Ministry of Higher Education and Scientific Research

University M'HAMED BOUGARA – Boumerdes

Institute of Electrical and Electronic Engineering

Department of Electronics

Final Year Project Report Presented in Partial Fulfilment of the Requirements for the Degree of

MASTER

In **Electronics**

Option: **Computer engineering**

Title:

Design and Implementation of a Web-Based Management System. Case of The Payroll Office of the "IGEE".

Presented by:

- **Ladgi Lina**
- **Fodil Sara**

Supervisor:

Dr. A. Zitouni

Registration Number..... /2021

Acknowledgements

Praise to God, the supreme for leading us during all these years of our study path, and for giving us the courage and the strength to realize this report successfully, despite all the difficulties, Alhamdulillah.

Our deep gratitude and thanks go to Dr. Abdelkader Zitouni, Our teacher and supervisor who had followed this work with great interest, we would like to express our deep respect and gratitude toward his presence despite all the responsibilities and duties that rest on his shoulders, his help and his precious orientation as well.

Secondly, we would like to thank Dr. Khalifa for his valuable course about databases that provide big help to realize this project.

Finally, may all people who helped us throughout this journey, find here an expression of our deep gratitude.

Thank you

Dedication

I dedicate this work to my family and all my friends and everyone who kept me moving forward through this journey.

Lina

I dedicate this work to my family who has been a constant source of support and encouragement during

the challenges of school and life.

And many friends who supported me

through the journey.

Sara

Abstract

As a variety of web technologies continues to grow, developers are able to create fast, responsive and modern applications.

The goal of this report is to lay the foundations for web-based employee control management system dedicated to the payroll office staff in the institute. Such a web application makes it possible to facilitate the work of the office by automating the management of the employees and the calculations. The developed application is easy to set up and maintain by users with little or no technical expertise.

This design work related to the application has been carried out using the development process and the modelling language, Unified Modelling Language (UML), for designing the application. As for the implementation, we have chosen to program the application with PHP (Hypertext pre-processor), MYSQL, in conjunction with the standard HTML, CSS, and JavaScript programming languages.

Keywords: Web application, PHP, MYSQL, HTML, CSS, JavaScript and UML.

Table of Contents

Table of Contents

Table of Contents

List of Abbreviations

- **CMS** Control Management System.
- **CSS** Cascading Style Sheets.
- **DBMS** Database Management System.
- **GIT** Global Income Tax
- **HTML** Hypertext Mark-up Language.
- **HTTP** Hypertext Transfer Protocol.
- **IP** Internet Protocol
- **MySQL** My Structured Query Language.
- **PHP** Hypertext Pre-processor.
- **SQL** Structured Query Language.
- **UML** Unified Modelling Language.
- **URL** Universal Resource Locator.
- **WAMP** Windows, Apache, MySQL, and PHP.

List of Figures

List of Tables

General Introduction

With the development of technologies and the worldwide expansion of the internet and the World Wide Web in the last few years, our lives have improved and most our daily activities have become easier to achieve and that all would be hard to imagine without web applications. In 1990s, the web was only used to share documents, the concept of web application appeared in the Java language in 1999 and by 2005 web applications have shifted from static to dynamic applications that are responsive and allow users to interact with them.

A web application is a program that is stored on a remote server and delivered over the internet through a browser interface. It is built with web technologies such as client side scripting languages Html, CSS, and JS, and server side scripting languages PHP and Python. It can be designed for a wide variety of uses and can be used by anyone; from an organization to an individual for numerous reasons, one of these reasons is to do payroll.

Each company which employs people must have a payroll office to take care of employees' management, documents reception and verification, salaries calculation and many more which is time and resources consuming. The payroll office of our institute is currently using old version software which generates a lot of problems such as bugs and latency.

This report aims at describing the whole process of the development of a dynamic and interactive web application from the initial to the final implementation to help the payroll office employees and facilitate their job.

This thesis is divided into three chapters. The first chapter states the objectives of this work and focuses on the role and tasks to be achieved by the payroll office of the IGEE institute, in addition it provides an overview about web applications, their functioning and the tools used to develop them. The second chapter describes the design of the web application using the unified modelling language (UML) and some generalities about databases along with the database used to store application's data. The last chapter presents the actual implementation and interfaces of the web application.

Chapter one

Project overview

Chapter One: Project Overview

I. Introduction

In this chapter we will first state the objectives of our project, then introduce web application and some concepts concerning web operations. We will also introduce the payroll office through a brief overview. Finally, we will define the tools and technologies used in building this web application.

II. Objectives and web application:

1.1 Subject presentation

Our subject is to design and implement a web application to help the institute payroll office. This theme is proposed and chosen in order to solve the problems the staff department encounters.

1.1.1 Problem statement

In order to implement our project, we conducted several conversations with the office staff that led us to discover some problems facing the department:

- No automated process exists in manipulating data such as addition, insertion and deletion.
- Each employee must submit a document of his attendance along the semester.
- Dealing with a lot of documents and files.
- The data stored on the software can only be accessed on the offices' computer and cannot be seen online.
- The software is difficult to use and contains a lot of bugs.

1.1.2 Objective

Our work aims at helping to solve the weaknesses mentioned above, and the objectives can be summarized as follows:

- Automate the process of the department staff's tasks such as calculations and management.
- Registrations and validation of the employee attendance along the year will be online.
- Updating of the employee information will be done on the web application.
- Make data access easier for the user.

1.2 Web application

1.2.1 Definition

A Web application (Web app) is an application program that is stored on a remote server and delivered over the Internet through a browser interface. Unlike traditional desktop application, which is launched by the operating system.

Web apps have several advantages over desktop applications. Since they run inside web browsers, developers do not need to develop web apps for multiple platforms. For example, a single application that runs in Chrome, will also work on both Windows and OS X. Developers do not need to distribute software updates to users when the web app is updated. By updating the application on the server, all users have access to the updated version **[1].**

1.2.2 How web applications work?

Web applications do not need to be downloaded since they are accessed through a network. Users can access a Web application through a web browser such as Google Chrome, Mozilla Firefox or Safari.

For a web app to operate, it needs a Web server, an application server, and a database. Web servers manage the requests that come from a client, while the application server completes the requested task. A database is used to store any needed information.

Web applications typically have short development cycles and can be achieved with small development teams. Most Web apps are written in JavaScript, HTML5, or Cascading Style Sheets (CSS). Client-side programming typically utilizes these languages to build an application front-end. Server-side programming is used to create the scripts a Web app will require to add functionality. Languages such as Php,Python, Java, and Ruby are commonly used in server-side programming **[2].**

1.2.3 Web server and web client

Web client: is a software running on the client side that is responsible for communication with the web server. The most commonly used web clients are web browsers such as Google chrome, Firefox, Mozilla and Safari.

When a user requests something from a server (via a URL) the web client creates a request and sends it to the server and then it parses the server response and presents it to the user.

Web server: is a software running on the server that processes the client requests and delivers the response back to the client. The primary function of a web server is to store, process and deliver web pages or data to clients upon request from a web browser. For example, Apache is one of the most widely used web server.

A web server respond to the client request in either of the following two ways:

- Sending the file to the client associated with the requested URL.
- Generating a response by invoking a script and communicating with database

1.2.4 HTML and HTTP

For communication between Web Servers and Web Clients to be established two essential tools must be provided:

- A common language for communication: **H**yper**T**ext **M**ark-up **L**anguage (HTML)
- A common communication protocol: **H**yper**T**ext **T**ransfer **P**rotocol (HTTP)

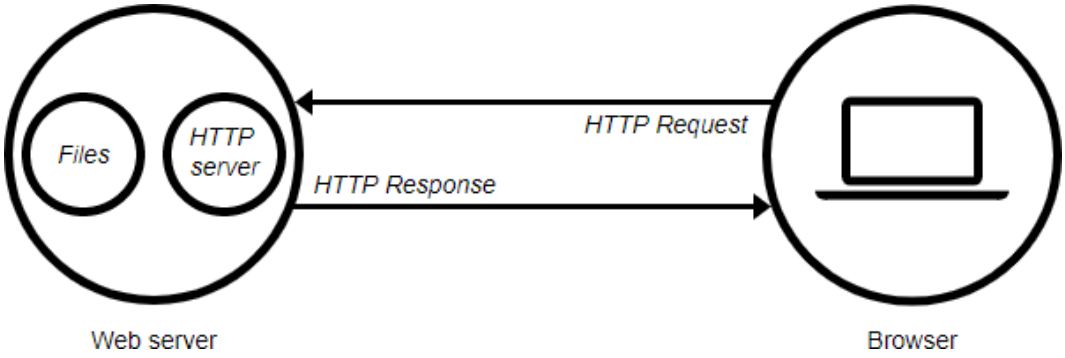


Figure 1.1 Web server and web client communication

HTTP runs on top of TCP/IP communication protocol governing the requests and responses that take place between the browser running on the end user's computer and the web server.

1.2.4.1 HTTP Request

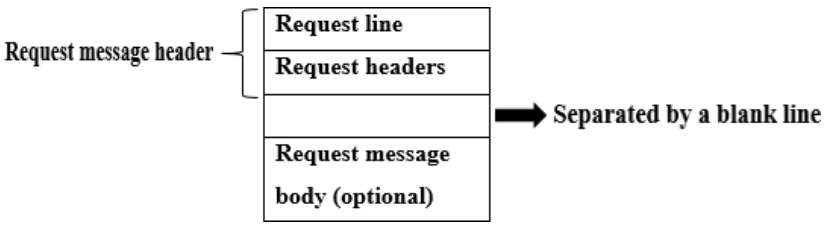


Figure 1.2 HTTP request message format

- **Request Line:** has the following parameters:
  - **Request-method:** Indicates the HTTP methods such as GET, POST, DELETE, PUT.
  - **URL**: Uniform Resource Locator to identify the resource on the server.
  - **HTTP Version**: Indicates the HTTP version.
- **Request Header:** provide information about the **request** context, so that the server can tailor the response.
- **Request Body** : Message content or Resource representation **[3].**

1.2.4.2 HTTP Response

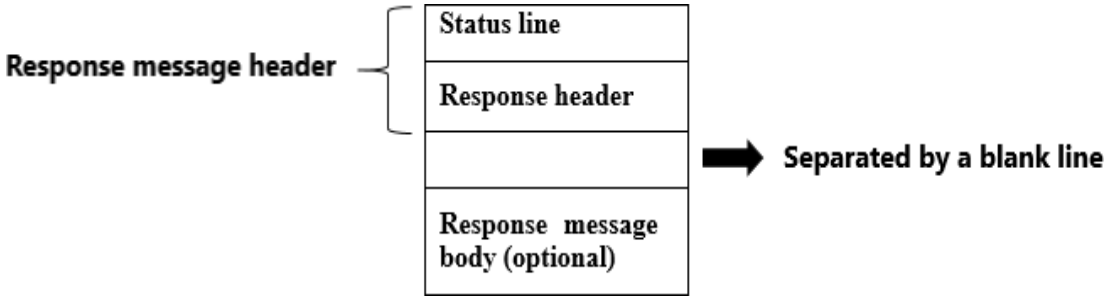


Figure 1.3 HTTP response message format

- **Status Line:** has the following parameters:
  - **HTTP-version:** Indicates the HTTP version.
  - **Status-code:** 3-digit that indicate the Server status for the requested resource.
  - **Reason-phrase:** gives a short explanation to the status code.
  - Examples of status line are:
    HTTP/1.1 200 OK
    HTTP/1.0 404 Not Found
    HTTP/1.1 403 Forbidden
- **Response Headers:** contains the date, size and type of file that the server is sending back to the client and also data about the server itself.
- **Response message body**: Contains the resource data that was requested by the client **[3].**

1.2.5 URL's

1.2.5.1 Definition

URL stands for *Uniform Resource Locator* is a specific type of Uniform Resource Identifier (URI).A URL represents the address of a given unique resource on the Web. Each resource on the web has its own unique address.

1.2.5.2 Format

A URL takes on a certain format through which you can create connections and/or access information using various protocols:

"protocol://domainOrAddress:Port/path/filename"

- **Protocol:** Indicates what protocol will be used to access the URL. Some common protocols include HTTP (Hyper Text Transfer Protocol) for transferring web content. The protocol is followed by ://
- **Domain or IP Address**: Specifies an IP address or domain where the resource is located.
- **Port:** Specifies the port to use for the connection. It is usually omitted, as default ports are used for various protocols such as port 80 for HTTP.
- **Path**: Specifies a subdirectory at the server where the resource is located. If the path is omitted, then the root directory at the server is checked for the resource.
- **Filename**: The actual name of the resource. If the filename is omitted, then a default filename is used such as index.html for an HTTP connection **[4].**

1.2.6 Static vs. Dynamic web pages

1.2.6.1 Static web pages

Static web pages are ones that are fixed and display the same content for every user, usually written exclusively in HTML.

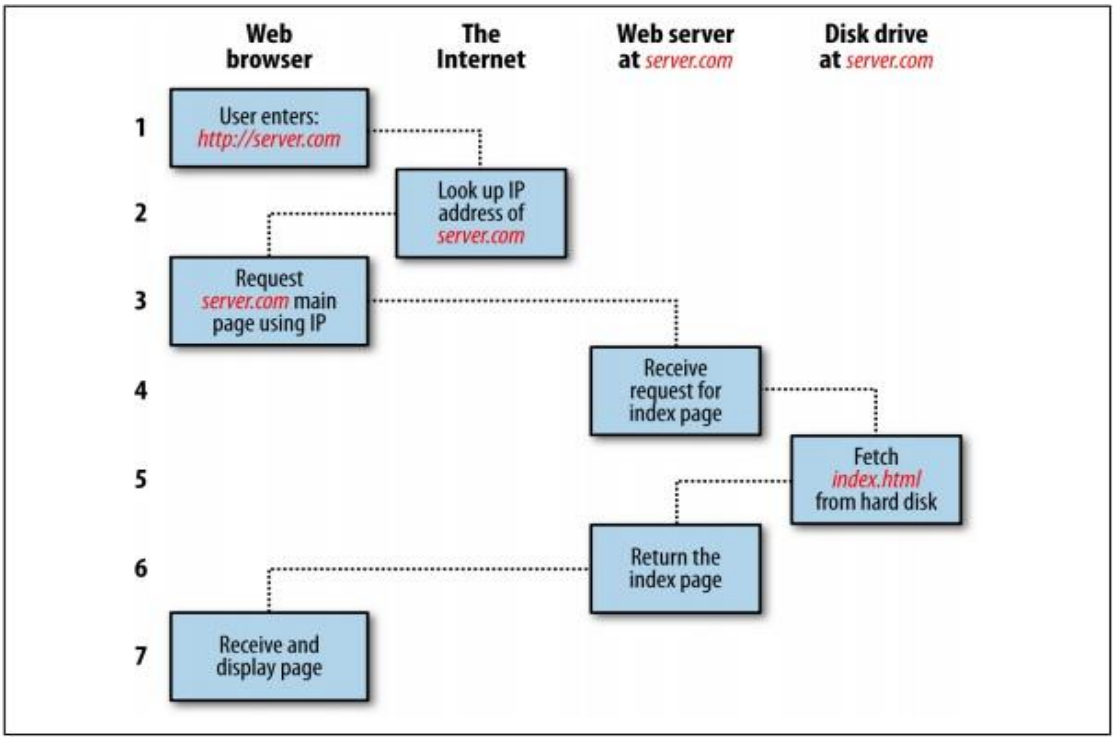


Figure 1.4 Static client/server communication procedure

1.2.6.2 Dynamic web pages

A dynamic website, on the other hand, is one that can display different content and provide user interaction, by making use of advanced programming languages(ex: PHP) and databases(using MySQL) in addition to HTML.

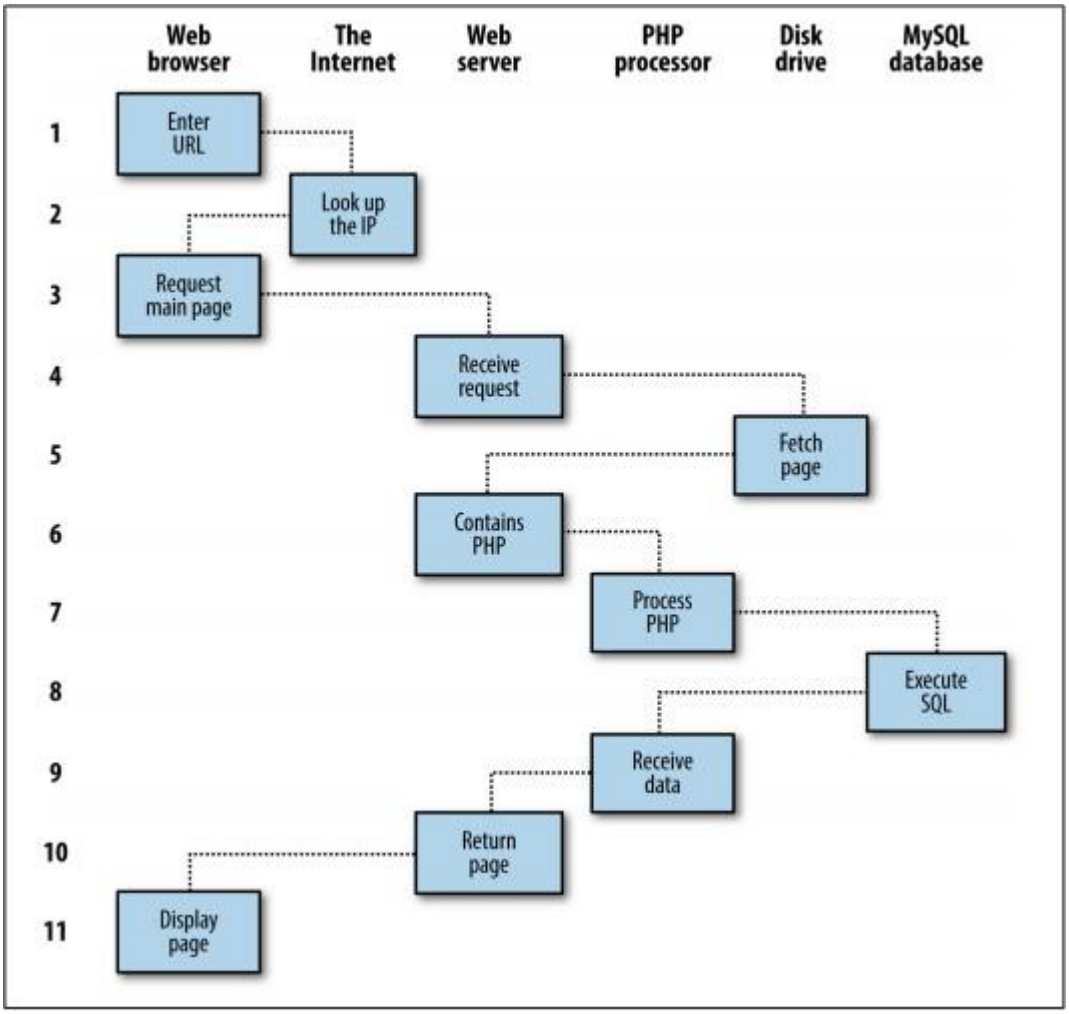


Figure 1.5 Dynamic client/server communication procedure

III. Part 2: Payroll office overview

2.1 Payroll office tasks

The office receives new employees wishing to work at the university, collects their documents and data, and prepares the list of employees that are entitled to receive payments, along with the amounts that each employee should receive, and the records of payments that were previously made to employees. Furthermore, it keeps track of time worked and prepares timesheets, calculates wages, produces paychecks and remits any tax payments necessary **[5].**

2.2 Types of employees

Internal employees: who are already teachers already at the university; to be eligible to work overtime they must first complete one of these:

- Six hours of lectures.
- Nine hours of recitations.
- Twelve hours of laboratory experiments.
- Twelve hours of English courses.

