by employees specifically for the attention of the system administrator or admin. These messages are intended to be viewed and responded to by the admin. The Messages entity by employee serves as a means for employees to communicate with the admin, seeking assistance, reporting issues, or providing relevant information related to their attendance or the system itself.

Attributes:

Column	Type	Null	Primary Key	Foreign Key
Id	Int	No	Yes	No
Emp_Id	Int	No	No	Yes
Date	Date	No	No	No
Time	Varchar(255)	No	No	No
Title	Varchar(255)	No	No	No
Description	Longtext	No	No	No

Table 5.6 Messages Attributes

5.3.2 ER Diagram

An Entity Relationship Diagram (ERD) is a type of diagram that lets you see how different entities (e.g. people, customers, or other objects) relate to each other in an application or a database. They are created when a new system is being designed so that the development team can understand how to structure the database. They can also be created on an existing system to help the team understand how the system works and to find and resolve any issues. Entity Relationship Diagrams use a specific set of symbols, such as shapes and arrows, to depict the system and database.

Components of an ER Diagram

An Entity Relationship Diagram is made up of many different components:

Entity

An entity is a thing that can have data stored about it. It can be a physical object (e.g. car, person), a concept (e.g. address) or an event (e.g. student enrolment in a course). They represent nouns. They are usually represented as rectangles on an ERD with the entity name inside the rectangle.

A strong entity has an identifier (a primary key) and does not depend on any other entities for it to exist. For example, a student may be a strong entity, as it can have a primary key and does not depend on any other entities for it to exist.

A weak entity is one that depends on a strong entity for existence. This means it has a foreign key to another entity. For example, an enrolment of a student may be a weak entity, as an enrolment cannot exist without a student.

Relationship

A relationship in an ERD defines how two entities are related to each other. They can be derived from verbs when speaking about a database or a set of entities.

Relationships in ERDs are represented as lines between two entities, and often have a label on the line to further describe the relationship (such as “enrols”, “registers”, “completes”).

There are several types of relationships that are represented on an ERD:

- One to one: One record of an entity is directly related to another record of an entity
- One to many: One record of an entity is related to one or more records of another entity.
- Many to many: Many records of one entity can be related to many records of another entity.

Attribute

An attribute is a property of an entity or something that can be used to describe an entity. They are often represented as ovals, or as entries inside an entity. There are several different types of attributes represented on an ERD:

- **Simple:** an attribute that cannot be split into other attributes, such as a first name.
- **Composite:** an attribute that can be split into other attributes, such as name being split into first, middle, and last name.
- **Derived:** an attribute that is calculated or determined from another attribute, such as the age of record being calculated from the created date.

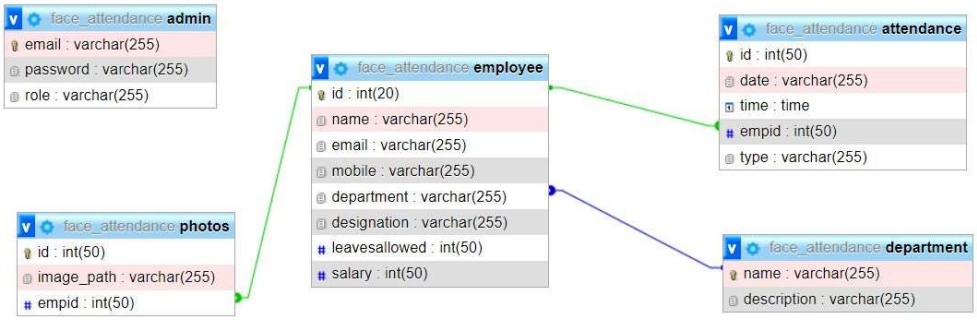
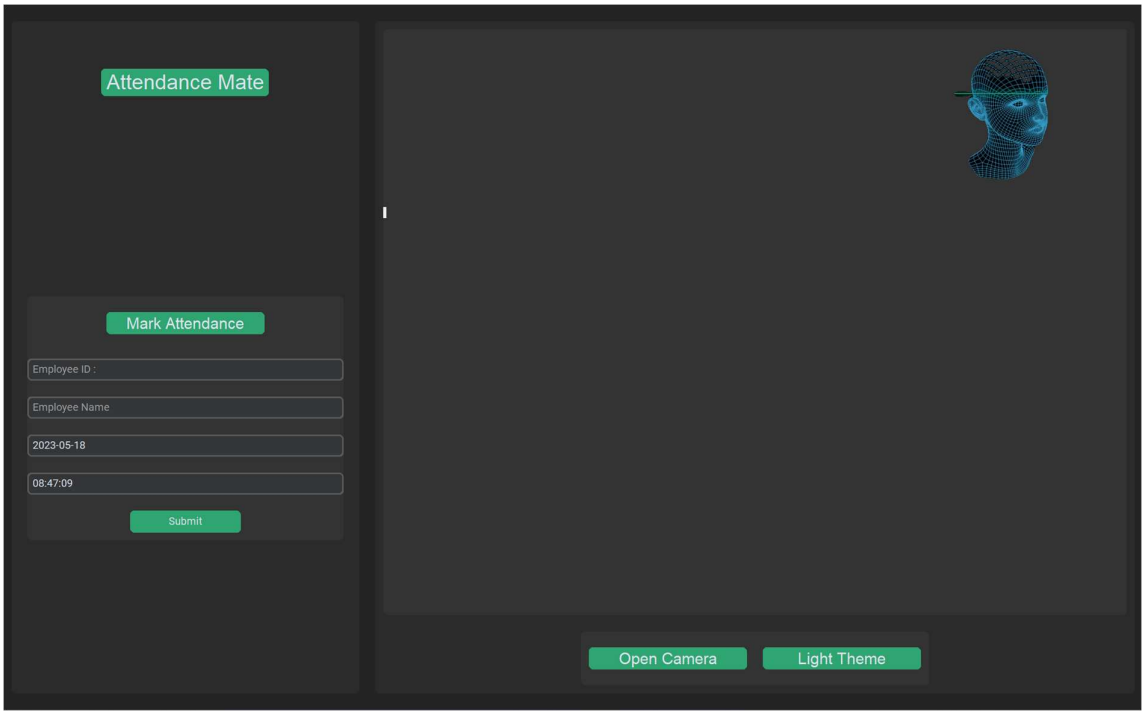


