Educational Institution

«Grodno State University Of Yanka Kupala»

Faculty of Mathematics And Informatics

Department of Modern Technologies of Programming

**Implementation « Development of Personnel Management System for a Private Company»**

Nkulu Kilumba

4th year student

173 «Information Technology Software»

Daily form of education

|  |  |
| --- | --- |
| «Admission to presentation»  Head of department  modern technologies of programming  Rudikova Lada  \_\_\_ \_\_\_\_\_\_\_ 2021 | Scientific advisor  Rudikova Lada,  head of modern software  technologies department,  Ph. D in physics and math,  association professor |

**Grodno, 2021**

**Contents**

[**THE AIM AND ISSUES OF GRADUATION WORK 4**](#_Toc65413858)

[**INTRODUCTION 6**](#_Toc65413859)

[Background of the project 6](#_Toc65413860)

[Research Significance 6](#_Toc65413861)

[Design Objectives 6](#_Toc65413862)

[**CHAPTER 1 ANALYSIS OF PERSONAL MANAGEMENT** 8](#_Toc65413863)

[1.1. System similar to this project 8](#_Toc65413864)

[Analysis of IceHrm 8](#_Toc65413865)

[Advantages of IceHrm 8](#_Toc65413866)

[Disadvantages of IceHrm 9](#_Toc65413867)

[Advantages 9](#_Toc65413868)

[Disadvantages 10](#_Toc65413869)

[1.2. Information subject area 10](#_Toc65413870)

[1.3. Project development language and tools 10](#_Toc65413871)

[1.4. Conclusion 12](#_Toc65413872)

[**CHAPTER 2 DESIGN OF PERSONAL MANAGEMENT SYSTEM** 13](#_Toc65413873)

[2.1. Project requirement 13](#_Toc65413874)

[2.2. General system architecture 14](#_Toc65413875)

[2.3. UML diagram for system function 15](#_Toc65413876)

[Use case diagram 15](#_Toc65413877)

[State-chart diagram 17](#_Toc65413878)

[Activity diagrams 20](#_Toc65413879)

[Component diagram 23](#_Toc65413880)

[Sequence diagrams 24](#_Toc65413881)

[2.4. System data model 25](#_Toc65413882)

[Conceptual data model 25](#_Toc65413883)

[Physical data model of system 26](#_Toc65413884)

[2.5. Data Warehouse 27](#_Toc65413885)

[Using star schema 27](#_Toc65413886)

[2.6. System Interface Design 29](#_Toc65413887)

[Personal management 29](#_Toc65413888)

[2.7. Conclusions 39](#_Toc65413889)

[**CHAPTER 3 DEVELOPMENT OF SYSTEM** 40](#_Toc65413890)

[3.1. The overall architecture of the system 40](#_Toc65413891)

[40](#_Toc65413892)

[3.2. The architecture of system 41](#_Toc65413893)

[3.3. The architecture of application server 43](#_Toc65413894)

[3.4. Development of database 44](#_Toc65413895)

[3.5. Interaction between Client and Server 45](#_Toc65413896)

[3.6. Module structure 46](#_Toc65413897)

[3.7. Implementation of the project 47](#_Toc65413898)

[3.7.1 User module 47](#_Toc65413899)

[3.7.2 Administrator module 48](#_Toc65413900)

[3.7.3 Manager module 56](#_Toc65413901)

[3.7.4 Clerks module 57](#_Toc65413902)

[3.8. Interface 59](#_Toc65413903)

[3.9. Conclusion 71](#_Toc65413904)

[SUMMARY 72](#_Toc65413905)

[LIST OF SOURCES 73](#_Toc65413906)

# **THE AIM AND ISSUES OF GRADUATION WORK**

Graduation aim – Development of a System for personal management for a private company

Tasks of this graduation work:

* Determine the requirements of the personnel management
* Complete the function design of the personnel management
* select software and technology to design a general model of the website;
* implement system development;
* system development testing and bug fixes;
* prepare presentations and reports.

# **INTRODUCTION**

## Background of the project

The software goes deeper every year and integrates into human life, without

exception in every successful business, companies, industries. The proposed development allows you to optimize and simplify to work with data. Personnel are the backbone of any company therefore their management plays a major role in deciding the success of an organization. Personnel Management Software makes it easy for the Manager to keep track of all records. Each employee in the database is associated with a position can be added and edited when need arises.

## **Research Significance**

In order to increase the quality of the work in every successful companies

have websites, admin or manager have access to data and manage it, nowadays system needs not only beautiful web pages, but also rigorous planning, paying attention to every small link, so as to avoid unnecessary mistakes in transactions. We will use HTML, JavaScript, php, and other technologies to edit web pages, JTDS technology to associate database with dynamic web pages, it establishes its own database, so that the required information can be timely saved and updated, and the administrator can better have a view on the evolution of each project in a timely manner.

## **Design Objectives**

Manual handling of employee information poses a number of challenges.

This is evident in procedures such as leave management where an employee is required to fill in a form which may take several weeks or months to be approved. The use of paper work in handling some of these processes could lead to human error, papers may end up in the wrong hands and not forgetting the fact that this is time consuming. A number of current systems lack employee self-service meaning employees are not able to access and manage their personal information directly without having to go through their HR departments or their managers. Another challenge is that multi-national companies will have all the employee information stored at the headquarters of the company making it difficult to access the employee information from remote places when needed at short notice.

The Personnel Management System for a Private Company project will

design an information system which mainly contains information of projects, users, and personal information, the system will help managers and employee to have access in time and fast.

The system consists of three parts: client website, management system and server.

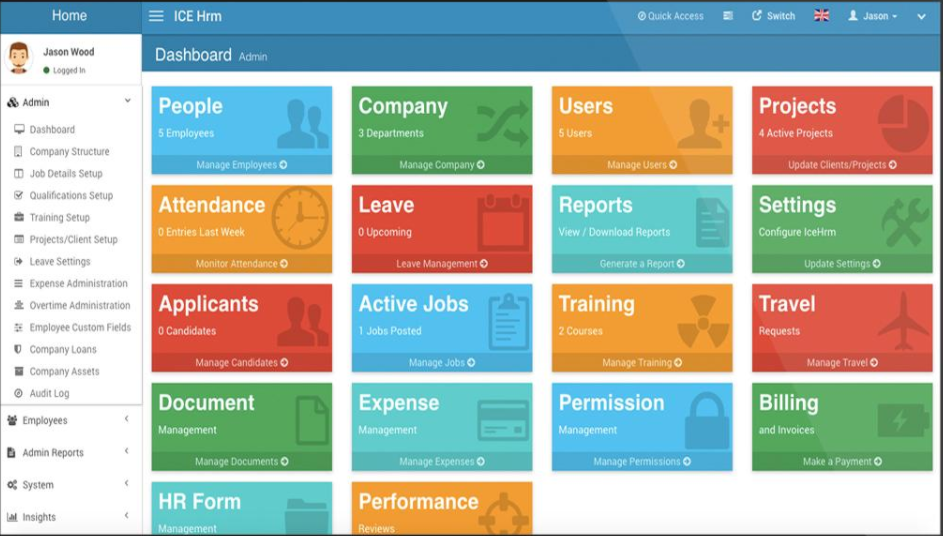
The customer website will show users the main functions provided by the system: project information, customers review. The management system will be used to update the data information of the system server, and to work on customers projects.

# **CHAPTER 1 ANALYSIS OF PERSONNEL MANAGEMENT**

## **System similar to this project**

## **Analysis of IceHrm**

IceHrm employee management system allows companies to centralize confidential employee information and define access permissions to authorized personnel to ensure that employee information is both secure and accessible

**Fig 1.1 IceHrm**

### Advantages of IceHrm

Friendly UI

Good looks

provide more convenient services for users

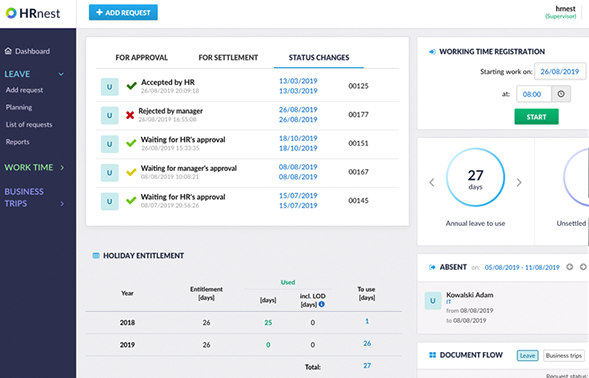
Users can modify their personal information instead of going to the headquarter

### Disadvantages of IceHrm

Limited as the functionality is not adaptable for others region IceHrm don’t have a mobile app and it’s the only negative feedback about the software

**Analysis of HRnest**

HRnest’s works on scheduling functionality automatically captures employee work hours and attendance and lets managers modify work schedules and assign tasks



**Fig 1.2 HRnest interface**

### Advantages

Easy and reliable registration of working time.

Good support in planning and accounting for work.

improve the organization of vacation applications and look after employee matters more efficiently.

### Disadvantages

The spreadsheet needed to complete the aggregate list is chaotic.

System has a trouble to recounting days off allowance when you change the working hours for the employee.

## **Information subject area**

In this project we have three major categories of roles, they are administrator

and user. And administrator have two types, user have two types.

* + - * Root Administrator is a higher system manager they can delete users, have access to data, interact with users, schedule appointment.
* Manager follow up on the project and work with the team
* Personnel: work on project assigned by his manager and send daily report of the evolution of the project

We have two types of user:

* public user can only browse and query basic information about company;
* Register user can schedule an appointment, chat with administrator and add feedbacks on his project about;

## **Project development language and tools**

**PHP -** is a server scripting language, and a powerful tool for making dynamic

and interactive Web pages. PHP is a widely-used, free, and efficient alternative to competitors such as Microsoft's ASP.

**HTML** – Hypertext Markup Language (**HTML**) is the standard markup language

for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript

**CSS - Cascading Style Sheets** (**CSS**) is a style sheet language used for

describing the presentation of a document written in a markup language such as HTML CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript CSS is designed to enable the separation of presentation and content, colours and fonts This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web  pages to share formatting by specifying the relevant CSS in a separate .css file which reduces complexity and repetition in the structural content as well as enabling the .css file to be cached to improve the page load speed between the pages that share the file and its formatting.

**SQL Server, SSMS** – SQL Server is a relational Database Management

System developed and promoted by Microsoft. SSMS (SQL Server Management Studio) is an integrated environment for managing the SQL Server infrastructure. Provides tools for configuring, monitoring, and managing SQL Server instances. In addition, it provides tools for deploying, monitoring, and upgrading data layer components, such as databases and data warehouses used by applications, to generate queries and scripts.

**JavaScript –** JavaScript often abbreviated as Js a programming language

commonly used in web development. It was originally developed by Netscape as a means to add dynamic and interactive elements to websites. ... Like server-side scripting languages, such as PHP and ASP, **JavaScript** code can be inserted anywhere within the HTML of a webpage.

**Node.js** – Node.js is a JavaScript runtime environment based on the Chrome

V8 engine. It uses an event-driven, non-blocking I/O model that makes it lightweight and efficient. In the Node environment, through the modular JavaScript code, plus functional programming, and without regard to browser compatibility issues, the latest ECMAScript standard can be used directly to meet the needs of development engineering.

## **Conclusion**

This chapter explain the background of the project, the scope of development,

and general functionality of the project. The advantages and disadvantages of the project are briefly analyzed with similar completed projects as an example. Finally, the technical and tool support involved in the development project is described

# **CHAPTER 2 DESIGN OF PERSONNEL MANAGEMENT SYSTEM**

## **Project requirement**

The system under development is mainly used to manage data of personnel, and

to work on our client’s projects and to interact with our client. The target audience of this application is people of all ages and professions, so the application must be intuitive and have a low entry threshold when used.

The main function of the system is to be able to work with our client’s projects,

regularly follow up on daily reports, schedule appointment with client and give them a feedback.

The system provides the following functions:

* Administrator: manage the information, add, delete, edit and can delete users
* Manager: schedule appointment with clients, work with personnel and assign task
* Personnel: work on their assigned project and write a daily reports
* Non registered user: can surf read information and can register
* Registered user: can take an appointment, chat with the manager if it’s needed view the evolution of his project and give feedback

## **General system architecture**



**Figure 2.1 General architecture of the system**

**Software architecture** refers to the fundamental structures of a **software**

system and the discipline of creating such structures and systems. Each **structure** comprises **software** elements, relations among them, and properties of both elements and relations.