External employees: are teachers who work at other universities and wish to work overtime, or unemployed who want to work by hours.

2.3 Salary (wages) calculation

Employees are paid based on the hours worked, specific hourly pay rate is set for each hourly employee based on their qualifications, To calculate an employee's total wages, we will multiply the pay rate by the number of hours the employee works **[6].**

Qualification	Pay Rate(Algerian Dinar)
Professor	960.00
Lecturer A	840.00
Lecturer B	750.00
Doctorate	750.00
Assistant professor A	750.00
Assistant professor B	720.00
Magister	720.00
Master	300.00
State engineer	300.00
Bachelor	225.00

Table 1.1 Teacher's pay rate based on their qualification

2.3.1 Net and gross pay

→ **Gross pay**: is an employee's total pay.
→ **Net pay:** is an employee's pay after all deductions are subtracted. It is an employee's take home pay. **[6]**

2.3.2 Deductions from teacher's salary:

Definition: Money to be deducted from an employee's total wages **[6].**

Two types of deduction are to be considered in our case:

→ **Social security (SS):** an amount of 9% will be deduced from the teacher's gross pay.
→ **GIT (Global Income Tax):** an amount of 10% will be deduced from the gross pay.

The two amounts deduced will be added and subtracted from the gross pay to get the net pay.

2.4 Options to run payroll

Three payroll options are available

1. **Do payroll by hand:** This option is the most time-consuming.
2. **Use a payroll accountant:** The accountant will take care of the entire payroll process.However, this option might be the most expensive.
3. **Use a payroll software:** Payroll business software can calculate wages and taxes, net and gross pay and many more things **[6].**

IV. Part 3: Tools and Technologies:

3.1 Development tools

3.1.1 Sublime Text

Sublime Text is a shareware cross-platform source code editor with a Python application programming interface (API). It natively supports many programming languages and markup languages, and functions can be added by users with plugins, typically community-built and maintained under free-software licenses **[7].**

3.1.2 PhpMyAdmin

PhpMyAdmin is a free software tool written in PHP, intended to handle the administrationof MySQL over the Web. phpMyAdmin supports a wide range of operations on MySQL and MariaDB. Frequently used operations (managing databases, tables, columns, relations, indexes, users, permissions, etc) can be performed via the user interface, while you still have the ability to directly execute any SQL statement **[8].**

3.1.3 WAMP

Stands for "Windows, Apache, MySQL, and PHP." WAMP is often installed as a software bundle (Apache, MySQL, and PHP). It acts like a virtual server on the computer and allows testing the codes without any consequences since it’s localized on the machine but not connected to the web. It is often used for web development and internal testing, but may also be used to serve live websites.

- **Apache**

The most important part of the WAMP package is Apache (or "Apache HTTP Server") which is used to run the web server within Windows. By running a local Apache web server on a Windows machine, a web developer can test webpages in a web browser without publishing them live on the Internet **[9].**

3.1.4 MySQL

MySQL is a very fast, robust, relational database management system (RDBMS). A database is used to efficiently store, search, sort, and retrieve data. The MySQL server controls the access to data to ensure that multiple users can work with it concurrently, to provide fast access to it, and to ensure that only authorized users can obtain access to the data.

Hence, MySQL is a multiuser, multithreaded server. It uses SQL (Structured Query Language), the standard database query language worldwide. MySQL has been publicly available since 1996 but has a development history going back to 1979 **[10].**

3.1.5 Web browser

A web browser, or simply "browser," is an application used to access and view websites. Common web browsers include Microsoft Internet Explorer, Google Chrome, Mozilla Firefox, and Apple Safari.

The primary function of a web browser is to render HTML, the code used to design or "mark up" webpages. Each time a browser loads a web page, it processes the HTML, which may include text, links, and references to images and other items, such as cascading style sheets and JavaScript functions. The browser processes these items, and then renders them in the browser window **[11].**

3.1.6 Edraw max

EdrawMax is a 2D business technical diagramming software which help create flowcharts, organizational charts, mind map, network diagrams, floor plans, workflow diagrams, business charts, and engineering diagrams with its built-in editable symbols and templates for a range of categories **[12].**

3.2 Programming languages

3.2.1 PHP

PHP (Hypertext Pre-processor) is a server-side scripting language designed specifically for the Web. Within an HTML page, you can embed PHP code that will be executed each time the page is visited. The PHP code is interpreted at the web server and generates HTML or other output that the visitor will see. PHP was conceived in 1994 and was originally the work of one man, Rasmus Lerdorf. It was adopted by other talented people and has gone through three major rewrites to bring us the board, mature product we see today **[10].**

3.2.2 HTML

HTML(Hypertext Mark-up language) it is a mark-up language "Hypertext" refers to the hyperlinks that an HTML page may contain, by clicking on them users are brought to new pages, "Markup language" refers to the way tags are used to structure the page layout and the elements within the page **[13].**

HTML allows the user to create and structure sections, paragraphs, headings, links, and block quotes for web pages and applications **[14].**

It can be assisted by technologies such as CSS and scripting languages such as JavaScript. These will be briefly explained in the next two sections.

3.2.3 CSS

CSS (Cascading Style Sheet) used to define styles for web pages, including the design, layout and variations in display for different devices and screen sizes **[15].**

CSS defines how html elements should be displayed but it is independent of HTML and can be used with any XML-based markup language.

3.2.4 JavaScript

JavaScript is a scripting language used to implement complex features on web pages, commonly found embedded in HTML code. It is an interpreted language. Thus, it does not need to be compiled. JavaScript renders web pages in an interactive and dynamic fashion which allows webpages to react to events, exhibit special effects, accept variable text, validate data, create cookies, and detect a user's browser **[16].**

V. Conclusion

Through this chapter, a general overview of the project has been introduced. First, we cited the objectives of the project with some generalities concerning web applications. Second, we mentioned a brief overview of the payroll office. Finally, we defined the tools used to develop a responsive and dynamic web application.

Chapter two

Design

Chapter Two: Design

2.1 Introduction

In order to develop a good web application that meets users' needs we will use modelling for better understanding the system we are building and its behavior. In this chapter, the unified modelling language is introduced accompanied with some of its diagram's types.

2.2 What's a model?

A model is a simplification of reality, it provides the blueprints of a system. Models may encompass detailed plans, as well as more general plans that give a 30,000-foot view of the system under consideration. A good model includes major elements that have broad effect and omits those minor elements that are not relevant to the given level of abstraction. Every system may be described from different perspectives using different models, and each model is therefore a semantically closed abstraction of the system **[17].**

2.3 The Unified Modeling Language (UML)

UML is a graphical language for visualizing, specifying, constructing, and documenting the artifacts of a software-intensive system. The UML provides a standard way to write a system's blueprints, covering conceptual things, such as business processes and system functions, as well as concrete things, such as classes written in a specific programming language, database schemas, and reusable software components. UML defines a number of diagrams that fall into two groups: behavioral diagrams and structural diagrams **[17].**

2.3.1 Structural diagrams:

Show the static structure of the system and its parts at different abstraction and implementation levels and how they are related to each other. The elements in a structural diagram represent the meaningful concepts of a system, and may include abstract, real world and implementation concepts **[18].**

2.3.2 Behavioral diagrams:

Show the dynamic behavior of the objects in a system, which can be described as a series of changes to the system over time **[18].**

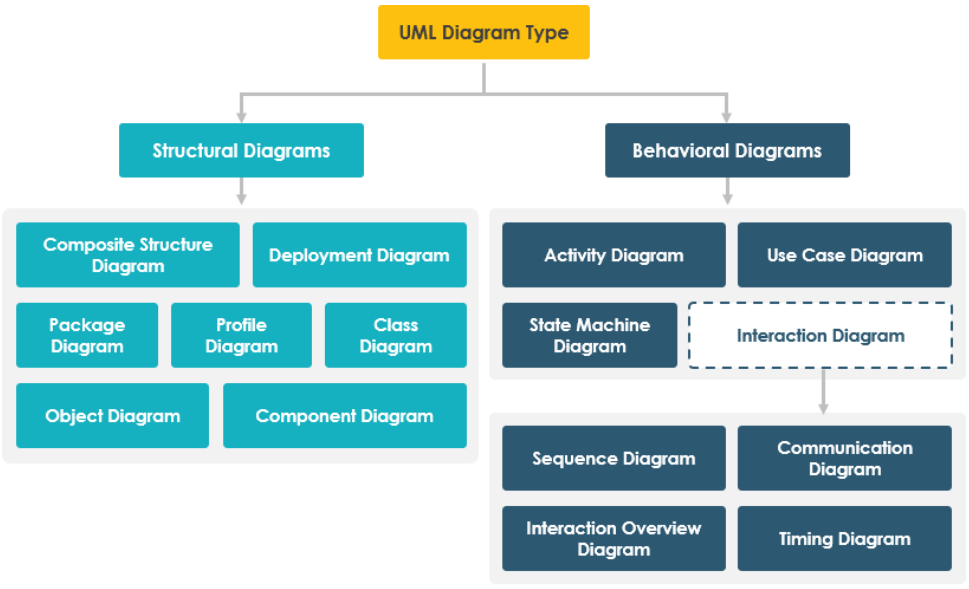


Figure 2.1 UML diagrams types

2.4 Modeling using UML (Use case model)

2.4.1 Use-case model

Is a model of how different types of users interact with the system to solve a problem. As such, it describes the goals of the users, the interactions between the users and the system, and the required behavior of the system in satisfying these goals **[19].**

2.4.2 Subject

Is a classifier (including subsystem, component, or even class) representing a business, software system, physical system or device under analysis, design, or consideration, having some behavior, and to which a set of use cases applies **[20].**

2.4.3 Actor

Specifies a role played by an external entity that interacts with the subject (e.g., by exchanging signals and data), a human user of the designed system, some other system or hardware using services of the subject. The standard UML notation for actor is "stick man" icon with the name of the actor above or below of the icon. Actor names should follow the capitalization and punctuation guidelines for classes **[21].**

2.4.4 Use case

Specifies a complete unit of useful functionality performed by one or more subjects to which the use case applies in collaboration with one or more actors, it yields (for complete use cases) an observable result that is of some value to those actors or other stakeholders. Use case is usually shown as an ellipse containing the name of the use case **[22].**

2.4.5 Relationships

- **Association**: an association between an actor and a use case indicates that the actor and the use case interact or communicate with each other **[23].**
- **Include:** A large use case could have some behaviors which may be detached into distinct smaller use cases to be included back into the base use case using include relationship. The purpose of this action is to simplify large use case by splitting it into several include use cases**.** Include relationship between use cases is shown by a dashed arrow with an open arrowhead from the including (base) use case to the included (common part) use case. The arrow is labeled with the keyword «include» **[24].**
- **Extend:** is a directed relationship that specifies how and when the behavior defined is supplementary (optional), extending use case can be inserted into the behavior defined in the extended use case**.** Extend relationship is shown as a dashed line with an open arrowhead directed from the extending use case to the extended (base) use case. The arrow is labeled with the keyword «extend» **[25].**

2.4.6 Use case diagrams

Are usually referred to as behavior diagrams and used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors). Each use case should provide some observable and valuable result to the actors or other system stakeholders **[26].**

2.5 Application modelling:

2.5.1 Actors:

There are two types of actors

- **Supervisor:** a single supervisor account gives its user the possibility to manage employees and modify certain settings on the application.
- **Employee:** an employee account provides a restricted use under the supervisor control, and used to manage their data.

2.5.2 Use cases

Table 2.1 Use cases of each actor

Actors	Use cases
Supervisor/Employee	1. Authentication: Logging in using a valid username and password.
Supervisor	2. Managing employees: Adds (creates accounts), removes (delete accounts). 3. Monitoring: consults teacher's information and hours worked. 4. Documents verification: enters the documents names submitted by employees to the database. 5. Edit settings: modifies hourly pay rate. 6. Printing documents: prints stand-in state of teachers and pay slips.
Employee	7. Manage account: Adds, edits and consults his data. 8. Filling worked hours: adds the number of total hours worked by date and time.

2.5.3 Use cases identification

1. **Authentication:** to protect users 'data authentication is used to decide whether the user is eligible to access the application.
2. **Managing employees:** Allows the supervisor to create and delete teachers' accounts.
3. **Monitoring:** Gives to the supervisor the possibility to consult teachers' information, worked hours, net salary and the total bill.
4. **Documents verification:** the supervisor can make enter documents names that have already been submitted by an employee.
5. **Edit settings:** supervisor can modify the rate of pay per hour.
6. **Document printing:** give the possibility to the supervisor to print a number of documents:
   - Stand-in state of all internal employees
   - Stand-in state of all external employees
   - Stand-in state of all employees
   - Pay slips of each employee
7. Managing account: allows the employee to see his data and modify it.
8. Filling up timesheet: register the number of hours worked along with the corresponding time, date and activity.

