People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
University M'Hamed Bougara – Boumerdès



Faculty of Sciences
Department of Computer Science

Domain : Mathematics and Computer Science
Field : Computer Science
Specialization : Information Systems (IS)

Final Graduation Project submitted for the degree of Bachelor's Degree in Computer Science

Theme

Biometric Attendance Management System

Presented by:
Djouaher Yacine
Saadi Mohamed
Brihmat Manel
Talbi Bassma Amina

Supervisor:
Mme. Sismail Warda

Defended on: .../.../ 2026 before the jury composed of:

Mme. Boughanem Hadjer : Examiner

Academic Year: 2025 - 2026

Acknowledgements

First and foremost, we express our deepest gratitude to God Almighty, whose guidance and blessings gave us the strength and determination to successfully complete this work.

We would like to extend our sincerest thanks to our supervisor, **Mme. Sismail Warda**, for her continuous support and guidance throughout the entire duration of this project. Her valuable advice, constructive feedback, and dedication were instrumental in shaping the quality of this work. Her expertise and pedagogical approach provided us with both scientific and methodological direction, for which we are truly grateful.

We also wish to express our appreciation to **Mme. Boughanem Hadjer**, the examiner of this thesis, for the interest she showed in our work and for her insightful remarks, which greatly contributed to enriching our reflection and improving the final outcome.

Our heartfelt gratitude goes to our families, whose unconditional moral support and constant encouragement throughout this journey meant more to us than words can express.

Finally, we warmly thank our friends and colleagues for their collaborative spirit, the enriching exchanges we shared, and the mutual support we offered one another. These shared moments not only helped the completion of this project but also made this experience truly memorable and rewarding.

Thank you all for your help, kindness, and friendship.

Yacine, Mohamed, Manel, and Bassma.

Abstract

Nowadays, many companies and institutions are looking for better ways to manage attendance and employee tracking. Traditional methods such as paper attendance sheets or card-based systems can cause several problems like human errors, data loss, and fake attendance registration. These methods are also slow and difficult to manage in large organizations.

This graduation project presents the development of a secure and automated Biometric Attendance Management System. The main goal of this system is to improve attendance tracking by using facial recognition technology instead of traditional methods. The system helps administrators monitor attendance in real time and reduces manual work.

The application is developed using a client-server architecture based on the MVC (Model-View-Controller) pattern. The user interface is built with Bootstrap 5, while the backend is developed using ASP.NET Core. The system also uses PostgreSQL for database management and Supabase for cloud file storage.

For biometric identification, the system integrates `face-api.js` to perform facial recognition directly in the browser. The webcam captures the employee's face, then the system compares it with stored facial data in the database. After successful verification, the application checks the employee's work schedule and records their attendance status such as "Present" or "Late".

The final system provides different features including employee management, attendance tracking, justification processing, holiday management, and reporting tools. By combining biometric technology with modern web technologies, the platform offers a reliable and scalable solution for attendance management while reducing fraud and administrative effort.

Keywords: Attendance Management, Facial Recognition, Biometrics, ASP.NET Core, MVC, PostgreSQL, face-api.js.

Contents

List of Figures . . . iii
List of Tables . . . iv
General Introduction . . . v

1 State of the Art **2**
1.1 Overview of Attendance Management Systems . . . 2
1.1.1 Definition and Objectives . . . 2
1.1.2 Evolution of Attendance Tracking Systems . . . 2
1.1.3 Comparison of Attendance Systems . . . 3
1.2 Biometric Time and Attendance Systems . . . 3
1.2.1 Main Components of Biometric Systems . . . 3
1.2.2 ZKTeco MB360 Device . . . 4
1.3 Analysis of Existing Systems and Their Limitations . . . 4
1.3.1 Main Functions of Attendance Systems . . . 4
1.3.2 Limitations of the Current System at UMBB . . . 5
1.4 Proposed System Architecture . . . 5
1.5 Synthesis . . . 5

2 Analysis and Design **7**
2.1 System Analysis . . . 7
2.1.1 Functional Requirements . . . 7
2.1.2 Non-Functional Requirements . . . 8
2.1.3 Identification of Actors . . . 8
2.1.4 Use Case . . . 9
2.1.5 Use Case Diagram . . . 10
2.1.6 The use cases description . . . 11
2.1.7 Sequence Diagrams . . . 12
2.2 Data Dictionary . . . 16
2.3 Class Diagram . . . 20

3 Implementation **24**
3.1 General Architecture of the Application . . . 24
3.1.1 MVC Model . . . 24

3.1.2 Deployment Diagram 25
3.2 Work Environment and Tools Used 26
3.2.1 Hardware Equipment 26
3.2.2 Software equipement 27
3.3 Application Interfaces 28
3.3.1 Dashboard 29
3.3.2 employee managment 29
3.3.3 Holidays 30
3.3.4 detailed reports 30
General Conclusion 32

List of Figures

1.1 ZKTeco MB360 Biometric Time Clock 4
2.1 Use Case Diagram: employee and attendance management 10
2.2 Sequence Diagram: authentification 13
2.3 sequence diagram - perform clocking 14
2.4 Sequence Diagram: process absence justification 15
2.5 Class diagram 20
3.1 Model–View–Controller (MVC) Architecture Schema 25
3.2 Deployment Diagram 26
3.3 Application Dashboard 29
3.4 Employee Management Interface 29
3.5 Holidays 30
3.6 detailed reports 30

List of Tables

1.1 Comparison of Attendance Systems 3
2.2 Description of Use Cases 12
2.3 Data Dictionary 19
3.4 Characteristics of the Machines Used 27

General Introduction

Nowadays, with the development of technology, many institutions use information systems to help manage administrative tasks. One of the important tasks is attendance management, which requires accurate recording, security, and good monitoring of employee attendance.

At the University M'Hamed Bougara of Boumerdes (UMBB), attendance management is based on a biometric card system. Although this system helps record employee attendance, it still has some limitations. Sometimes, synchronization problems can cause missing attendance records.In addition, the current card-based system presents security and reliability limitations, since cards can be forgotten, lost, stolen, or used by another person, which may compromise the accuracy of attendance verification.

This raises the following research question:

> **How can the current attendance system at UMBB be improved by creating a more secure, centralized, and accessible solution that provides reliable synchronization and better real-time attendance monitoring?**

To solve this problem, our graduation project focuses on the design and implementation of a **Biometric Attendance Management System** using facial recognition technology. The main objective of this solution is to improve attendance management by overcoming the limitations of the current card-based system through a web-based platform that offers better accessibility, reliability, and security.

This report is divided into three chapters:

- **Chapter 1 – State of the Art:** presents attendance management systems, biometric technologies, and the limitations of the current system used at UMBB.
- **Chapter 2 – Analysis and Design:** presents the functional analysis and UML modeling of the proposed system.
- **Chapter 3 – Implementation:** presents the architecture of the application, the development tools used, and the main interfaces of the developed system.

Chapter 1

State of the Art

State of the Art

Attendance management has changed a lot over the years. Today, many organizations are moving from traditional attendance methods to automated systems because they are more practical and reliable. In institutions and companies, keeping track of employee attendance is important because it affects productivity, salary calculation, and human resource management. Traditional methods, such as paper attendance sheets or simple badge systems, often cause problems like lost data, human mistakes, and attendance fraud, such as one employee signing in for another.

The main purpose of this chapter is to present an overview of attendance management systems and their development over time. It will discuss the evolution of attendance tracking technologies, explain the role of biometric systems in identity verification, present the ZKTeco MB360 biometric device used in this project, and analyze the limitations of the attendance system currently used at our institution. This study helps justify the development of the web-based platform proposed in our project.

1.1 Overview of Attendance Management Systems

1.1.1 Definition and Objectives

An attendance management system is a system used to record, monitor, and manage employee attendance automatically. Its main objective is to replace manual attendance methods with more accurate and real-time tracking solutions. In many organizations, attendance systems are connected with Human Resource Management systems to help in salary calculation, absence management, and employee supervision.[1]

1.1.2 Evolution of Attendance Tracking Systems

Attendance tracking systems have developed over time because of technological progress:[2].

- **Mechanical Systems:** In the past, attendance was recorded manually using paper sheets or punch card machines. These methods were simple but often caused errors and required a lot of administrative work. They were also easy to manipulate.
- **Electronic Systems:** Later, electronic systems such as RFID cards and PIN codes were introduced. These systems improved attendance recording, but they still had weaknesses because cards could be lost, stolen, or shared between employees, which made fraud possible.
- **Biometric Systems:** Modern attendance systems use biometric technologies, such as fingerprint or facial recognition, to identify employees. Since biometric information is unique to each person, these systems improve security and reduce fraud by ensuring that the actual employee is present.