The **figure 2.1** is the architecture of the project. This project has two systems,

they are the private company system and the management system It has user layer, server layer and data layer three layers.

The first is user layer is user interface, customers using UI to view and

manage system data. And according the http request send data to the server layer.

The second is service layer provides routing transformations, data object

models, SQL map, and linking databases through Sqlsrv to use JDBC.

The third is data layer includes DBMS. It’s an online object storage server.

# **UML diagram for system function**

### **Use case diagram**

A use case diagram at its simplest is a representation of a user's interaction

with the system that shows the relationship between the user and the different use cases in which the user is involved.

In the personnel management system, there are four major user roles, namely administrator, manager, personnel and visitors.



**Figure 2.2 Personnel management use case including visitors**

The figure 2. is used to illustrate the relationship between the four roles of the system and the management personal system. Some visitor can register and view information registered visitor can schedule appointment, give feedbacks as well as read the information

****

**Figure 2.3 Personnel management use case**

The figure 2. shows the most operations users and administrators can do for

managing or assigning tasks, personnels information and project information

### **State-chart diagram**

A state diagram is used to represent the condition of the system or part of

the system at finite instances of time. It’s a behavioral diagram and it represents the behavior using finite state transitions. State diagrams are also referred to as **State machines** and **State-chart Diagrams**. These terms are often used interchangeably. So simply, a state diagram is used to model the dynamic behavior of a class in response to time and changing external stimuli. We can say that each and every class has a state but we don’t model every class using State diagrams. We prefer to model the states with three or more states.



**Figure 2.4 Personnel management login state diagram**

The figure 2.4 shows the login state-chart diagram, it shows the states when

user login an account. If the account is an administrator account, manager or user account the system will direct to the appropriate page if the inputs entered are wrong it will show the error message



**Figure 2.5 Personnel management login state diagram**

The figure 2.5 shows the administrator account state-chart activity and

describe the general state management between components

### **Activity diagrams**

Activity diagrams are graphical representations of workflows of stepwise

activities and actions with support for choice, iteration and concurrency in the unified modelling language activity diagrams are intended to model both computational and organizational processes (i.e., workflows), as well as the data flows intersecting with the related activities. Although activity diagrams primarily show the overall flow of control, they can also include elements showing the flow of data between activities through one or more data stores

**Figure 2.4 activity diagram admin**

The figure 2.4 shows the admin most activities like view information, add new user



**Figure 2.5 activity diagram visitors**

The figure 2.5 shows the visitors activities who can use the system without having a registered account and the registered visitors who can schedule an appointment and give a feedback

**Figure 2.6 activity diagram user**

The figure 2.6 shows the user activities who use the system to manage and to work with tasks

### **Component diagram**

A component diagram, also known as a UML component diagram, describes

the organization and wiring of the physical components in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required function is covered by planned development



**Figure 2.7 Component diagram**

### **Sequence diagrams**

A **sequence diagram** shows object interactions arranged in time sequence. It depicts the objects involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the logical view of the system under development. Sequence diagrams are sometimes called **event diagrams** or **event scenarios**.

****

**Figure 2.8 Admin sequence diagram**

## **System data model**

### **Conceptual data model**

**Conceptual ERD** models the business objects that should exist in a system

and the **relationships** between them. A **conceptual model** is developed to present an overall picture of the system by recognizing the business objects involved. It defines what **entities** exist, NOT which tables

****

**Figure 2.9 conceptual data model**

### **Physical data model of system**

**Physical ERD** represents the actual design of database. It deals with conversion

from logical design into a **schema** level design that will be transformed into **relational** database. When modeling a **physical ERD**, Logical **ERD** is treated as base, refinement occurs by defining primary keys, foreign keys and constraints



**Fig 2.10 physical data model**

# Data Warehouse

A data warehouse is a database of a different kind: an OLAP (online analytical processing) database. A data warehouse exists as a layer on top of another database or databases (usually OLTP databases). The data warehouse takes the data from all these databases and creates a layer optimized for and dedicated to analytics, a data warehouse (DW or DWH), also known as an enterprise data warehouse (EDW), is a system used for reporting and data analysis and is considered a core component of business intelligence DWs are central repositories of integrated data from one or more disparate sources. They store current and historical data in one single placethat are used for creating analytical reports for workers throughout the enterprise. The data stored in the warehouse is uploaded from the operational system (such as marketing or sales). The data may pass through an operational data source and may require data cleansing for additional operations to ensure data quality before it is used in the DW for reporting.

### **Using star schema**

The star schema separates business process data into facts, which hold the measurable, quantitative data about a business, and dimensions which are descriptive attributes related to fact data. Examples of factual data include scenic spot information, the country to which the scenic spot belongs and the time of its existence. Relevant visitors visit, geographical location.



**Figure 2.11 DWH using star schema**

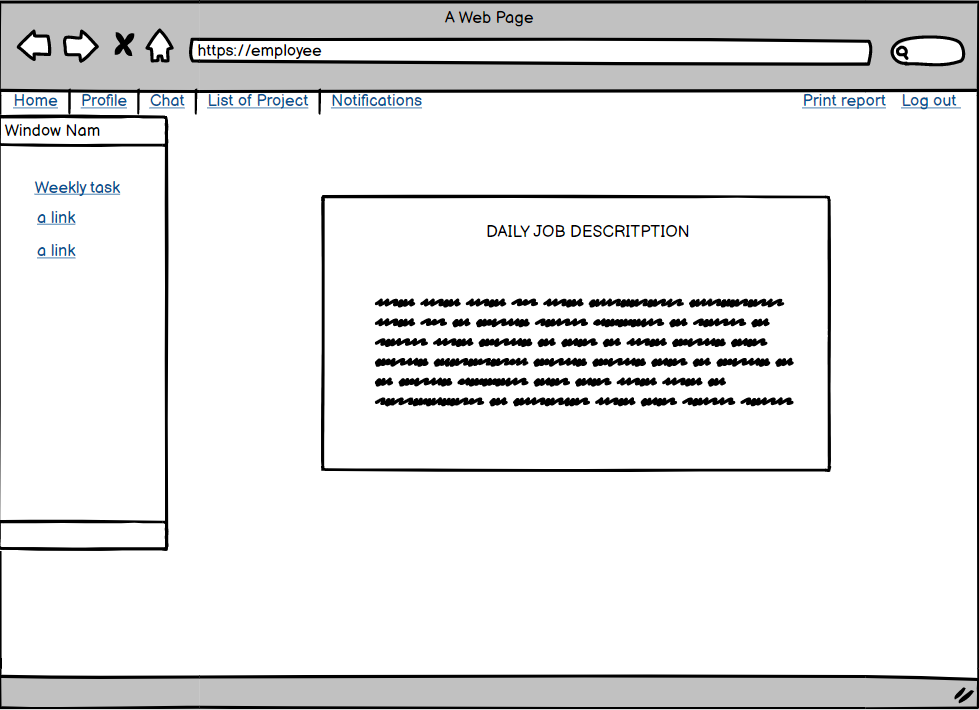


**Figure 2.12 DWH using star schema**

## **System Interface Design**

### Personnel management

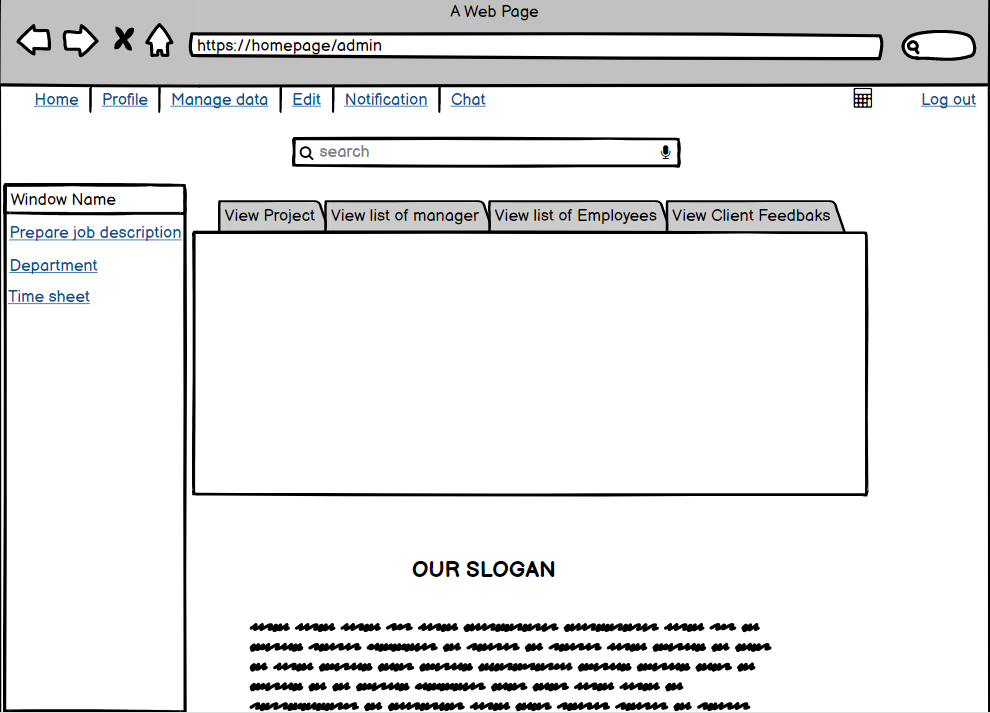
This section describes the basic layout of the pages of each management control in the Personnel Management System.



**Figure 2.16 Personnel management system homepage**

The figure 2.16 shows the basic information of how the systems present

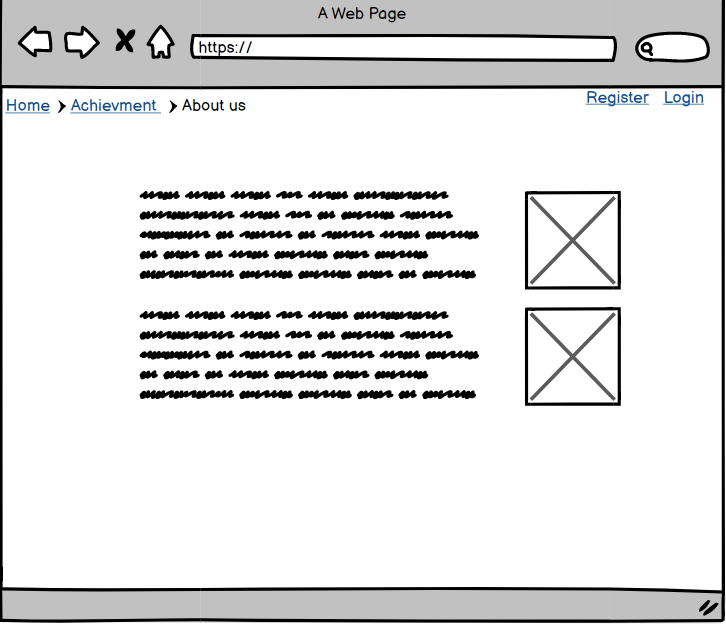
information to administrators, this page mainly describes the daily job description and other functions, in this page click the list of projects the administrator can have more information