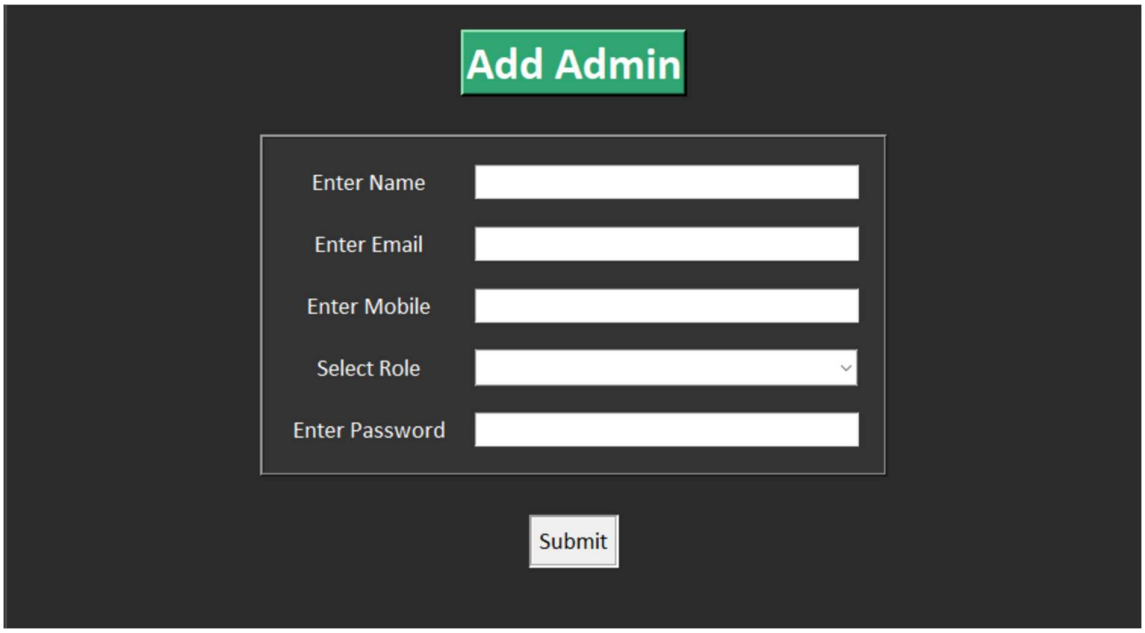
Figure 5.13 ER Diagram

5.4 Interface Design
5.4.1 Input Interface Design

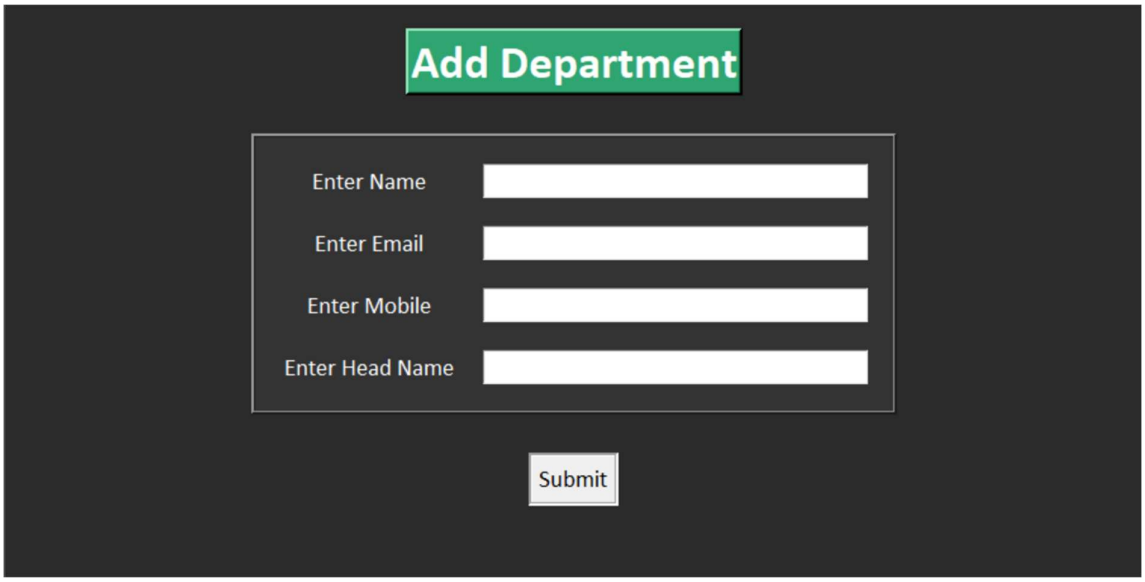
a) Mark Attendance



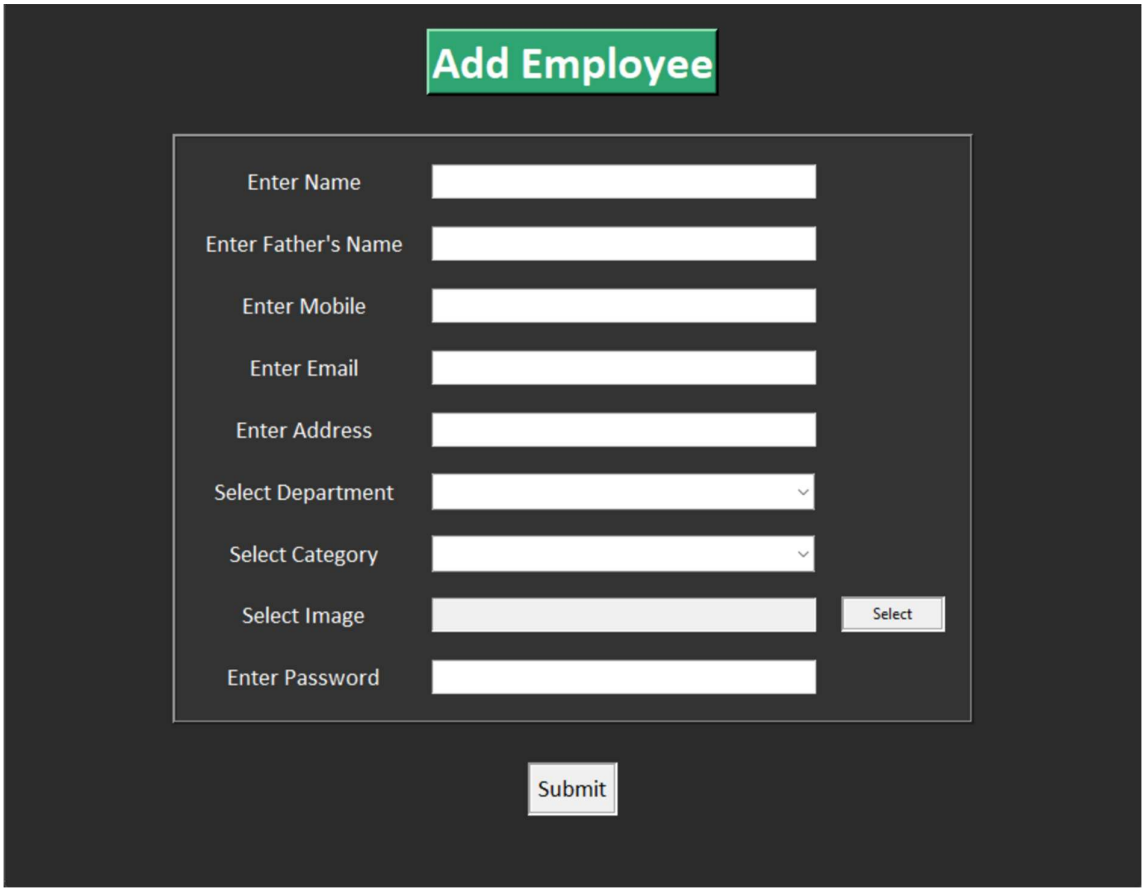
b) Add Admin



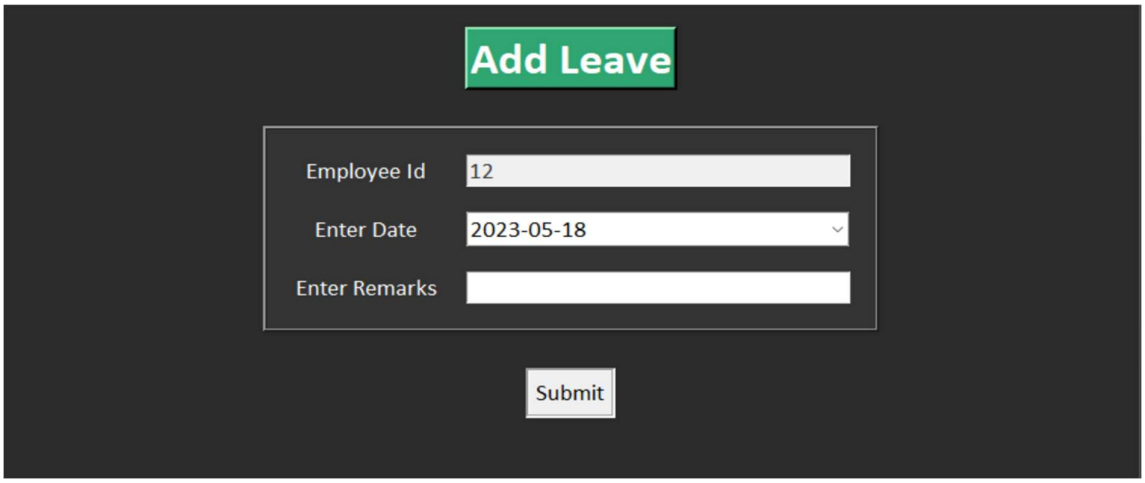
c) Add Department



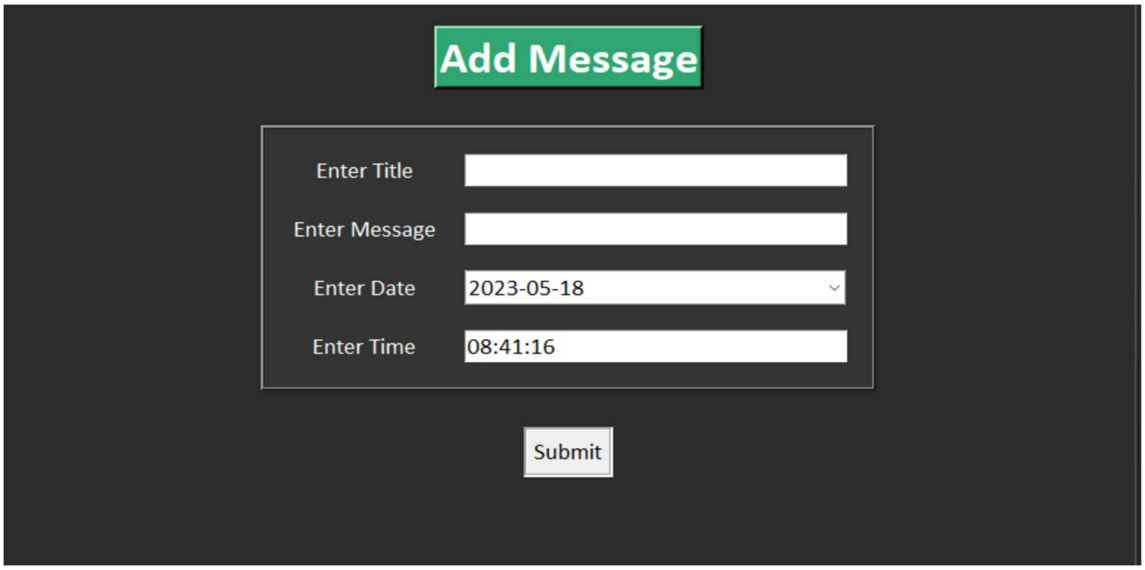
d) Add Employee



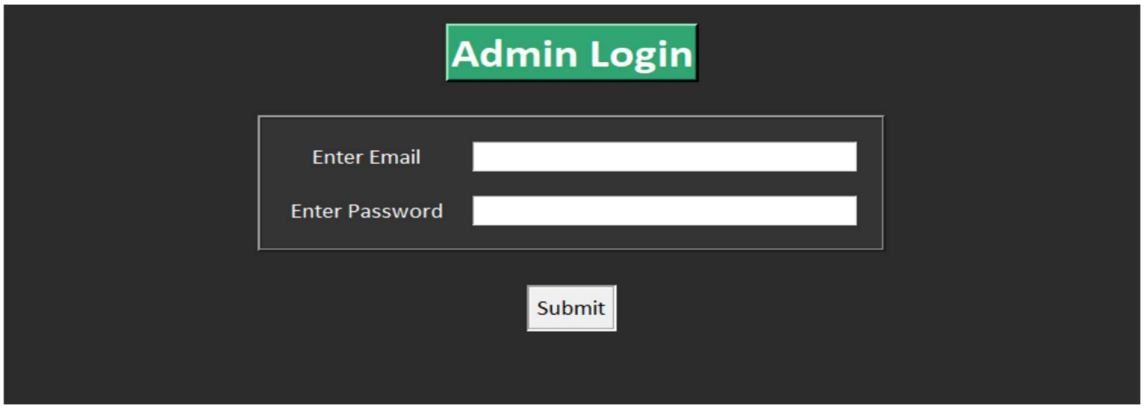
e) Request Leave



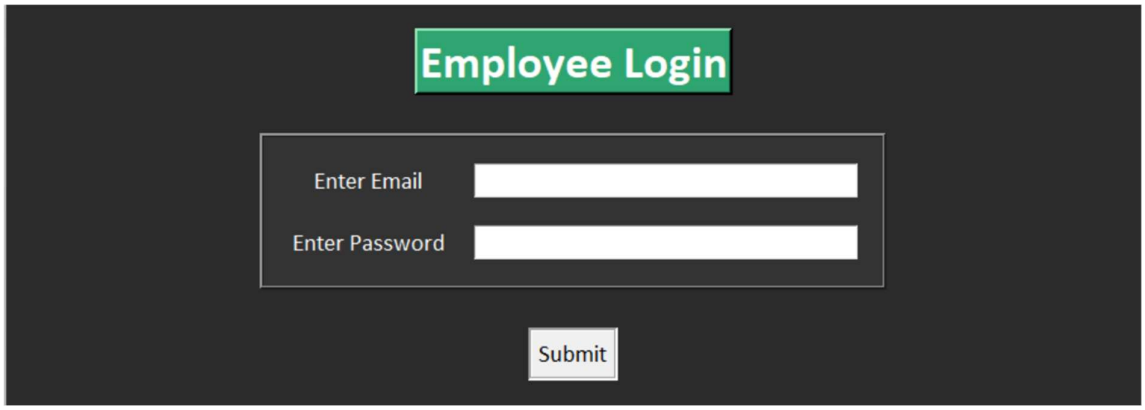
f) Add Message