1.1.3 Comparison of Attendance Systems

The following table presents a comparison between different attendance systems[3]:

Criteria	**Mechanical Systems**	**Electronic Systems (RFID)**	**Biometric Systems**
Data Accuracy	Low	Medium	High
Fraud Prevention	Weak	Weak	Strong
Administrative Effort	High	Medium	Low
System Type	Paper-based	Electronic	Web-based/Centralized

Table 1.1: Comparison of Attendance Systems

1.2 Biometric Time and Attendance Systems

1.2.1 Main Components of Biometric Systems

A biometric attendance system works by collecting and verifying biological information from users. Usually, these systems contain three main components:[4]

1. **Identification Sensor:** This part captures biometric data such as fingerprints or facial images.
2. **Processing and Storage Unit:** This unit processes the biometric data and stores attendance records locally to avoid losing information during network problems.

3. **Communication Interface:** This allows the device to connect to the central system through technologies such as TCP/IP or Wi-Fi for data synchronization.

1.2.2 ZKTeco MB360 Device

The biometric device used in this project is the ZKTeco MB360 terminal[5]. This device supports different authentication methods, including facial recognition, fingerprint scanning, PIN codes, and RFID cards.



Figure 1.1: ZKTeco MB360 Biometric Time Clock

The system starts with an enrollment process where employee biometric information is recorded and saved in the system. When an employee enters or leaves the workplace, they verify their identity using one of the available methods. If the identity matches the saved information, the system records attendance information, including the employee ID, date, time, and verification method.

1.3 Analysis of Existing Systems and Their Limitations

1.3.1 Main Functions of Attendance Systems

A Time and Attendance Management System (TAMS) helps organizations transform attendance data into useful information. In general, these systems provide several important functions, including:[6]

- **Schedule Management:** Creating work schedules and shifts for employees.

- **Attendance Monitoring:** Tracking employee presence and checking attendance records in real time.
- **Report Generation:** Producing reports about attendance, lateness, overtime, and absences.

1.3.2 Limitations of the Current System at UMBB

The attendance system currently used at the University M'Hamed Bougara of Boumerdes (UMBB) has several limitations:

- **Synchronization Problems:** Sometimes, the system fails to synchronize attendance data from biometric devices, which can lead to missing records.
- **Limited Accessibility:** The software is based on an old desktop system that requires installation on specific computers. This makes remote access difficult and limits real-time monitoring.

1.4 Proposed System Architecture

To solve the problems of the current system, this project proposes a modern web-based solution using a Service-Oriented Architecture (SOA). In this approach, the system and database are centralized on a web server, allowing administrators to access attendance information through a web browser. This solution makes the system easier to use because there is no need to install software on each computer. It also improves accessibility, system maintenance, and real-time synchronization with biometric devices.

1.5 Synthesis

This chapter presented the main concepts related to attendance management systems and biometric technologies. It explained the evolution of attendance systems from traditional methods to modern biometric solutions.

The chapter also analyzed existing attendance management platforms and identified their main limitations. These observations helped justify the development of the proposed solution presented in this project.

Chapter 2
Analysis and Design

Analysis and Design

This chapter is the main part of our technical study. It presents the conceptual design of the **Biometric Attendance Management System**. The goal is to give a clear idea about how the system works and how its different parts are organized. To do this, we use the **Unified Modeling Language (UML)** to describe both the structure and the behavior of the application before starting the implementation phase.

2.1 System Analysis

Before starting the design of our system, it is important to understand and study the users' needs and expectations. This step helps us identify the different tasks that the system must perform as well as the constraints that should be respected. In general, the requirements are divided into two categories: functional requirements and non-functional requirements.

2.1.1 Functional Requirements

Functional requirements describe the main functionalities that the system must provide to users. In our project, the functional requirements are the following:

- **Administrator :**
  - Authenticate to the system.
  - Manage employees (add, edit, and archive employees).
  - Manage teams and schedules.
  - Manage holidays and non-working days.
  - Consult attendance logs and reports.
  - Validate leave requests and absence justifications.

- **Employee :**
  - Perform clock-in and clock-out using facial recognition.
  - Consult attendance records.
  - Submit leave requests.
  - Submit justifications for absences or delays.
- **Biometric Terminal :**
  - Capture employee facial data.
  - Recognize employees using facial recognition.
  - Send attendance records to the system database.

2.1.2 Non-Functional Requirements

Non-functional requirements define the quality constraints and conditions that the system must respect. In our project, the non-functional requirements are the following:

- **Performance :** the system must recognize employees and record attendance in real time without significant delay.
- **Security :** access to the system must be protected by authentication using email and password.
- **Reliability :** the system must ensure accurate and stable attendance recording.
- **Ergonomics :** the interface must be simple and easy to use for administrators and employees.
- **Compatibility :** the application must work on modern web browsers.

2.1.3 Identification of Actors

Based on the system requirements and the functional scope, we identify the following actors:

- **Administrator:** The main actor responsible for system configuration, managing employee records, and monitoring attendance logs.
- **Employee:** The actor who interacts with the biometric terminal to perform clocking (check-in/check-out).
- **Biometric Terminal:** A secondary actor representing the hardware interface (fingerprint) that captures and transmits clock-in/out logs to the system to be processed.

2.1.4 Use Case

A use case represents a set of action sequences performed by the system that produce an observable result of interest for a particular actor. It allows for a description of what the future system must do, without specifying how it will do it .[7]

In our study, we will use two terms to specify the difference between our types of use cases:

- **C.R.U.D:** is a computing acronym (standing for **Create, Read, Update, Delete**) that refers to the four basic operations for data persistence, specifically the storage of information within a database.
- **Developed:** This is a use case representing a process containing at least one decision loop, potentially an ordinary flow, and several extensions or exceptions.