**Figure 2.17 the management information page**

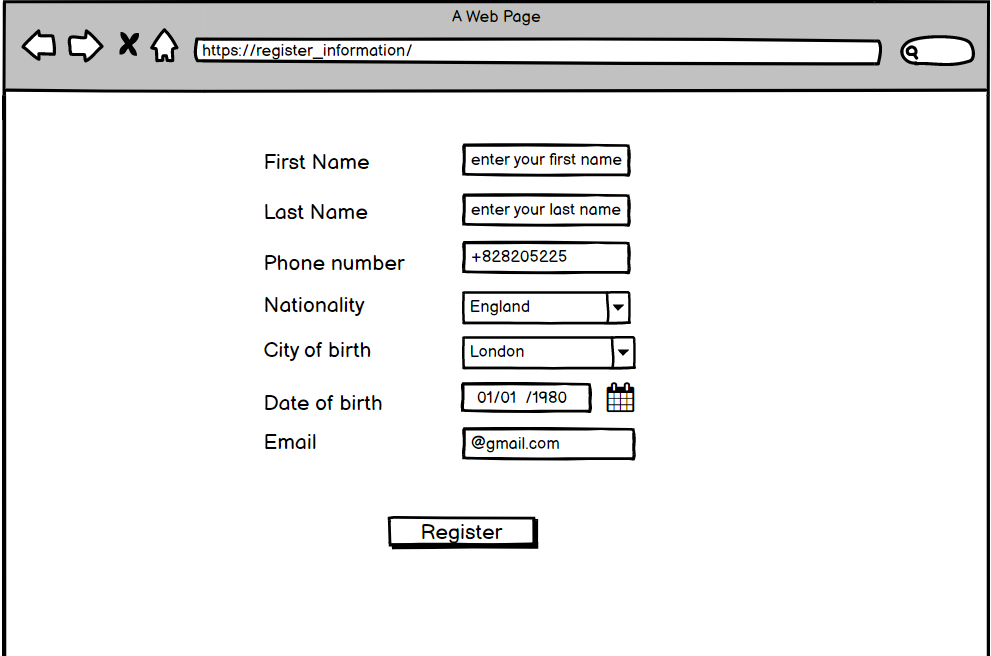
This figure 2.17 shows the management page information this page allow

the administrator to manage data like add users, update information, view the evolution of each project, delete read clients feedbacks. Click the prepare job description to assign task to manager. Click the button department to view information of all manager’s and personnel. Click the button timesheet to view the information



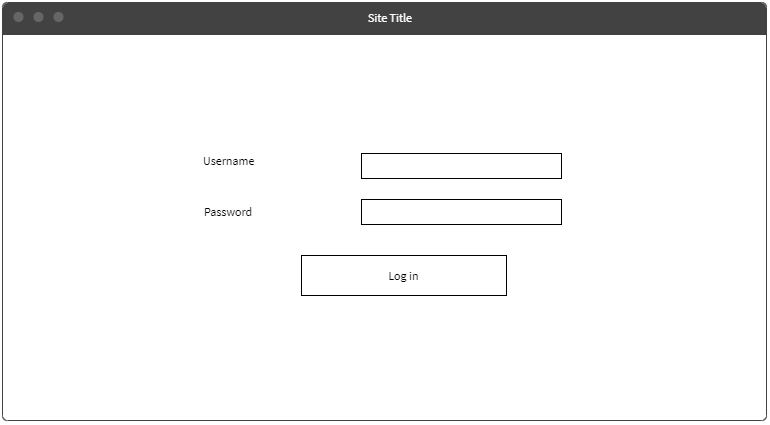
**Figure 2.18 non-register client home page**

This figure 2.18 show the non-register client homepage and some functionalities they can only surf and read information, if they want to register, they have to click on the button “register” and fill-up the form



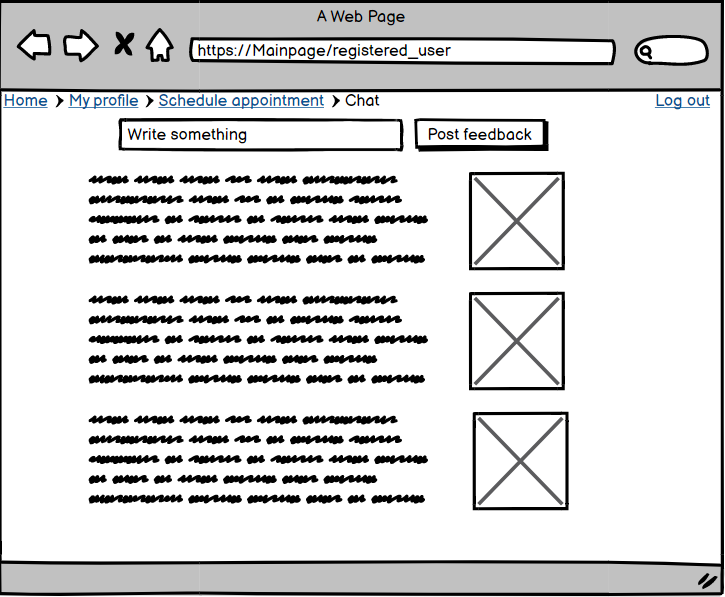
**Figure 2.19 registration page**

The figure 2.19 shows the user page registration to the system by entering information in a provided text field.



**Figure 2.20 login page**

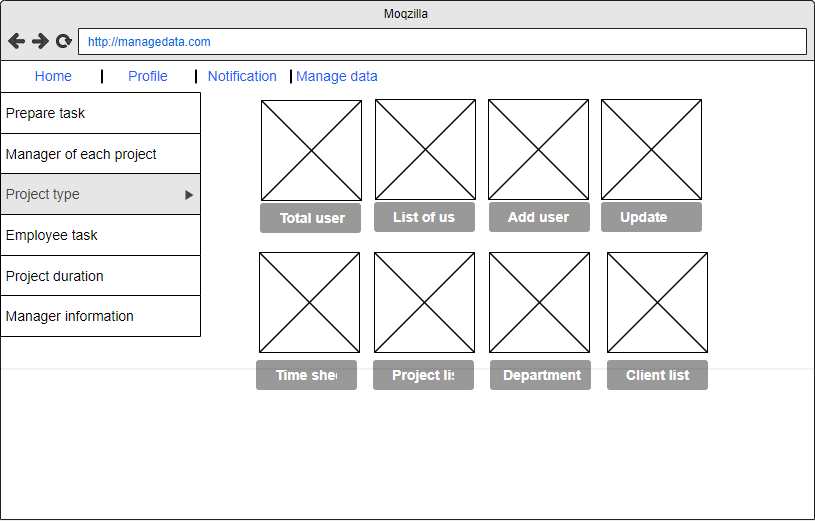
This figure 2.20 In this interface, there will be a button log in. the registered user will have to enter is email and password



**Figure 2.21 client home page**

The figure 2.21 describe the home page on the user side it’s mainly used by

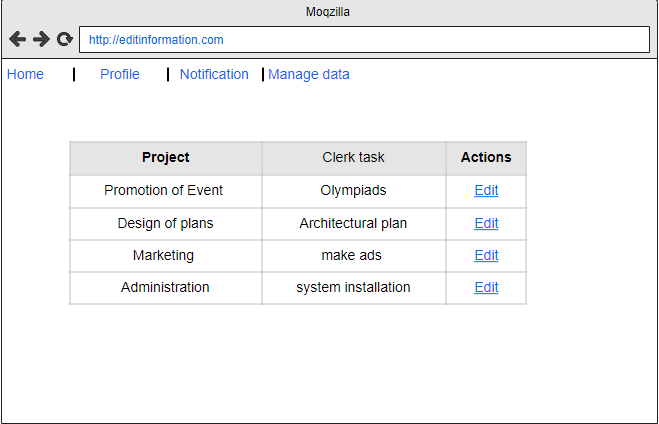
registered users it’s a news presentation interface that will show users the details of the news, the news are users feedbacks their comments according to their experience with the company and their satisfaction concerning our services



**Figure 2.22 Administrator manage page**

The figure 2.21 describe the manage page interface used by administrators

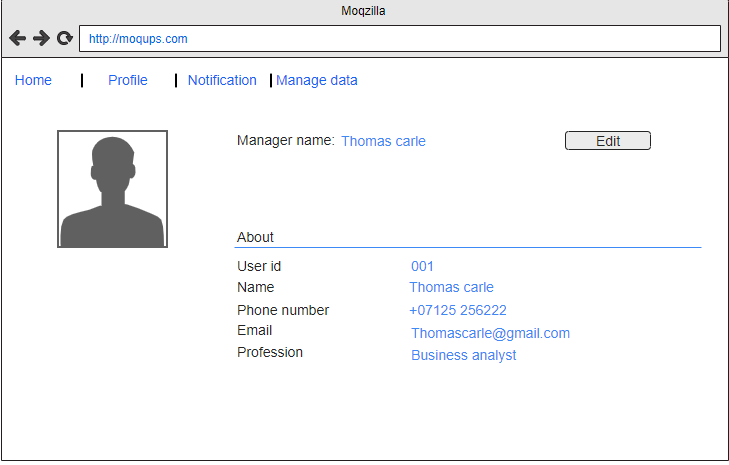
this interface offers to administrators the possibility to have all the possible functionalities in one single page and when the administrators click the button will open a modal containing information or redirect to the appropriate page



**Figure 2.23 Administrator Edit page**

The figure 2.23 describe the edit information page on the administrator side

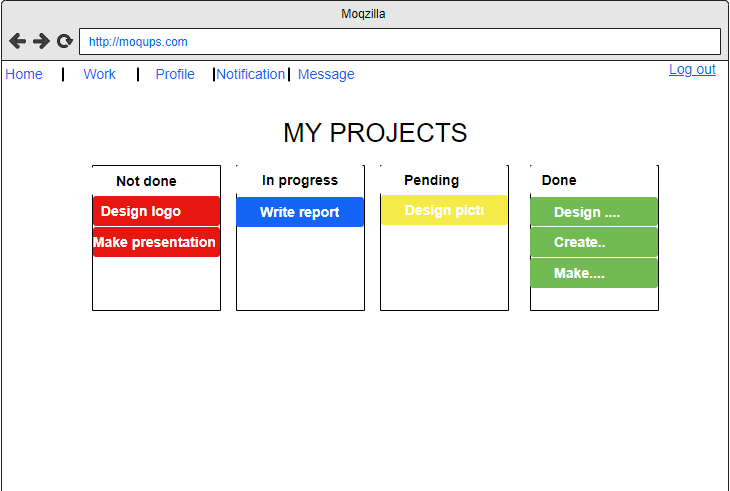
the interface provides the ability to edit data such as project name or personnel task on clicking the edit button a modal will be displayed containing the input form in which we can edit the current information



**Figure 2.24 Administrator profile page**

The figure 2.24 describe the profile page, this interface give the ability to edit

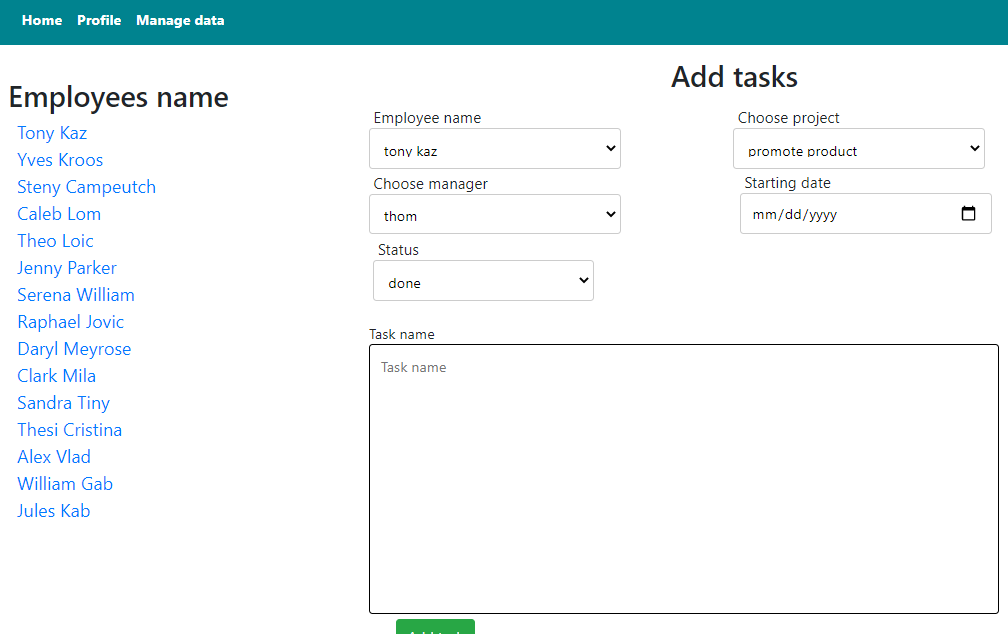
some personal information such as profile picture, phone number, email through the edit button on click a modal will be displayed containing the input form in order to edit current information



**Figure 2.25 manager work page**

The figure 2.25 describe the manager work page, this interface gives the

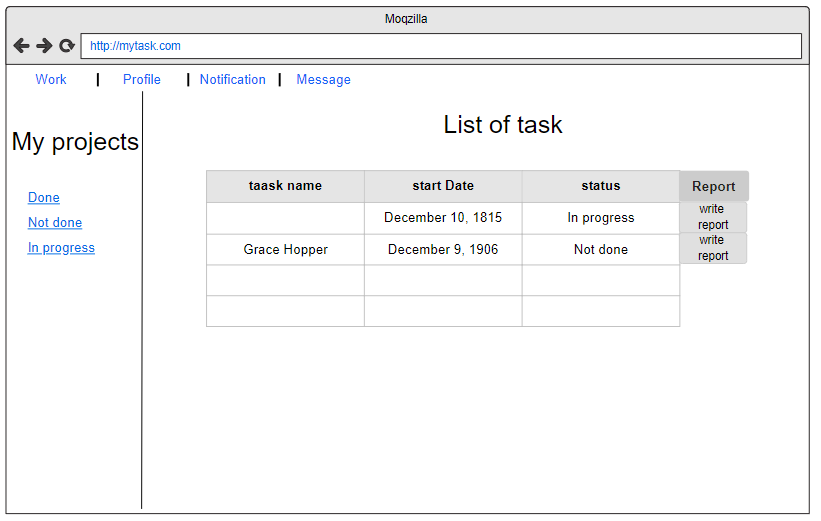
ability to the manager to have a clear view of the evolution of each project those that are not done, in progress, pending, and done the manager can click on any project to read the daily reports wrote by personnel.