2.4 Application's use case diagrams

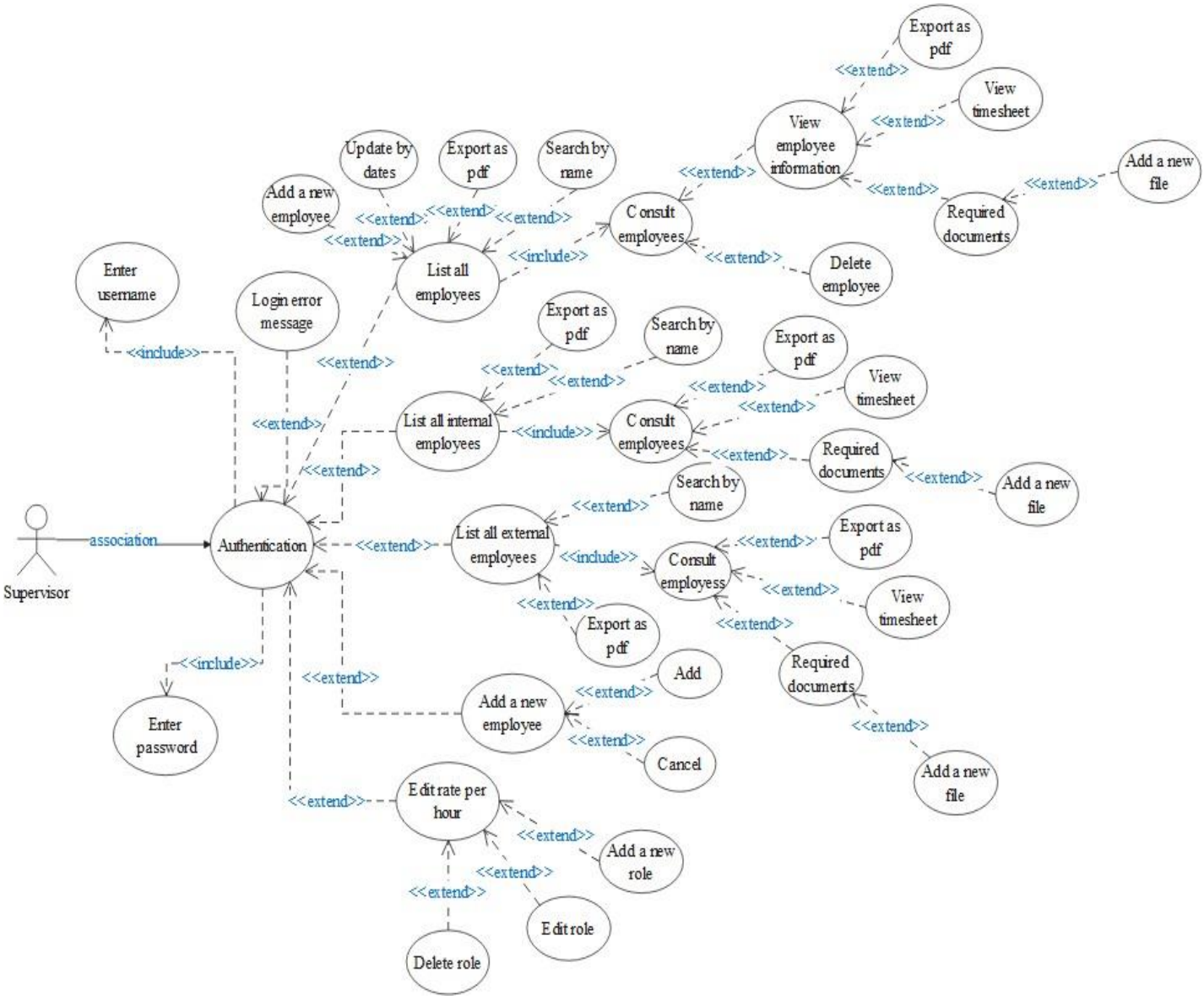


Figure 2.2 Supervisor use case diagram

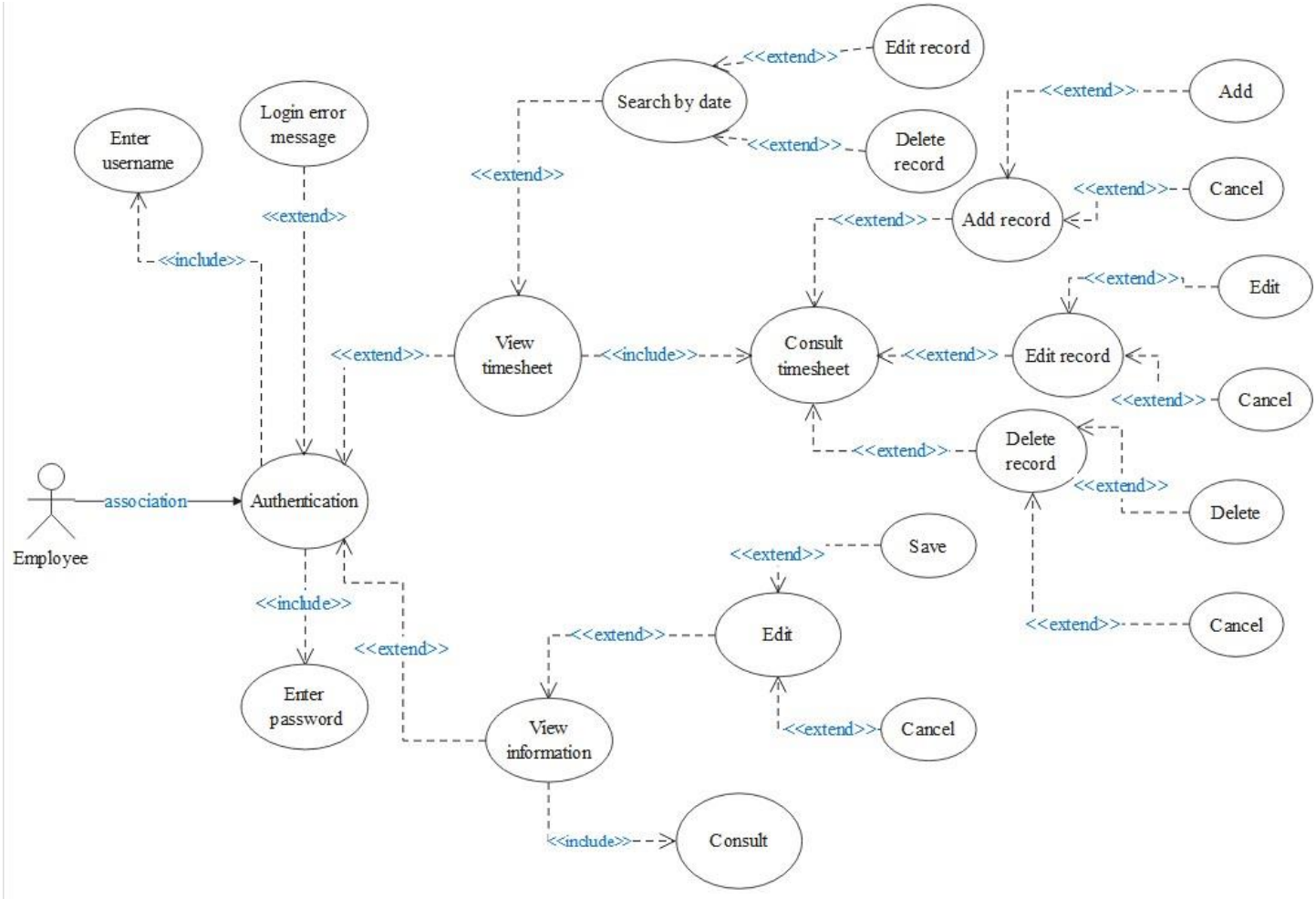


Figure 2.3 Employee use case diagram

2.5 Textual description of use cases

Use case 01: Authentication

Table 2.2 Textual description for authentication use case

Use case name	Authentication
Actor	Supervisor, Employee
Objective	Authenticate to get access to application's account
Precondition	Browser and access to internet
Scenario	1.The user launches the application via a browser 2.The application request a username and password 3. The user enters his username and password 4. The system sends an authentication query to the server to check the correctness of the information entered. 5. The server verifies the query and returns favorable answer 6. The user accesses the application
Alternative	If the username or password is wrong,an error message is displayed to the user.(return to 3)

Use case 02: Managing employees' accounts

Table 2.3 Textual description for managing employees' accounts use case

Use case name	Managing employees' accounts
Actor	Supervisor
Objective	Manage employees' accounts to enable each employee to have his personal account.
Precondition	Authentication to supervisor account
Scenario	1. The supervisor accesses to Add new employee interface or "list of employees interfaces to delete an employee's account. 2. The supervisor enters full name, username and password for employee to add an employee or press the delete button on the list of employees interface to delete an employee's account. 3. The system sends a query to the server for processing. 4. The system creates new account or deletes an existing account.
Alternative	If the username already exists an error message will be displayed to the user.(return to 2)

Use case 03: Monitoring

Table 2.4 Textual description for monitoring use case

Use case name	Monitoring
Actor	Supervisor
Objective	Monitoring to verify submitted documents and employees' timesheets
Precondition	Authentication to supervisor account
Scenario	1. The supervisor accesses to list of all employees or list of internal employees or list external employees interfaces. 2. The supervisor presses the view info button on an employee's cell. 3. The system sends a query to the server for processing. 4. The application shows the entire employee's data.
Alternative	If the query fails, an error message will be displayed to the user. (return to 1)

Use case 04: Documents verification

Table 2.5 Textual description for documents verification use case

Use case name	Documents verification
Actor	Supervisor
Objective	Document verification to ensure that employee has submitted all required documents.
Precondition	Authentication to supervisor account
Scenario	1. The supervisor accesses to list all employees or list internal employees or list external employees interfaces. 2. The supervisor clicks on view info on one of the previous interfaces and then required documents. 3. The supervisor can add a submitted document by selecting a document from the options list and press add new document.
Alternative	✓ If a document is already added to the list it cannot be added again. ✓ An empty document option cannot be added. (return to 3)

Use case 05: Edit settings

Table 2.6 Textual description for edit settings use case

Use case name	Edit settings
Actor	Supervisor
Objective	Consult roles and their associated rates and add, edit or delete a rate.
Precondition	Authentication to supervisor account
Scenario	1. The supervisor accesses the dashboard home and click on edit rate per hour link. 2. The supervisor clicks on the edit button that corresponds to the rate that needs to be changed.
Alternative	If the edition fails, an error message will be displayed to the user. (return to 2)

Use case 06: Documents printing

Table 2.7 Textual description for documents printing use case

Use case name	Documents printing
Actor	Supervisor
Objective	Salary verification to ensure that the calculation are accurate.
Precondition	Authentication to supervisor account
Scenario	1. The supervisor accesses to list all employees or list internal employees or list external employees or employee data interfaces 2. The supervisor clicks on the export as pdf link on one of the previous interfaces.
Alternative	If the query fails, an error message will be displayed to the user. (return to 1)

Use case 07: Managing account

Table 2.8 Textual description for managing account use case

Use case name	Managing account
Actor	Employee
Objective	Information consultation and edition to ensure that all the information seen by the supervisor are accurate.
Precondition	Authentication to employee account
Scenario	1. The employee access the dashboard home and clicks on view information 2. The employee press the edit button so he can access the form responsible for editing his information.
Alternative	If the query fails an error message will be displayed to the user. (return to 2)

Use case 08: Filling up timesheet

Table 2.9 Textual description for filling up timesheet use case

Use case name	Filling up timesheet
Actor	Employee
Objective	Reviewing and editing the employee timesheet.
Precondition	Authentication to employee account
Scenario	1. The employee access to its dashboard home and clicks on view timesheet. 2. The employee can add edit or delete a record according to his needs by pressing the appropriate button.
Alternative	If the query fails an error message will be displayed to the user. (return to 2)

2.6.1 Modeling using UML (sequence diagrams)

A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems [27].

2.6.1 Messages

Communication between objects is depicted using messages. The messages appear in a sequential order on the lifeline. We represent messages using arrows. Lifelines and messages form the core of a sequence diagram [27].

2.6.2 Sequence Fragments:

Make it easier to create and maintain accurate sequence diagrams. It is represented as a box, called a combined fragment, which encloses a portion of the interactions within a sequence diagram. The fragment operator (in the top left cornet) indicates the type of fragment [28].

→ Messages and sequence fragments used in the sequence diagrams are classified in the following table.

Table 2.10 Messages and sequence fragments used in sequence diagrams

Messages and Sequence fragments	Definition
Actor	represents a type of role where it interacts with the system and its objects, it is represented using a stick person notation. We can have multiple actors in a sequence diagram [27].
Lifelines	depicts an individual participant, each instance in a sequence diagram is represented by a lifeline. Lifeline elements are located at the top in a sequence diagram. The standard in UML for naming a lifeline follows the following format – Instance Name : Class Name [27].
Call message	is a kind of message that represents an invocation of operation of target lifeline [29]
Self Message	Certain scenarios might arise where the object needs to send a message to itself. Such messages are called Self Messages [27].
Return Message	Reply messages are used to show the message being sent from the receiver to the sender. The interaction moves forward only when a reply message is sent by the receiver [27].
Loop fragment operator	It represents a loop. The loop operand will be repeated a number of times [29].
Alternative fragment operator	It represents a choice of behavior. At most one of the operands will be chosen [29].
Reference fragment operator	Refers to an interaction defined on another diagram. The frame is drawn to cover the lifelines involved in the interaction. You can define parameters and a return value [28].

2.6.1 Application's sequence diagrams

Note: several sequence diagrams have not been mentioned in order to avoid repetition because of the processes similarities.

Sequence diagram 01: Authentication

When the user (employee/supervisor) want to access to the application, he must log- in by providing his username and password. The system sends a query to the server for verification. If these information exists in the database, the user can access his dashboard successfully, otherwise an error message page will be displayed.

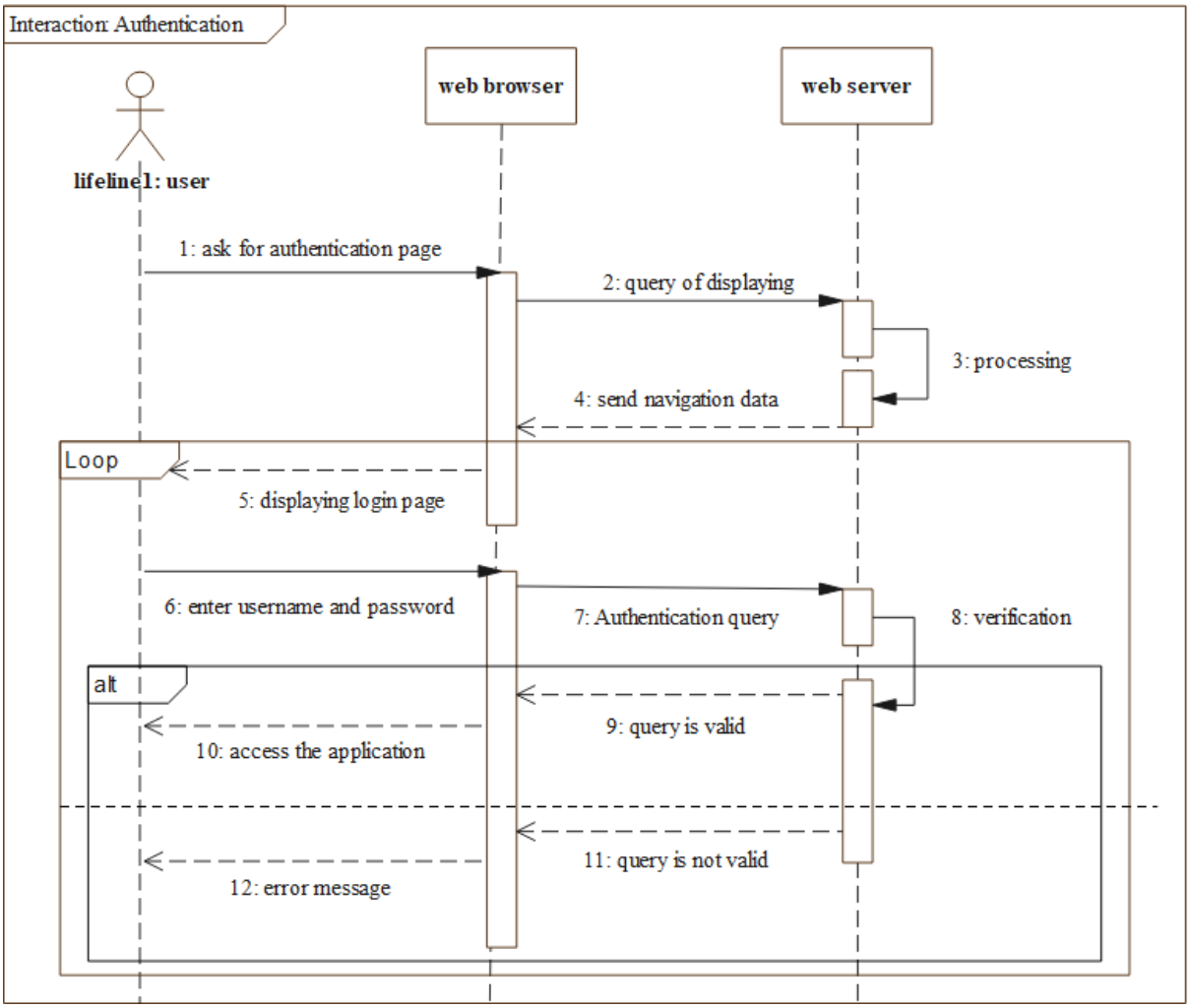


Figure 2.4 Authentication sequence diagram

Sequence diagram 02: Add employee

After authentication, the supervisor can delete or add employees' accounts and saves them in the database. This matter will be done by accessing the "add a new employee" page and filling the form that contains: the username and the password of that employee account. The browser then, sends a query to the server for processing and saving. If a failure occurs, an error message will be displayed; otherwise the employee is added successfully.

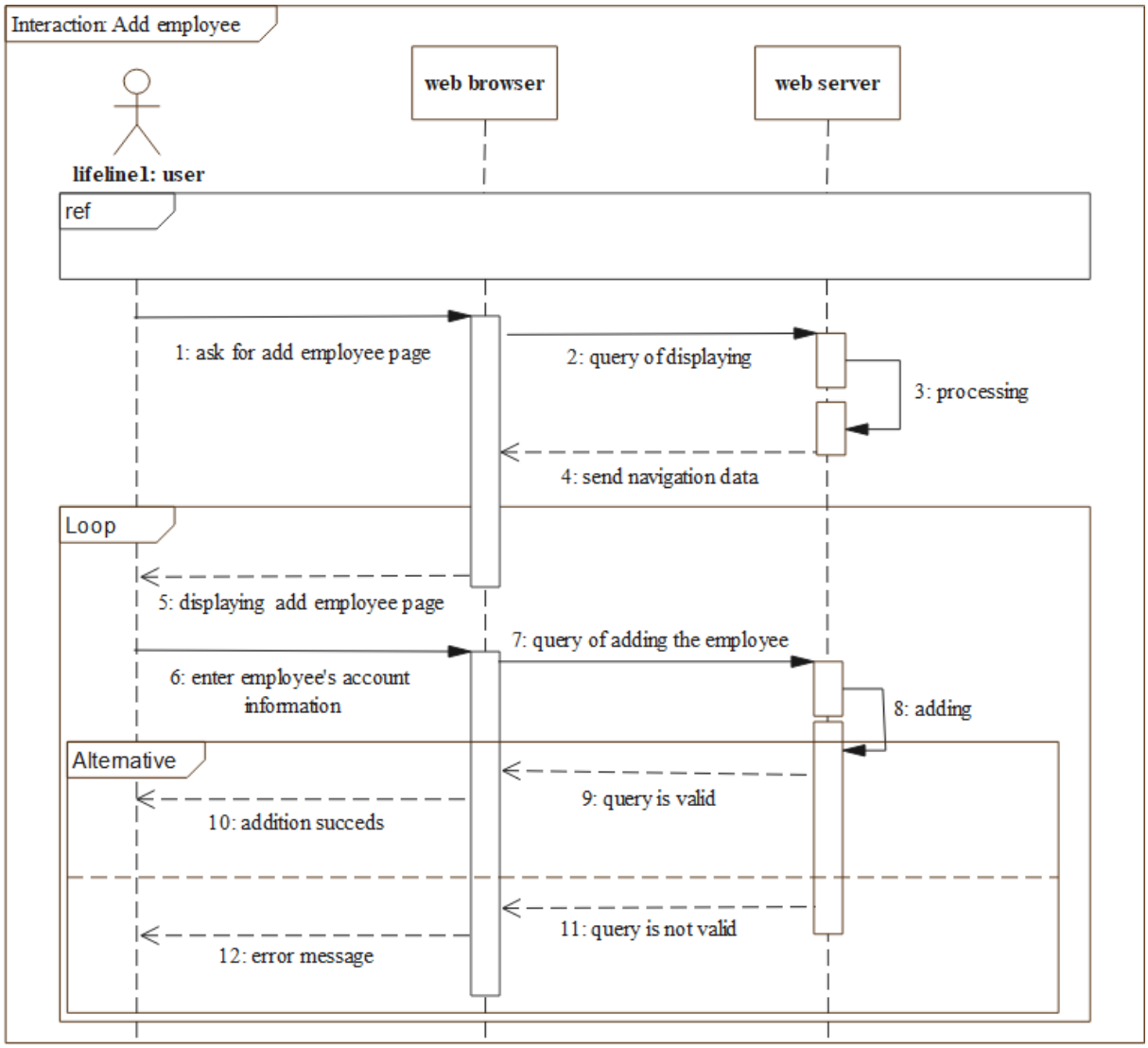


Figure 2.5 Add employee sequence diagram

Sequence diagram 03: Search by name

The supervisor can review the list of all employees or either the external or the internal ones by choosing the appropriate page from the dashboard home. Once he accesses one of the previous pages, he can search for an employee table by providing his name using “search by name”. The browser sends a query to the server for processing. If the query fails an error message will be displayed; otherwise the result of the search page will be shown.