2.1.5 Use Case Diagram

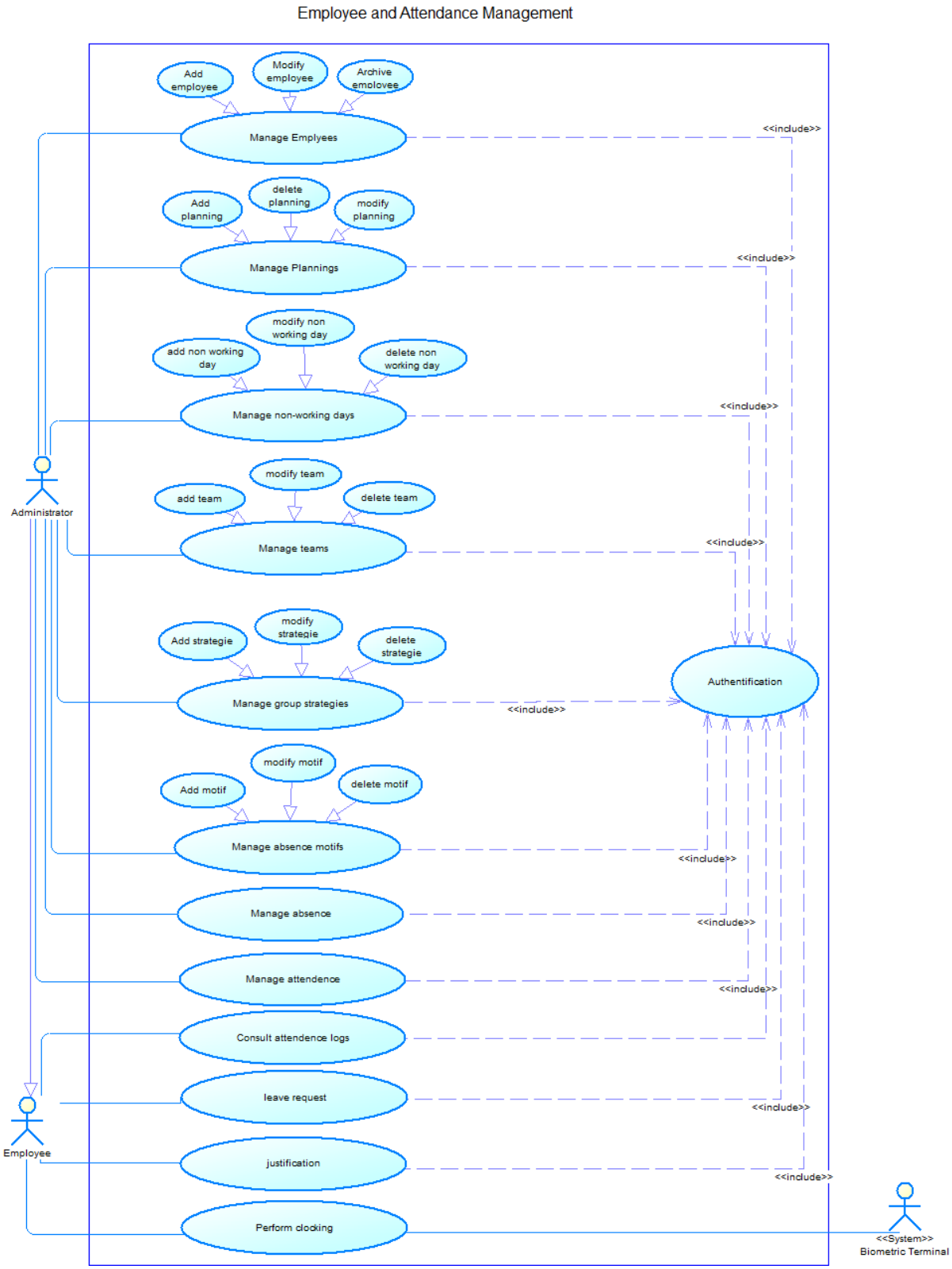


Figure 2.1: Use Case Diagram: employee and attendance management

2.1.6 The use cases description

Use Case	Description	Rules	Type
Manage Group Strategies	Add, edit, or delete a group strategy and assign personnel to it.	threshold entry time must be before threshold exit time	C.R.U.D
Manage Teams	Add, edit, or delete a team and assign staff members to it.	A team must have at least one staff member to be active	C.R.U.D
Manage Personnel	Add, edit, or archive a personnel record and assign them to a specific team and strategy.	A staff member is assigned to only one team and one strategy.	C.R.U.D
Manage Plannings	Define the time range for a schedule and the corresponding working hours.	A schedule is assigned to a team. An assigned schedule cannot be deleted.	C.R.U.D
Management of Non-working Days	Define the rest days or holidays for personnel.	A non-working day cannot overlap with an already defined holiday	C.R.U.D
Management of Absence Motifs	Add, delete, or edit reasons/motifs for absences.	Default motifs (sickness, family event) cannot be deleted	C.R.U.D
Consult Attendance Logs	View the clock-in/out records uploaded automatically		Developed
Manage Absence	Assign a motif to an absence and validate the absence based on that motif.	An absence can have only one motif. An absence cannot be assigned to a non-working day.	Developed
Manage Attendance	Retrieve the number of hours worked, total cumulative delays, and absences for a staff member for a given date range.	Includes administrator authentication.	Developed

Use Case	Description	Rules	Type
Perform Clocking	Identify via the biometric terminal to record entry or exit times.	clocking must be within 30 minutes of scheduled time to be valid	Developed
Leave Request	Submit a request for a leave of absence for a specific date range, pending administrator approval.	A leave request cannot overlap with an already approved request. A staff member can only submit a request for themselves.	Developed
Justification	Submit a written justification for an absence or a late arrival to the administrator.	A justification must be linked to an existing absence or anomaly record. Only one justification per absence is allowed.	Developed

Table 2.2: Description of Use Cases

2.1.7 Sequence Diagrams

The sequence diagram is a type of UML diagram that graphically represents the inter actions between different objects or actors within a system, as a function of time. It shows the chronological order of messages exchanged between these different elements during the execu tion of a specific scenario or use case. The sequence diagram helps visualize and understand complex interactions within a system, making it easier to design, document, and communicate dynamic application behavior[7]

a)" Authentication" Sequence Diagram

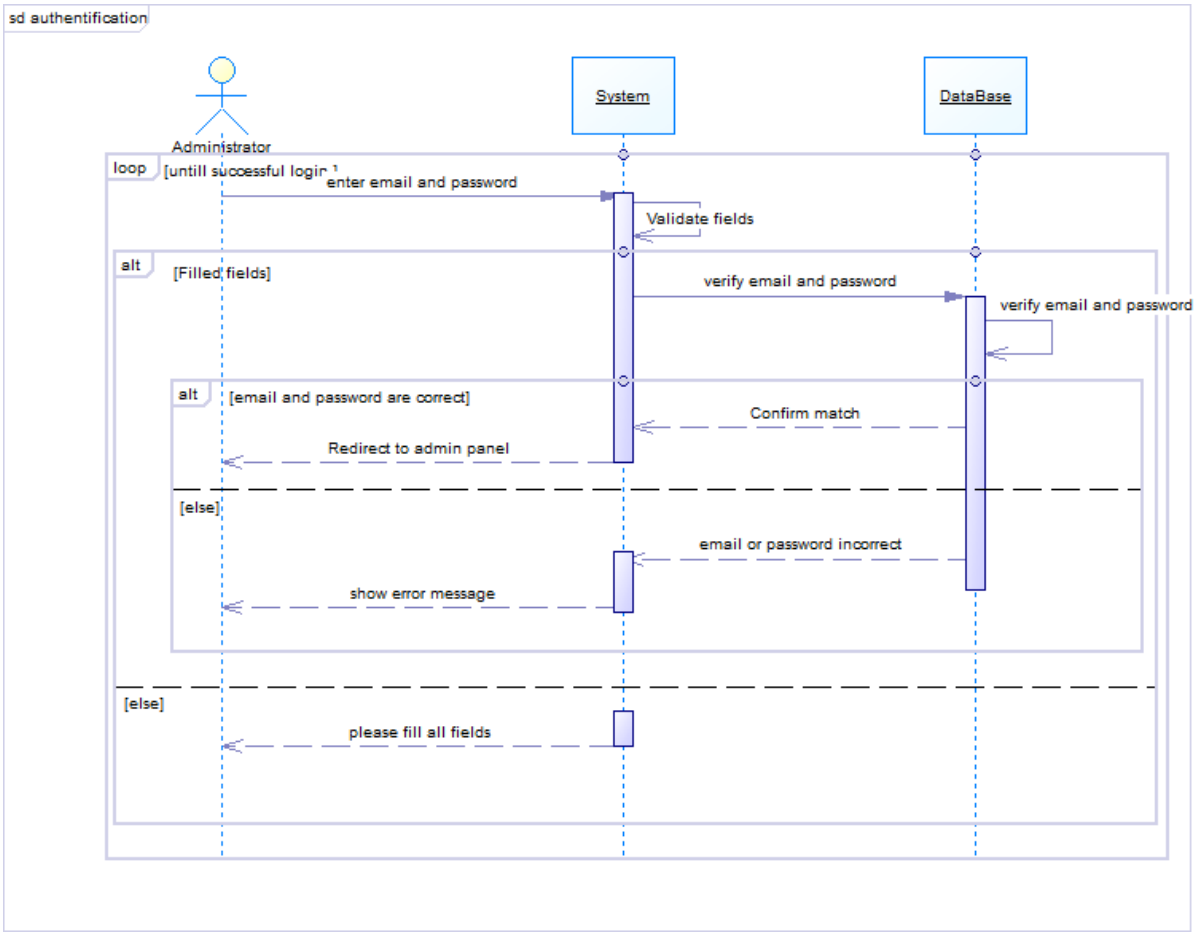


Figure 2.2: Sequence Diagram: authentification

The authentication process begins when the Administrator enters their email and password, which are submitted to the System for field validation. **If the fields are filled:** The System forwards the credentials to the Database for verification. **If the email and password are correct**, the Database confirms the match and the System redirects the Administrator to the admin panel. **If the credentials are incorrect**, the Database returns a mismatch and the System displays an error message to the Administrator. **If the fields are empty:** The System immediately prompts the Administrator to fill in all required fields