**Figure 2.26 Assign task page**

The figure 2.26 describe the assign task page, the interface provides a clear

view to administrators so he can assign task for every projects, select to whom will he assign a task, select for each project a manager and a clerk to work on clients projects, on the left side of the interface there is a list of clerks when the administrator clicks on the link the will redirect to a page containing clerks information such as name, email, phone number background and his skills, on the right side there a set of selects where you can select the project, employee’s name, manager’s name and the start date and below a text area to write tasks or description and the button save to save the data



**Figure 2.27 My tasks page**

This figure 2.27 describe the personnel’s interface, this interface provides a

clear view of the clerk’s task and the ability to write report for each project on click the button write, on the left side the clerk can view the history of achieved task on click any button on the left side.

## **Conclusions**

In the second chapter we have described the main basic needs of project

development. This chapter includes general structures of use case, sequence diagram, component diagram, state-chart activity, activity diagram etc. the chapter also contain the user interface design

# **CHAPTER 3 DEVELOPMENT OF SYSTEM**

## **The overall architecture of the system**

# å¾çæè¿°

**Figure 3.1 – The overall architecture of the system**

All in line server is enough for the personnel management system. Because it's a graduation project, not an online project. It's OK to have file servers, databases, and applications all deployed on a single machine.

But if it is online projects with more and more users, access is getting larger and larger, hard disk, CPU, memory and so on are beginning to be tight. A local server is not enough. At this time, we need to evolve the server, which is the future.

## **The architecture of system**



**Figure 3.2 – The physical architecture of system**

Figure 3.2 is the physical architecture of private company system. It has three

level and three parts.

**Presentation layer**: This layer is a user interface and the user obtains the

website interface through the http request. It has two parts: one is the user system and the other is the management system. Using phpStorm and since IntelliJ IDEA has a built-in web server that can be used to preview and debug the application. This server is always running and does not require any manual configuration

**Business logic layer**: This business logic layer and this layer completes the

data processing of a series of activities. In order to ease of development symphony is used and it’s using ODBC + SQL Server JDBC:JTDS to connect database.

**Data layer**: This layer is only for storing data in database, having a database

Will provide user to query data.

For feature, using Bootstrap to develop the front-end interface will make the website interface more scalable. If you need to add new features, you only need to add new components, routing and change the component related to the added function.

**Framework: rest** or **restful api** design (representational state transfer) is designed to take advantage of existing protocols. while **rest** can be used over nearly any protocol, it usually takes advantage of http when used for web **apis**, in our project we decide to choose the rest api framework in other to retrieve data through http protocol, it’s done by getting the id of the logged user

## **The architecture of application server**

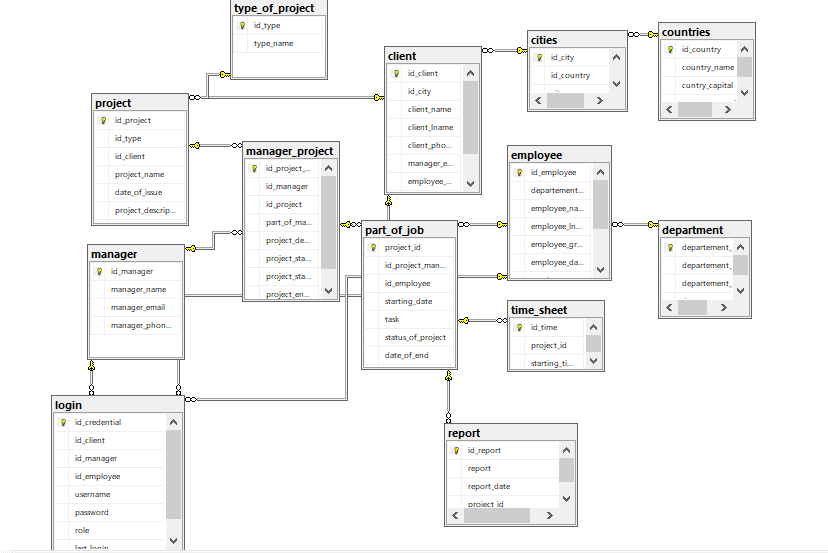
the architecture of applications built on the basis of the three levels



**Figure 3.3 – The architecture of application server**

This figure 3.3 describe the architecture of the application server which is based on the three levels

## **Development of database**



**Figure 3.4 Database Structure After Creation**

Figure 3.4 is a screenshot of the views of the tables and relationships in the

Database. There are fifteen tables in this database, they are type of project, countries, cities, time sheet, login, report, department, manager, project, part of job, client. Manager of project, employee,

## **Interaction between Client and Server**

 Frontend & Backend are two most popular terms used in web development.

These terms are very crucial for web development but are quite different from each other. Each side needs to communicate and operate effectively with the other as a single unit to improve the website’s functionality.

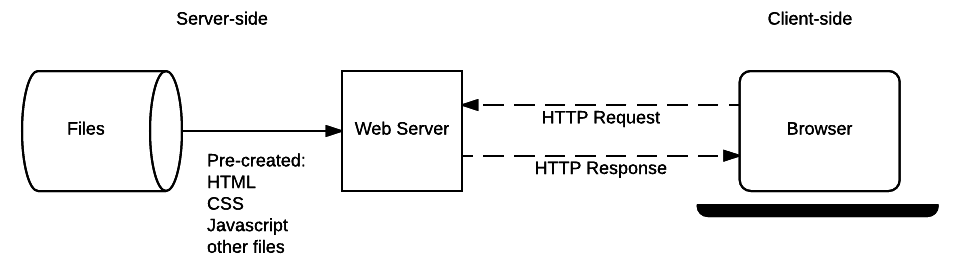
The core idea is that front–end HTML pages call back–end RESTFUL API

interface through PHP and interact with data source

Client Server: Apache which establish a connection between the **server** and

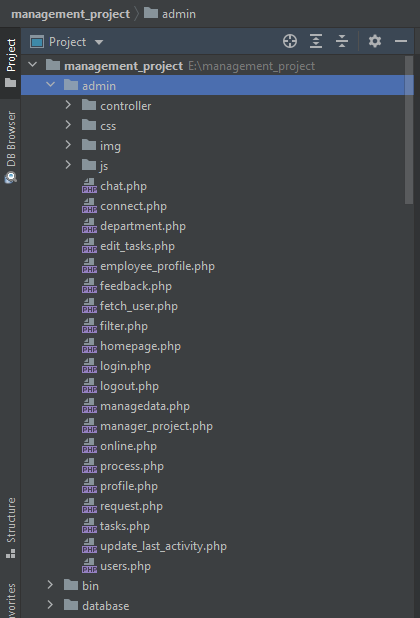
the browsers

Application server: tomcat



**Figure 3.4 – Client and Server diagram**

## **Module structure**

The web-application is structurally divided into several modules (packages), so let's consider each of them in more detail.

**Figure 3.6 Structure of the Web app**

Figure 3.6 describes the structure of the web application which is divided in

modules and every modules contain pages

## **Implementation of the project**

### **User module**

In the user module the aim is to implement a user interface in order for them

to use in a more convenient way the system the user system provides several functions such as login, registration, schedule appointment read other user’s feedback, personal information

#### **Listing: user login**

<?php

$servername = "DESKTOP-0G0R3RG\\SQLEXPRESS";

$connection= array('Database' => "employee","UID"=>"employee","PWD"=>"1111" );

$con=sqlsrv\_connect($servername,$connection);

session\_start();

if(isset($\_POST['Login']))

{

if(empty($\_POST['username']) || empty($\_POST['password']))

{

header("location:login.php?Empty= Please Fill in the Blanks");

}

else

{

$id=$\_POST['id'];

$name=$\_POST['username'];

$id=$\_POST['id'];

$pass=$\_POST['password'];

$sql="select \* from login where username='$name' and password='$pass'";

$params = array();

$options = array( "Scrollable" => SQLSRV\_CURSOR\_KEYSET );

$query = sqlsrv\_query( $con, $sql , $params, $options );

$num\_rows = sqlsrv\_num\_rows($query);

$row=sqlsrv\_fetch\_array($query);

if(isset($row['id\_credential']) and $row['id\_credential']>0 )

{

$\_SESSION['username']=$name;

if($row['role'] == "manager-a") {

header("location:homepage.php?id=" . $row['id\_credential']);

}

else if($row['role'] == "manager")

{

header("location:../manager/homepage.php?id=" . $row['id\_credential']);

}

else

{

header("location:../employee/myproject.php?id=" . $row['id\_credential']);

}

}

else

{

header("location:login.php?Invalid= Please Enter Correct User Name and Password ");

}

}

}

?>

#### **Listing: update last login time**

<?php

$servername = "DESKTOP-0G0R3RG\\SQLEXPRESS";

$connection= array('Database' => "employee","UID"=>"employee","PWD"=>"1111" );

$con=sqlsrv\_connect($servername,$connection);

$\_SESSION=$\_GET["id"];

$value=$\_SESSION;

$time=time()+10;

$sql="UPDATE login SET last\_login=$time where id\_credential='$value'";

$params = array();

$options = array( "Scrollable" => SQLSRV\_CURSOR\_KEYSET );

$query = sqlsrv\_query( $con, $sql , $params, $options );

$num\_rows = sqlsrv\_num\_rows($query);

?>

### **Administrator module**

The administrator module is a user interface which gives to administrators

possibility to work and to access personal information, edit his own profile, assign tasks, edit tasks, work with client, delete users, and manage the information that

are provided from the database

#### **Listing: Add project**

<?php

$servername = "DESKTOP-0G0R3RG\\SQLEXPRESS";

$connection= array('Database' => "employee","UID"=>"employee","PWD"=>"1111" );

$con=sqlsrv\_connect($servername,$connection);

if(isset($\_POST['save'])) {

$project\_name = $\_POST['project\_name'];

$task = $\_POST['task\_name'];

$employee\_name = $\_POST['employee\_name'];

$manager\_name = $\_POST['manager\_name'];

$status = $\_POST['status'];

$start\_date = $\_POST['start\_date'];

$sql = "INSERT INTO part\_of\_job (project\_id,id\_project\_manager, id\_employee,starting\_date, task,status\_of\_project)VALUES('12','$manager\_name','$employee\_name','$start\_date','$task','$status') ";

$stmt= (sqlsrv\_query($con,$sql));

if ( $stmt )

{

$something = "Submission successful.";

}

else

{

$something = "Submission unsuccessful.";

die( print\_r( sqlsrv\_errors(), true));

}

$output=$something;

/\* Free statement and connection resources. \*/

sqlsrv\_free\_stmt( $stmt);

sqlsrv\_close( $con);

header("Location: edit\_tasks.php");

}

?>

#### **Listing: Delete**

if(isset($\_GET['delete'])) {

$project\_id=$\_GET['delete'];

$sql1="DELETE project\_id,id\_manager,id\_employee,starting\_date,task,status\_of\_project,date\_of\_end FROM part\_of\_job where project\_id='$project\_id' ";

$stmt1=(sqlsrv\_query($con,$sql1));

if ( $stmt1 )

{

$something = "Submission successful.";

}

else

{

$something = "Submission unsuccessful.";

die( print\_r( sqlsrv\_errors(), true));