g) Admin Login

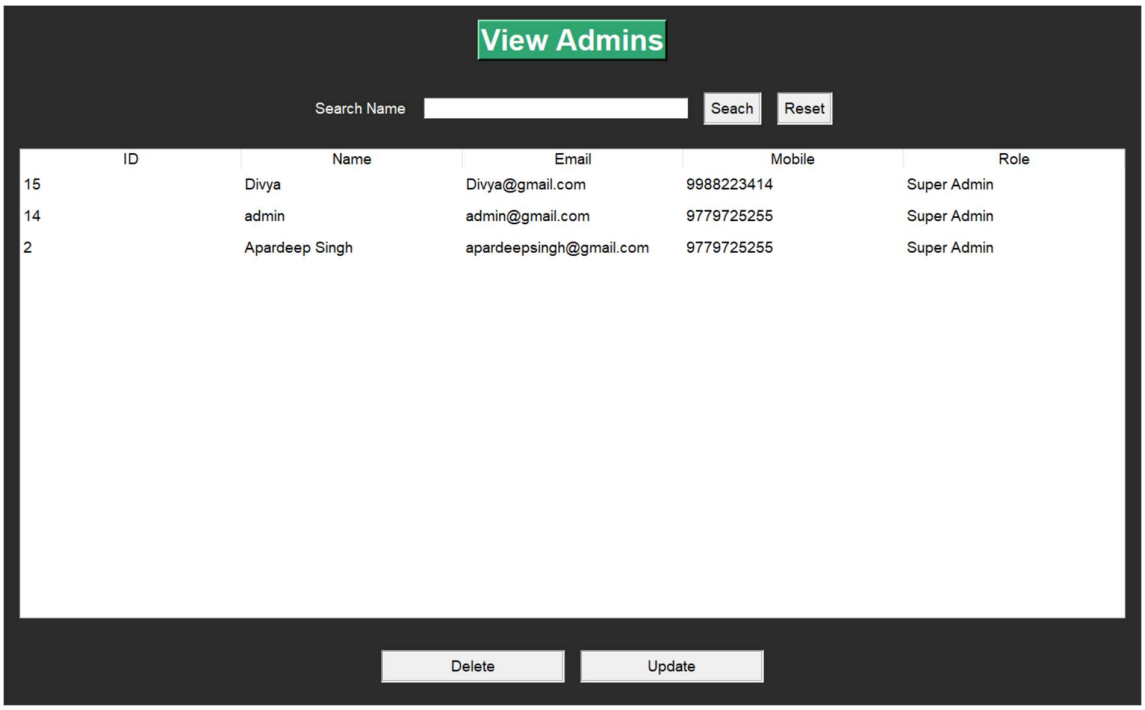


h) Employee Login

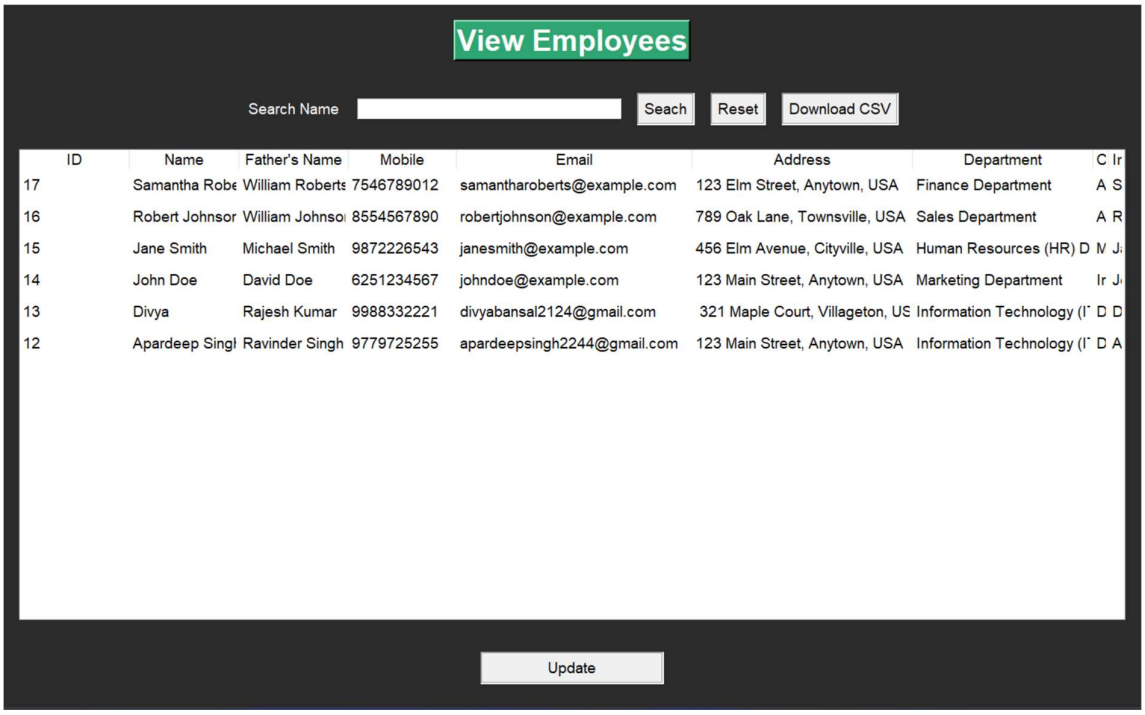


5.4.2 Output Interface Design

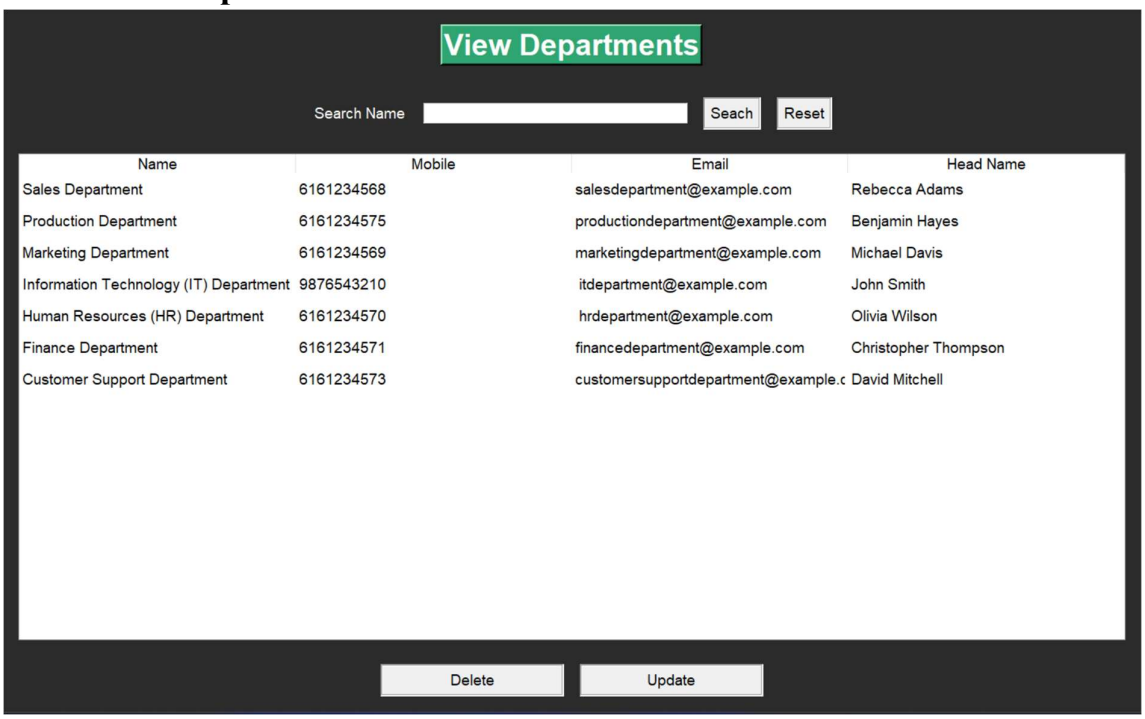
1. View Admins



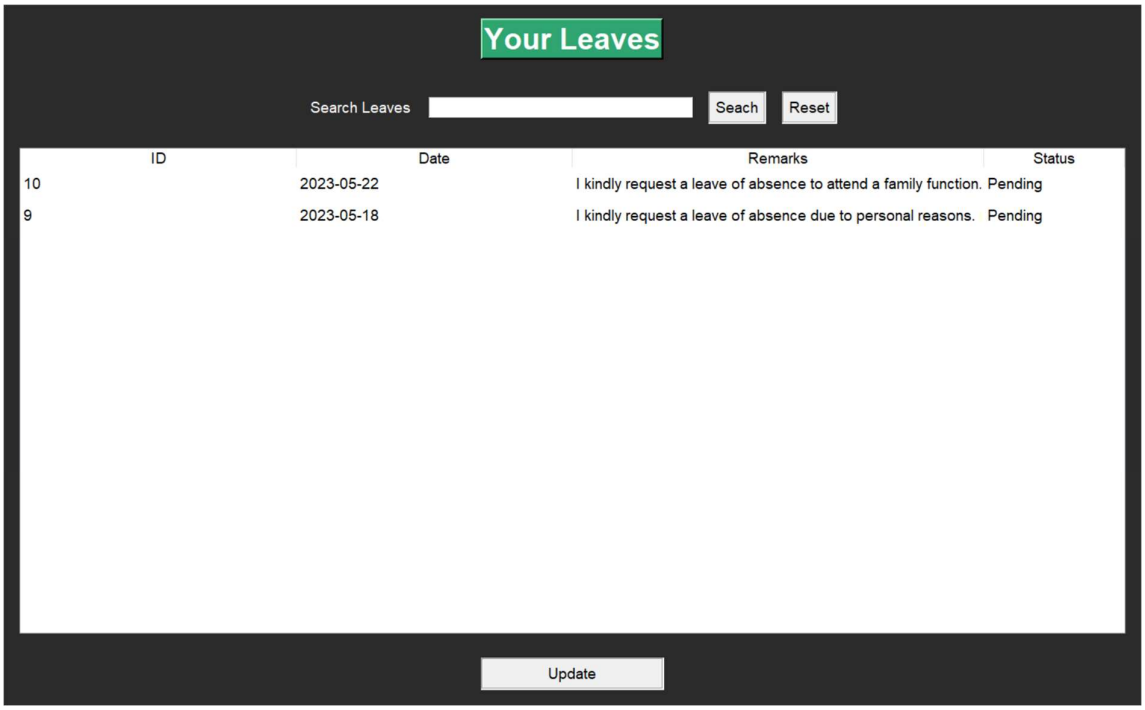
2. View Employees



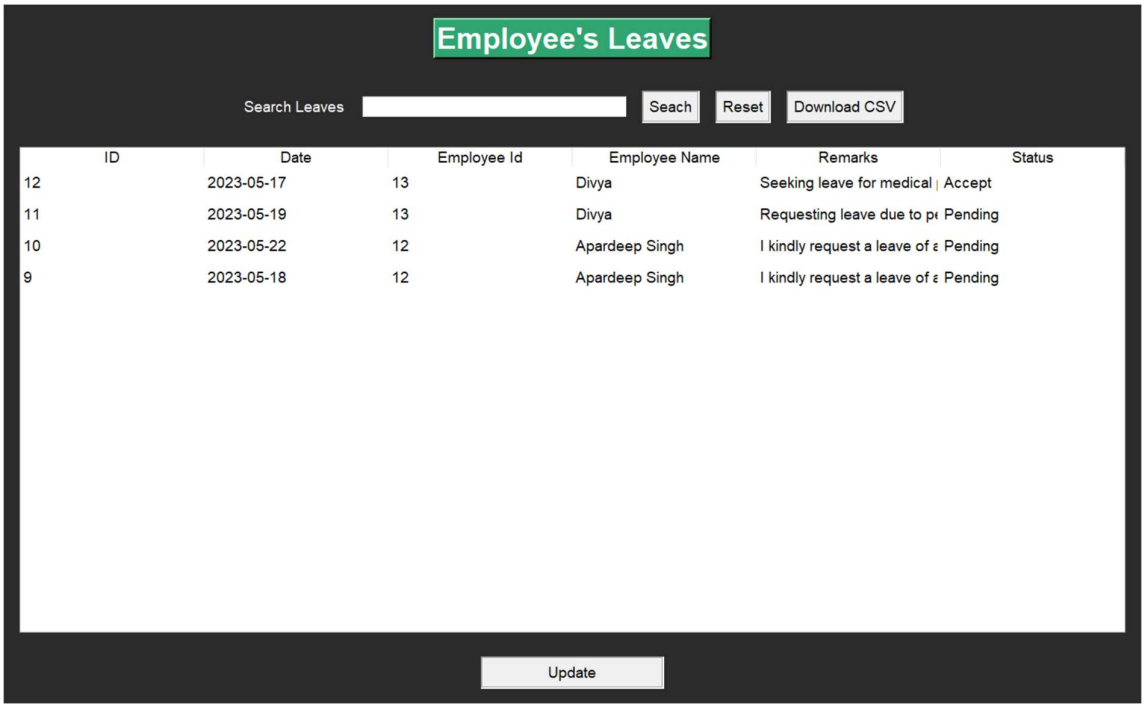
3. View Departments



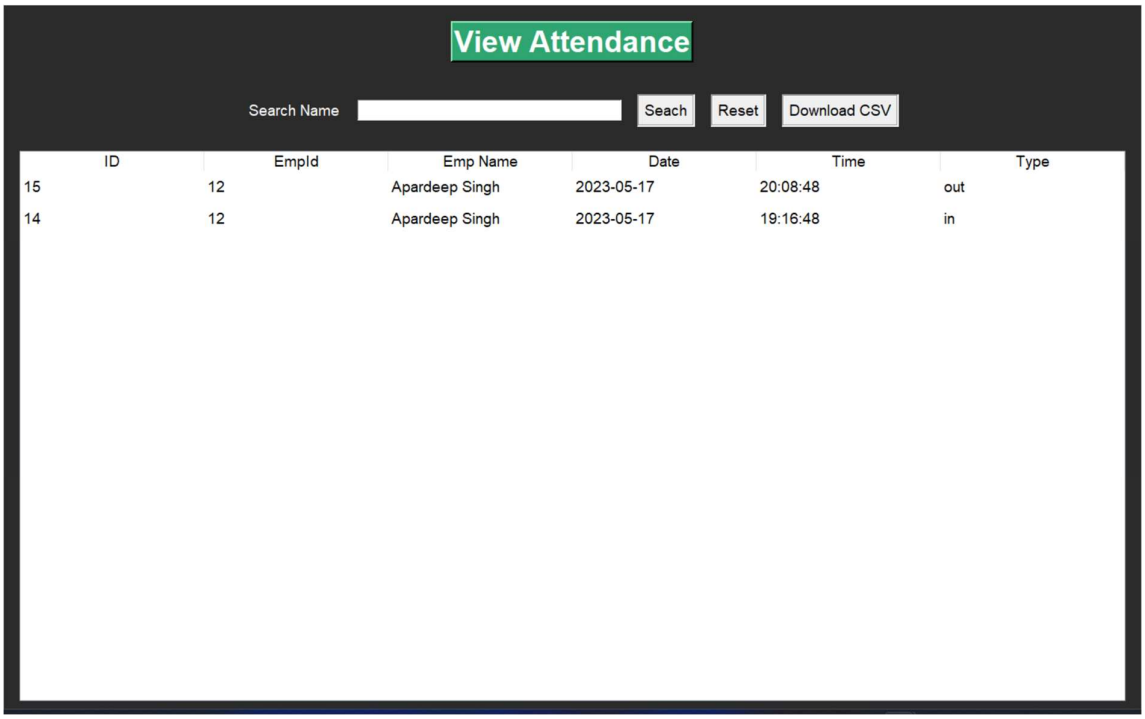
4. View Leaves(Employee Side)



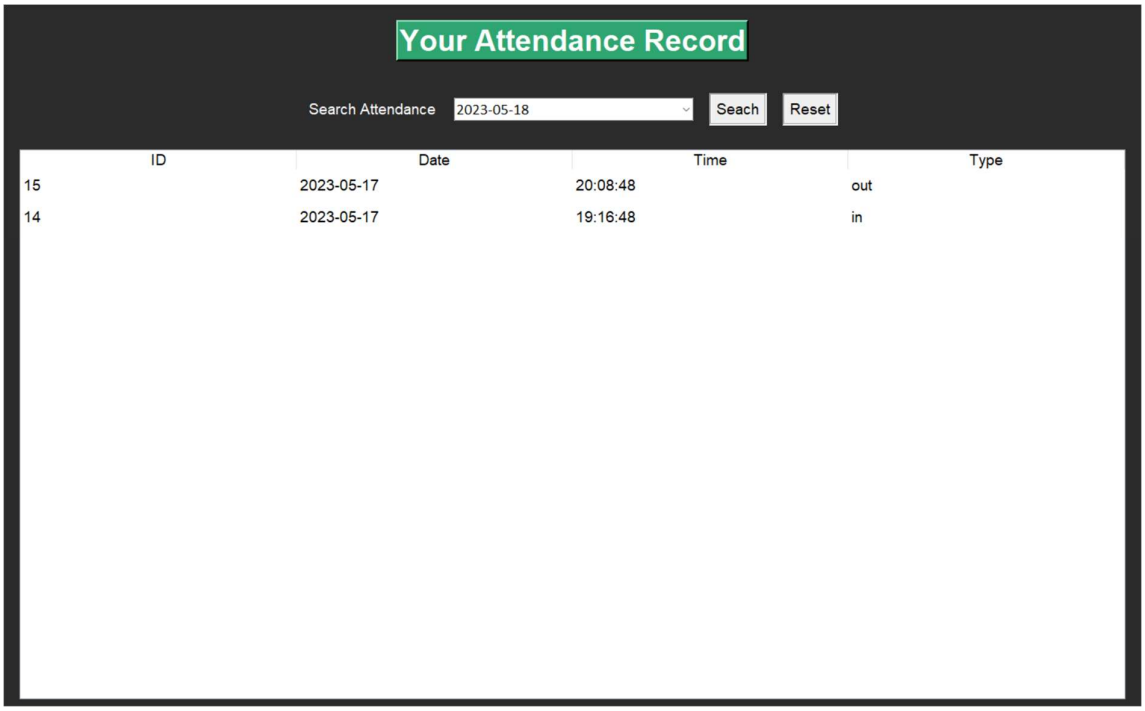
5. View Leaves(Admin Side)