b)" perform clocking" sequence diagram

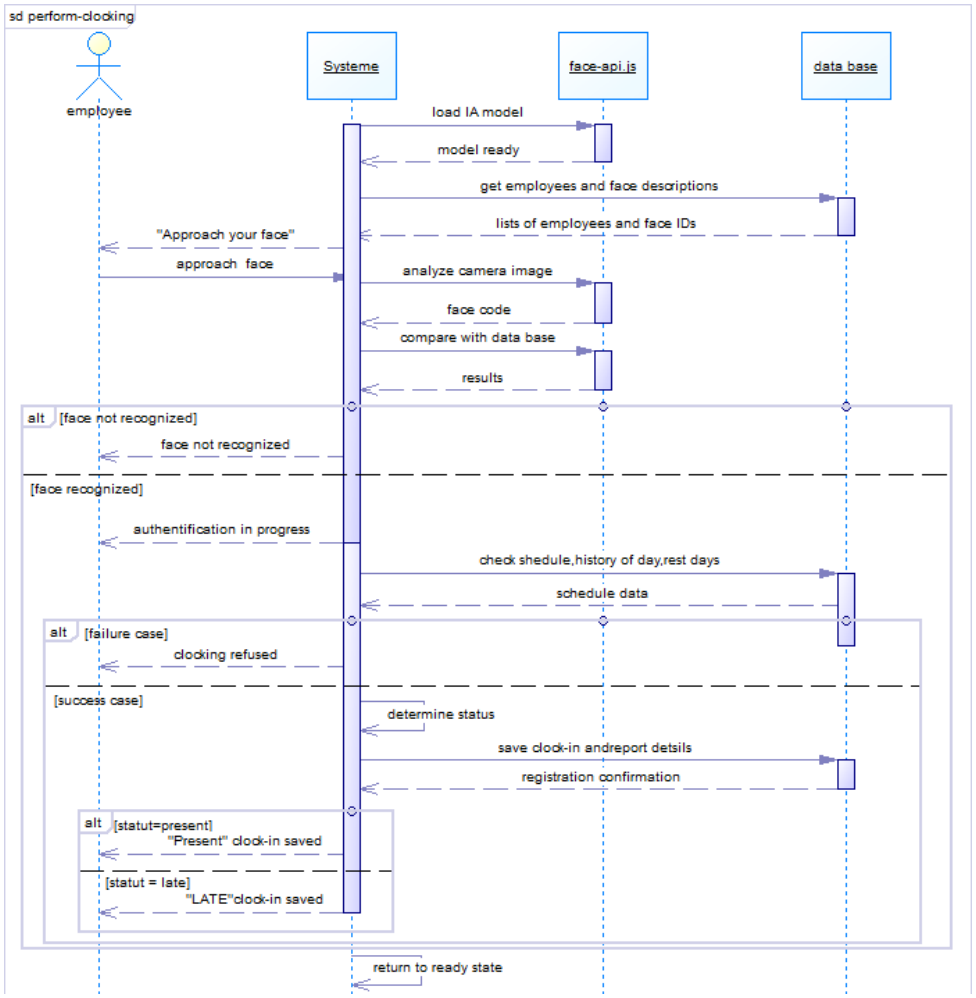


Figure 2.3: sequence diagram - perform clocking

The biometric clocking process starts by loading the AI models and retrieving all employees' facial descriptors from the database. Once ready, the employee approaches the camera, the system captures their face and uses `face-api.js` to compare it against stored descriptors.

If the face is recognized: The system checks the employee's schedule, determines their status as Present or Late, then saves the record to the database.

If the face is not recognized or the schedule check fails, the clocking is refused. Finally, the system resets and waits for the next employee.

c) "Process absence Justification" Sequence Diagram

The justification process begins when the administrator accesses the justification page after authentication. The system retrieves the list of justifications from the database and displays them.

The administrator then selects a pending justification to review its details and attached document.

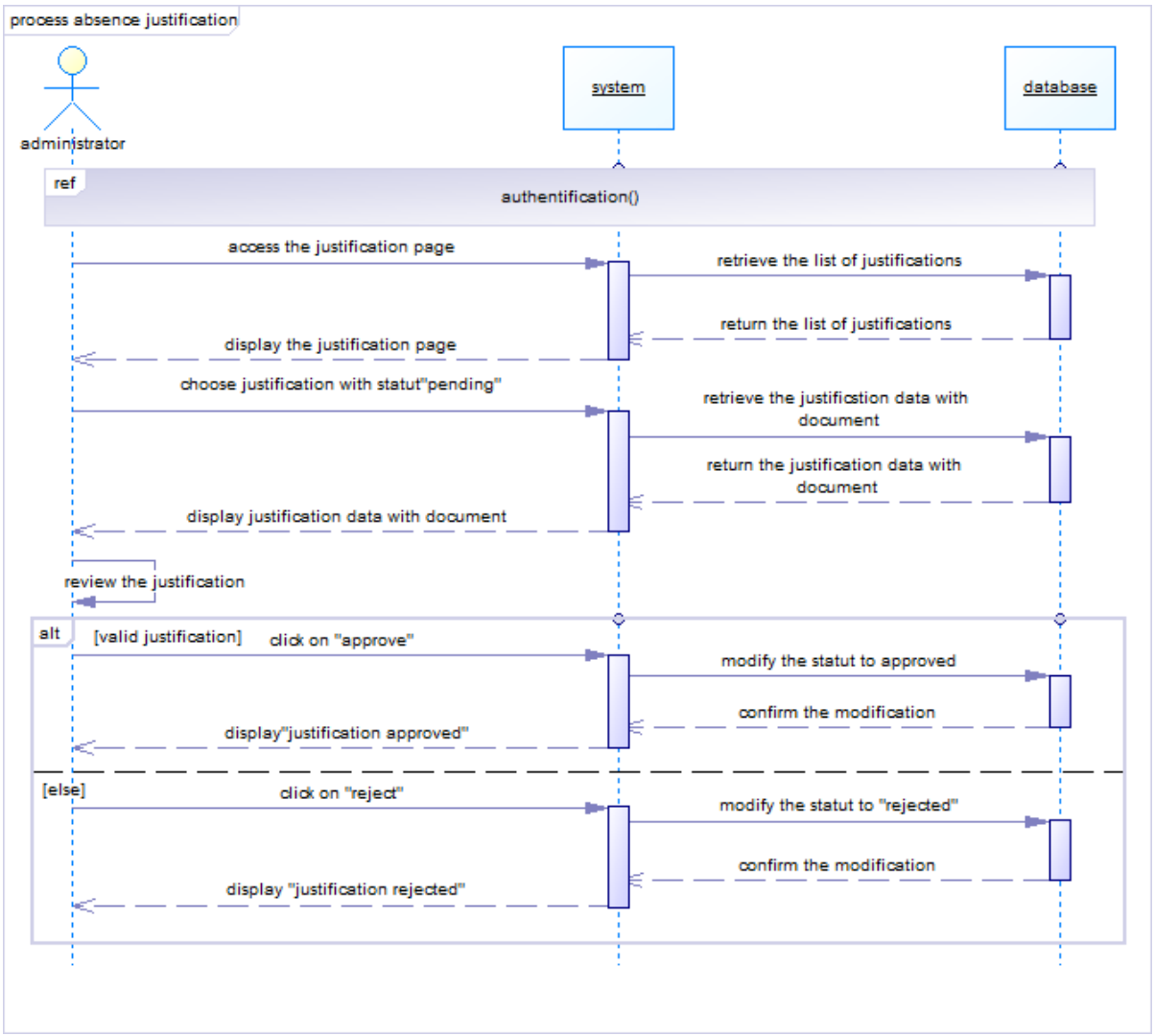


Figure 2.4: Sequence Diagram: process absence justification

If the justification is valid, the administrator approves it, and the system updates its status to "approved." Otherwise, the administrator rejects it, and the status is changed to "rejected." In both cases, the database confirms the modification and the system displays a success message.