}

$output=$something;

/\* Free statement and connection resources. \*/

sqlsrv\_free\_stmt( $stmt1);

sqlsrv\_close( $con);

header("Location: edit\_tasks.php");

}

#### **Listing: Manager tasks information**

<?php

$servername = "DESKTOP-0G0R3RG\\SQLEXPRESS";

$connection= array('Database' => "employee","UID"=>"employee","PWD"=>"1111" );

$con=sqlsrv\_connect($servername,$connection);

$output='';

$rec=$\_POST['id'];

$sql1="select manager\_name,employee\_lname,employee\_name,task from part\_of\_job,manager,manager\_project,employee where part\_of\_job.project\_id=$rec and part\_of\_job.id\_employee=dbo.employee.id\_employee and

part\_of\_job.id\_project\_manager=dbo.manager\_project.id\_project\_manager and dbo.manager\_project.id\_project\_manager=manager.id\_manager

";

$params = array();

$options = array( "Scrollable" => SQLSRV\_CURSOR\_KEYSET );

$query = sqlsrv\_query( $con, $sql1 , $params, $options );

$num\_rows = sqlsrv\_num\_rows($query);

while($row=sqlsrv\_fetch\_array($query,SQLSRV\_FETCH\_ASSOC))

{

$output.="

<div class='tab-content' style='height: 120px; width: inherit;border-bottom: 1px solid black;' >

<div>

<select style='height: 32px;width:150px;float: right;margin-top: 15px;margin-right: -390px;outline: none;'>

<option>Done</option>

<option>Not Done</option>

<option>Progress</option>

</select>

<input type='submit' name='update' value='Update' class='tab-content' style='height: 32px;width:80px;float: right;margin-top: 15px;margin-right: 20px;outline: none;color:white; background-color: black;border-radius: 10%;'>

</div>

<span style='color: black;font-size:20px; font-family: Calibri Light;text-transform: uppercase'>Manager of project : ".$row['manager\_name']."</span><br>

<span style='font-size: 20px;color: black;font-family: Calibri Light;text-transform: uppercase'>Employee :</span><a href='profile.php?id=$rec' style='font-size: 20px;color: lightskyblue;font-family: Times New Roman;text-transform: uppercase'> ".$row['employee\_name']. ' '.$row['employee\_lname']."</a><br>

<span style='font-size: 20px;color: black;font-family: Calibri Light;text-transform: uppercase'>Task : ".$row['task']."</span>

</div>";

}

echo $output;

$id=$\_POST['id'];

$sql='select employee\_name,employee\_lname,substring([report],0,250) as report from part\_of\_job,employee,report where part\_of\_job.project\_id=report.project\_id

and dbo.employee.id\_employee=part\_of\_job.id\_employee and report.project\_id='.$id.'

';

$params1 = array();

$options1= array( "Scrollable" => SQLSRV\_CURSOR\_KEYSET );

$query1 = sqlsrv\_query( $con, $sql , $params1, $options1 );

$num\_rows1 = sqlsrv\_num\_rows($query1);

$d='';

while($row1=sqlsrv\_fetch\_array($query1,SQLSRV\_FETCH\_ASSOC)) {

$d="

<div class='' style = 'height:100px;width: 750px;text-transform: initial;border-radius: 50px;margin-top:5px;background-color:cadetblue;color:white;font-family:Calibri Light; border-radius:2%;float: left;margin-left: 2px;padding: 0;color: white;' >

<p style = 'font-size: 25px;margin-bottom: -4px;text-transform: capitalize' > " . $row1['employee\_name'] . ' ' . $row1['employee\_lname'] . " </p >

<a href = 'detail.php' style = 'color: white;' > " . $row1['report'] . "...etc click to read more </a >

</div >";

}

echo $d;

?>

#### **Listing: Admin list of managers**

<div id="menu1" class="container tab-pane fade"><br>

<h3>List of managers</h3>

<table class="table">

<thead>

<tr>

<th>Id no</th>

<th>Manager name</th>

<th>Manager email</th>

<th>Manager phone number</th>

</tr>

</thead>

<tbody>

<?php

$sql\_manager="select \* from manager";

$query = sqlsrv\_query( $con, $sql\_manager , $params, $options );

while ($emp =sqlsrv\_fetch\_array($query,SQLSRV\_FETCH\_ASSOC)) {

?>

<tr>

<td>

<?php echo $emp['id\_manager']?>

</td>

<td>

<?php echo $emp['manager\_name']?>

</td>

<td>

<?php echo $emp['manager\_email']?>

</td>

<td>

<?php echo $emp['manager\_phone\_number']?>

</td>

</tr>

</tbody>

<?php

}

sqlsrv\_free\_stmt($query);

?>

</table>

#### **Listing: Admin assign task**

<form method="GET">

<div class="row">

<div class="col-md-4">

<div class="profile-img">

<img src="https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcS52y5aInsxSm31CvHOFHWujqUx\_wWTS9iM6s7BAm21oEN\_RiGoog" alt=""/>

</div>

</div>

<div class="col-md-6">

<div class="profile-head">

<h5>

Manager <?php echo $emp['manager\_name']?>

</h5>

<h6>

<!-- Qualification -->

</h6>

<p class="proile-rating">RANKINGS : <span>8/10</span></p>

<ul class="nav nav-tabs" id="myTab" role="tablist">

<li class="nav-item">

<a class="nav-link active" id="home-tab" data-toggle="tab" href="#home" role="tab" aria-controls="home" aria-selected="true">About</a>

</li>

</ul>

</div>

</div>

<div class="col-md-2">

<input type="submit" class="profile-edit-btn" name="btnAddMore" value="Edit Profile"/>

</div>

</div>

<div class="col-md-8">

<div class="tab-content profile-tab" id="myTabContent">

<div class="tab-pane fade show active" id="home" role="tabpanel" aria-labelledby="home-tab">

<div class="row">

<div class="col-md-6">

<label>User Id</label>

</div>

<div class="col-md-6">

<p><?php echo $emp['id\_manager'];?></p>

</div>

</div>

<div class="row">

<div class="col-md-6">

<label>Name</label>

</div>

<div class="col-md-6">

<p><?php echo $emp['manager\_name'];?></p>

</div>

</div>

<div class="row">

<div class="col-md-6">

<label>Email</label>

</div>

<div class="col-md-6">

<p><?php echo $emp['manager\_email']?></p>

</div>

</div>

<div class="row">

<div class="col-md-6">

<label>Phone</label>

</div>

<div class="col-md-6">

<p><?php echo $emp['manager\_phone\_number']?></p>

</div>

</div>

<div class="row">

<div class="col-md-6">

<label>Profession</label>

</div>

</div>

</div>

</div>

</div>

</div>

<?php

}

?>

</form>

#### **Listing: Project description and duration**

<div class="pop" data-toggle="modal" data-target="#myModal10">

<div class="modal fade" id="myModal10" >

<div class="modal-dialog" >

<div class="modal-content" style="width: 750px; height: inherit;float: left">

<div class="tab-content">

<div id="client-list" class="container tab-pane active"><br>

<h3>Project duration</h3>

<table class="table">

<thead>

<tr>

<th>Project name</th>

<th>Start date</th>

<th>End date</th>

<th>Total days</th>

<th>Total Month</th>

</tr>

</thead>

<tbody>

<?php

$sql4="SELECT

project\_name,

date\_of\_issue,

date\_of\_end,

DATEDIFF(day,date\_of\_issue, date\_of\_end) As days,

DATEDIFF(MONTH,date\_of\_issue, date\_of\_end) As month

FROM project,part\_of\_job where part\_of\_job.project\_id=project.id\_project";

$params = array();

$options = array( "Scrollable" => SQLSRV\_CURSOR\_KEYSET );

$query4 = sqlsrv\_query( $con, $sql4 , $params, $options );

$num\_rows = sqlsrv\_num\_rows($query4);

while ($emp4 =sqlsrv\_fetch\_array($query4,SQLSRV\_FETCH\_ASSOC)) {

?>

<tr>

<td><?php echo $emp4['project\_name'];?></td>

<td><?php

$date=$emp4['date\_of\_issue'];

echo date\_format($date,"Y-m-d");

?>

</td>

<td><?php

$date=$emp4['date\_of\_end'];

echo date\_format($date,"Y-m-d");

?>

</td>

<td><?php echo $emp4['days'];?> days</td>

<td><?php echo $emp4['month'];?> months</td>

</tr>

</tbody>

<?php

}

sqlsrv\_free\_stmt($query4);

?>

</table>

</div>

</div>

#### **Listing: list of managers**

<table class="table">

<thead>

<tr>

<th>Manager ID</th>

<th>Manager name</th>

<th>Manager email</th>

<th>Manager phone number</th>

</tr>

</thead>

<tbody>

<?php

$sql5="select \* from manager";

$params = array();

$options = array( "Scrollable" => SQLSRV\_CURSOR\_KEYSET );

$query5 = sqlsrv\_query( $con, $sql5, $params, $options );

$num\_rows = sqlsrv\_num\_rows($query5);

while ($emp5 =sqlsrv\_fetch\_array($query5,SQLSRV\_FETCH\_ASSOC)) {

?>

<tr>

<td><?php echo $emp5['id\_manager'];?></td>

<td><?php echo $emp5['manager\_name'];?> days</td>

<td><?php echo $emp5['manager\_email'];?> months</td>

<td><?php echo $emp5['manager\_phone\_number'];?> months</td>

</tr>

</tbody>

<?php

}

sqlsrv\_free\_stmt($query5);

?>

</table>

### **Manager module**

The manager module is a user interface which provides functionalities to

manager to work with clerk have a regular follow-up, read daily clerks report as each manager as one or more project to manage

#### **Listing: clerks tasks**

<form action="process.php" method="GET">

<h2 style="text-align: center">List of tasks</h2>

<table class="table">

<thead>

<tr>

<th>Project name</th>

<th>Project manager</th>

<th>Employee name</th>

<th>Task</th>

<th>Date of Begin</th>

<th>Status</th>

<th>Date of End</th>

<th>Edit</th>

<th>Delete</th>

</tr>

</thead>

<tbody>

<?php

while ($emp =sqlsrv\_fetch\_array($query,SQLSRV\_FETCH\_ASSOC)) {

?>

<tr>

<td><?php echo $emp['project\_id']?></td>

<td><?php echo $emp['id\_project\_manager']?></td>

<td><?php echo $emp['id\_employee']?></td>

<td><?php echo $emp['task']?></td>

<td><?php

$date=$emp['starting\_date'];

echo date\_format($date,"Y-m-d");?></td>

<td><?php echo $emp['status\_of\_project']?></td>

<td><?php

$date=$emp['date\_of\_end'];

echo date\_format($date,"Y-m-d");

?></td>

<td>

<a href="#edit-data" class="edit" data-toggle="modal" ><i class="fa fa-pen"></i><span>Edit</span></a>

</td>

<td><a href='process.php?delete=<?php echo $emp['project\_id'];?>' name="delete" class="delete"><i class="fa fa-trash"></i><span>Delete</span></a></td>

</tr>

</tbody>

<?php

}

sqlsrv\_free\_stmt($query);

?>

</table>

</form>

### **Personnel’s module**

#### **Listing: send report to manager**

<?php

$servername = "DESKTOP-0G0R3RG\\SQLEXPRESS";

$connection= array('Database' => "employee","UID"=>"employee","PWD"=>"1111" );

$con=sqlsrv\_connect($servername,$connection);

if(isset($\_POST['save'])) {

$id=$\_GET['id'];

$v=$id;

echo $v;

$task = $\_POST['task\_name'];

$project\_id=$\_POST['project\_id'];

$start\_date =date('Y-m-d');

$sql = "INSERT INTO report(report,report\_date,project\_id)VALUES('$task','$start\_date','$project\_id') ";

$stmt= (sqlsrv\_query($con,$sql));

if ( $stmt )

{

$something = "Submission successful.";

}

else

{

$something = "Submission unsuccessful.";

die( print\_r( sqlsrv\_errors(), true));

}

$output=$something;

/\* Free statement and connection resources. \*/

sqlsrv\_free\_stmt( $stmt);

sqlsrv\_close( $con);

header('location:myproject.php?id='.$v.'');