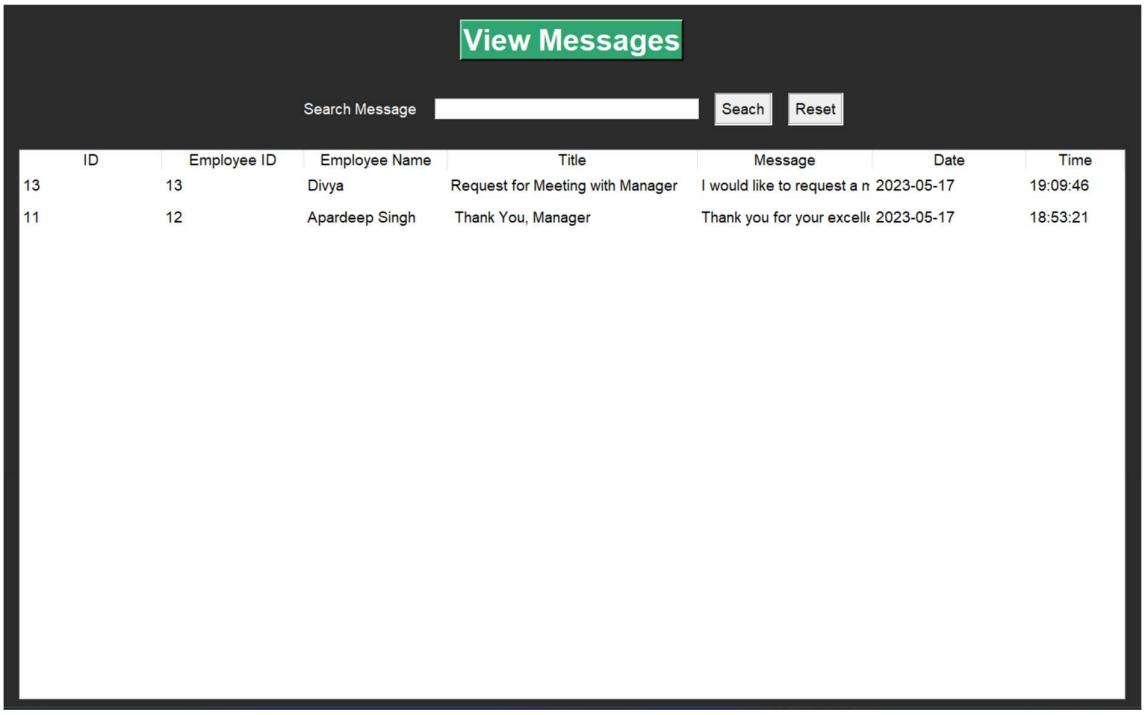
6. View Attendance(Admin Side)



7. View Attendance(Employee Side)



8. View Message



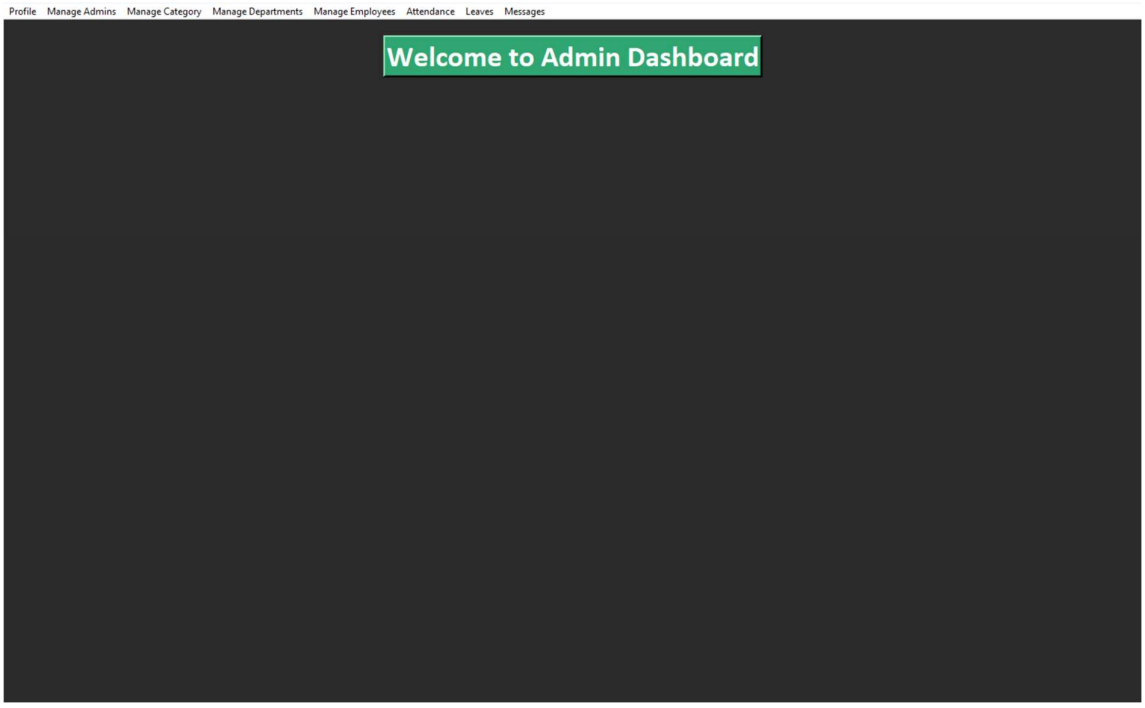
Chapter 6: Implementation

1. main.py

"The main.py file is responsible for initializing and launching the user interface of our face attendance system. It utilizes the tkinter module to create a Fullscreen window with a menu bar that provides options for admin login, employee login, mark attendance and exiting the application. The interface features a dynamically resized image displayed on a canvas, providing an aesthetically pleasing background for the system. This file showcases the project's ability to create a visually engaging and user-friendly interface using tkinter and image manipulation techniques."

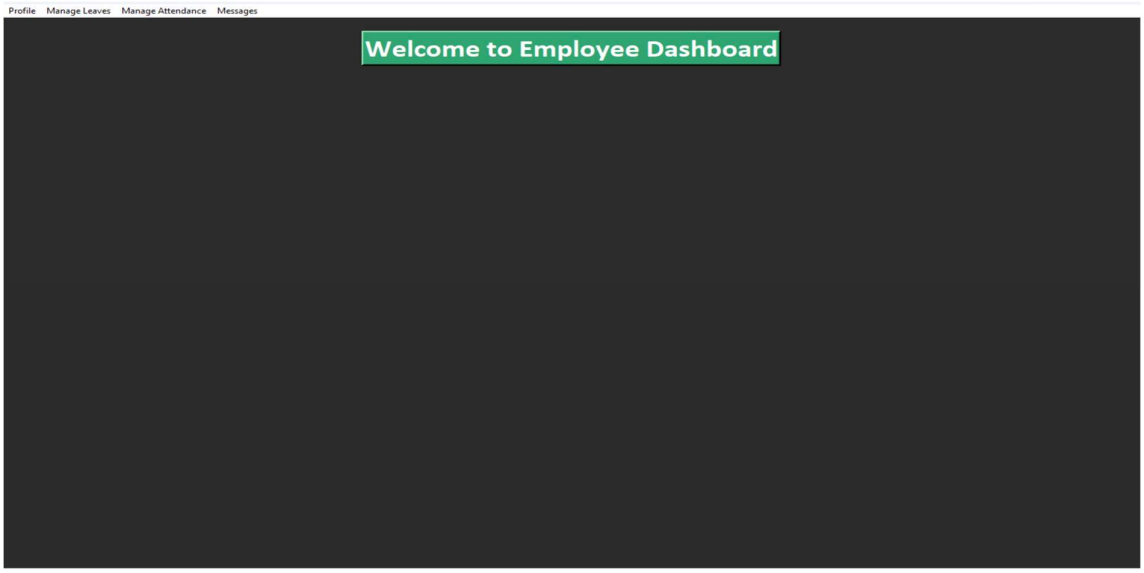
2. admin_dashboard.py

After logging in from the main.py file, administrators are directed to the Admin Dashboard module. This module serves as a centralized hub for system management, offering a user-friendly interface to access various administrative features. Administrators can efficiently perform tasks such as user and admin management, category and department management, employee management, attendance monitoring, leave management, and utilize the messaging system. The Admin Dashboard provides administrators with a seamless and convenient experience for effectively managing the face attendance system.



3. employee_dashboard.py

After logging in from the main.py file, employees are directed to the Employee Dashboard. This dashboard, powered by tkinter, provides employees with an intuitive interface to manage their attendance-related tasks. It includes features such as profile management, leave requests and viewing, attendance viewing, and a messaging system. The Employee Dashboard ensures a seamless and efficient experience for employees to conveniently handle their attendance-related activities.



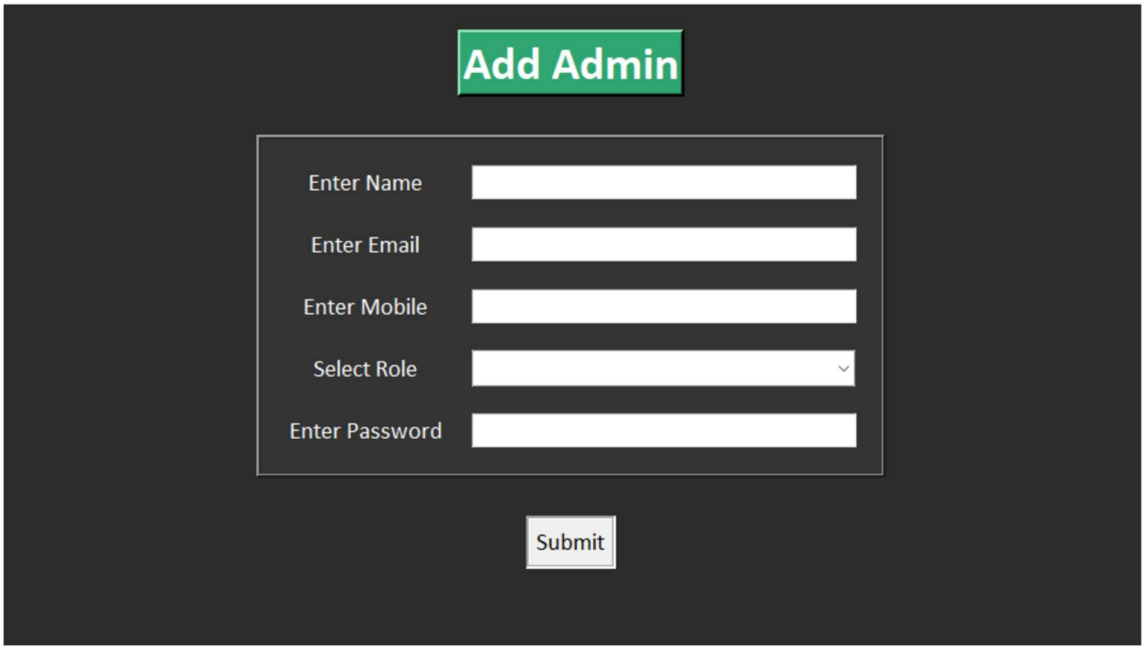
4. addAdmin.py

The addAdmin.py file is responsible for providing functionality to add an admin through a user-friendly interface. It utilizes the tkinter library for creating the graphical user interface. After launching the addAdmin.py file, a window titled "Add Admin" is displayed, featuring input fields and a submit button.

The user interface includes input fields for the admin's name, email, mobile number, role, and password. Additionally, there is a submit button to initiate the process of adding the admin. It includes validation checks to ensure that the entered details are valid, such as a valid email format and a non-empty value for each field.

Upon successful validation, the admin details are inserted into the database using SQL queries. A success message is displayed to inform the user that the admin has been added successfully. The input fields are cleared for the next entry.

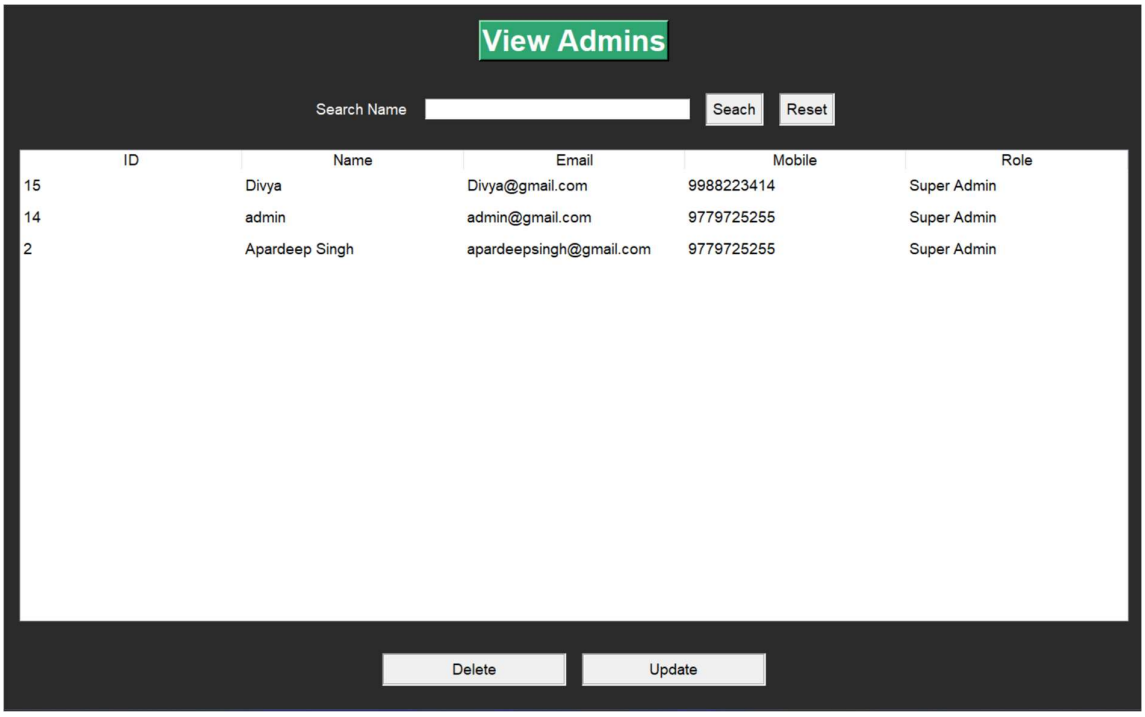
This implementation of the addAdmin.py file allows administrators to conveniently add new admins to the system by providing accurate and valid details.