2.2 Data Dictionary

Entity	Attribute	Description	Type
Schedule	idSchedule	Unique identifier of the schedule.	int
	ScheduleName	Name given to the specific schedule.	String
	StartTime	Standard expected start time of work.	time
	EndTime	Standard expected end time of work.	time
Public_Holiday	idHoliday	Unique identifier of the public holiday.	int
	HolidayName	Designation or name of the official holiday.	String
	HolidayDate	Specific calendar date of the holiday.	Date
Team	idTeam	Unique identifier of the working team.	int
	TeamName	Descriptive name of the team.	String
	StartDate	Date when this team profile was created.	Date
Group_Strategy	idGroup	Unique identifier of the strategy profile.	int
	GroupName	Descriptive label of the tracking strategy.	String
	EntryThreshold	Authorized margin or grace period for arrival.	time
	ExitThreshold	Authorized margin or buffer for exit time.	time
Employee	idEmployee	Unique identifier assigned to each employee.	int
	LastName	Employee family name.	String
	FirstName	Employee given first name.	String
	email	Corporate communication mail account.	String
	password	Encrypted verification string for profile access.	String
FaceID	idFace	Unique tracking index for the biometric profile.	int
	FaceDescriptor	Serialized tensor array for identity comparison.	String

Entity	Attribute	Description	Type
	PhotoReference	Filepath or server location of facial snapshot.	String
	CreationDate	Initial enrollment snapshot date.	DateTime
Employee_Archive	idArchive	Sequential key index for deleted tracking lists.	int
	idEmployee	Logged primary identifier of the worker.	int
	lastName	Historical archival record of employee last name.	String
	firstName	Historical archival record of employee first name.	String
	email	Archived address string.	String
	Team_id	Linked past operational team index assignment.	int
	Group_id	Historical configuration constraint profile indicator.	int
	reason	Specific legal/operational justification for archival.	String
	description	Narrative notes logging administrative cleanup actions.	String
	ArchiveDate	Precise timestamp tracking exit removal.	DateTime
	ArchivedBy	Verification key logging active execution actor.	String
Admins	idAdmin	Primary identification index of an administrator.	int
	username	Alphanumeric account access name.	String
	password	Security credentials for panel verification.	String
	email	System operations notification mailbox.	String
	fullname	Real identity of system administrator.	String
	createdAt	Timestamp defining administrative setup action.	DateTime

Entity	Attribute	Description	Type
Device	idDevice	Hardware network device tracking index key.	int
	deviceName	Descriptive identifier for physical site terminal.	String
	ipAddress	Static network endpoint definition string.	String
Clocking	idClocking	Master tracking reference index for hardware logs.	int
	ClockingDate	Chronological transaction checkpoint record.	DateTime
	type	Categorization value checking Entry or Exit direction.	String
Report_Detail	idDetail	Execution row tracking key for data compiling.	int
	detailDate	Operational reference calendar timeframe mark.	Date
	detailType	Status marker categorizing overall presence posture.	String
Absence	idAbsence	Exception management identification indicator.	int
	Isjustified	Logical confirmation boolean evaluating validity.	boolean
	reason	Structural diagnostic explanation notes fields.	String
Lateness	idLateness	System transaction latency key identifier.	int
	duration	Total deviation in minutes.	int
Leave	idLeave	Approved timeline transaction entity identity key.	int
	StartDate	Valid operational inception calendar boundary.	Date
	EndDate	Final formal expiration constraint checkpoint.	Date

Entity	Attribute	Description	Type
	LeaveType	Explicit categorization string classification option.	String
Leave_Requests	idRequest	Formal authorization log index pointer item.	int
	LeaveType	System tracking categorization profile statement.	String
	StartDate	Targeted inception boundary statement value.	Date
	EndDate	Target termination calendar parameter selection.	Date
	Reason	Explanatory text log outlining justification.	String
	Status	Verification phase evaluating current approval phase.	String
	requestDate	Chronological filing creation time capture entry.	DateTime
Justifications	idJustification	Document tracking unique database master index code.	int
	reason	Text metadata summary logging context description.	String
	filepath	Storage reference system pointer locating physical file.	String
	submissionDate	Ingestion execution transaction time tracking logging.	DateTime
	status	Administrative assessment monitoring check mark field.	String

Table 2.3: Data Dictionary

2.3 Class Diagram

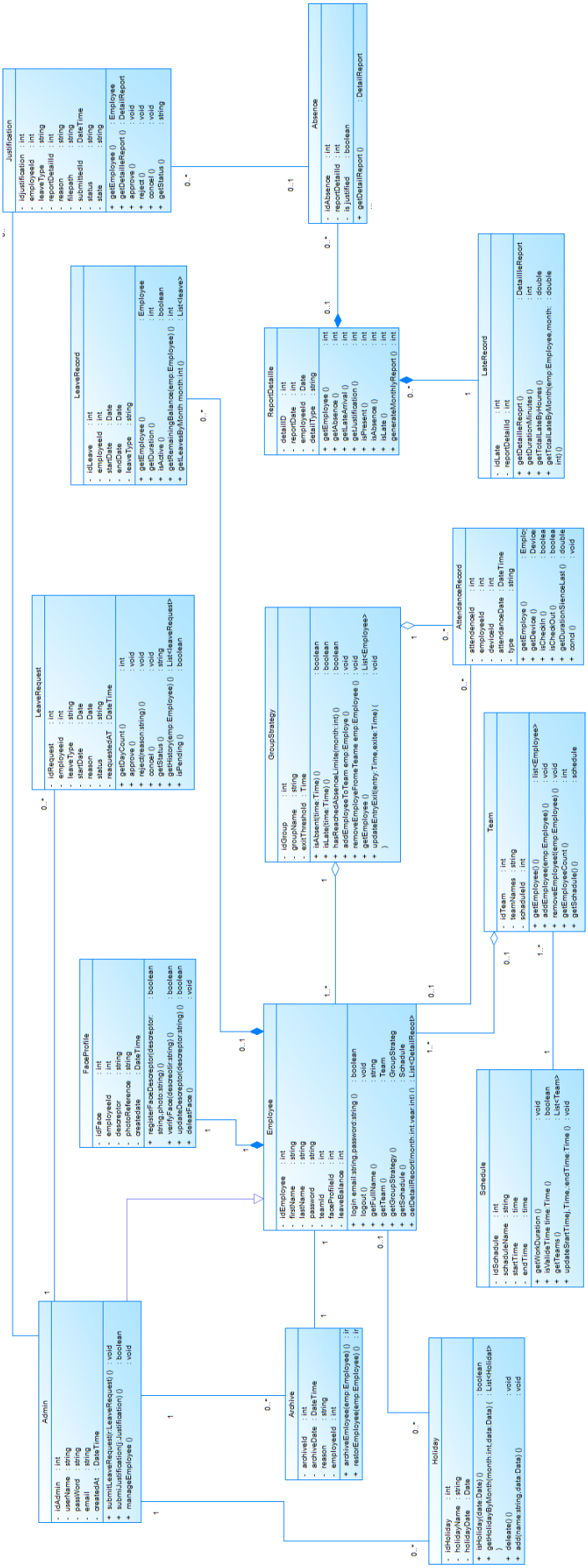


Figure 2.5: Class diagram

Relational Model

1. **Schedule** (idSchedule, scheduleName, startTime, endTime)
2. **Public_Holiday** (idHoliday, holidayName, holidayDate)
3. **Team** (idTeam, teamName, startDate,#idSchedule)
4. **Group_Strategy** (idGroup, groupName, entryThreshold, exitThreshold)
5. **Employee** (idEmployee, , lastName, firstName, email, password,#idTeam,#idGroup)
6. **FaceID** (idFace, faceDescriptor, photoReference, creationDate,#idEmployee)
7. **Device** (idDevice, deviceName, ipAddress)
8. **Clocking** (idClocking, ,clockingDate, type,#idDevice, #idEmployee)
9. **Report_Detail** (idDetail, detailDate, detailType,#idEmployee)
10. **Absence** (idAbsence, isJustified, reason,#idDetail)
11. **Lateness** (idLateness, duration,#idDetail)
12. **Leave** (idLeave, startDate, endDate, leaveType,#idEmployee)
13. **Leave_Requests** (idRequest, leaveType, startDate, endDate, reason, status, requestDate, #idEmployee)
14. **Justifications** (idJustification, , reason, filePath, submissionDate, status,#idEmployee,#idDetail)
15. **Employee_Archive** (idArchive, idEmployee, lastName, firstName, email, idTeam, idGroup, reason, description, archiveDate, archivedBy)
16. **Admins** (idAdmin, username, password, email, fullname, createdAt)

Conclusion

In this chapter, we analyzed and modeled the Biometric Attendance Management System through UML diagrams and a detailed data dictionary. These results serve as a solid foundation for the implementation phase presented in the next chapter.

Chapter 3

Implementation

Implementation

This chapter presents the technical and architectural choices adopted to ensure the performance, security, and maintainability of the system. We describe the overall architecture of the solution as well as the organization of the different application layers.

We then present the hardware equipment used during development. We also present the architecture adopted for structuring the application, based on the MVC model.

Finally, some screenshots of the main interfaces will be presented to illustrate the operation of the application and the results obtained.

3.1 General Architecture of the Application

3.1.1 MVC Model

In our application, we adopted the MVC (*Model–View–Controller*) architecture. This is a design pattern that structures an application into three distinct parts to separate the data, the user interface, and the control logic [8].

According to InterviewBit [8], MVC architecture is a software design pattern that separates an application into three main logical components: the Model, the View, and the Controller. Each of these components is built to handle specific development aspects of an application. MVC is one of the most frequently used industry-standard web development frameworks to create scalable and extensible projects.