}

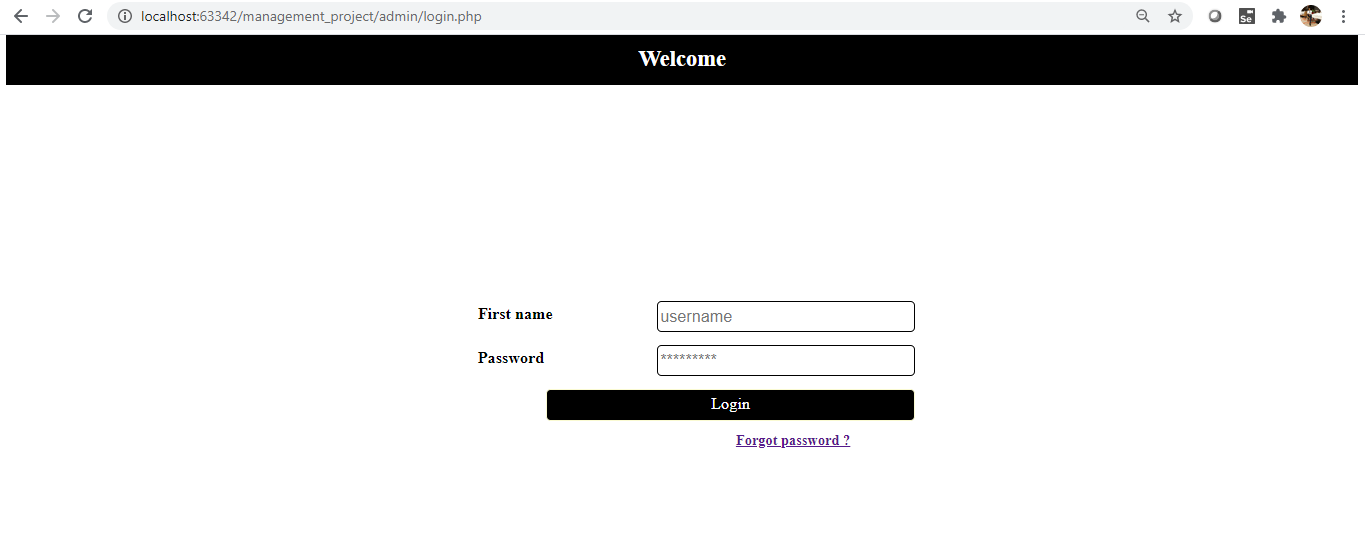
?>

## 

## Interface

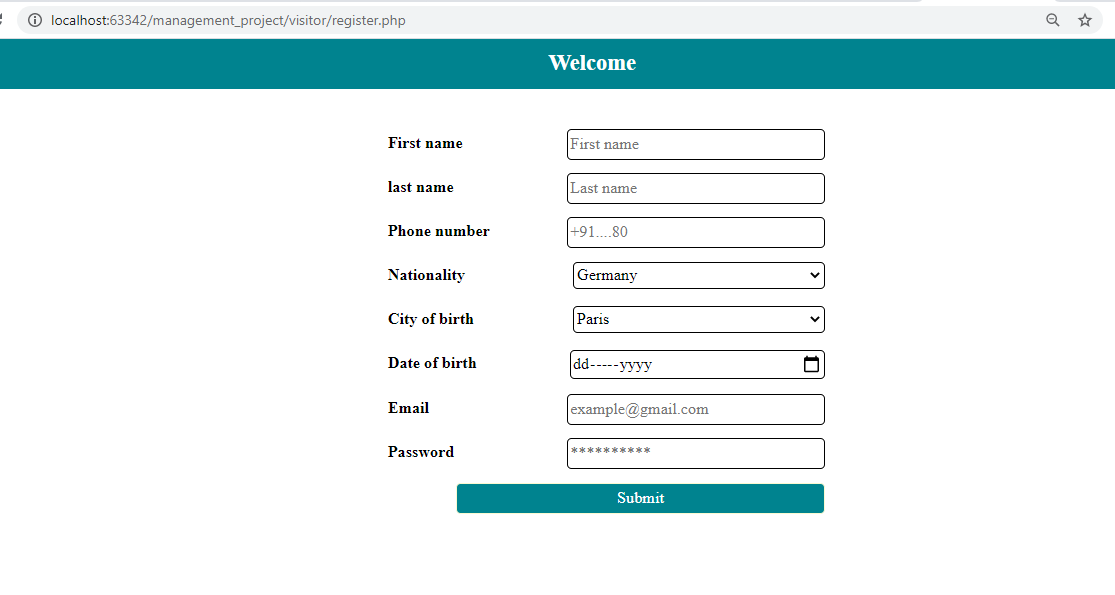
Login interface. Users enter their own account password for login. If there is

no registration, you can click the registration button in the lower right corner to jump to the registration interface. If the user forgets the password, click the forget the password button below to retrieve the password

**Figure 3.7 – Login interface**

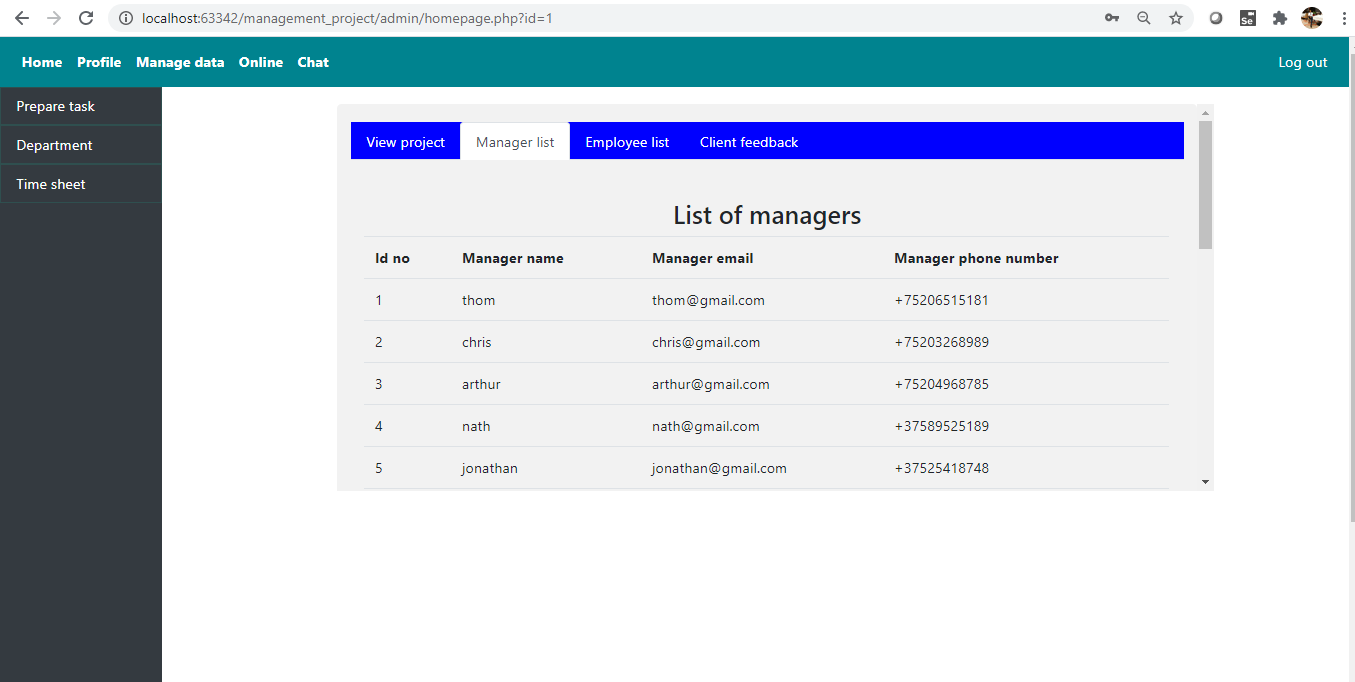
As we can see on the figure 3.7 above it/s mainly describe the login interface

for the register users for those who are not registered they will have to click on the registration button and they will be redirected to the registration page and they will have to fill out the form and provide some information that are required

**Figure 3.8 – Registration interface**

The figure 3.8 above is the registration page on this registration page non-

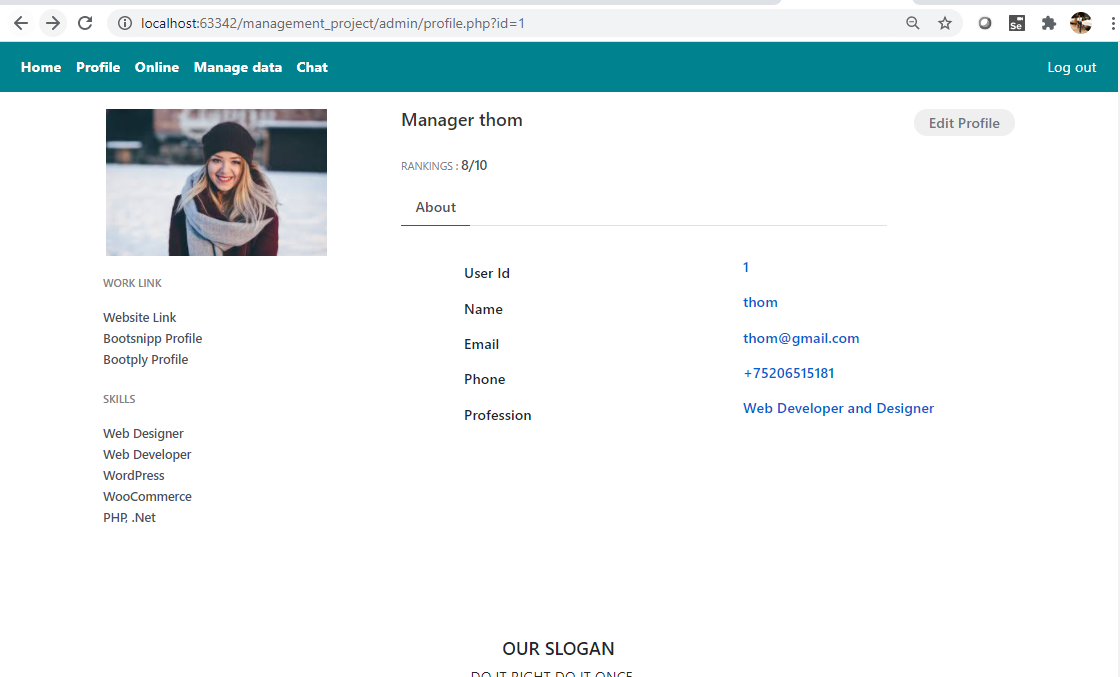
registered users will have to fill out the form and provide their information such as first name, last name, phone number, nationality, date of birth, email and create a personal and unique password, and after that the new registered user will be redirected on the registered home page



**Figure 3.10 – Administrator interface**

The figure 3.10 above is the administrator main interface “home page” once

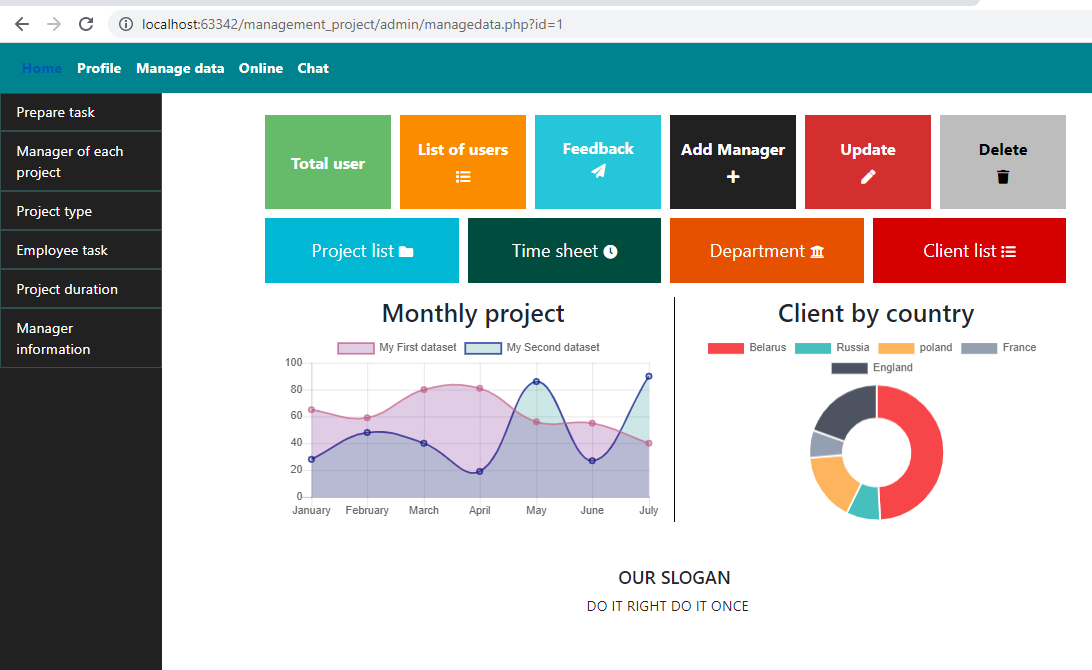
The administrator has entered his own credential information on the login page the system will redirect to the homepage, on this page there are several functionalities provided by the system such as projects list, managers list, personnels list, client feedback the administrator can click on any tabs and the information will be displayed, this page is more a single page application, as example on department click a modal will be displayed containing information of each department manager of the department, email of the department etc…, on the navigation bar we have profile, manage data, online manager, message ,log out