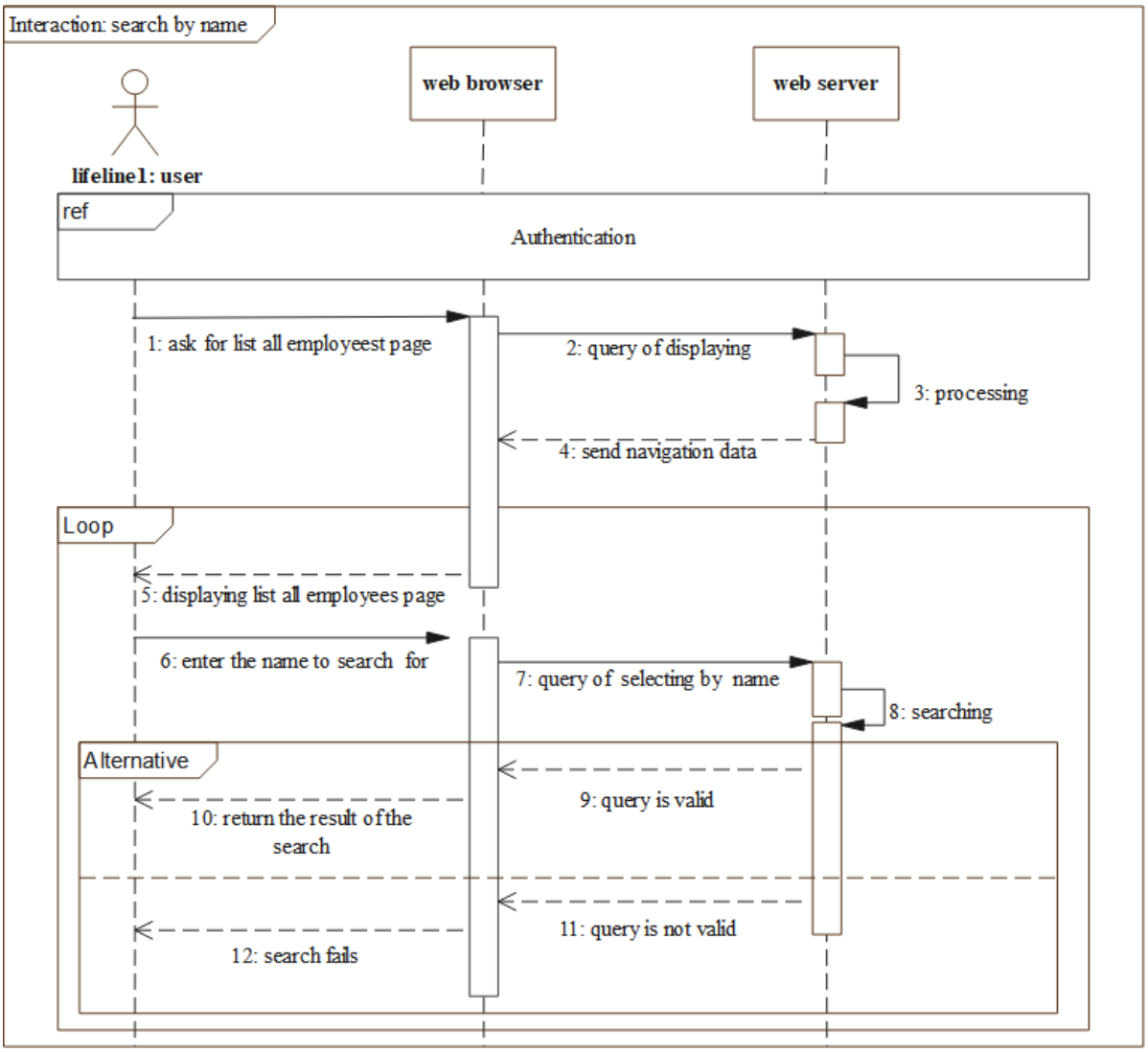


Figure 2.6 Search by name sequence diagram

Sequence diagram 04: Required documents

Once the supervisor has accessed the page where the employees are listed, he has can review the information, the timesheet of the selected period and the required documents that to be submitted by each employee. The bellowed sequence diagram will be about the latter. The browser sends a query to the server for processing and saving. If the query fails, an error message will be displayed; otherwise the result of the each of the previously mentioned pages will be shown.

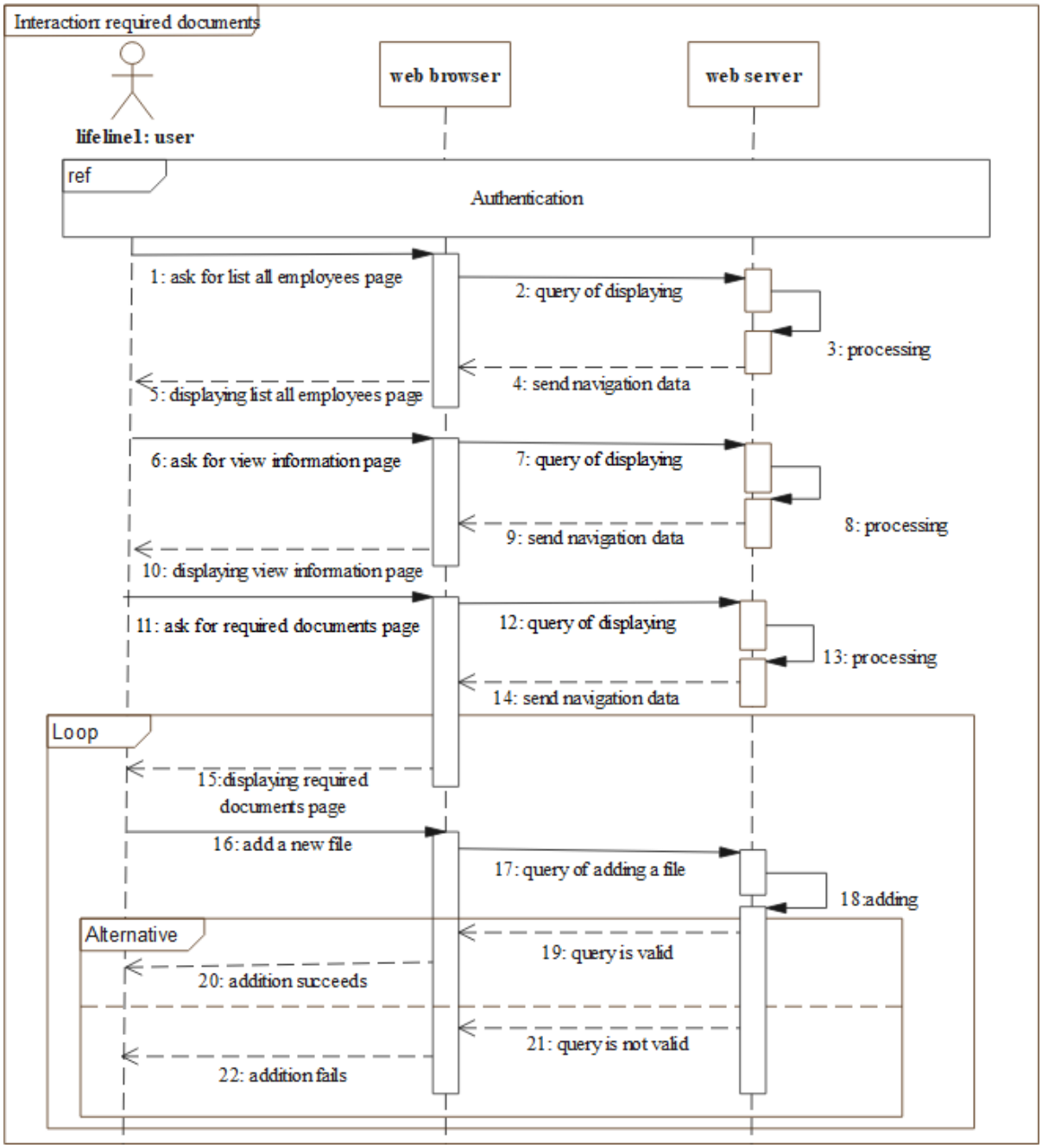


Figure 2.7 Required documents sequence diagram

Sequence diagram 05: Edit hourly pay rate

In addition to the ability of the supervisor to manage and review the employees account, he can choose to add, edit or delete an hourly rate that corresponds to a certain degree of the employee. This sequence diagram is an example of one of these mentioned operations “edit rate “noted that the rest of them have the same process. The browser sends a query to the server for processing and saving. If the query fails an error message will be displayed otherwise the result of the each of the previously mentioned operations will be shown.

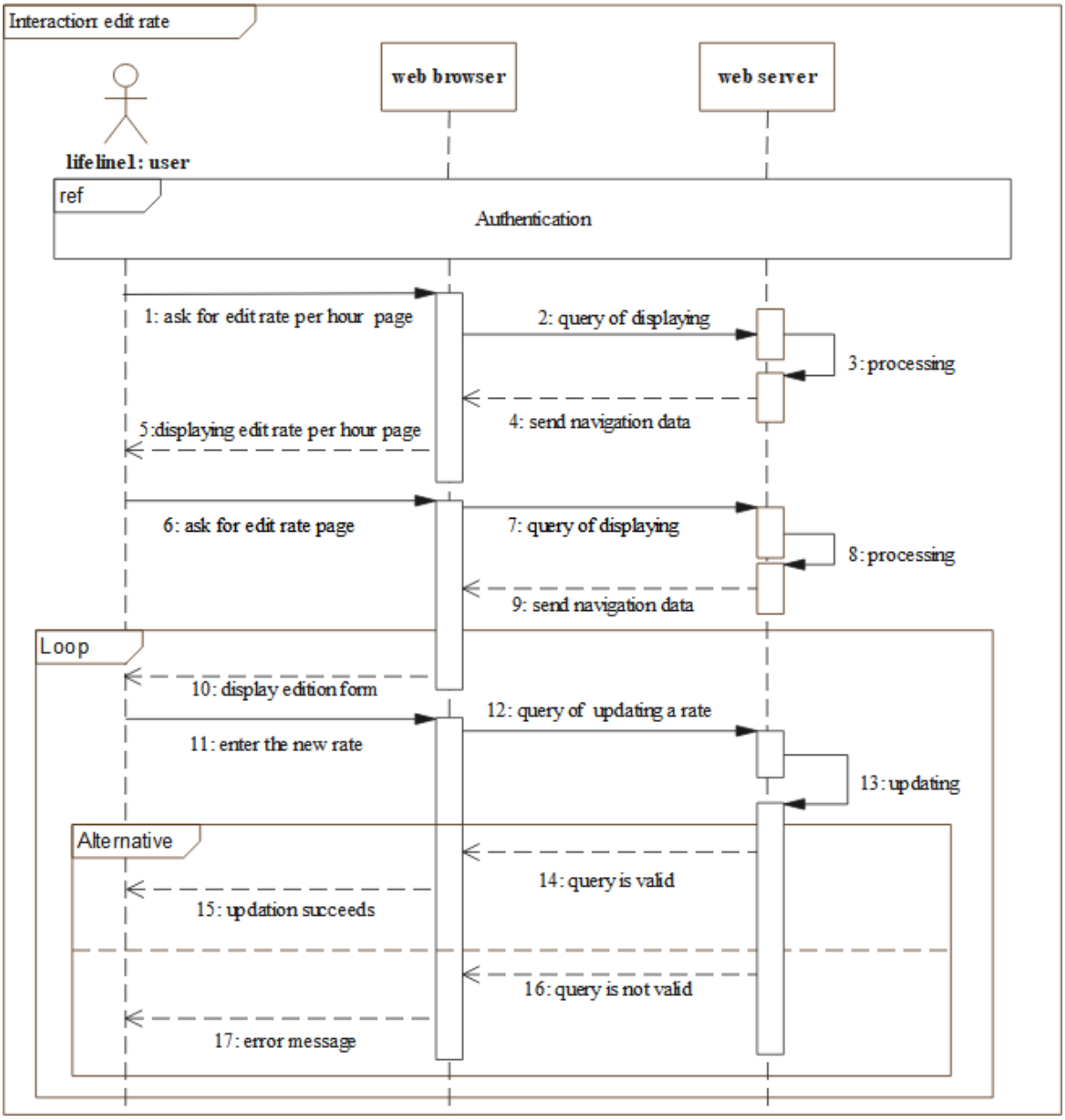


Figure 2.8 Edit rate per hour sequence diagram

Sequence diagram 06: Add a new record

Once the employee has successfully logged in, he can access his timesheet to adjust it based on his schedule. In order to achieve such adjustments, the employee has the choice to add, edit or delete a record that contains the start, the finish times and the activity of each session. In this sequence diagram, the process of "add new record" has been chosen. The browser sends a query to the server for processing and saving. If the query fails an error message will be displayed otherwise the record will be added successfully.

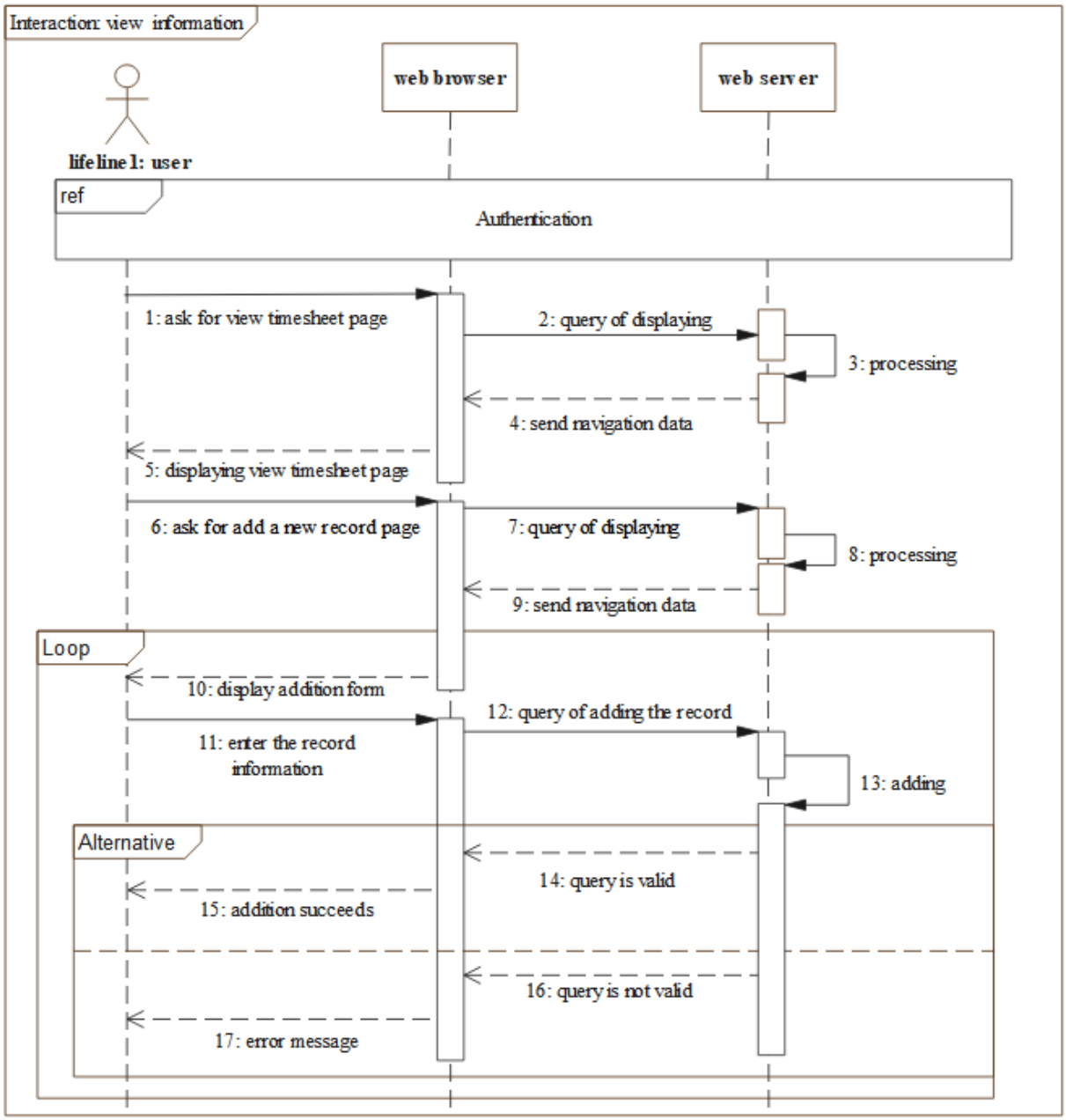


Figure 2.9 View information sequence diagram

Sequence diagram 07: Edit information

In addition to the timesheet page, the employee have access to his information page that can be edited using “edit” button. After filling the modification form the browser sends a query to the server for processing and saving. If the query fails an error message will be displayed; otherwise the information will be edited successfully.

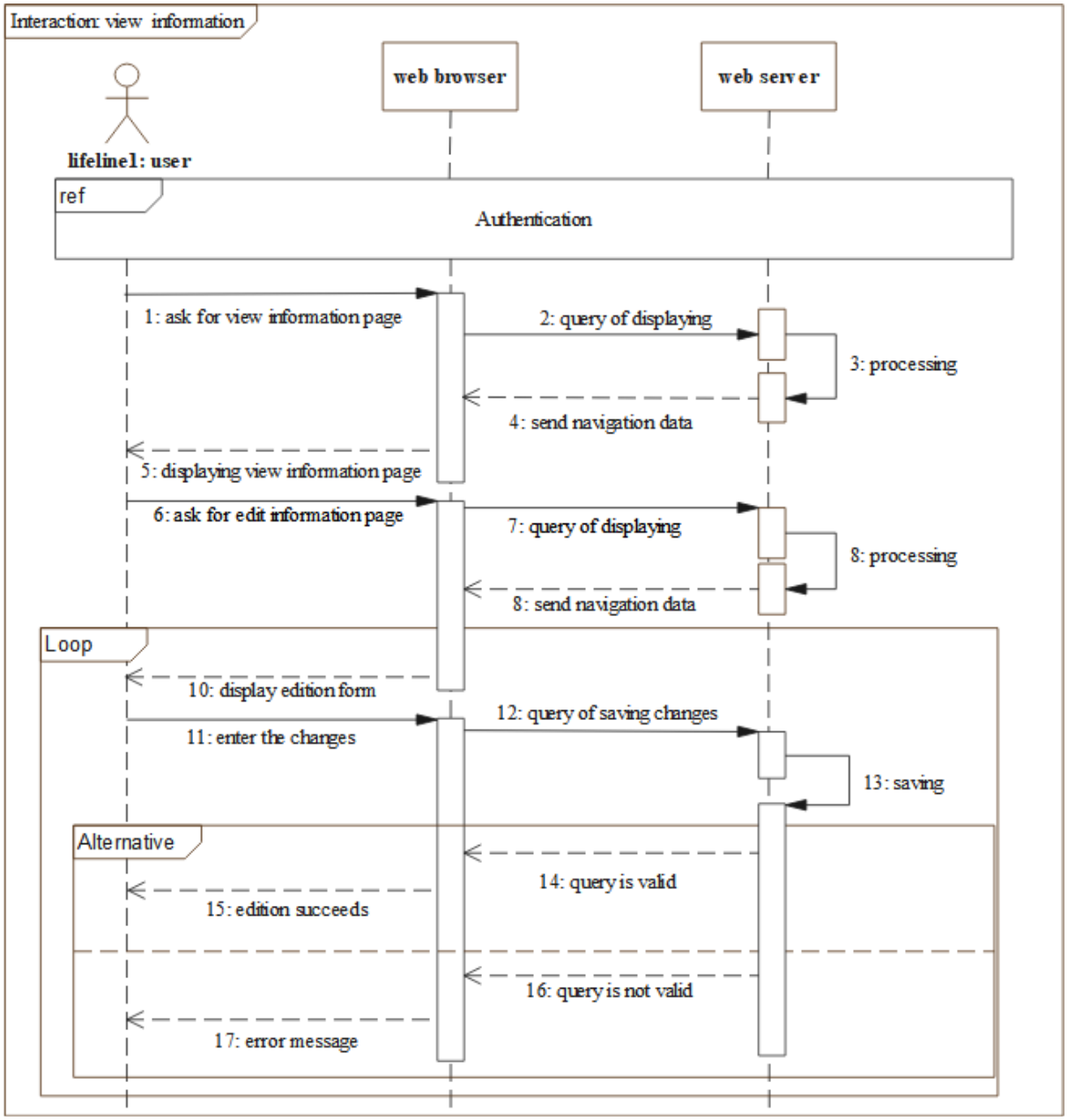


Figure 2.10 Edit information sequence diagram

2.7 Introduction to databases

2.7.1 Definition

A database is a collection of information that is organized so that it can be easily accessed, managed and updated. Computer databases typically contain aggregations of data records or files that can be managed by database management system (DBMS) **[30].**

A database management system (DBMS) is a collection of programs that enables users to create and maintain a database. The DBMS is a general-purpose software system that facilitates the processes of defining, constructing, manipulating, and sharing databases among various users and applications **[31].**

2.7.2 Non-relational model

A *non-relational database* is a database that does not use the tabular schema of rows and columns found in most traditional database systems. Instead, non-relational databases use a storage model that is optimized for the specific requirements of the type of data being stored. For example, data may be stored as simple key/value pairs, as JSON documents, or as a graph consisting of edges and vertices **[32].**

There are four non-relational types: document data store, column-oriented database, key-value store and graph database.