This approach allows for better code organization, easier maintenance, and makes the application more scalable.

- **The Model** manages the data and business logic. It also handles interactions with the database and processing of management rules. The Model component corresponds to all the data-related logic that the user works with. This can represent either the data that is being transferred between the View and Controller components or any other business logic-related data [8].
- **The View** corresponds to the interface displayed to the user. It presents the data and manages interaction with the user. The View component is used for all the UI logic of the

application [8].

- **The Controller** acts as an intermediary between the Model and the View. It receives user actions, processes the data via the Model, then returns the result to the View. Controllers act as an interface between Model and View components to process all the business logic and incoming requests, manipulate data using the Model component, and interact with the Views to render the final output [8].

In our project, this organization facilitates exchanges between the *frontend*, the *backend*, and the database, while maintaining a clear and easily scalable structure

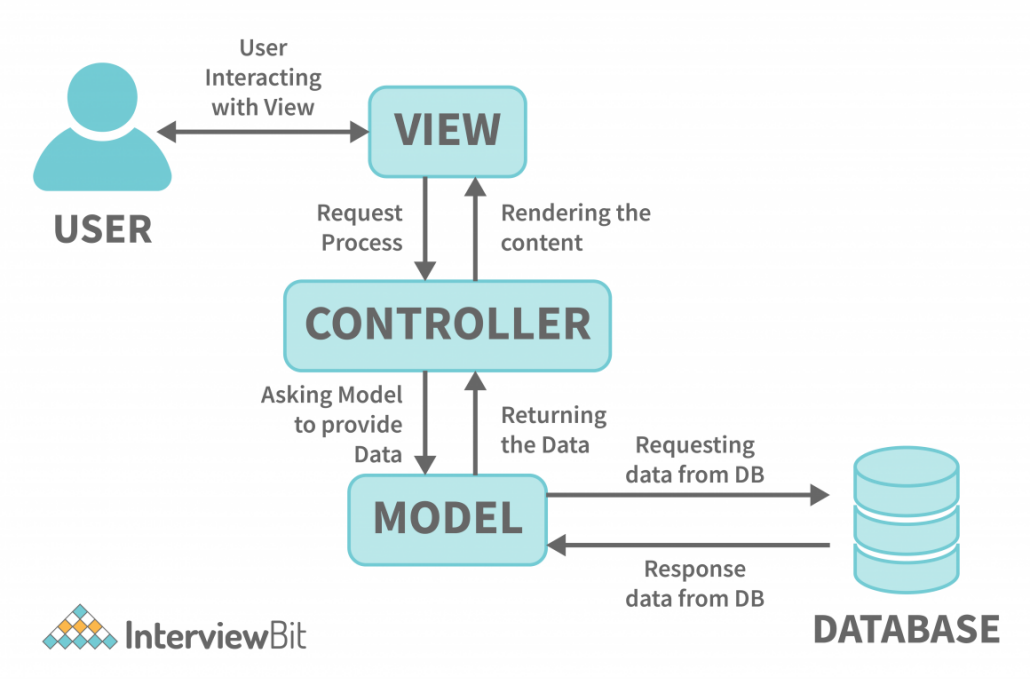


Figure 3.1: Model–View–Controller (MVC) Architecture Schema

3.1.2 Deployment Diagram

The deployment diagram below illustrates the physical architecture of the facial recognition attendance system. It highlights three main nodes: the user workstation running the web interface via a browser, the local server hosting the ASP.NET Core backend, and the database server.

Our application uses **face-api.js**, a JavaScript library built on top of TensorFlow.js that implements several convolutional neural network models for face detection, facial landmark detection, and face recognition directly in the browser or Node.js environment. This enables real-time AI-powered facial recognition without requiring a dedicated server-side ML infrastructure.

Exchanges between the *frontend* and *backend* are performed via HTTP/REST requests in JSON format, while communication with the database is ensured via SQL over TCP/IP. The face-api.js module runs entirely in the browser, processing the webcam stream locally for face detection and recognition.

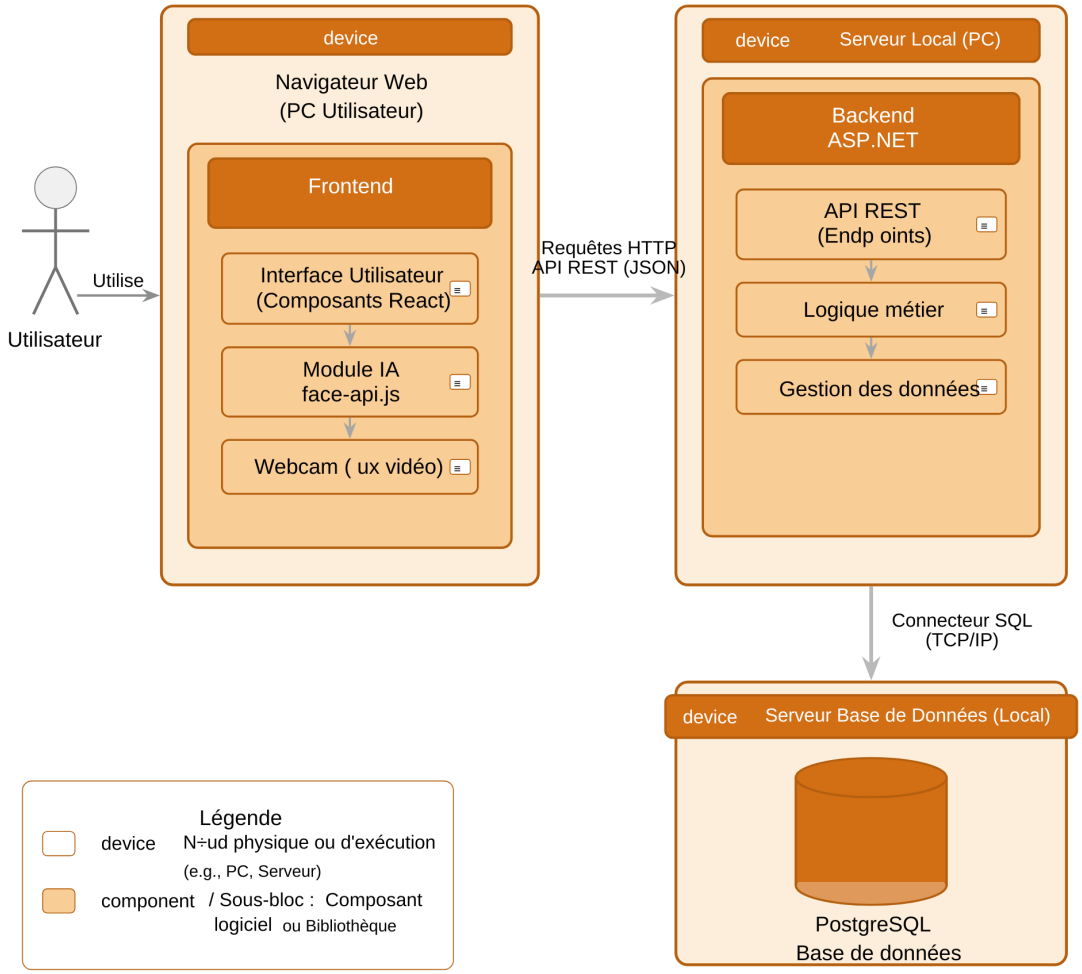


Figure 3.2: Deployment Diagram

3.2 Work Environment and Tools Used

3.2.1 Hardware Equipment

The following table presents the characteristics of the physical equipment we used to carry out this project. These two Lenovo machines served as server and client respectively in the deployed architecture.

Machine Name	Processor	RAM Memory	System Type	Operating System	Used as
Lenovo	AMD Ryzen 5 PRO 2500U w/ Radeon Vega Mobile Gfx	10.9 GB (11 GB)	64-bit	Windows 10	**SERVER**
Lenovo	Intel(R) Core(TM) Ultra 5 125U	15.3 GB (16 GB)	64-bit	Windows 11	**Client**

Table 3.4: Characteristics of the Machines Used

3.2.2 Software equipement

ASP.NET Core

ASP.NET Core 8 is an open-source, cross-platform framework developed by Microsoft for building modern, cloud-based, and internet-connected applications. It is the latest long-term support release of ASP.NET Core, offering improved performance, minimal APIs, and enhanced support for building RESTful web services and MVC web applications [9].