**Figure 3.11 – Profile interface**

The above figure 3.11 describes the profile interface “profile page” Contain

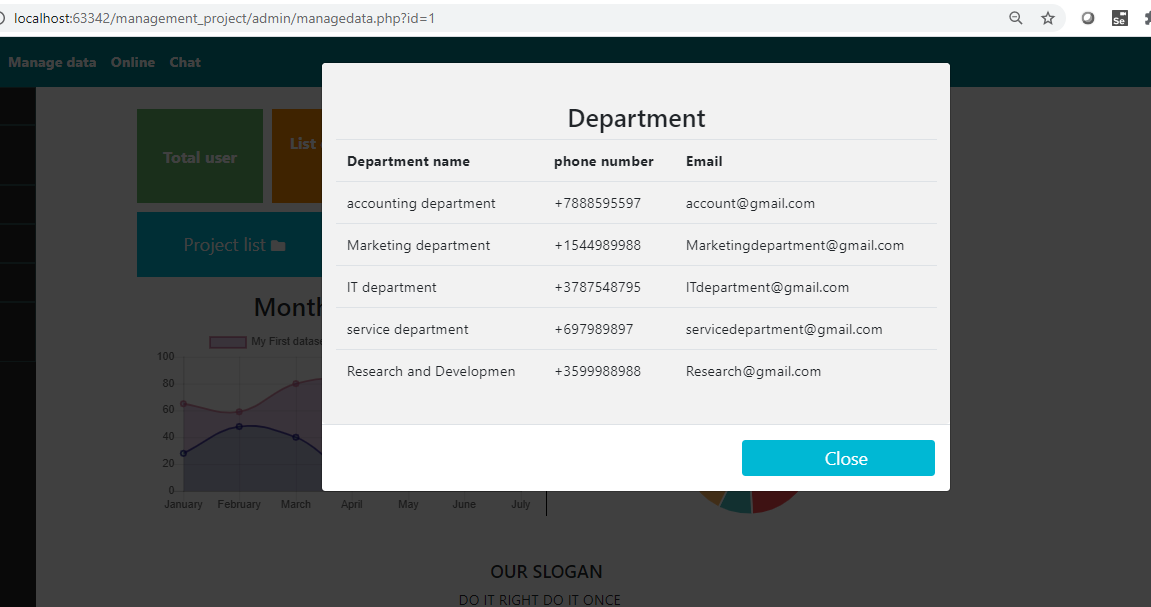
personal information and the administrator can change is mail, phone number, picture, to access this page the administrator has to click on the link named profile on the navigation bar at the top of the page it’s



**Figure 3.12 – Manage data interface**

The figure 3.12 above describe the manage data information on this interface

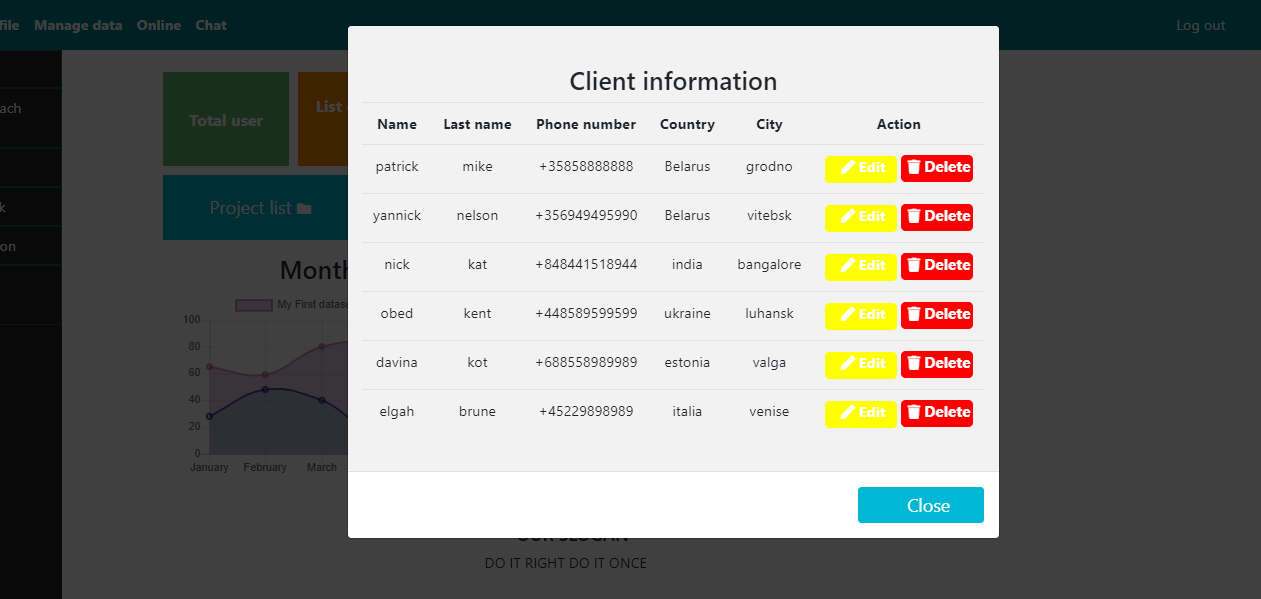
the administrator works with data, such as assign task, add new user, delete user, delete project, update information, this page is a one single page application on click on any button a modal will be displayed, on this page most of the work is done, it’s also gives a statical view of monthly project and clients by country



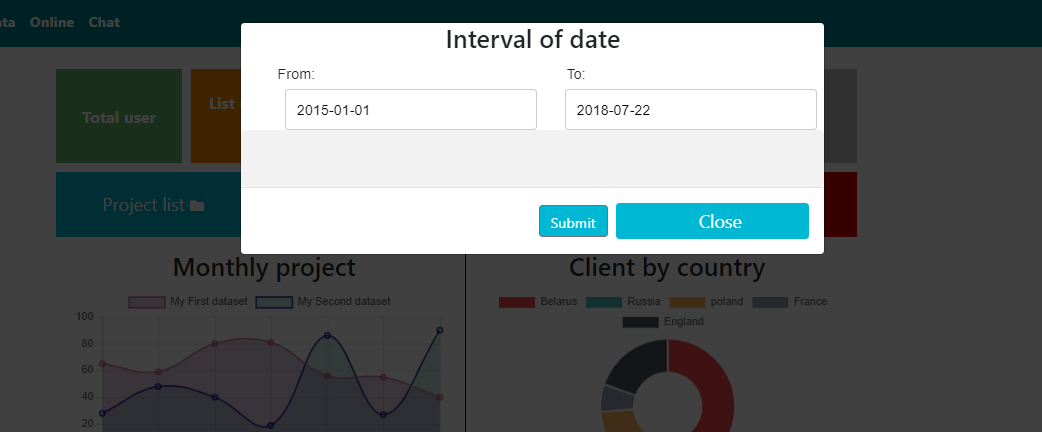
**Figure 3.13 – Department interface**

The figure 3.13 above display the department interface information such as

department name, phone number, mail. The close button is to close the modal page to access the department information we have to click on the button department



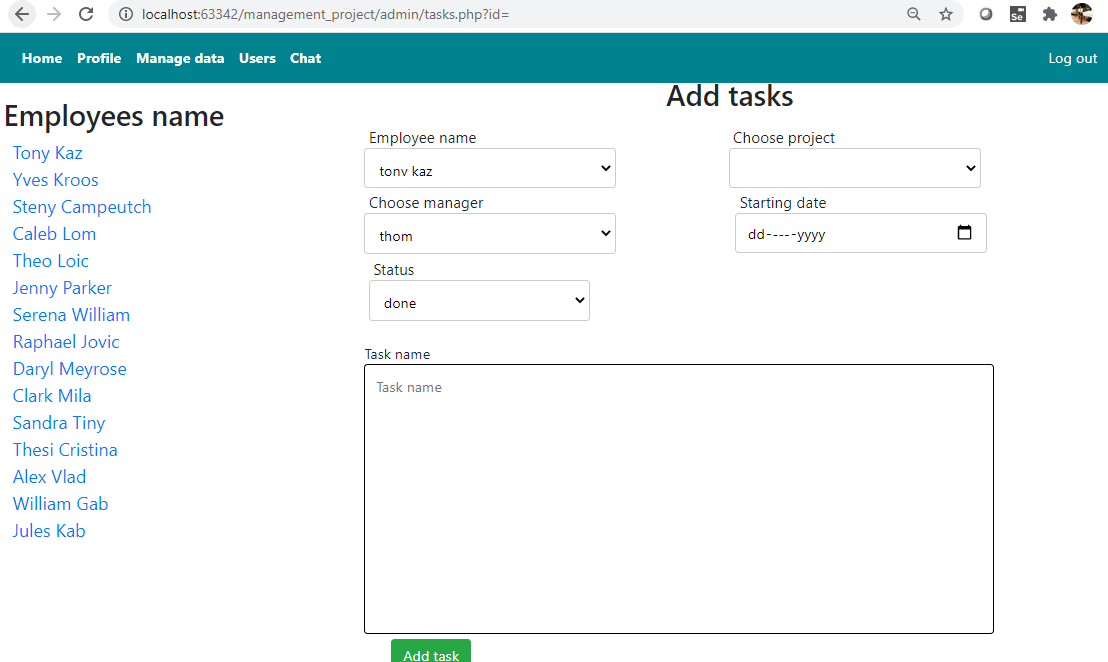
**Figure 3.14 – client information interface**