5. view_admin.py

From the admin dashboard admin gets the view admin option. The code of this file can be divided into the following sections:

1. Treeview: The ttk.Treeview component is used to display a tabular view of administrators. Column headings are defined, and the style for the Treeview is configured.
2. Data Retrieval: The getValues method retrieves administrator data from a database and inserts it into the Treeview component.
3. Event Handling: Event handlers are included for actions like double-clicking a row, updating an administrator's details, deleting an administrator, and searching for administrators by name.
4. Update Window: The openUpdateWindow method opens a new window for updating an administrator's details.
5. Database Operations: The code includes SQL statements to update or delete administrators based on user actions.
6. Main Execution: The Main class is instantiated, and the main event loop is started.



6. addEmployee.py

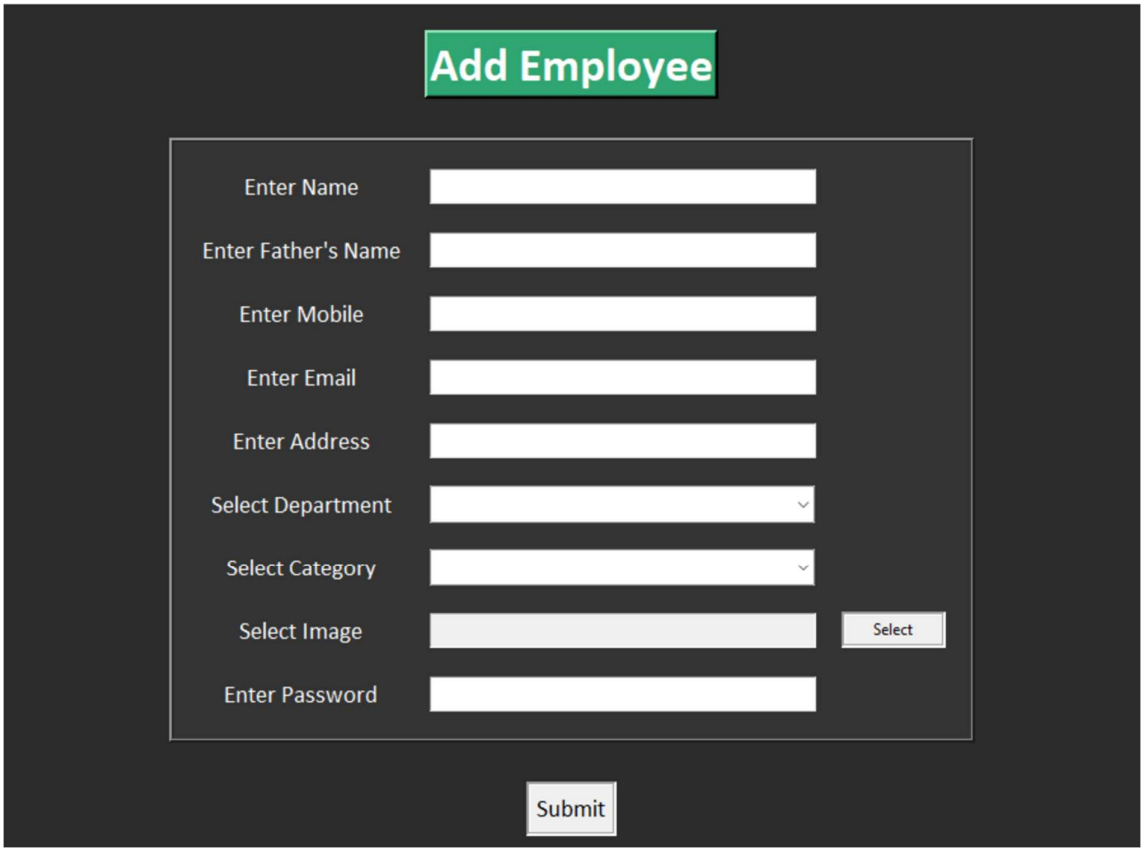
The addEmployee.py file is responsible for providing functionality to add an employee through a user-friendly interface. It utilizes the tkinter library for creating the graphical user interface. After launching the addEmployee.py file, a window titled "Add Employee" is displayed, featuring input fields and a submit button.

Multiple labels and entry fields are added to capture employee details such as name, father's name, mobile number, email, address, department, category, image selection, and password. The labels and entry fields are arranged using the grid layout manager to ensure proper alignment.

Two comboboxes are used to select the department and category from predefined values. The values are fetched using the getDepartments() and getCategories() methods that are connect to the database and fetch the available department and category names.

A "Select Image" button is provided to choose an image file using the file dialog. When an image is selected, it is processed using OpenCV's cascade classifier to detect faces. If a face is detected, the image is saved in the "employees_Images" directory with a unique name. The selected image's name is displayed in the corresponding entry field.

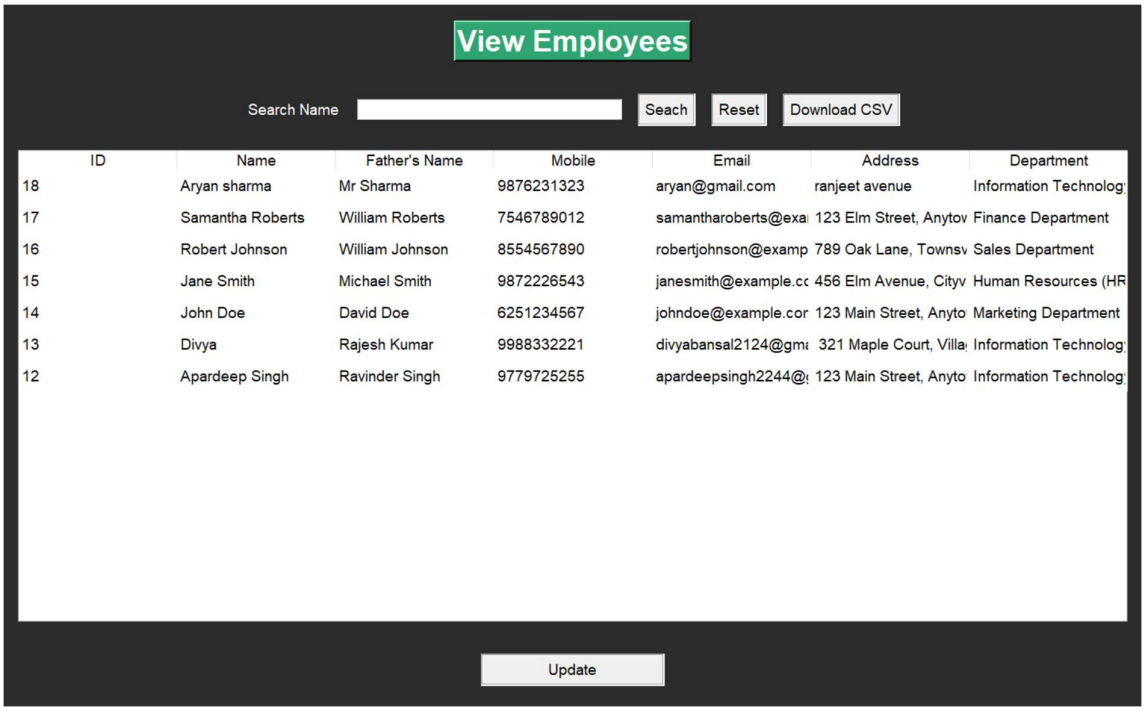
The submitForm() method is triggered when the "Submit" button is clicked. It retrieves all the entered values from the entry fields and performs validation checks. If all the validations pass, the employee's details are inserted into the database using an SQL insert query. A success message is displayed, and the entry fields are cleared for the next entry.



7. view_employee.py

The view_employe.py file is a part of the admin dashboard, It provides the functionality to view and manage employees' information. Here's a brief explanation of this file:

1. The Main class is defined, which represents the main window of the employee view functionality.
2. The GUI elements such as labels, entry fields, buttons, and treeview (table) are created and configured using Tkinter.
3. The getValues method is called to populate the employee table with data from the database. It retrieves employee information and displays it in the table.
4. The openUpdateWindow method is called when a user double-clicks on a row in the employee table. It opens a new window to update the employee's details. The selected employee's information is pre-filled in the entry fields for editing.
5. The selectImage method is called when the user clicks on the "Select" button to choose a new image for the employee. It opens a file dialog to select an image file, performs face detection using OpenCV, and saves the image in the employees' images folder. The selected image's name is displayed in the entry field.
6. The getDepartments and getCategories methods retrieve the department and category names from the database and return them as a list. These lists are used to populate the respective dropdown menus in the update window.
7. The generateCSV method is called when the user clicks on the "Download CSV" button. It opens a folder dialog to choose a destination folder, creates a CSV file, and writes the employee attendance data into the file.
8. The updateEmployee method is called when the user clicks on the "Submit" button in the update window. It retrieves the updated employee details from the entry fields, validates the mobile number and email address, renames the image file if a new image is selected, updates the employee's information in the database, and shows a success message.
9. The delEmployee method is called when the user clicks on the "Delete" button. It confirms the deletion with a popup message and deletes the selected employee from the database.

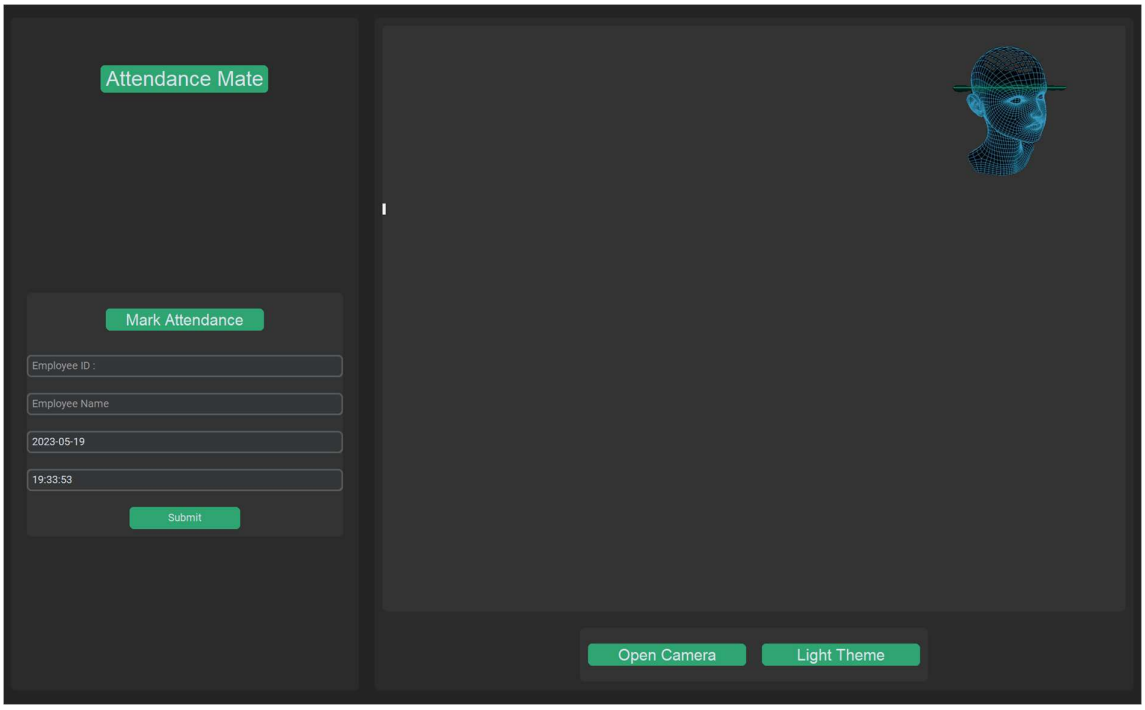


8. markAttendance.py

The markAttendance.py module is a crucial component of our Face Attendance System project. It integrates face recognition functionality into the employee dashboard, allowing for automated attendance marking. Here's a concise explanation of the code:

- The module utilizes the Tkinter library for creating a graphical user interface.
- It relies on OpenCV for video capturing and face detection, as well as PIL for image processing.
- DeepFace, a face recognition library, is employed to compare captured faces with stored employee images.
- The module establishes a connection with the database using a separate connection.py module.
- Upon starting, the user can begin video capturing and initiate face recognition.
- The captured video frames are continuously processed, with faces detected and highlighted.
- Once the user captures an image, the system compares it with stored employee images.
- If a match is found, the employee's information is retrieved from the database and displayed.
- A success message is shown using Tkinter's messagebox to indicate the successful marking of attendance.