Key	Document
1001	{ "CustomerID": 99, "OrderItems": [{ "ProductID": 2010, "Quantity": 2, "Cost": 520 }, { "ProductID": 4365, "Quantity": 1, "Cost": 18 }], "OrderDate": "04/01/2017" }
1002	{ "CustomerID": 220, "OrderItems": [{ "ProductID": 1285, "Quantity": 1, "Cost": 120 }], "OrderDate": "05/08/2017" }

Figure 2.11 Non-relational database example

2.7.3 Relational model

The relational model represents the database as a collection of relations. Informally, each relation resembles a table of values that consists of rows and columns **[31].**

The columns of the table hold attributes of the data, and each row usually has a value for each attribute and could be marked with a unique identifier called a primary key, and rows among multiple tables can be made related using foreign keys.

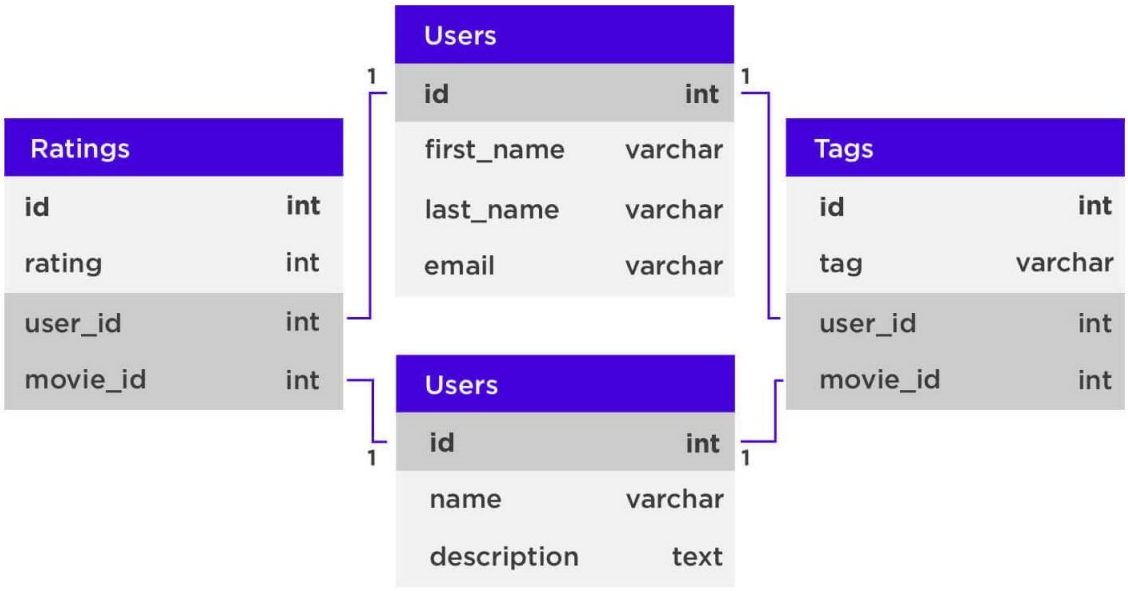


Figure 2.12 Relational database example

2.7.3.1 Relational Model Concepts

→ **Tables**: relations are saved in the table format. It is stored along with its entities. A table has two properties rows and columns.

→ **Attribute:** the columns heads in a Table. Attributes are the properties which define a relation.

→ **Column:** The column body which represents the set of values for a specific attribute.

→ **Tuple**: It is nothing but a single row of a table, which contains a single record.

→ **Relation Schema:** A relation schema represents the name of the relation with its attributes.

→ **Degree:** The total number of attributes which in the relation.

→ **Cardinality:** Total number of rows present in the Table.

→ **Relation instance**: Relation instance is a finite set of tuples in the RDBMS system. Relation instances never have duplicate tuples.

→ **Relation key**: Every row has one, two or multiple attributes, which is called relation key.

→ **Attribute domain**: Every attribute has some pre-defined value and scope which is known as attribute domain.**[33]**

2.7.3.2 Relational model constraints

Relational constraints are referred to conditions or rules that must be present for a valid relation, also known as integrity constraints. Those rules ensure the accuracy and reliability of the data in the database.

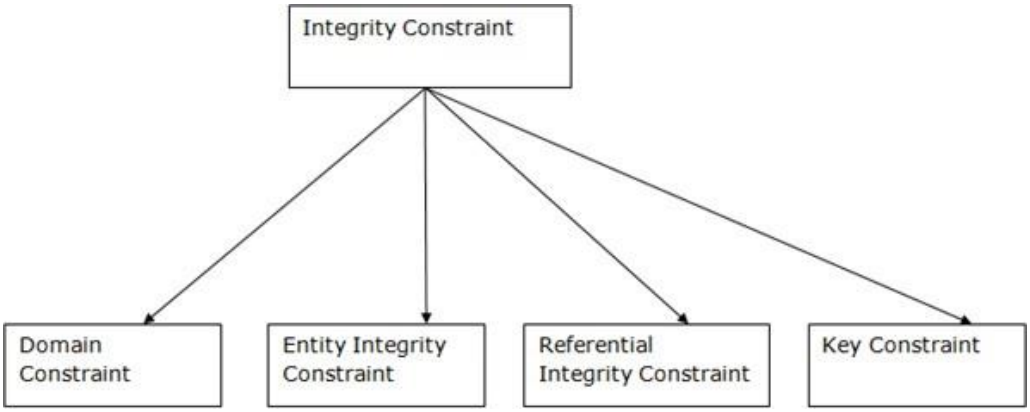


Figure 2.13 Relational database constraints

a. Domain constraint

→ Domain constraints can be defined as the definition of a valid set of values for an attribute.

→ The data type of domain includes string, character, integer, time, date, currency, etc. The value of the attribute must be available in the corresponding domain.

b. Entity integrity constraints

→ The entity integrity constraint states that primary key value can't be null.

→ This is because the primary key value is used to identify individual rows in relation and if the primary key has a null value, then we can't identify those rows.

c. Referential Integrity Constraints

→ A referential integrity constraint is specified between two tables.

→ In the Referential integrity constraints, if a foreign key in Table 1 refers to the Primary Key of Table 2, then every value of the Foreign Key in Table 1 must be null or be available in Table 2.

d. Key constraints

→ Keys are the entity set that is used to identify an entity within its entity set uniquely.

→ An entity set can have multiple keys, but out of which one key will be the primary key. A primary key can contain a unique and null value in the relational table.**[34]**

2.7.4 Database Schema

Along this application the database has taken a salient role in controlling and managing the information used by the user and ensuring the flexibility and responsivity of the application. The database schema is comprised of four tables that are represented as follow:

2.7.4.1 Employee table

It has consisted of 13 attributes:

- **id:** identification number that needs to be inserted in any table and considered as the primary key.
- **full_name:** is the full name of the employee.
- **username:** is the username that has been provided by the supervisor to any employee that attends to use the application.
- **password:** is the password of the account that is provided the employee to access the application.
- **Role:** corresponds to the teaching state or the academic diploma of the employee.
- **Max_hours:** is the number of hours that an employee can work extra shift per week.
- **State:** is used to define whether an employee is an external or an internal employee.
- **Pay:** is used to define whether an employee is paid using bank or ccp.
- **account_nbr:** is the employee's account that his payment is deposited to.
- **date_from:** is the start date from when an employee is going to be paid for the total hours worked in that period.
- **date_to:** is the finish date that an employee is going to be paid for the total hours worked in that period.
- **hours:** is the total hours that an employee has worked during a specific period.
- **salary:** is the payment of the employee for the total hours worked during a specific period before tax deduction.

Field	Type	Null	Key	Default	Extra
id	int(20)	NO	PRI	NULL	auto_increment
full_name	varchar(20)	YES		NULL	
username	varchar(50)	YES		NULL	
password	varchar(255)	YES		NULL	
role	varchar(20)	YES		NULL	
max_hours	int(10)	YES		NULL	
state	tinyint(1)	YES		NULL	
pay	tinyint(1)	YES		NULL	
account_nbr	int(30)	YES		NULL	
date_from	date	YES		NULL	
date_to	date	YES		NULL	
hours	int(10)	YES		NULL	
salary	int(10)	YES		NULL	

Figure 2.14 Employee table

2.7.4.2 Time table

It has consisted of 6 attributes:

- **id:** identification number that needs to be inserted in any table and considered as the primary key.
- **employee_id:** is a foreign key that references to the employee corresponds to the timesheet represented by this table.
- **date:** is the date of the working day of the employee.

- **time_from:** is the start time of a working hour of the employee.
- **time_to:** is the end time of a working hour of the employee.
- **activity:** corresponds to the teaching activity during that working hour.

Field	Type	Null	Key	Default	Extra
id	int(20)	NO	PRI	NULL	auto_increment
employee_id	int(20)	NO	MUL	NULL	
date	date	YES		NULL	
time_from	time	YES		NULL	
time_to	time	YES		NULL	
activity	varchar(255)	YES		NULL	

Figure 2.15 Time table

2.7.4.3 Document Table

It consists of 3 attributes:

- **id:** identification number that needs to be inserted in any table and considered as the primary key.
- **employee_id:** is a foreign key that references to the employee corresponds to the documents represented by this table.
- **doc:** represents the list of the required documents that needs to be submitted by the employee.

Field	Type	Null	Key	Default	Extra
id	int(20)	NO	PRI	NULL	auto_increment
employee_id	int(20)	NO	MUL	NULL	
doc	varchar(255)	YES		NULL	

Figure 2.16 Document table

2.7.4.4 Rates & roles table

- **id:** identification number that needs to be inserted in any table and considered as the primary key.
- **role:** represents the list of the roles or degrees that an employee can own one of them.
- **rate:** is to the rate or salary of each worked hour associated with the corresponding degree.

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
role	varchar(20)	YES		NULL	
rate	int(10)	YES		NULL	

Figure 2.17 Rate and roles table

2.7.5 Application relation model:

The following figure shows our database model:

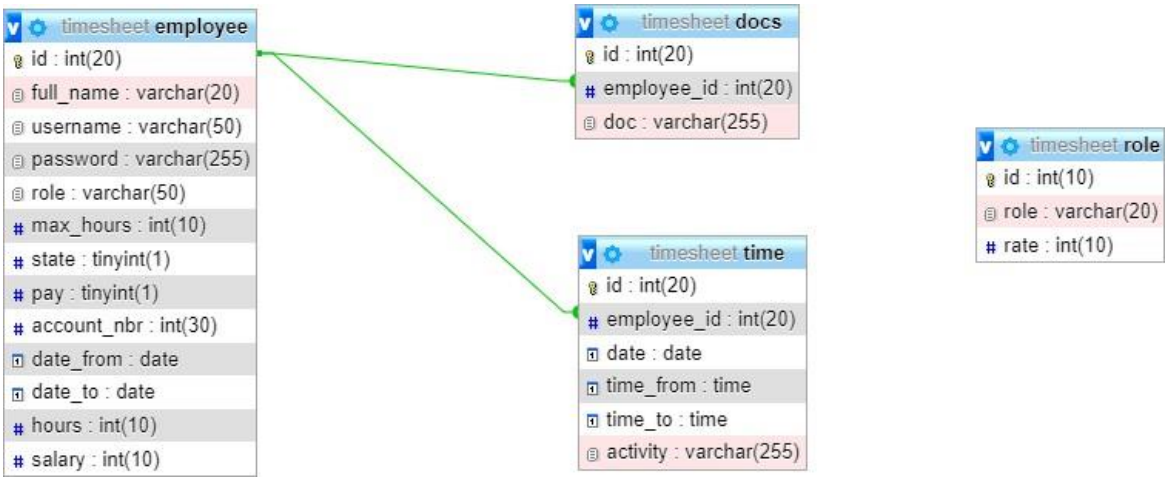


Figure 2.18 Tables and their relationships

2.8 Conclusion

In this chapter we have modeled our web application using UML diagrams this is to make it easier to communicate it with other people especially non-developers first, we have summarized the details of our application and shown how entities interact with the systems to achieve their goals using use case diagrams, then we used sequence diagrams to show in which order the application's operations are carried out. Finally, we have defined database models (relational and non-relational) and how we have used the relational model to create our application's database.

Chapter three

Implementation

Chapter three: Implementation

3.1 Introduction

In the previous chapters, we have identified the tools used to build our project followed by the description of how the several entities interact with each other to give the users a flexible and responsive web application. In this chapter, we are going to give a presentation of the application and its functionalities accompanied with some user interfaces.

3.2 Interfacing with the application

After implementing the application with the tools specified in the previous chapters, we will present in this part the different interfaces, the user will be dealing with.

Note: all the names and the numbers in this chapter are selected randomly in order to test the functionality of the web application.

3.2.1 Home interface

This following interface is the first page that will be displayed to the user once he enters the application. The user can login as a supervisor or as an employee according to his role in the department:

Figure 3.1 Home page interface

3.2.2 Login interface

The following interface corresponds to the login page that will be displayed to the user when he attends to login as a supervisor or as an employee:

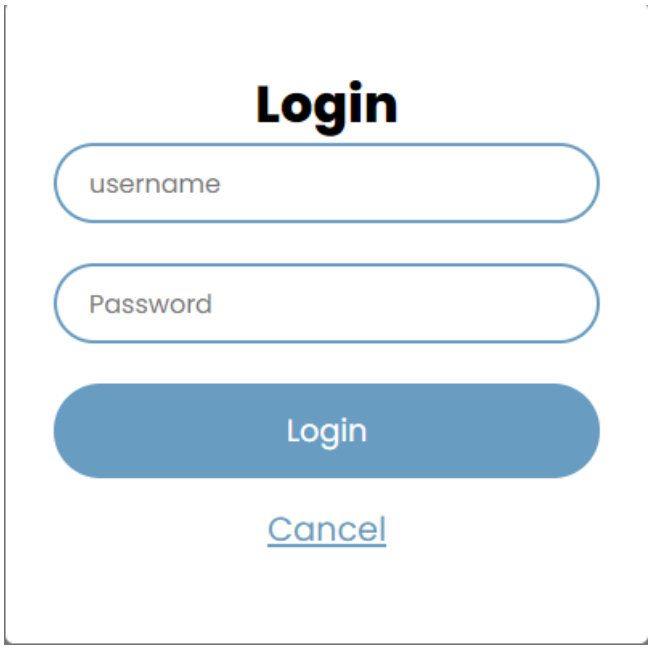


Figure 3.2 Login page interface

3.2.3 Supervisor Dashboard

3.2.3.1 Supervisor home interface

Once the user has logged in as a supervisor the following interface will be displayed to him, and he can choose the page he wants to enter according to his needs:

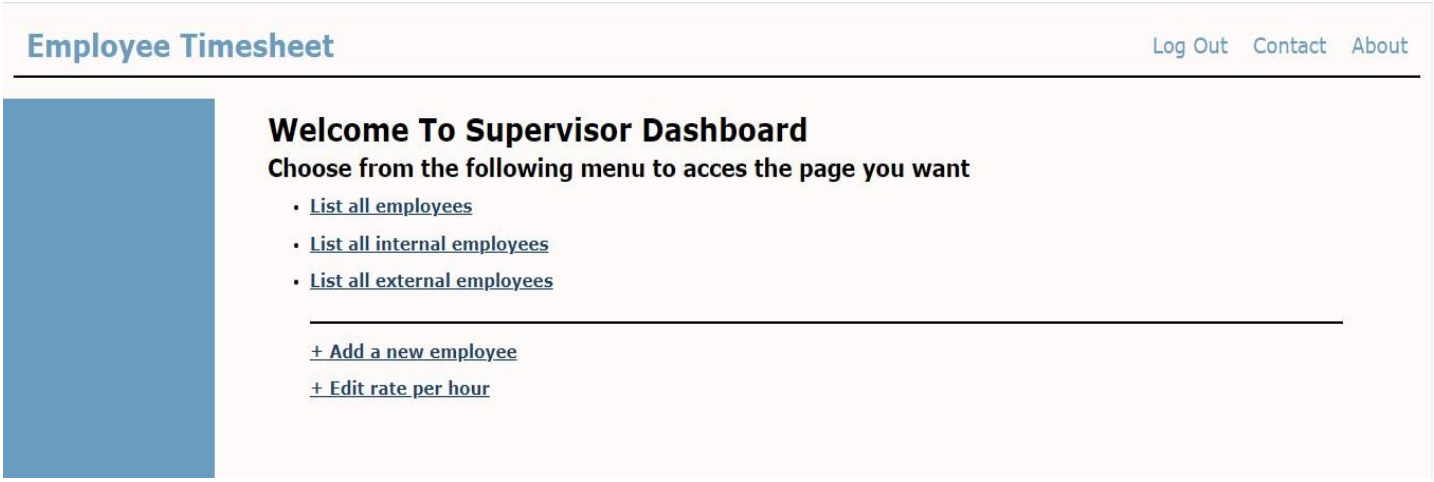


Figure 3.3 Supervisor home interface

3.2.3.2 Edit rate per hour interface

This interface represents a table of the rate per hour according to each employee degree and each one can be edited or deleted:

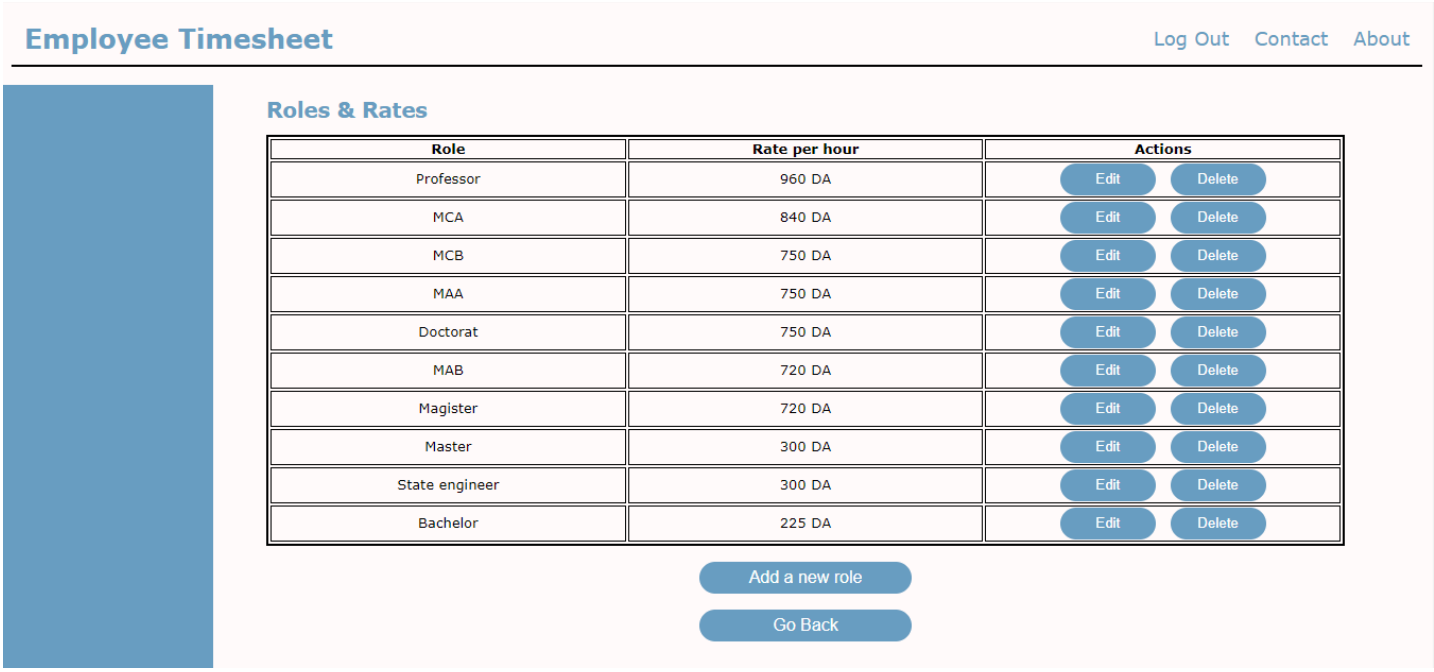


Figure 3.4 Edit rate per hour interface

→ **Add a new role interface**

This button allows the supervisor to create a new role and assigned an hourly rate for it:

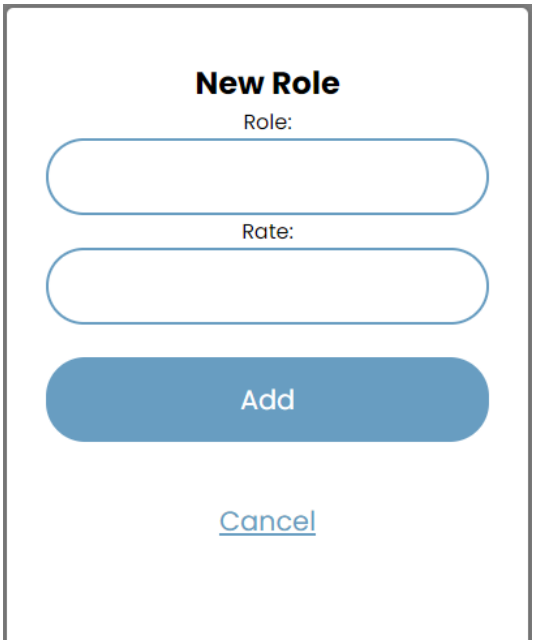


Figure 3.5 Add new role interface

→ **Edit role interface**

In this page, the supervisor can edit either the rate associated with each role:

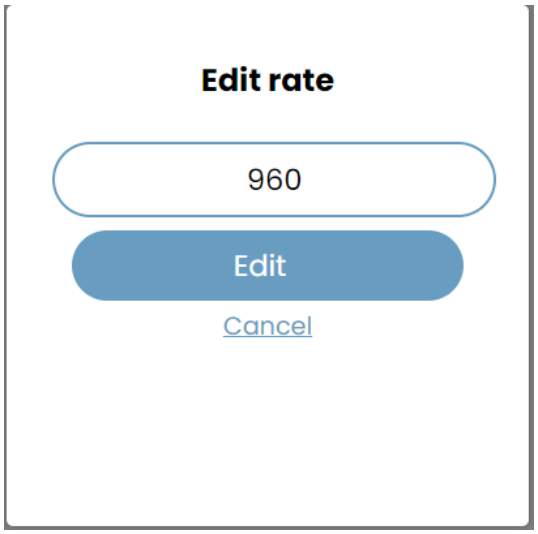
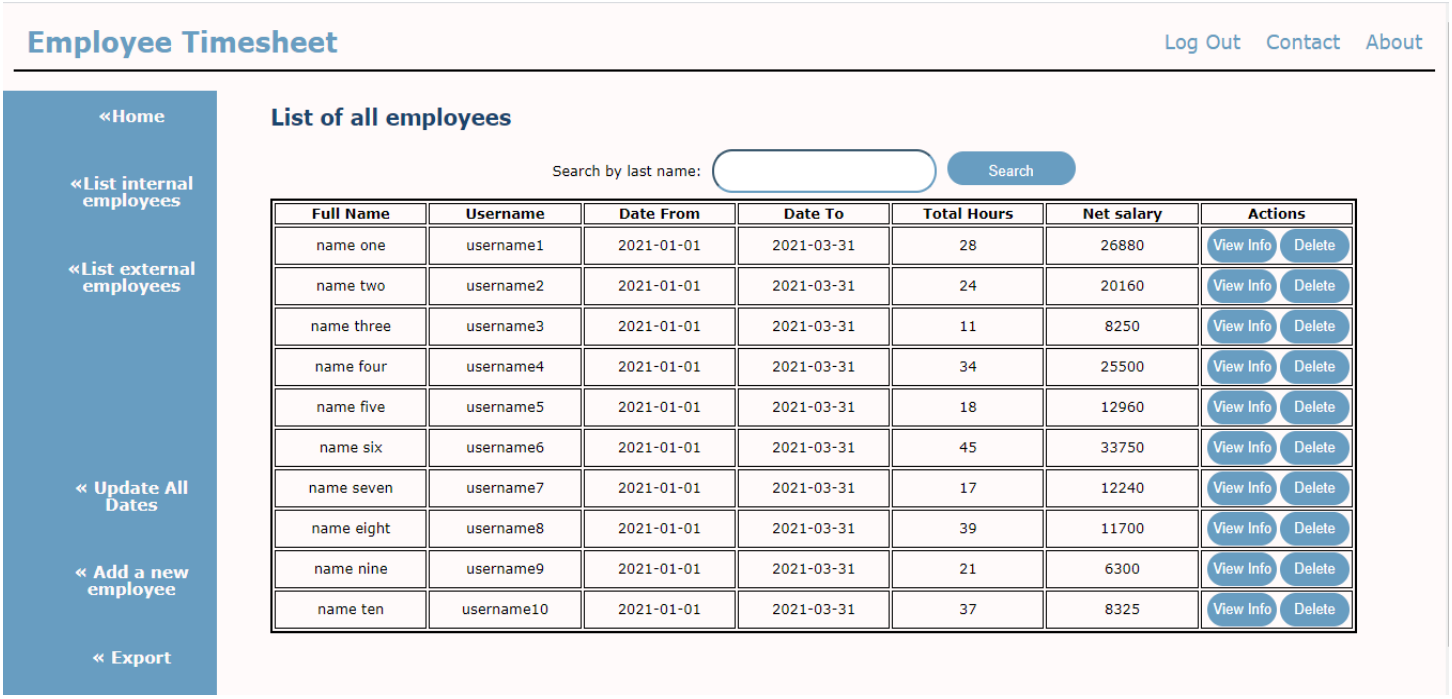


Figure 3.6 Edit rate interface

3.2.3.3 List of all employees interface

In this page, the supervisor is able to see the list of all employees linked with their *username*, the total *worked hours* in a selected *period* (using *update all dates* button) and the *salary* for that period. In addition, the supervisor can delete, search for or add an employee and also view the personal information related to each one in the list. As a final report the supervisor can export all of this information as a pdf file (using *export as pdf*).



Full Name	Username	Date From	Date To	Total Hours	Net salary	Actions
name one	username1	2021-01-01	2021-03-31	28	26880	View Info Delete
name two	username2	2021-01-01	2021-03-31	24	20160	View Info Delete
name three	username3	2021-01-01	2021-03-31	11	8250	View Info Delete
name four	username4	2021-01-01	2021-03-31	34	25500	View Info Delete
name five	username5	2021-01-01	2021-03-31	18	12960	View Info Delete
name six	username6	2021-01-01	2021-03-31	45	33750	View Info Delete
name seven	username7	2021-01-01	2021-03-31	17	12240	View Info Delete
name eight	username8	2021-01-01	2021-03-31	39	11700	View Info Delete
name nine	username9	2021-01-01	2021-03-31	21	6300	View Info Delete
name ten	username10	2021-01-01	2021-03-31	37	8325	View Info Delete

Figure 3.7 List of all employees interface

a. Add a new employee interface

By filling the following form, the supervisor can provide each new employee with an account to access his dashboard

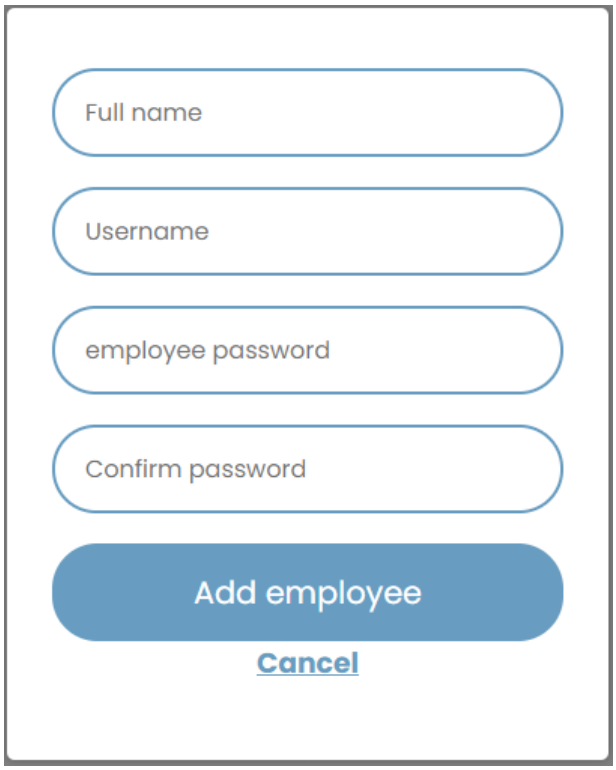


Figure 3.8 Add employee interface

b. Update all dates interface

In this page, the supervisor can select the start and the end of the period to calculate the total worked hours and salary for each employee:

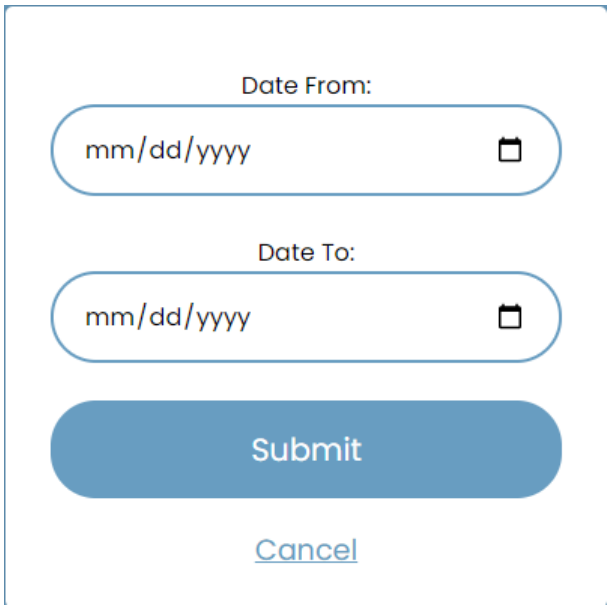


Figure 3.9 Update all dates interface

c. Export as pdf interface

"Export as pdf" button will leads the supervisor to a pdf file that contains final report of all employees with their roles and their worked hours during the period that he selected previously in addition to all detailed information about the salary and tax deduction:

Ministry of Higher Education and Scientific Research
University of M'HAMED BOUGUERRA BOUMERDES

STAND-IN STATE

Temporary teacher
All
Date From : 2021-01-01 To 2021-03-31

N	Id	Full Name	Degree	Hours	Gross	Ret SS (9%)	Taxable	Ret GIT (10%)	Net to pay
1	1	name one	Professor	28	26880	2419.2	24460.8	2446.08	22014.72
2	2	name two	MCA	24	20160	1814.4	18345.6	1834.56	16511.04
3	3	name three	MCB	11	8250	742.5	7507.5	750.75	6756.75
4	4	name four	MAA	34	25500	2295	23205	2320.5	20884.5
5	5	name five	MAB	18	12960	1166.4	11793.6	1179.36	10614.24
6	6	name six	Doctorat	45	33750	3037.5	30712.5	3071.25	27641.25
7	7	name seven	Magister	17	12240	1101.6	11138.4	1113.84	10024.56
8	8	name eight	Master	39	11700	1053	10647	1064.7	9582.3
9	9	name nine	State engineer	21	6300	567	5733	573.3	5159.7
10	10	name ten	Bachelor	37	8325	749.25	7575.75	757.575	6818.175

page1/2

Ministry of Higher Education and Scientific Research
University of M'HAMED BOUGUERRA BOUMERDES

STAND-IN STATE

Temporary teacher
All
Date From : 2021-01-01 To 2021-03-31

Total	274	166065	14945.85	151119.15	15111.915	136007.235

Figure 3.10 Export as pdf page

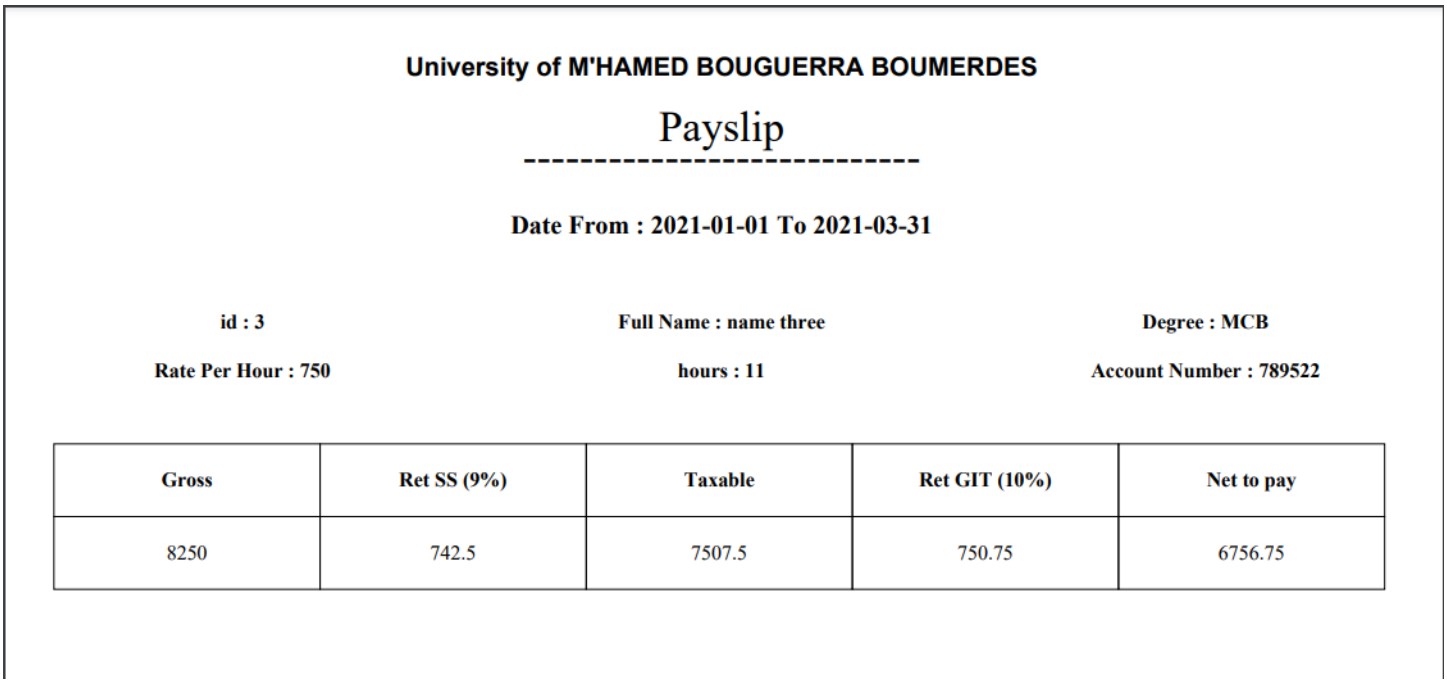
d. View info interface

The following figure shows the information related to each employee that can be seen by the supervisor and can be exported as pdf with additional detailed information:

Figure 3.11 View info interface

→ **Export as pdf interface**

This button will link the supervisor to a detailed report that contains all the calculations related to the salary and tax deduction of each individual employee:



University of M'HAMED BOUGUERRA BOUMERDES

Payslip

Date From : 2021-01-01 To 2021-03-31

id : 3 **Full Name : name three** **Degree : MCB**

Rate Per Hour : 750 **hours : 11** **Account Number : 789522**

Gross	Ret SS (9%)	Taxable	Ret GIT (10%)	Net to pay
8250	742.5	7507.5	750.75	6756.75

Figure 3.12 Export information as pdf page

→ **View timesheet interface**

This button will link the supervisor to the detailed timesheet of the employee during the previously selected period:

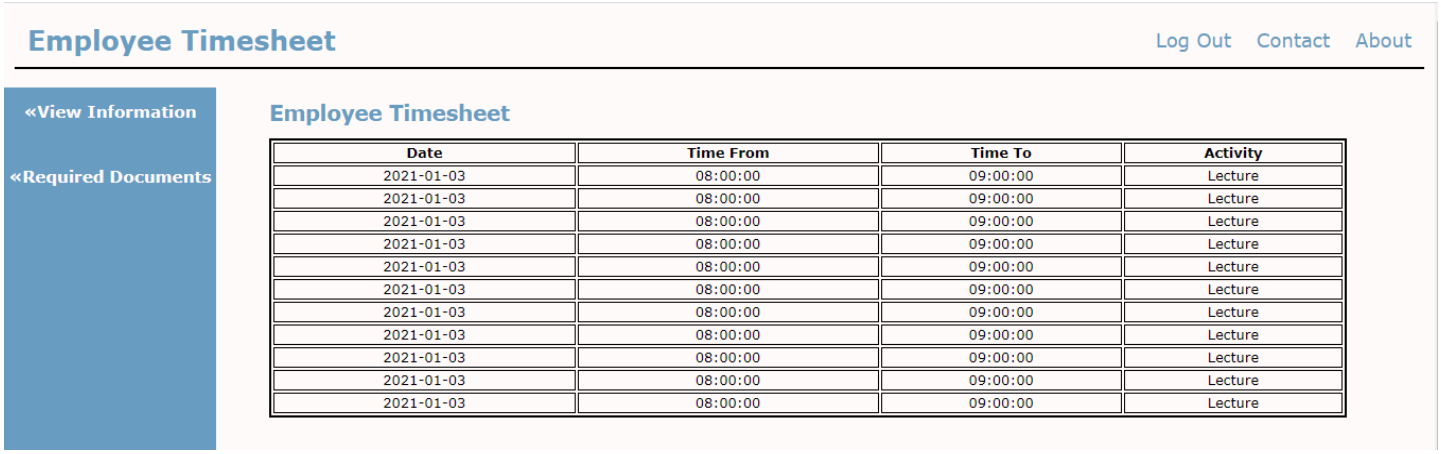


Figure 3.13 View timesheet interface

→ **Required documents interface**

This interface represents the list of the required documents that needs to be submitted by the employee which can be expanded by adding new files correspond to the state of the employee

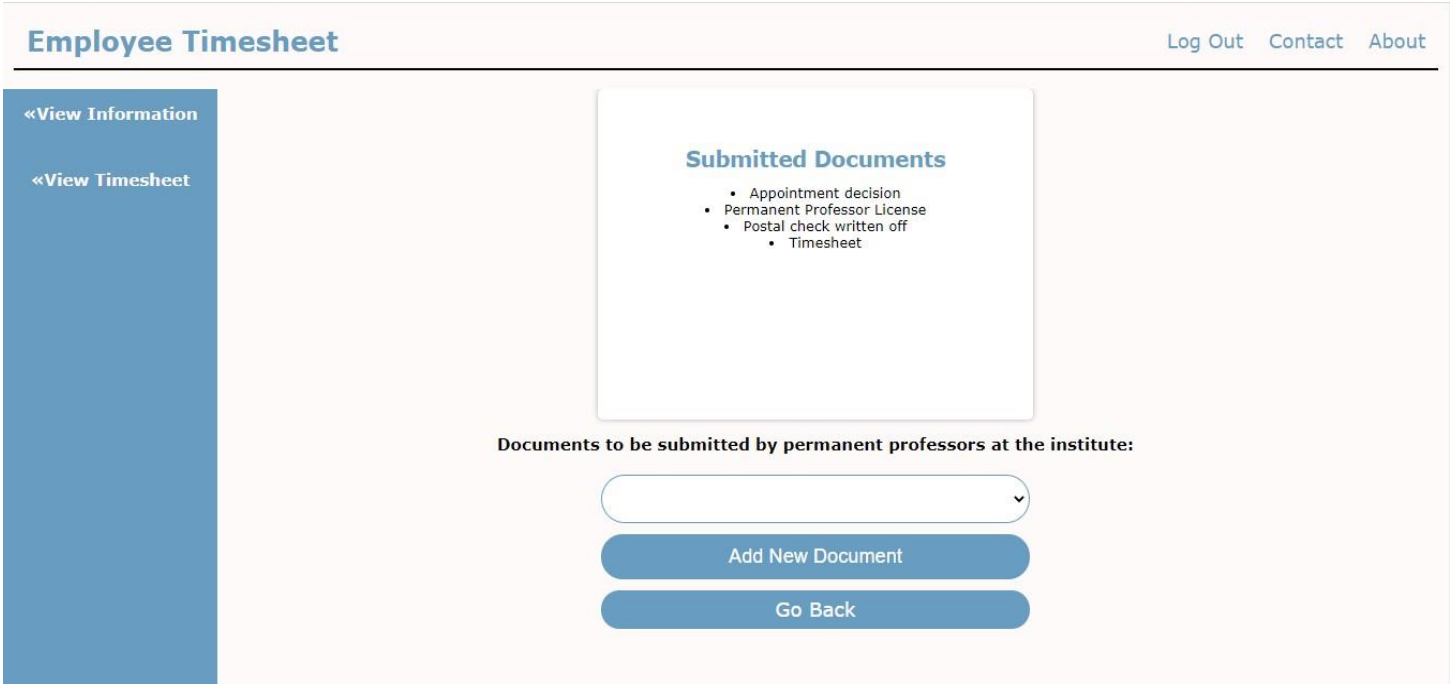


Figure 3.14 Required Documents interface

3.2.3.4 List of external employees interface

The following interface list all the external employees each associated with his *username*, the total *worked hours* in the previously selected *period* (using *update all dates* button) and the *salary* for that period. In addition to this the supervisor can delete or search for an employee and also view the personal information related to each one in the list. As a final report the supervisor can export all of these information as a pdf file (using *export as pdf*).

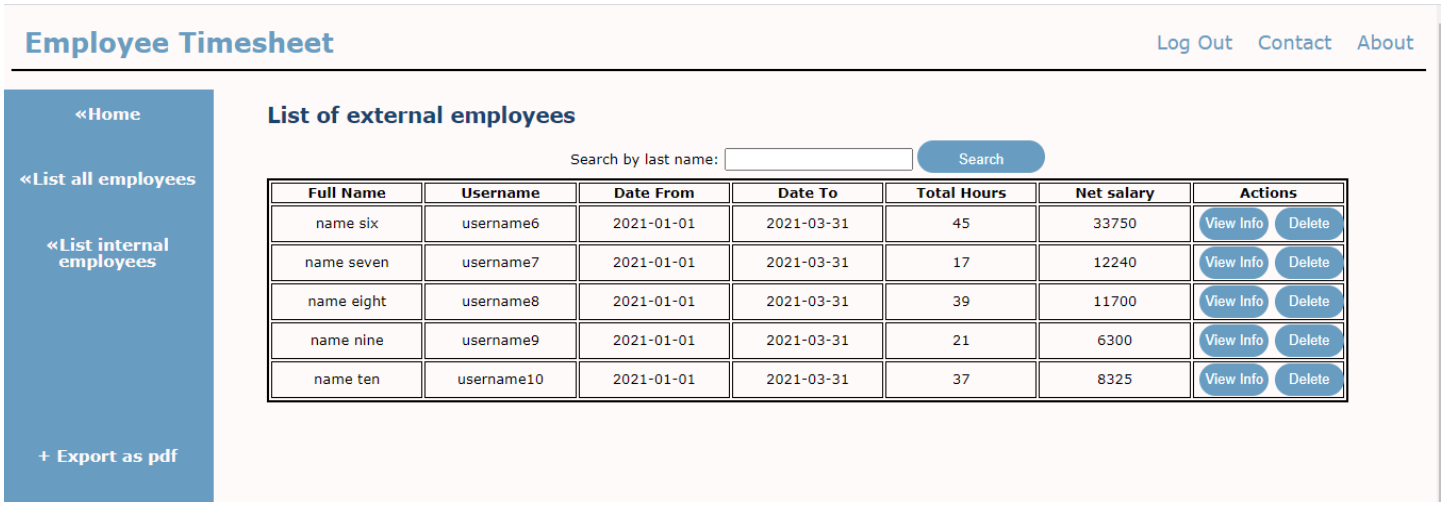


Figure 3.15 List of external employees interface

3.2.3.5 List of internal employees interface

The following interface list all the internal employees each associated with his *username*, the total *worked hours* in the previously selected *period* (using *update all dates* button) and the *salary* for that period. In addition to this the supervisor can delete or search for an employee and also view the personal information related to each one in the list. As a final report the supervisor can export all of these information as a pdf file (using *export as pdf*).

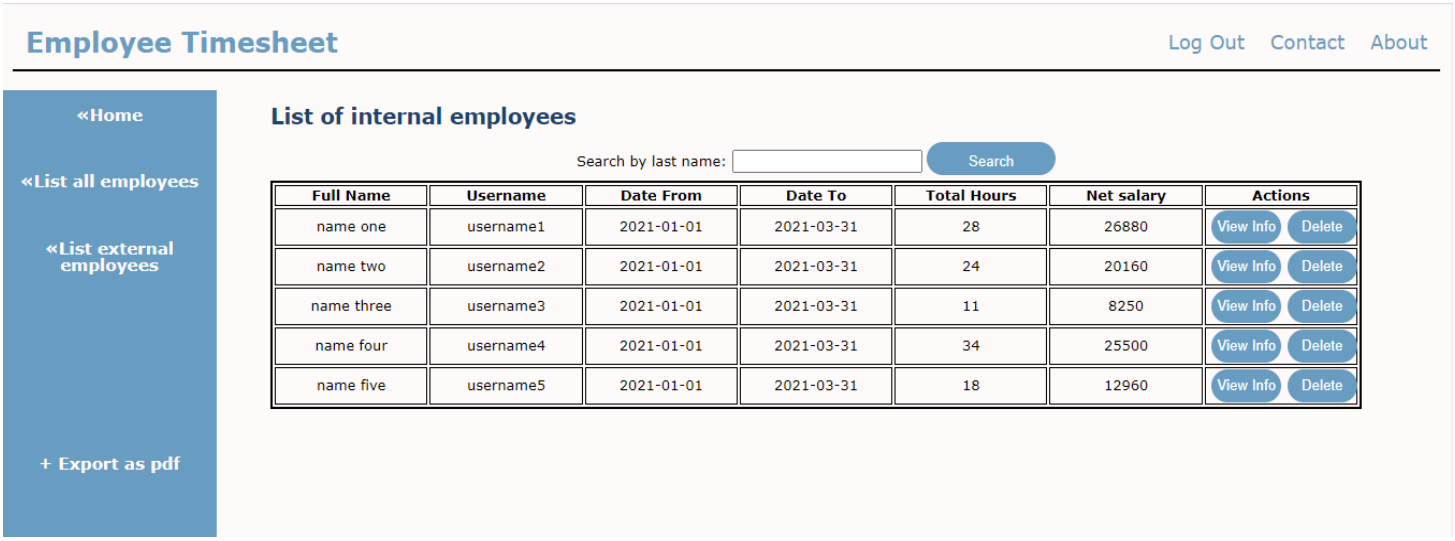


Figure 3.16 List of internal employees interface

3.2.4 Employee Dashboard

3.2.4.1 Employee home interface

Once the user has logged in as an employee the following interface will be displayed to him, and he can choose the page he wants to enter according to his needs:

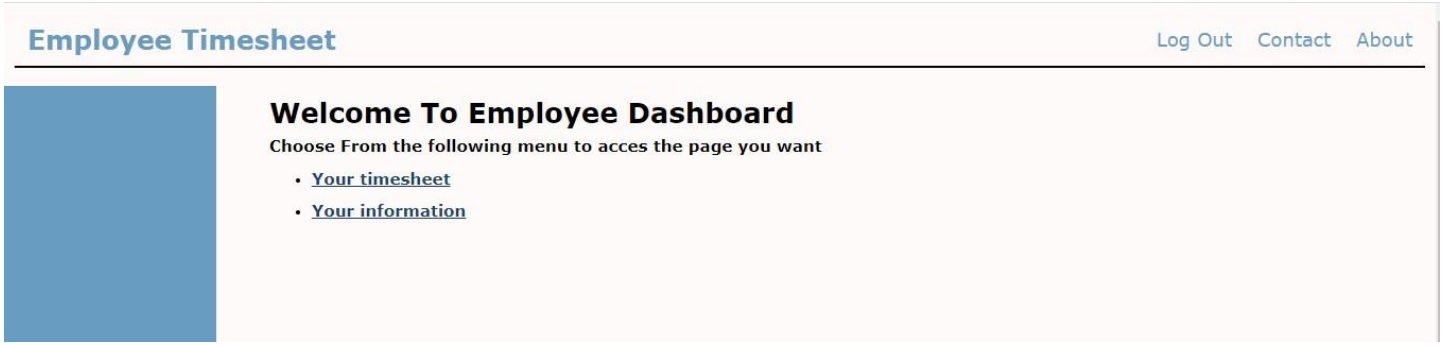


Figure 3.17 Employee home interface

3.2.4.2 Timesheet interface

In this page, the employee can look at all his working hours each associated with the activity, the start and the finish time of that hour. Besides this, the employee can add, delete, or edit a row that contains all the information mentioned below. At the end, he can search for a date and that will returns all his working hours during that date:

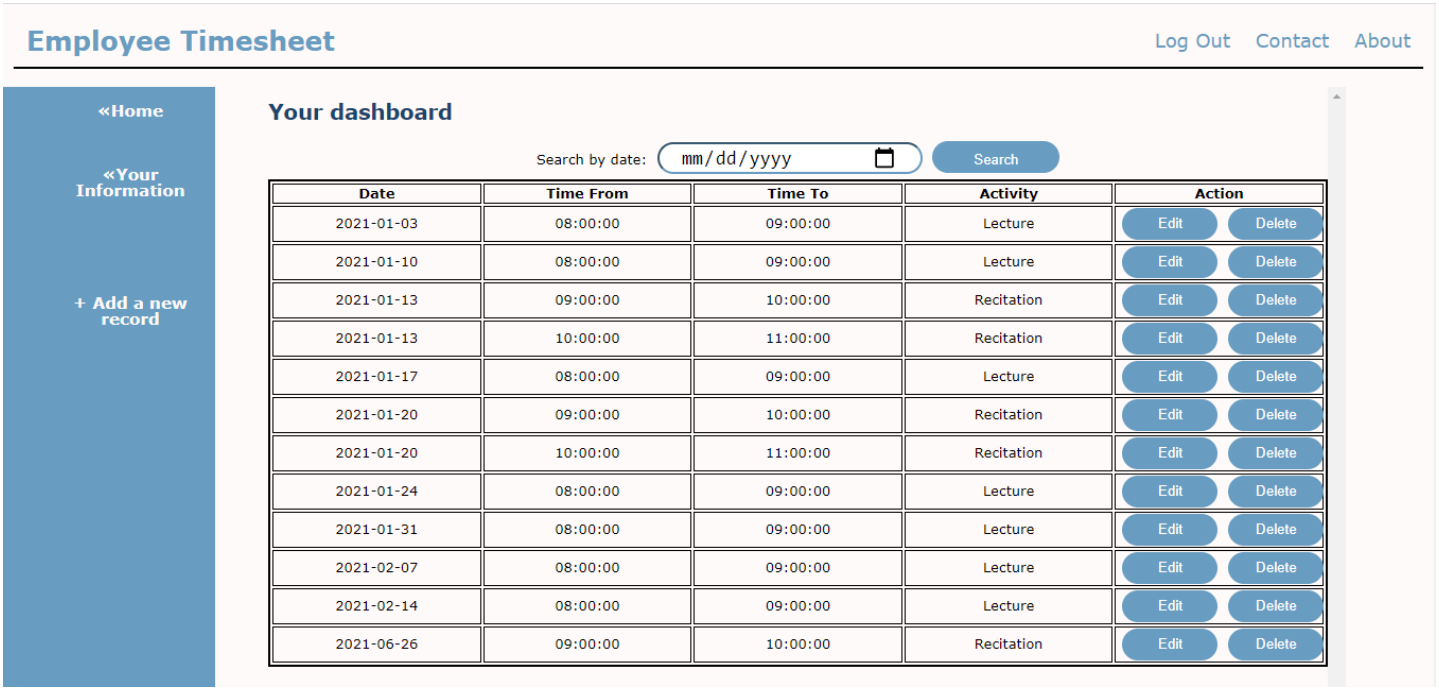


Figure 3.18 Employee timesheet interface

→ **Add a new record interface**

The employee should fill the following form in order to add a new record that corresponds to a new hour in his timesheet that requires the following information

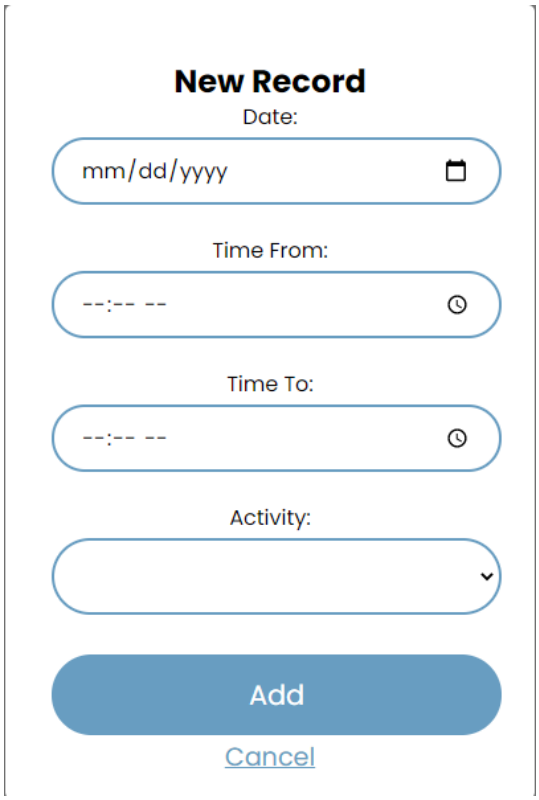


Figure 3.19 Add a new record interface

→ **Edit record interface**

This interface represents the form of updating a record that requires the following information:

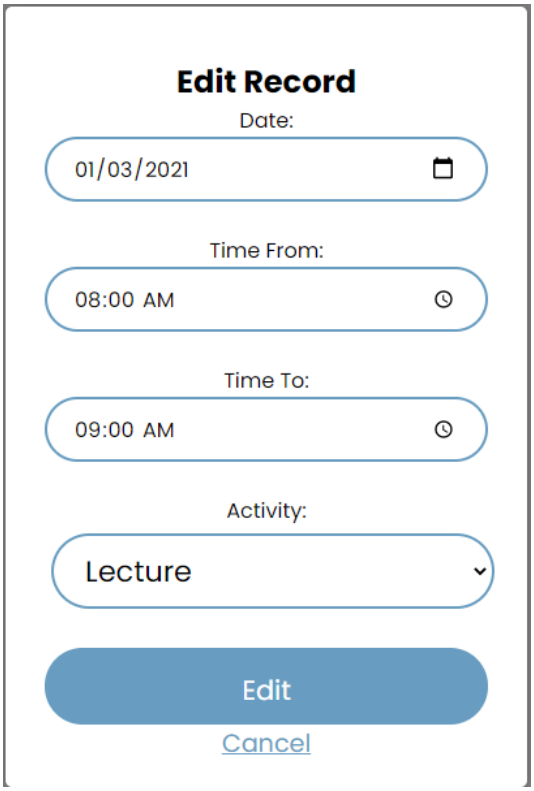


Figure 3.20 Edit record interface

→ **Search by date interface**

By selecting the date in *select by date* form the employee will be leaded to the timesheet corresponds to that date:

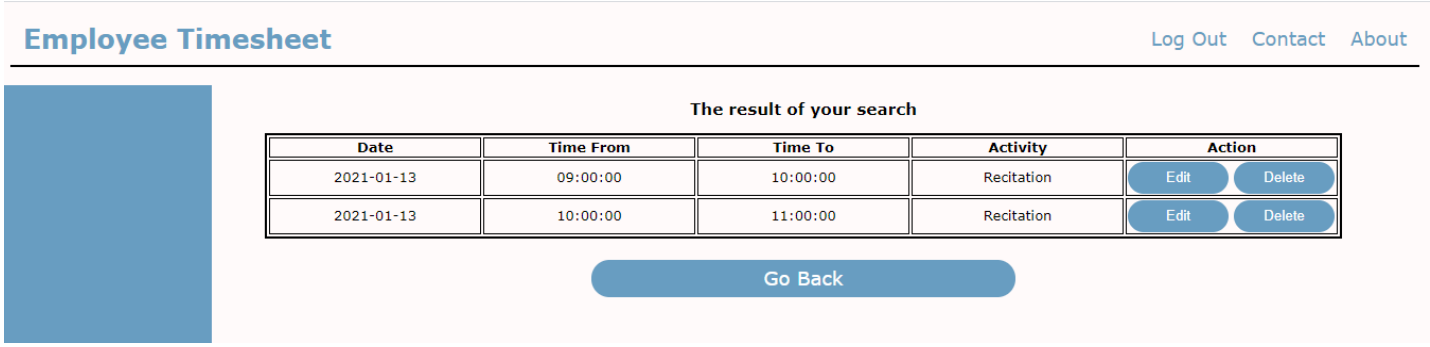


Figure 3.21 Search by date interface

3.2.4.3 Employee information interface

In the following interface, the employee can look at all his information note that those information can be edited by the employee so they are displayed to the supervisor.