**Figure 3.15– Search interface**

The figure 3.15 above describes the search interface administrators search

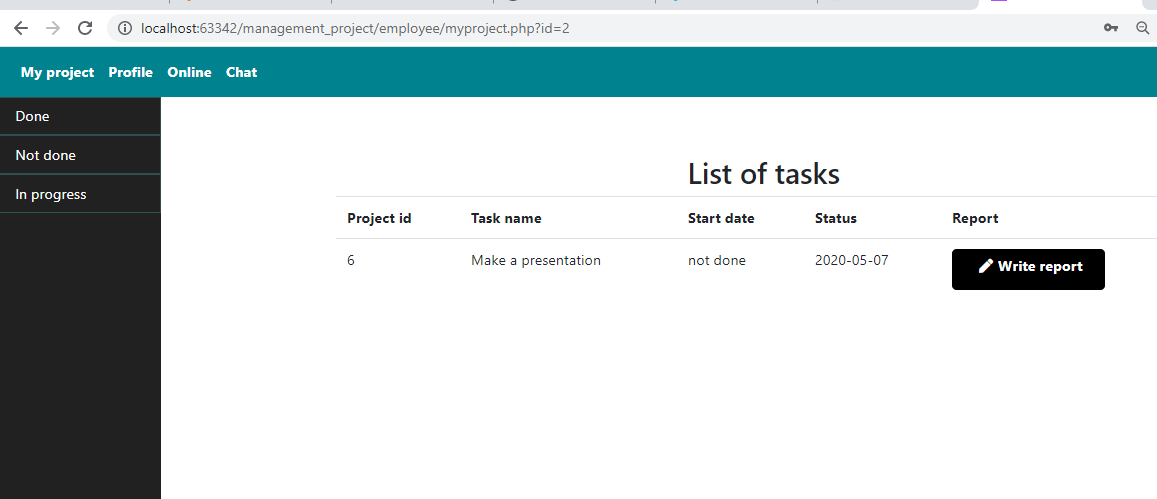
project on the interval of two dates



**Figure 3.16– Assign task interface**

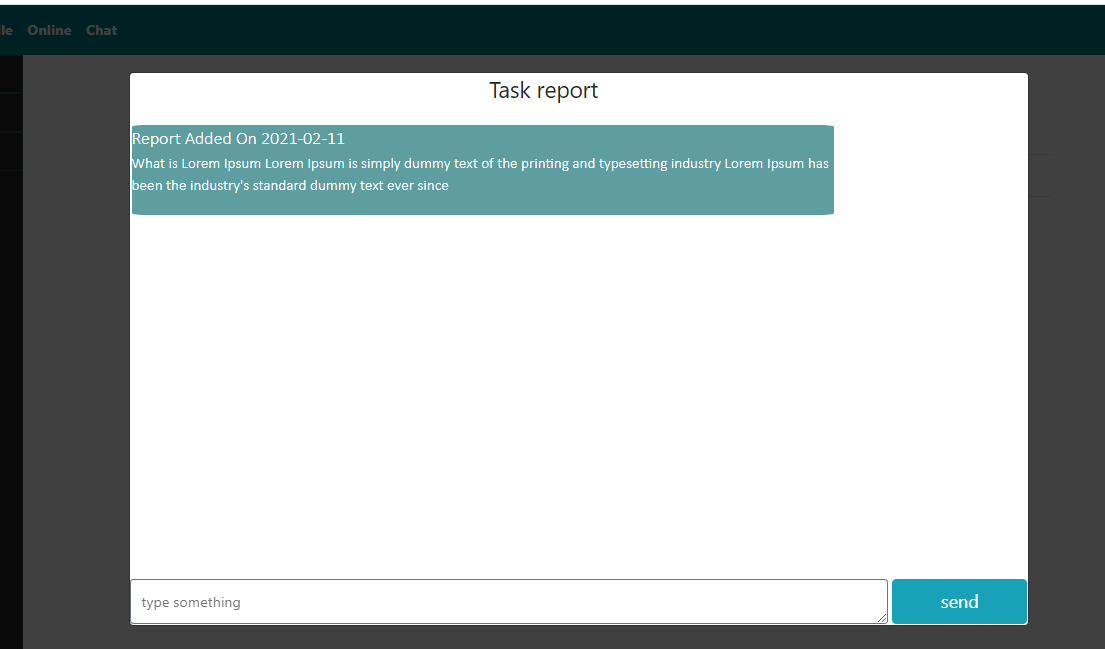
The figure 3.16 above describe the assign task interface the administrator will

have to select the project name on click on the select button where projects are listed, as well as to choose manager who will be in charge of the project and the personnel, on the left side there is a list of clerks the manager can click on any name to have more information of the clerk, on click a modal page will be displayed containing information such as personnel’s background, how many projects the clerks is on, which project did he work on etc…



**Figure 3.17–Personnel Main interface**

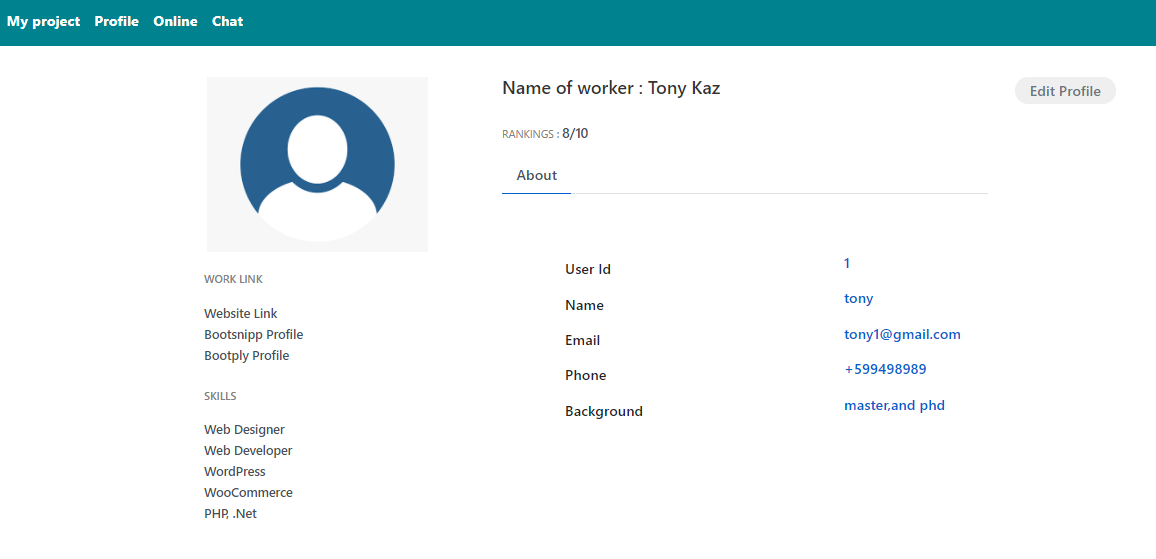
The figure 3.17 above describe the personnel home page interface on this page the clerk has the possibility to write report on the current project that’s working on for that he has to click on the black button “write report”



**Figure 3.18– Report interface**

The figure 3.18 above describe the report interface clerk has write a daily

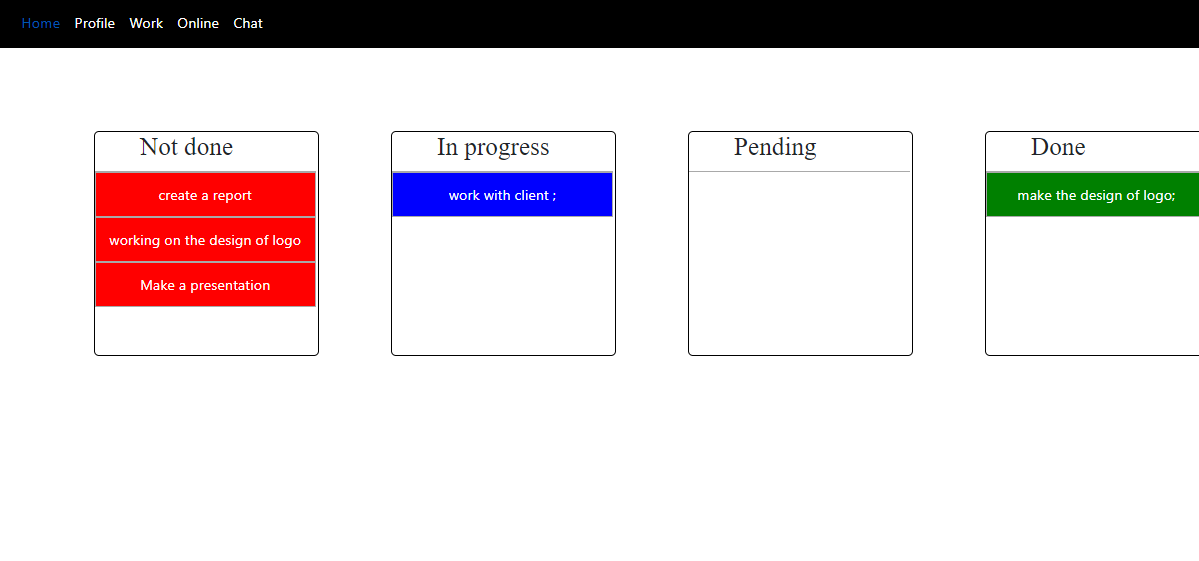
report every working day if he has one or more task not done and after writing the report the clerk has to click the send, and the added report will be displayed on the page



**Figure 3.19– profile interface**

The figure 3.19 above describe the clerk profile interface the clerk can change

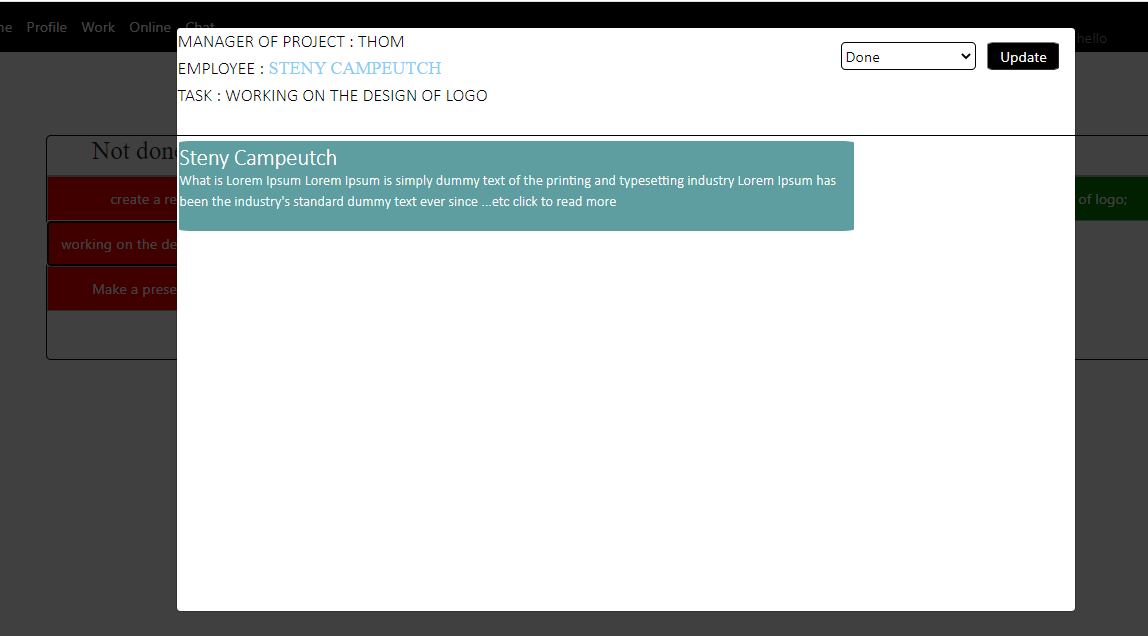
his phone number, email and profile picture



**Figure 3.20– Manager work space interface**

The figure 3.20 above describe the manager work space interface on this

interface the manager works with personnel,s the projects are classified according to their status those that are not done are display on red color, those that are in progress are displayed on blue color, those that are in pending process will be displayed on yellow color and those that are done will be displayed on green color, and each block is a button the manager can click and read the daily report and change the status if the task is completed as it should be done

**Figure 3.21– Manager work interface**

The figure 3.21 above describe the manager interface work space to access

this page the manger has to click on any project that he wants to have information, and on this figure, we can see that the manager has a clear information of the project and he can update the status of the project depending on the evolution of the project

## **Conclusion**

In this chapiter we have described the physical architecture of the system as

well as the framework used on this project, we mainly talked about the implementation of the project and for our we’ve used as main language php and its features, symphony, rest api architecture, to achieve the overall of the system

# SUMMARY

In this document, the development process of the private company management

system is introduced, from the development background of the system to the analysis of the system requirements, to the detailed design of the system, and finally to the implementation and testing. This is an indispensable process sequence for project development, the graduation work of private company & management system aimed to developed a software that meets the requirements

* The following tasks have been solved:
* Find and analyze existing solutions.
* The main functions and capabilities of software products are highlighted.
* The choice of software for system implementation is reasonable.
* Design web application architecture and database.
* Choose the software and technology used to achieve the purpose and thesis.

# 

# LIST OF SOURCES

[1] Patterns of Enterprise Application Architecture is a programming book written by Martin Fowler.

[2] PHP and MySQL Web Development Book & CD Edition by [Luke Welling](https://www.amazon.com/s/ref=dp_byline_sr_book_1?ie=UTF8&field-author=Luke+Welling&text=Luke+Welling&sort=relevancerank&search-alias=books) (Author), [Laura Thomson](https://www.amazon.com/Laura-Thomson/e/B001IGJV0A/ref=dp_byline_cont_book_2)

[3] The Art of Software Modeling Book by Benjamin A. Lieberman December 26, 2006

[4] [**JavaScript & JQuery: Interactive Front-End Web Development**](https://geni.us/BSWCCoZ) Author**–** Jon Duckett, latest edition

[5] MySQL Introduction Book by Antun Peicevic, March 2016

[6] REST API Design Rulebook Book by Mark Masse, 2011

[7] HTML & CSS: Design and Build Web Sites Book by Jon Duckett

[8] Node.js Web Development Book by David Herron