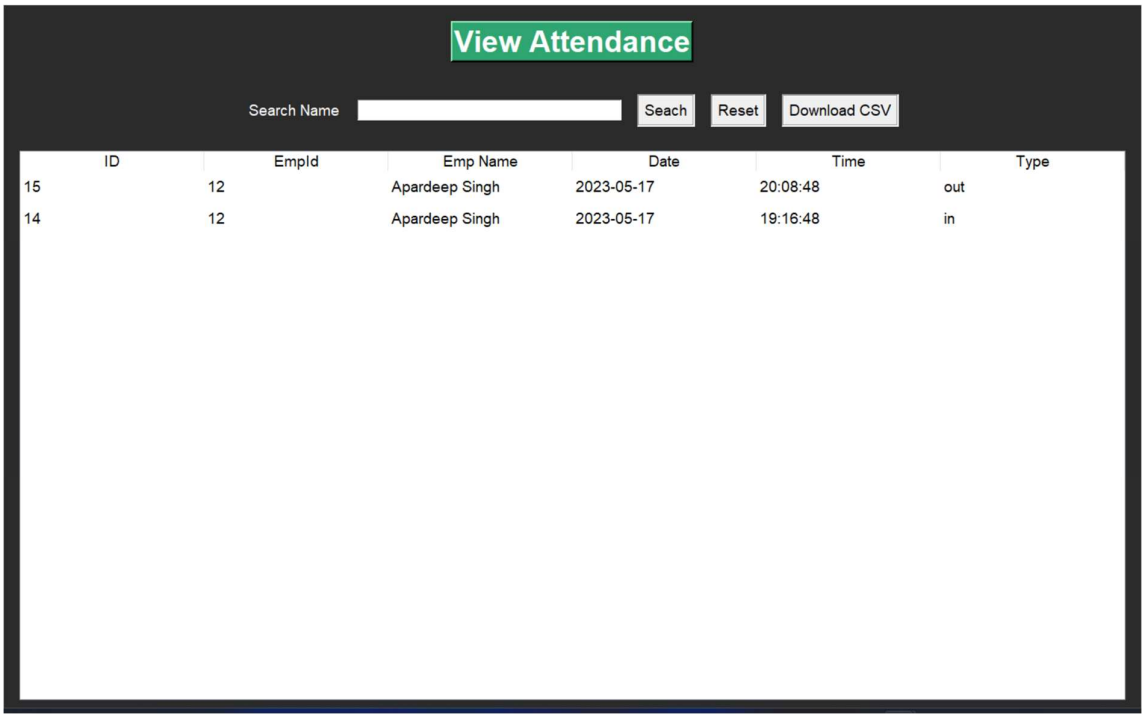
By incorporating this code into our project, we enable the face attendance system to identify employees in real-time, providing a convenient and accurate method for attendance tracking.



9. View_attendance.py

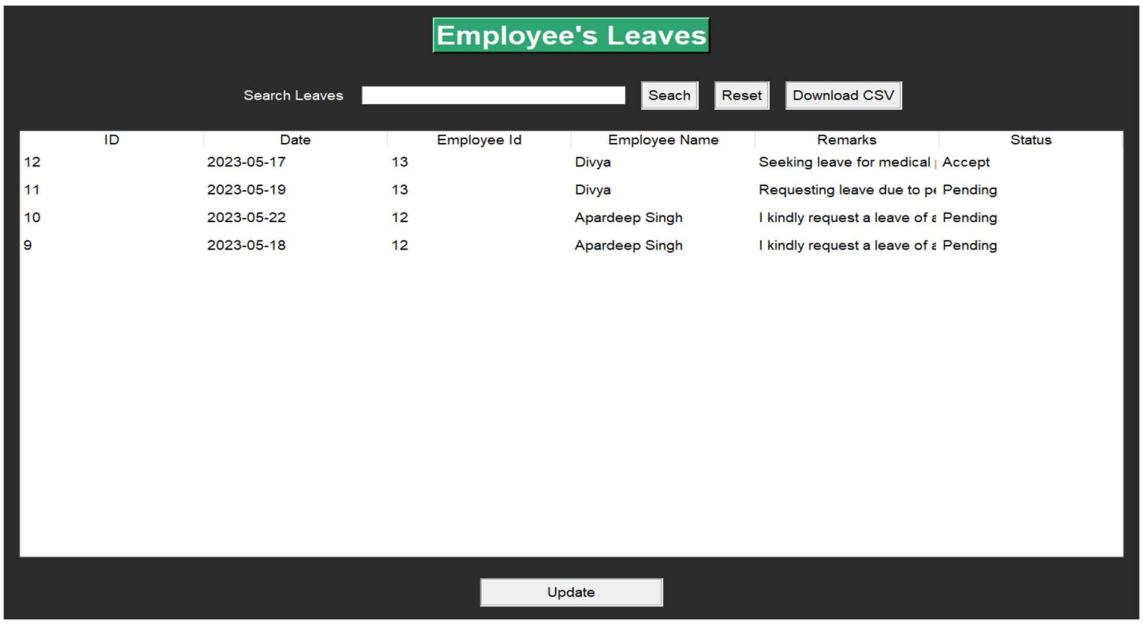
The ‘view_attendance.py’ enables administrators to view and manage attendance records efficiently. Here's a concise explanation of the file:

- The module utilizes the Tkinter library to create a graphical user interface for the admin dashboard.
- It incorporates various Tkinter widgets such as labels, entry fields, buttons, and a Treeview widget for displaying attendance records.
- The module establishes a connection with the database using a separate connection.py module.
- The main window of the admin dashboard is created with a labeled header, search functionality, and buttons for resetting and downloading attendance records in CSV format.
- The attendance records are displayed in a tabular format using the Treeview widget, with columns for ID, Employee ID, Employee Name, Date, Time, and Type.
- The module provides functionality to search for attendance records based on the employee's name, and the results are dynamically updated in the table.
- The “download CSV” button allows the admin to download the attendance records in CSV format by selecting the desired directory using the file dialog.
- The Reset function clears the search input field and displays all the attendance records again.



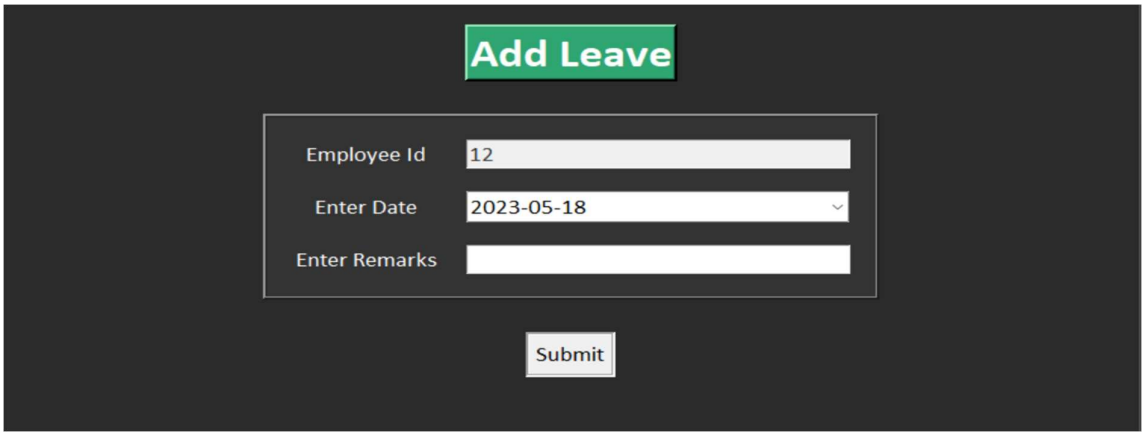
10. View_leave.py

view_leave.py is a script that creates an admin dashboard interface using Tkinter. It allows administrators to view and manage employee leaves. The interface includes search functionality, a table to display leave records, and options to update leave status and generate a CSV report. Admin can accept or cancel the employee's leave as per the need.



11.Addleave.py

- This module allows employees to select a date using the calendar widget, and provide remarks for their leave request.
- Once the employee submits the leave request by clicking the "Submit" button, the module validates the input data and inserts the leave details into the database.
- If any required field is left empty, a warning message is displayed.
- Upon successful submission, a success message is shown to the employee, confirming that the leave request has been recorded.



Chapter 7: Testing

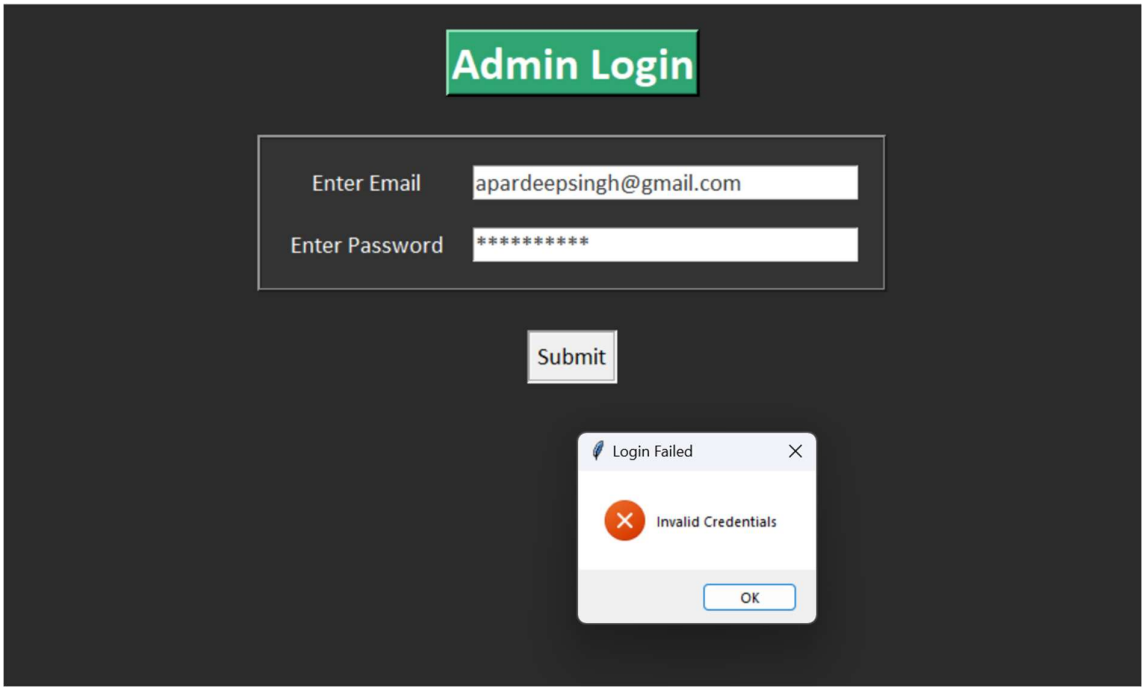
7.1 Test Plan

The testing plan includes planning for several functions like:

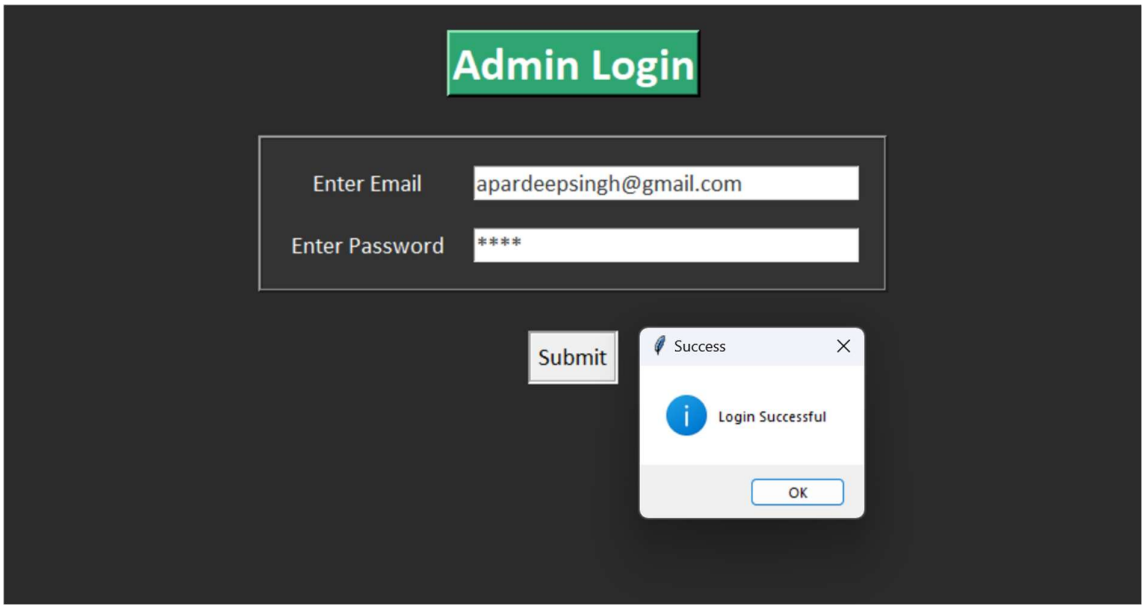
- Login feature
- Registration of employee
- Face detection
- Face recognition from registered database
- Unknown face detection
- Attendance entry in Excel sheet
- Email notification

7.2 Test Cases

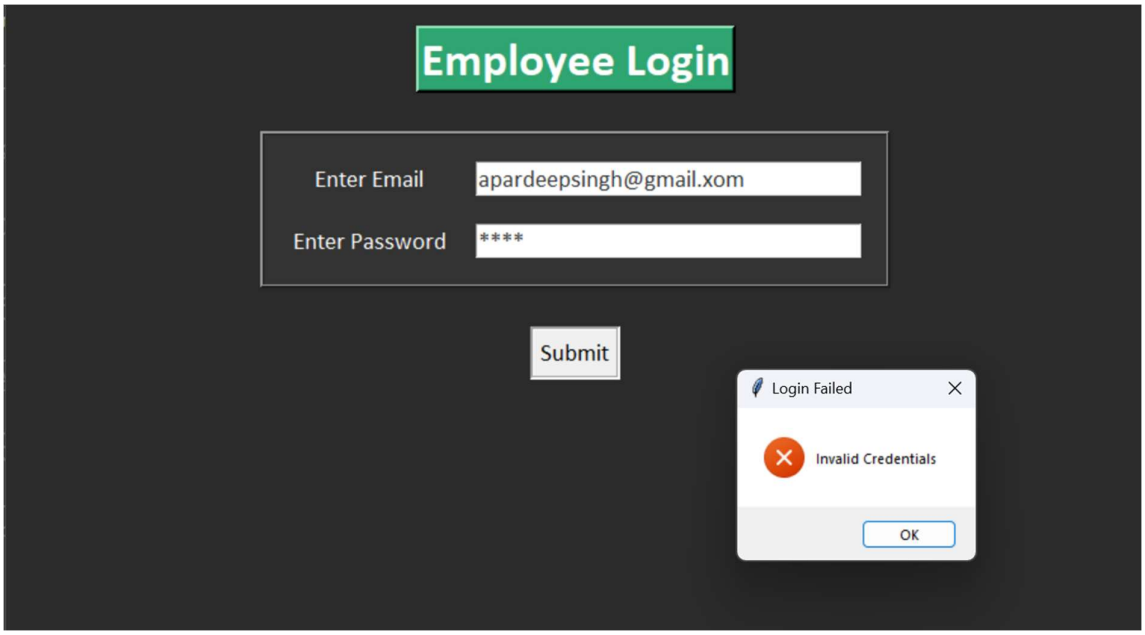
1. **Login Feature:**
    - **Test invalid credentials(Admin):** Validate that the system rejects login attempts with incorrect or invalid credentials.