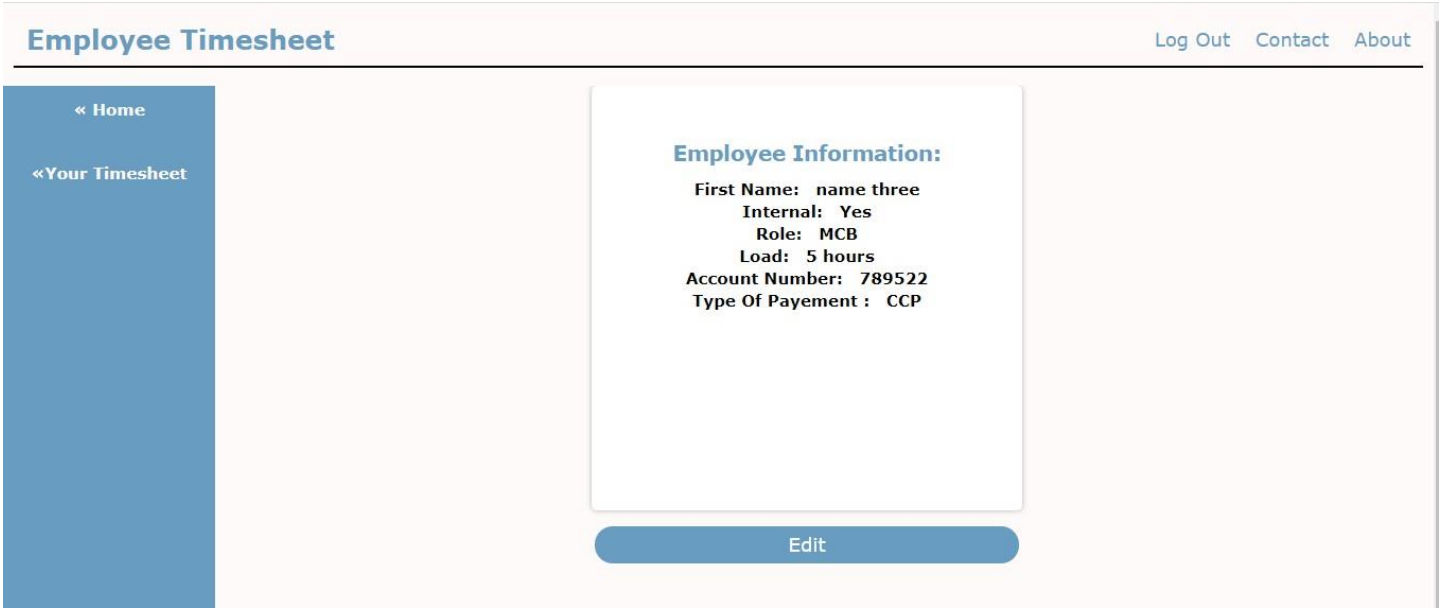


Figure 3.22 Employee information interface

→ **Edit information interface**

The following form is used by the employee in order to edit his information

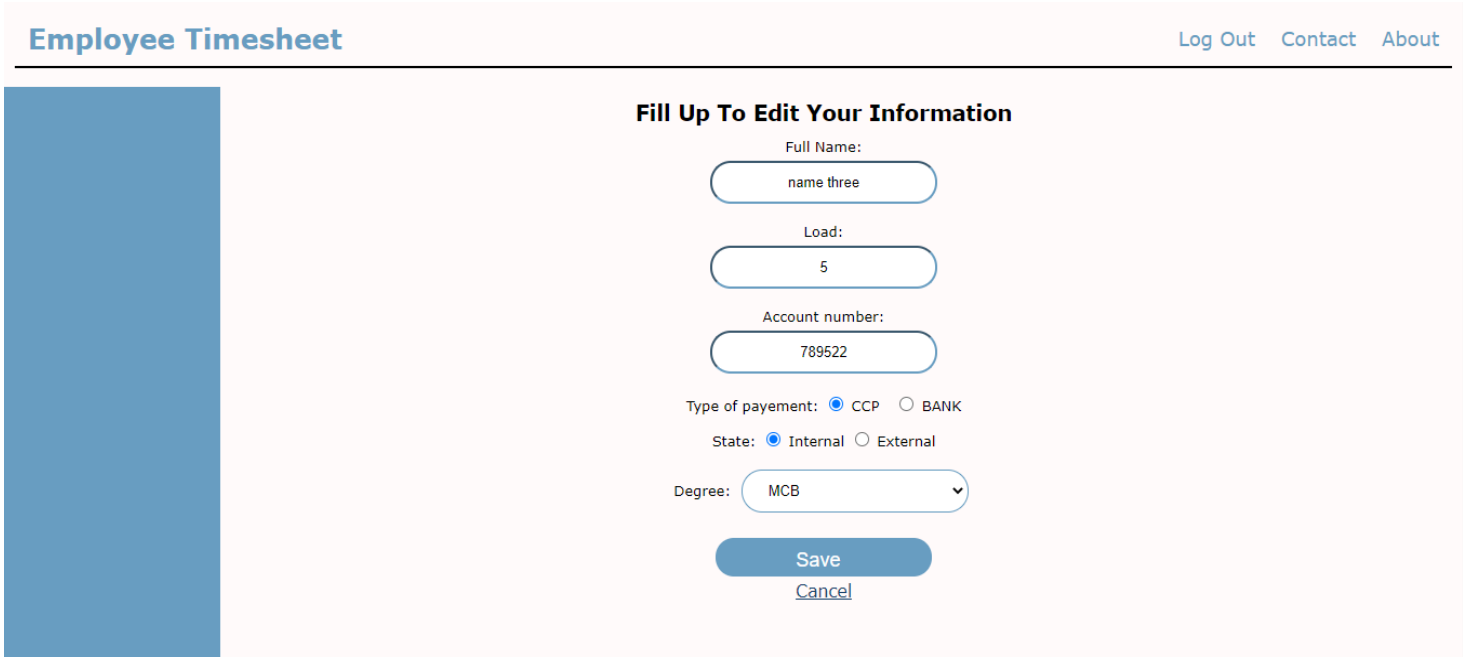


Figure 3.23: Edit information interface

3.2.5 Contact us interface

This page represents a hypothetical contact page that can be used to contact the office responsible for managing the weال application in case of any problem or enquiries:

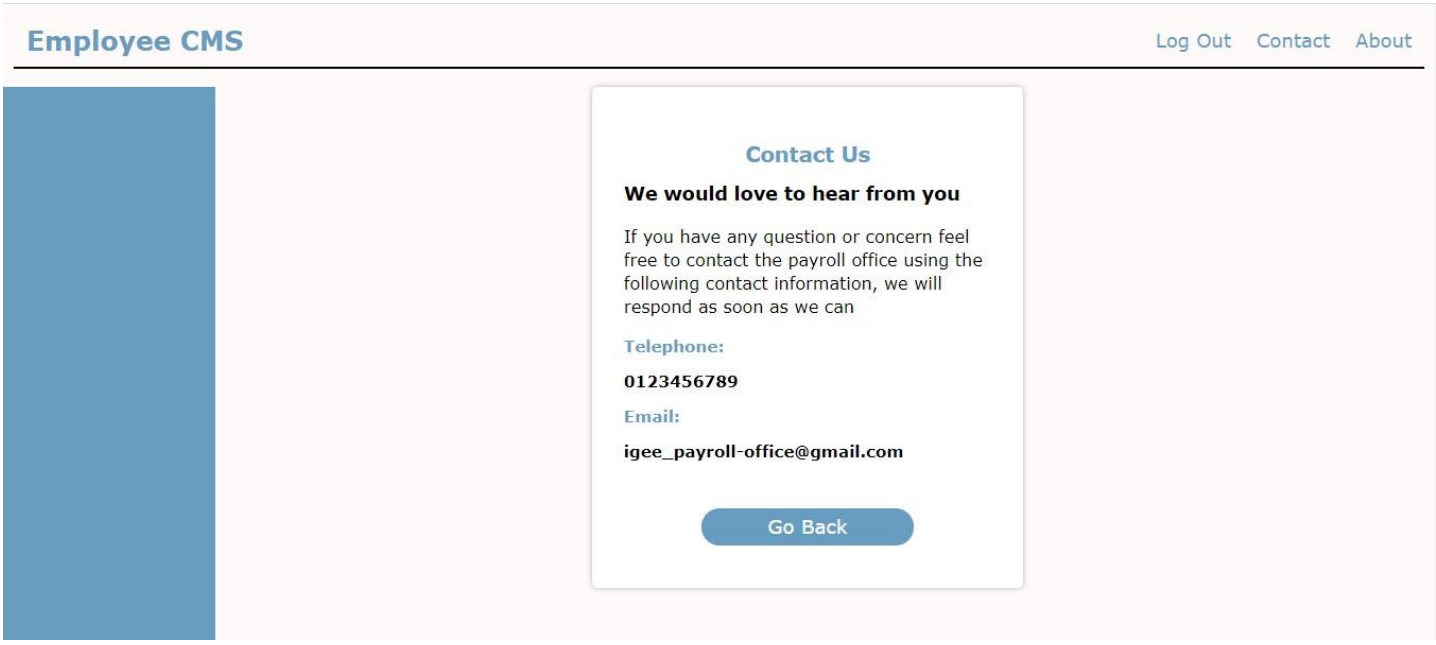


Figure 3.24 Contact us interface

3.2.6 About us interface

This interface provides a small overview about the web application

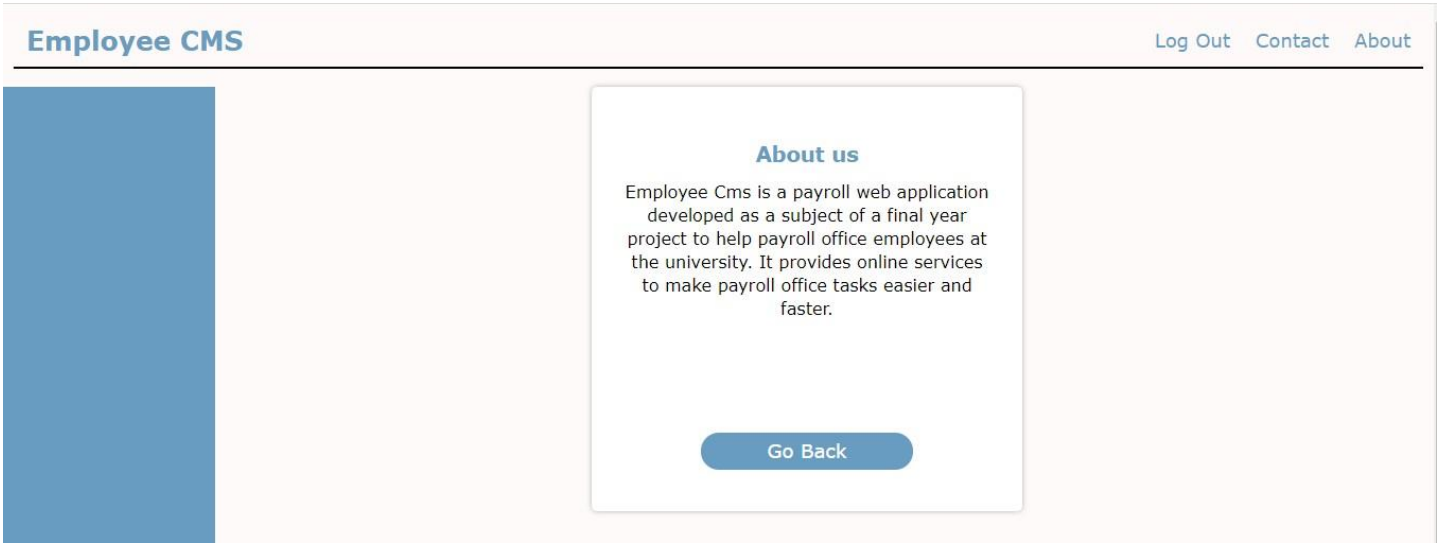


Figure 3.23 About us interface

3.3 Conclusion

In this chapter, we have given a brief explanation considering the different user's interfaces in our web application. As we went through this illustration, we have seen that those user interfaces are designed in such a simple and an easy way to use and provide a rapid access to all required information and services.

General Conclusion

The focus of this work was to use programming languages and available technologies to design and develop a web application intended for managing the work of the payroll office in the institute. After several discussions with the office staff, we managed to collect and analyze the problems encountered by them.

The main purpose of this project is to give technical solutions that are able to automate the salary calculations, and to manage the employees by providing them with accounts so every employee can update his timesheet and information to give the ability to the office to review those data in an easy and organized way and store them in a secured and centralized database.

We have implemented a web application, so it can be accessed from anywhere at any time using the internet.

This project, which falls in the field of web development, was very interesting and introduced us to new concepts, and allows us to improve our knowledge and skills in the programming field.

Future Work

Any project no matter how great, easy and flexible it is, it never escape the fact that it could be enhanced by adding more functionalities.

The first version of the employee CMS is achieved; but there are many other features that can be added in order to enhance the developed application and get a better version. Some of the proposed enhancements is to give the employee the ability to submit his documents and sign them online in a secure manner.

Webography

[1] https://techterms.com/definition/web_application. Consulted (28/05/2021)

[2] https://searchsoftwarequality.techtarget.com/definition/Web-application-Web-app. Consulted (28/05/2021)

[3] https://www.tutorialspoint.com/restful/restful_messages.htm. Consulted (28/05/2021)

[4] http://www.landofcode.com/html-tutorials/url-format.php. Consulted (29/05/2021)

[5] https://en.wikipedia.org/wiki/Payroll. Consulted (13/06/ 2021)

[6] https://www.patriotsoftware.com/blog/payroll/what-is-payroll/. Consulted (13/06/ 2021)

[7] https://en.wikipedia.org/wiki/Sublime_Text. Consulted (29/05/2021)

[8] https://www.phpmyadmin.net/. Consulted (26/06/2021)

https://techterms.com/definition/wamp. Consulted (29/052021)

[11] https://techterms.com/definition/web_browser. Consulted (29/052021)

[12] https://en.wikipedia.org/wiki/Edraw_Max. Consulted (12/06/ 2021)

[13] https://techterms.com/definition/html. Consulted (12/06/ 2021)

[14] https://www.hostinger.com/tutorials/what-is-html. Consulted (13/06/ 2021)

[15] https://www.w3schools.com/css/css_intro.asp. Consulted (13/06/2021)

[16] https://www.techopedia.com/definition/3929/javascript-js. Consulted (13/06/ 2021)

[18] https://www.uml-diagrams.org/uml-23-diagrams.html. Consulted (27/06/ 2021)

[19] https://www.utm.mx/~caff/doc/OpenUPWeb/openup/guidances/concepts/use_case_model_CD178AF9.html. Consulted (27/06/ 2021)

[20] https://www.uml-diagrams.org/use-case-subject.html. Consulted (27/06/ 2021)

[21] https://www.uml-diagrams.org/use-case-actor.html. Consulted (27/06/ 2021)

[22] https://www.uml-diagrams.org/use-case.html. Consulted (27/06/ 2021)

[23] https://www.uml-diagrams.org/use-case-actor-association.html. Consulted (27/06/ 2021)

[24] https://www.uml-diagrams.org/use-case-include.html. Consulted (27/06/ 2021)

[25] https://www.uml-diagrams.org/use-case-extend.html. Consulted (27/06/ 2021)

[26] https://www.uml-diagrams.org/use-case-diagrams.html. Consulted (27/06/ 2021)

[27] https://www.geeksforgeeks.org/unified-modeling-language-uml-sequence-diagrams/. Consulted (27/06/ 2021)

[28] https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-sequence-diagram/. Consulted (27/06/ 2021)

[29] https://www.visual-paradigm.com/VPGallery/diagrams/Sequence.html.

Consulted (27/06/ 2021)

[30] https://searchdatamanagement.techtarget.com/definition/database.

Consulted (13/07/ 2021)

[32] https://docs.microsoft.com/en-us/azure/architecture/data-guide/big-data/non-relational-data. Consulted (13/07/ 2021)

[33] https://www.guru99.com/relational-data-model-dbms.html. Consulted (13/07/ 2021)

[34] https://www.javatpoint.com/dbms-integrity-constraints. Consulted (13/07/ 2021)

Bibliography

[10] Luke Welling,Laura Thompson,"Php and MySQL Web Development Tool", 2nd edition. Page 3

[17] Grady Booch, James Rumbaugh, and Ivar Jacobson, Unified Modelling Language User Guide, 2nd edition

[31] Ramez Elmasri, Shamkant B. Navathe, "Fundamentals of Database Systems", 6th edition. Page5

People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research

University M'Hamed
BOUGARA - Boumerdes

Institute of Electrical and
Electronic Engineering

Authorization for Final Year Project Defense

Academic year: 2020/2021

The undersigned supervisor: ..... A. ZITOUNI .....
authorizes the student(s):

Fadil ..... Sara ..... Option... computer

Ladgi ..... Lina ..... Option... computer

to defend his / her / their final year Master program project entitled:

Design and Implementation of a Web Based Management System - Case of the Payroll Office of The IGEE

during the ☑ July ☐ September session.

Date: 28 /06/ 2021

The Supervisor

The Department Head