PostgreSQL

PostgreSQL is a powerful, open-source object-relational database management system (ORDBMS) that uses and extends the SQL language. Known for its reliability, feature robustness, and performance, it supports advanced data types and is fully ACID-compliant, making it suitable for a wide range of applications.[10]

Supabase

Supabase is an open-source Backend-as-a-Service (BaaS) platform that provides cloud file storage, authentication, and real-time database capabilities built on top of PostgreSQL. It offers developers a scalable and easy-to-use infrastructure for storing and serving files and data . [11]

Entity Framework



Entity Framework is an open-source object-relational mapper (ORM) for .NET developed by Microsoft. It enables developers to work with relational databases using .NET objects, eliminating the need for most of the data-access code that developers typically need to write. It supports a code-first approach, allowing database schemas to be defined directly through C# classes .[12]

Bootstrap 5



Bootstrap 5 is the latest version of the world's most popular open-source CSS framework for developing responsive and mobile-first web interfaces. It provides a comprehensive collection of pre-built UI components, a flexible grid system, and utility classes, enabling developers to build consistent and visually appealing web applications efficiently .[13]

Visual Studio



Visual Studio is an integrated development environment (IDE) developed by Microsoft. It is used to develop computer programs, as well as websites, web apps, web services, and mobile apps. In this project, Visual Studio was used as the primary development environment for writing, debugging, and compiling the C# backend code with ASP.NET Core 8 and managing Entity Framework migrations efficiently [15].

face-api.js

face-api.js is a JavaScript library built on top of TensorFlow.js that implements several convolutional neural network models for face detection, facial landmark detection, and face recognition directly in the browser or Node.js environment. It enables real-time AI-powered facial recognition without requiring a dedicated server-side ML infrastructure .[14]

3.3 Application Interfaces

Here we present the main interfaces of the developed application.