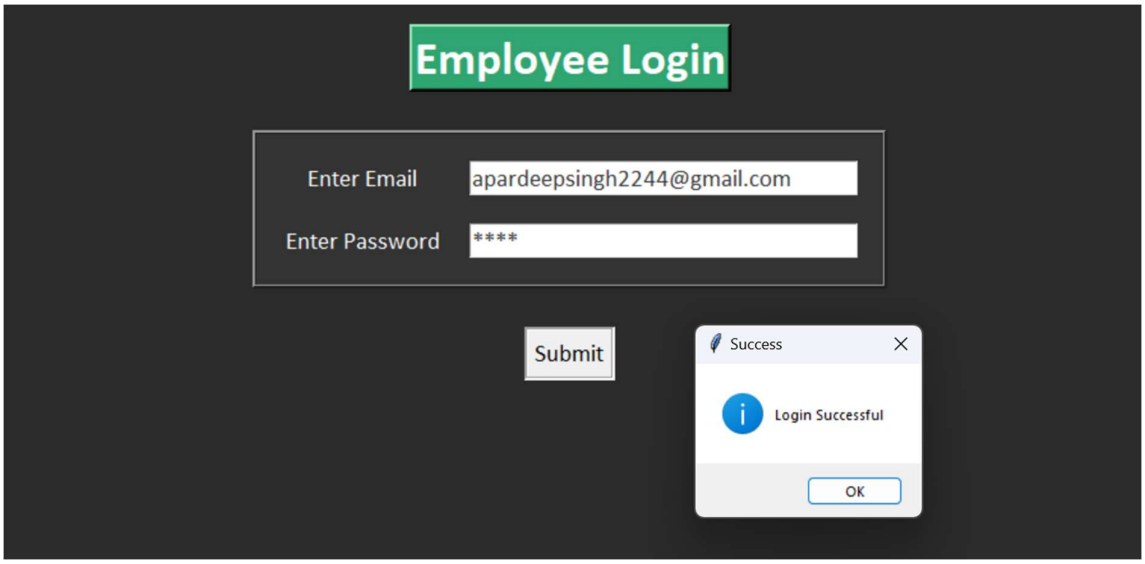
- **Test valid credentials(Admin):** Verify that users can successfully log in with valid username and password combinations.



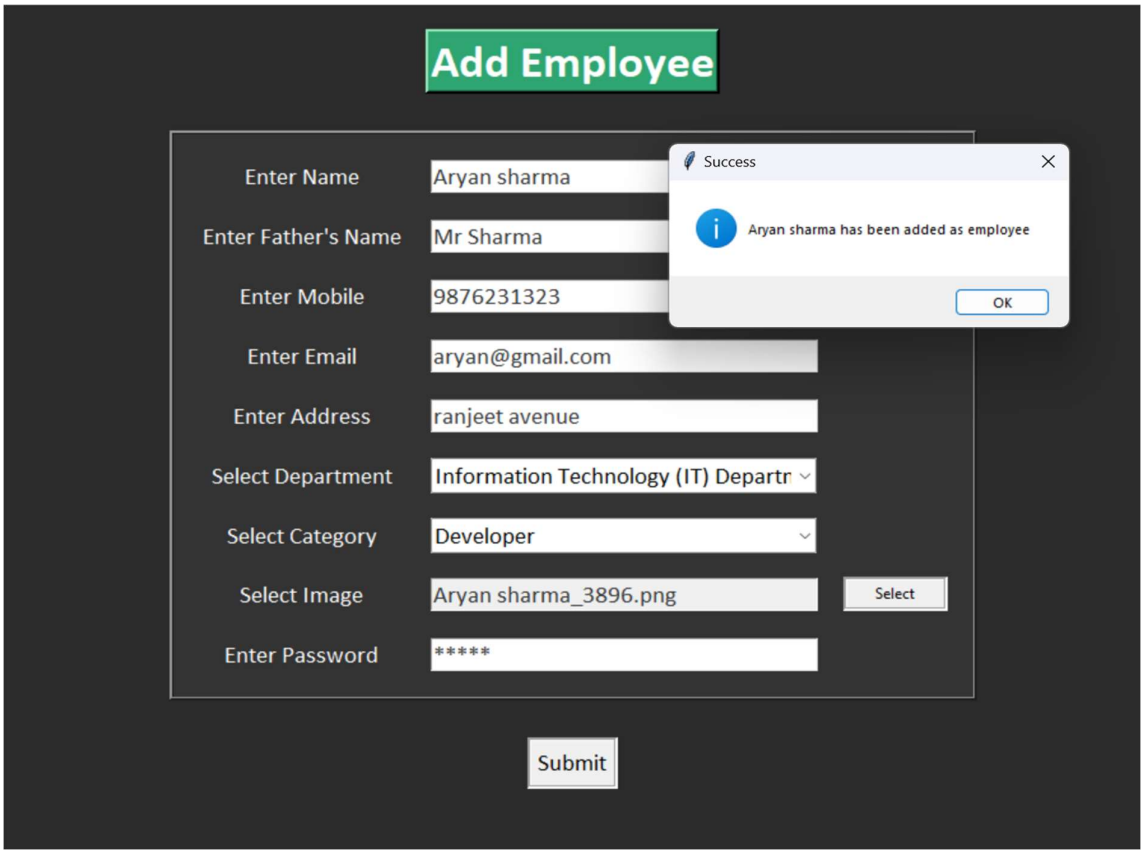
- **Test invalid credentials(Employee):** Validate that the system rejects login attempts with incorrect or invalid credentials.



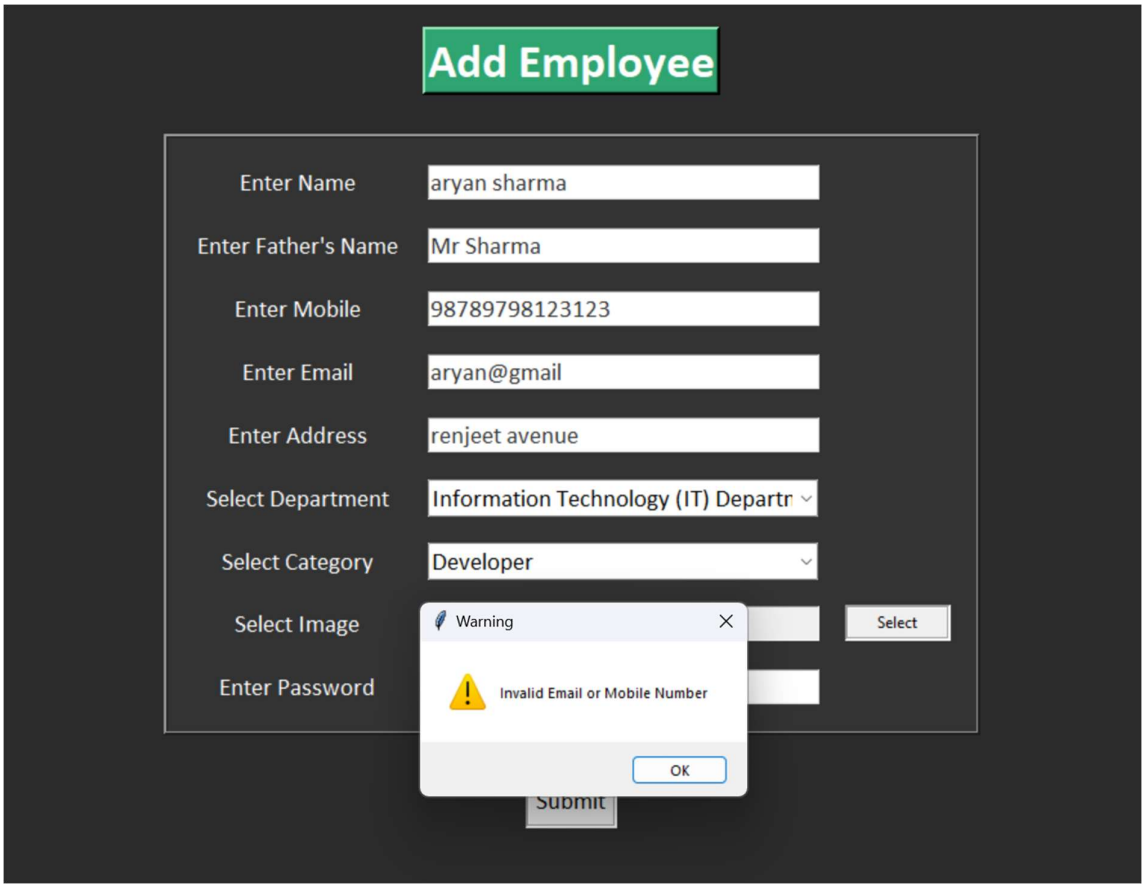
- **Test valid credentials(Employee):** Verify that users can successfully log in with valid username and password combinations.



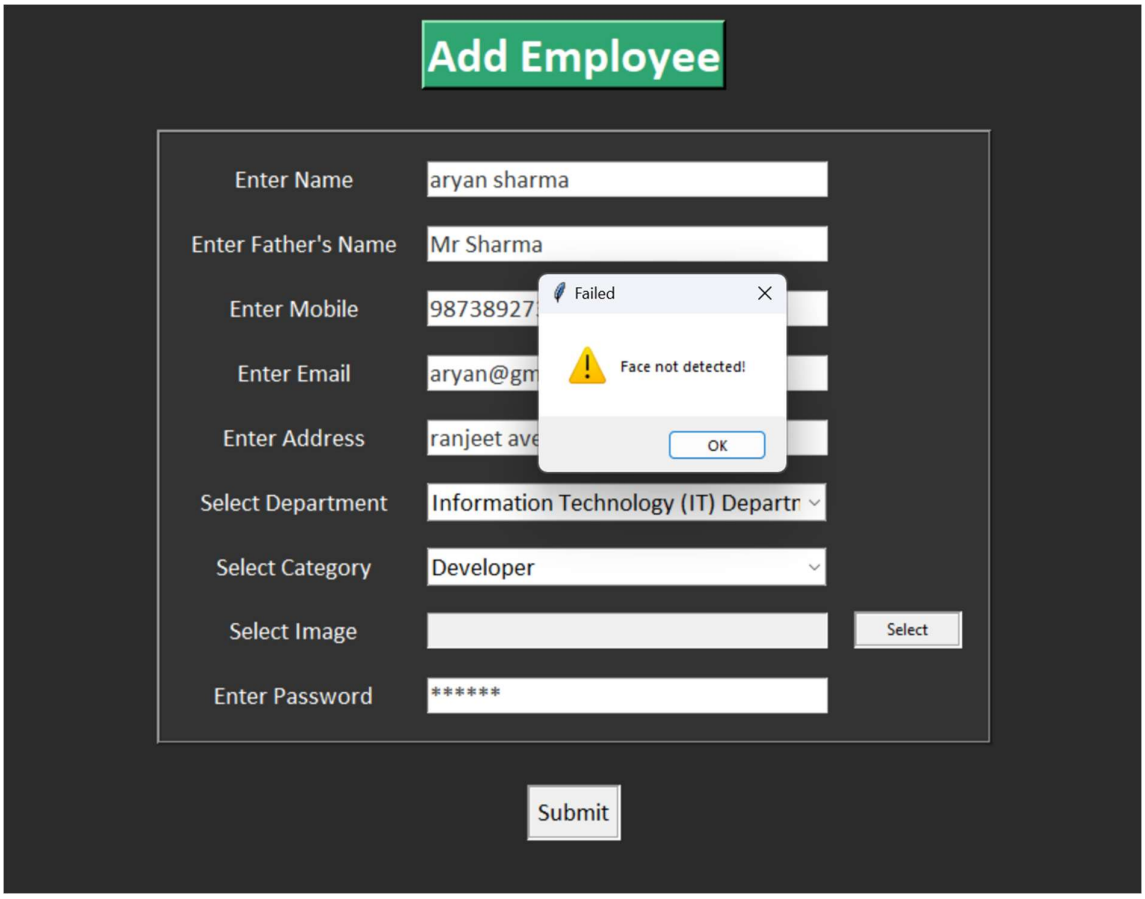
2. **Registration Employee:**
   - **Test registration process:** Verify that employees can be registered successfully by providing valid information.



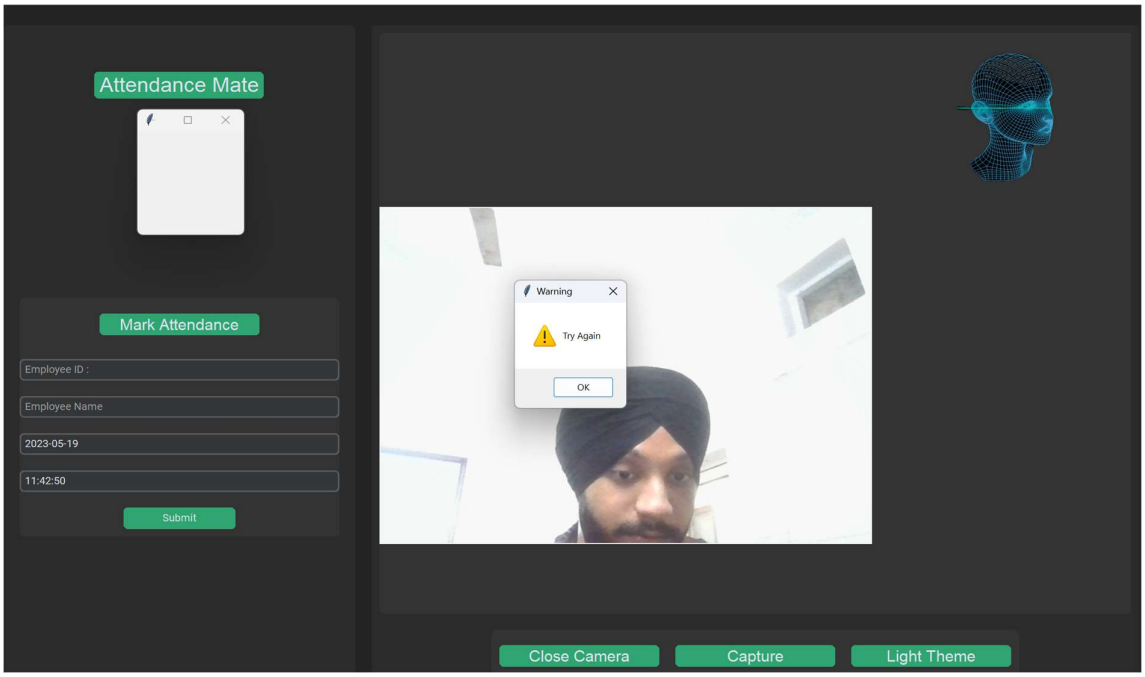
- **Test validation:** Validate that the system enforces proper validation rules for required fields and data formats.



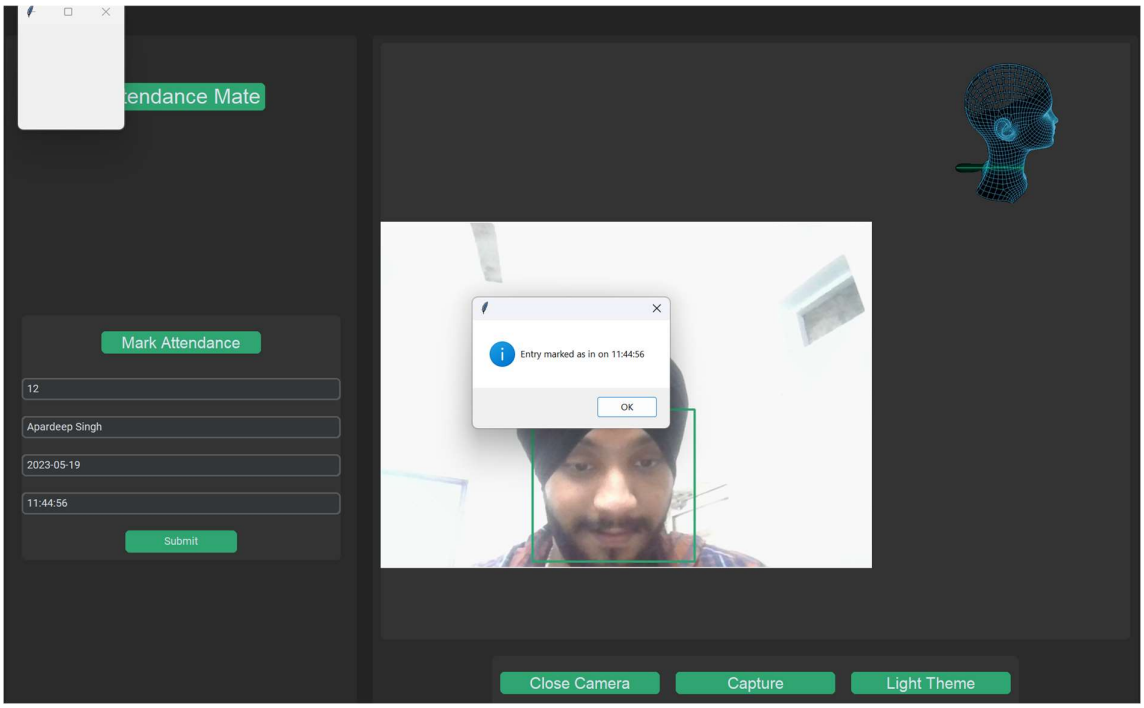
3. **Face Detection:**
   - **Test face detection algorithm:** Validate the accuracy and reliability of the face detection mechanism under various lighting conditions, angles, and facial expressions.



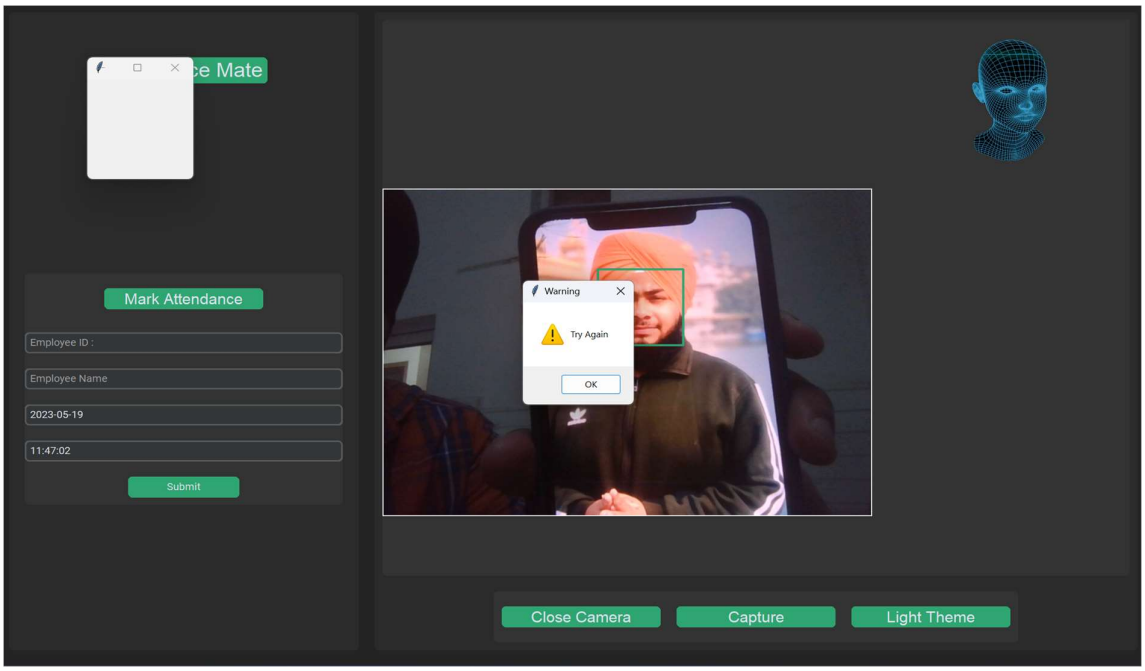
4. **Face recognition from registered database:**
    - **Test face recognition accuracy:** Validate the accuracy of the face recognition algorithm by comparing the detected faces with the registered employee database.



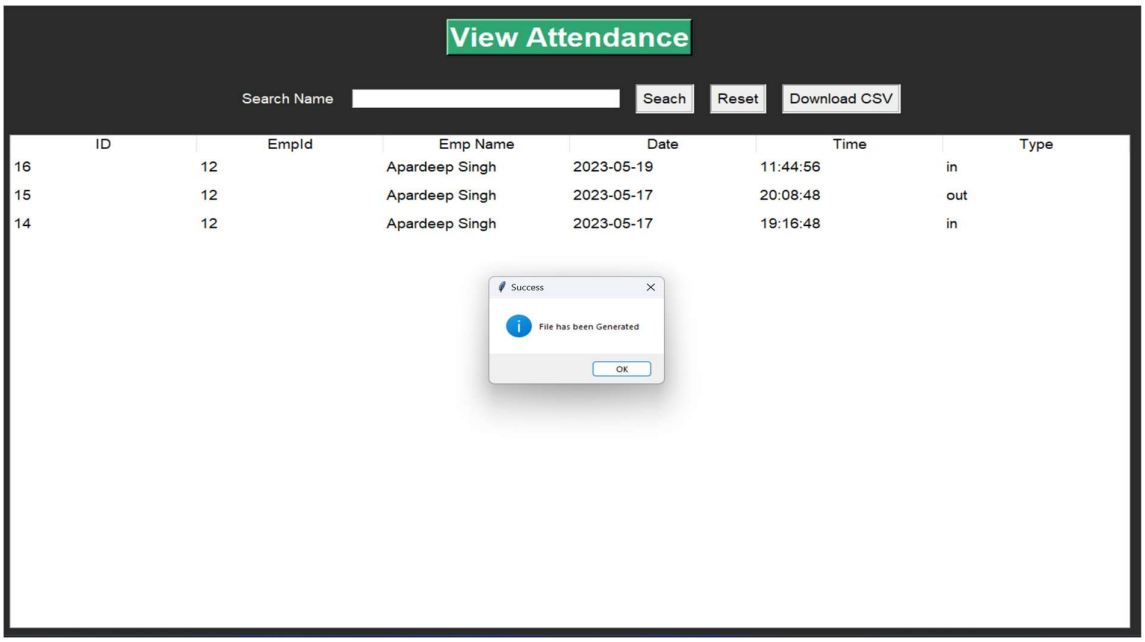
- **Test identification and verification:** Verify that the system correctly identifies registered employees and distinguishes them from unknown individuals.



5. **Unknown face detection:**
    - **Test unknown face detection accuracy:** Validate the system's ability to detect and flag unknown faces that are not present in the registered employee database.



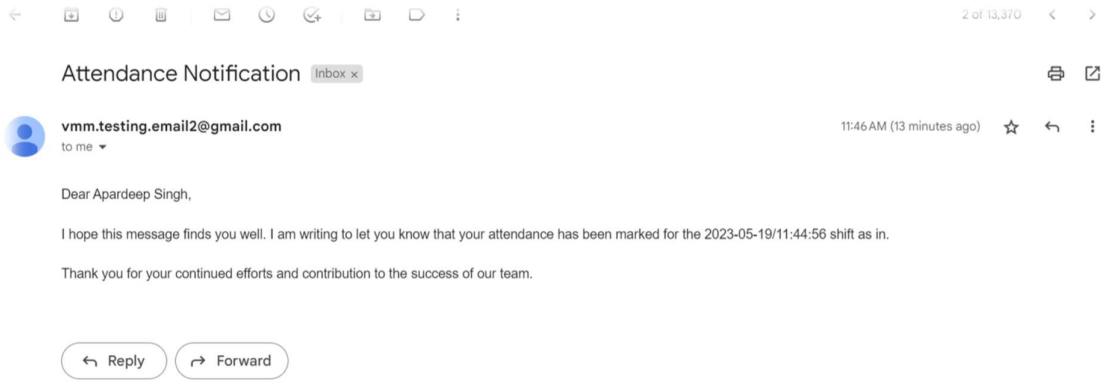
6. **Attendance entry in Excel sheet:**
    - **Test attendance recording:** Ensure that employee attendance is accurately recorded in the Excel sheet, including date, time, and employee details.



- **Test data integrity:** Verify that the attendance entries are correctly synchronized with the employee records and maintain data consistency.

	A	B	C	D	E	F	G
1	Attendance ID	Employee ID	Employee Name	Date	Time	Type	
2	16	12	Apardeep Singh	19-05-2023	11:44:56	in	
3	15	12	Apardeep Singh	17-05-2023	20:08:48	out	
4	14	12	Apardeep Singh	17-05-2023	19:16:48	in	
5							
6							
7							

7. **Email notification:**
   - **Test email sending:** Validate that the system sends email notifications to designated recipients for specific events, such as employee registration or attendance updates.



Bibliography

The following articles were referred during the development of this project.

1. OpenCV. OpenCV Documentation.[https://pypi.org/project/deepface/]

2. GeeksforGeeks. [https://www.geeksforgeeks.org/python-gui-tkinter/]

3. Stack Overflow. [https://stackoverflow.com/questions/28518072/play-animations-in-gif-with-tkinter]

4. Tutorialspoint. [https://www.tutorialspoint.com/create-a-date-picker-calendar-in-tkinter]

5. GitHub repositories. [https://github.com/opencv/opencv/blob/master/data/haarcascades/haarcascade_frontalface_default.xml]