3.3.1 Dashboard

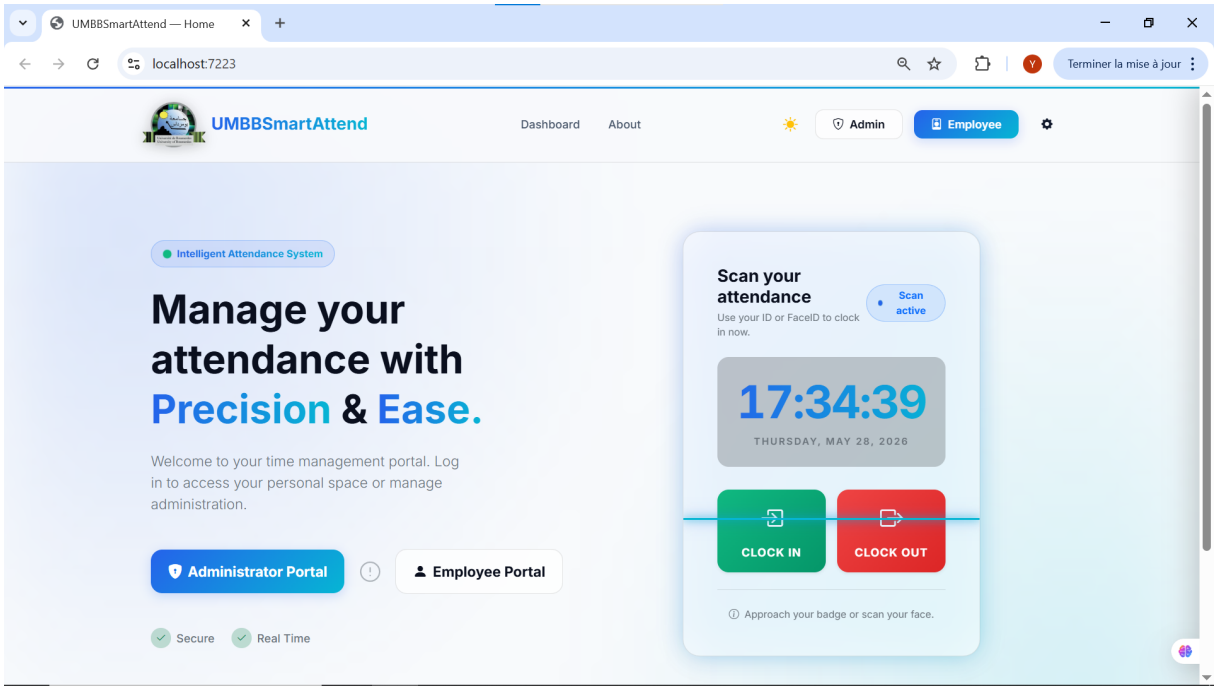


Figure 3.3: Application Dashboard

3.3.2 employee managment

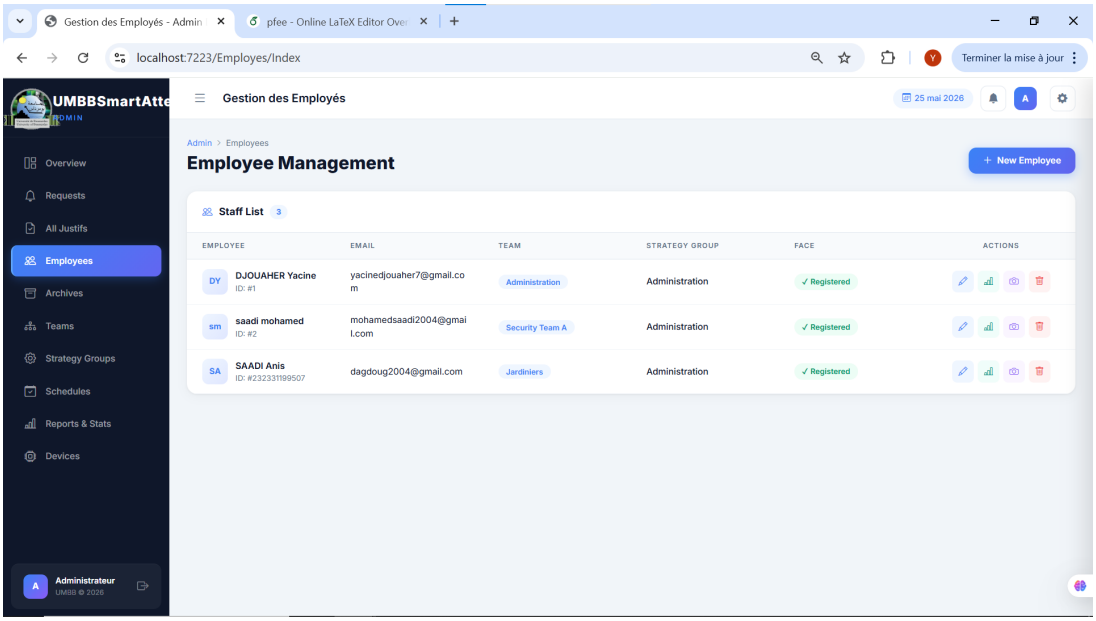


Figure 3.4: Employee Management Interface

3.3.3 Holidays

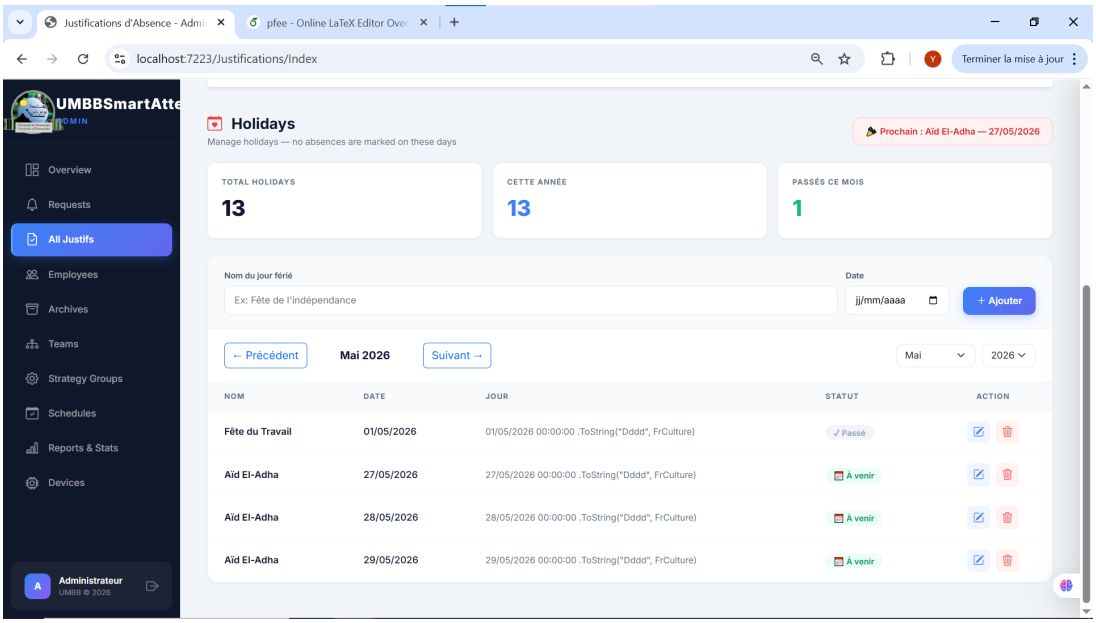


Figure 3.5: Holidays

3.3.4 detailed reports

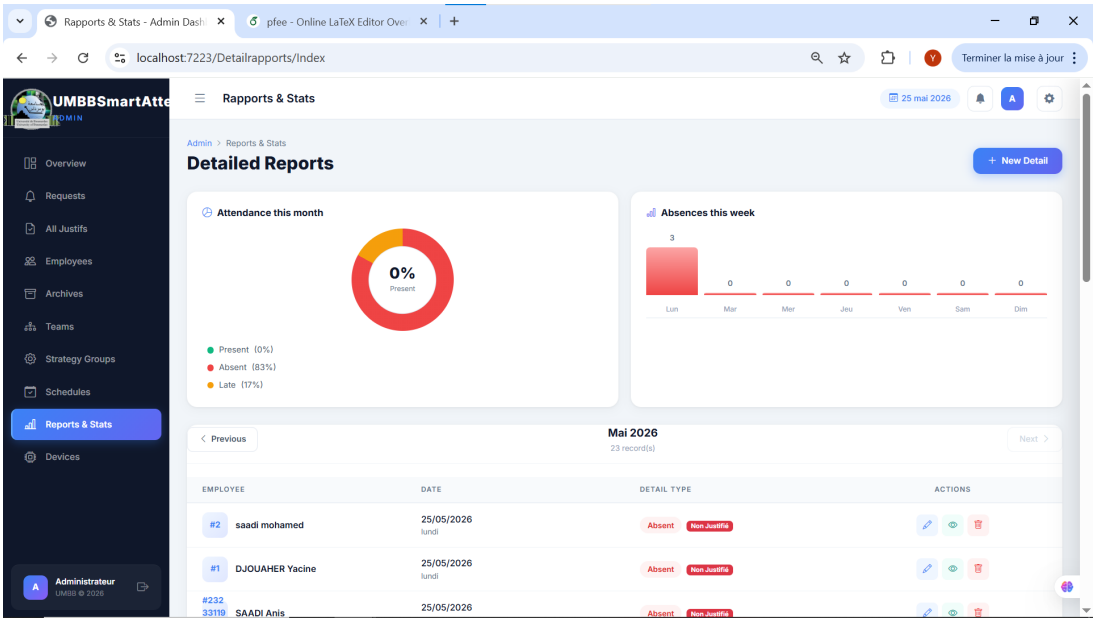


Figure 3.6: detailed reports

Conclusion

In conclusion, this chapter presented the practical implementation of our Biometric Attendance Management System. We showcased the main user interfaces, demonstrating how administrators can easily manage employee records, configure holidays, and consult detailed tracking reports. The successful development of these features proves that the system is fully operational, user-friendly, and capable of replacing traditional, manual methods with a secure and automated environment.

General Conclusion

This project set out to address a real administrative challenge faced by the Faculty of Sciences at the University M'Hamed Bougara Boumerdes: the unreliability and vulnerability of the existing card-based attendance system. In response, we designed and implemented a **Biometric Attendance Management System** that leverages facial recognition technology to automate and secure the process of tracking employee presence.

Throughout the development process, we successfully delivered a fully functional web-based application that covers the complete attendance management lifecycle, from employee enrollment and automated clocking to absence handling, leave management, and report generation. The system eliminates the risk of proxy clocking and identity fraud, providing the administration with reliable, real-time attendance data.

Despite the results achieved, time limitations prevented us from implementing several additional features we had envisioned for the system:

- **Mobile Application:** A dedicated mobile application for employees would improve accessibility, allowing staff to consult their records and submit requests directly from their smartphones.
- **Integrated Chatbot:** An AI-powered assistant for both employees and administrators, enabling users to get instant answers to their questions without contacting HR manually.
- **Predictive Holiday Management:** An AI model capable of automatically predicting and pre-populating public holidays for the entire year, relieving the administrator from repetitive manual entry.
- **Automated Payroll Reporting:** An AI-powered module capable of automatically generating and sending detailed monthly attendance reports to the payroll responsible, reducing manual effort and minimizing errors in salary calculation.
- **Enterprise System Integration:** Connecting the platform with third-party systems such as HRMS and payroll software, enabling broader workplace automation and seamless data exchange across the organization.

Overall, this project allowed us to apply the knowledge acquired throughout our academic training in a concrete and meaningful way, bridging the gap between theoretical concepts and real-world software engineering challenges.

Bibliography

[1] Michael Armstrong and Stephen Taylor. *Armstrong's Handbook of Human Resource Management Practice*. Kogan Page Publishers, 16th edition, 2023.

[2] K. Ishaq and S. Bibi. *IoT Based Smart Attendance System Using RFID: A Systematic Literature Review*. International Journal of Advanced Computer Science and Applications, 2023. URL: `https://arxiv.org/abs/2308.02591`. [Visited on: May 2026].

[3] N. S. Ali, A. H. Alhilali, H. D. Rjeib, H. Alsharqi, and B. Al-Sadawi, "Automated attendance management systems: systematic literature review," *International Journal of Technology Enhanced Learning*, vol. 14, no. 1, pp. 37–65, 2022. `https://www.researchgate.net/publication/358178607_Automated_attendance_management_systems_systematic_literature_review`[Visited on: May 2026]

[4] I. Apostolescu and M. Shafiei, Eds., *Husserl and Leibniz: Metaphysics, Monadology and Phenomenology*, `https://link.springer.com/book/10.1007/978-3-031-78880-2`[Visited on: May 2026]

[5] ZKTeco. *ZKTeco MB360 Multi-Biometric Time & Attendance and Access Control Terminal User Manual*. Manuals+, 2024. URL: `https://manuals.plus/asin/B073XFWDHS`. [Visited on: May 2026].

[6] A. Mishra and S. Tarar. *Design and Architecture of Automated Time and Attendance Management Systems in Enterprise Environments*. International Journal of Information Technology and Computer Science, 2020. URL: `https://www.mecs-press.org/ijitcs/ijitcs-v12-n3/v12n3-4.html`. [Visited on: May 2026].

[7] Martin Fowler. *UML Distilled: A Brief Guide to the Standard Object Modeling Language*. Addison-Wesley Professional, 4th edition, 2024.

[8] InterviewBit. *MVC Architecture: Model, View, Controller Framework Design Explained*. InterviewBit Blog, 2025. URL: `https://www.interviewbit.com/blog/mvc-architecture/`. [Visited on: May 2026].

[9] Jonathan Danylko. *ASP.NET 8 Best Practices*. Packt Publishing, 2023.

[10] PostgreSQL Global Development Group. *PostgreSQL 17 Official Core Documentation*. PostgreSQL Documentation Project, 2025. URL: `https://www.postgresql.org/docs/`. [Visited on: May 2026].

[11] Supabase Inc. *Supabase Architecture and Open-Source Backend-as-a-Service Documentation.* Supabase Ingestion Docs, 2026. URL: `https://supabase.com/docs`. [Visited on: May 2026].

[12] Julie Lerman and Arthur Vickers. *Modern Entity Framework Core: Data Access for .NET.* O'Reilly Media, Inc., 2024.

[13] Bootstrap Team. *Bootstrap v5.3 Official Documentation: Responsive Frontend Framework.* Bootstrap Framework Docs, 2025. URL: `https://getbootstrap.com`. [Visited on: May 2026].

[14] R. Moughten. *Browser-Based Machine Learning with Face-api.js and TensorFlow.js.* Packt Publishing, 2nd edition, 2024.

[15] Microsoft. *Visual Studio 2022 v17.12 Integrated Development Environment Documentation.* Microsoft Learn Docs, 2025. URL: `https://learn.microsoft.com/en-us/visualstudio/`. [Visited on: May 2026].

[16] A. Madani and S. Oulaceb. *Conception d'un système de gestion de données de pointage biométrique.* Mémoire de Licence Professionnelle, Université M'Hamed Bougara - Boumerdès (UMBB), 2016.