MINISTRY OF EDUCATION AND SCIENCE OF THE RUSSIAN FEDERATION

Federal State Autonomous Educational Institution of Higher Education
South Ural State University (National Research University)
School of Electrical Engineering and Computer Science
Department of System Programming

THESIS IS CHECKED	ACCEPTED FOR THE DEFENSE
Reviewer, Head of Bokareva Enterprise	Head of the department, Dr. Sci., Prof.
___________ S.A. Bokareva	__________ L.B. Sokolinsky
“___”___________ 2018	“___”___________ 2018

DEVELOPMENT OF PERSONNEL MANAGEMENT SYSTEM

GRADUATE QUALIFICATION WORK
SUSU–02.04.02.2018.308-590.GQW

Supervisor

Cand. Sci., Assoc. Prof.

__________ A.T. Latipova

Author,
the student of the group CE-219

__________ Z.A. Jaffar

Normative control

___________ O.N. Ivanova

“___”___________ 2018

Chelyabinsk–2018

TABLE OF CONTENTS

ACKNOWLEDGEMENT

Apart from the efforts of me, the success of any project depends largely on the encouragement and guidelines of many others. I take this opportunity to express my gratitude to the people who have been instrumental in the successful completion of this project.

I would like to show my greatest appreciation to Assistant Professor. Alina T. Latipova. I can’t say thank you enough for her tremendous support and help.

I feel motivated and encouraged every time I attend her meeting.

Without her encouragement and guidance this project would not have materialized.

The guidance and support received from all the members who contributed and who are contributing to this project, was vital for the success of the project. I am grateful for their constant support and help.

INTRODUCTION

The goal of personnel management (PM) information systems is to store, maintain and analyze necessary data related to organization’s human resources. Such systems may boost effectiveness of providing information about employees, organizational policies and procedures, etc. Managers can use a system of this kind to track staff development.

There are many companies, governmental and non-governmental organizations. Some companies have a lot of different employees with a large number of departments which are in charge of specific tasks. For example, there can be the department of accounting which is in charge of salaries of the employees in the company. When employees are working in a company with a large number of staff, they may have difficulties to know personal information about themselves (e.g. information about their position, overtime, sick leaves, etc.). That is why a PM system should provide access for employees to their personal information [1].

Most organizations use corporate information systems (CIS) for accounting and tax administration [3]. CIS usually contain excessive data for personnel management and direct connection to CIS database may be dangerous from aspect of information security. Thus, a PM system should be a separate system which has an exporting and importing mechanism for data exchange with CIS.

Lastly, management of personnel affairs can be a new thing to the organization which can cause big changes [20]. Management of personnel affairs can help the employer to manage human resources effectively. It also can give advantages to the organization in many perspectives.

Statement of the problem.

Use of paper transactions in personnel management leads to consumption time and effort of employees and their managers, so using this program will be a great improvement to the organization [21]. Like any information technology application, the program facilitates and speeds up personnel management and employees can view only own personal data any time without waste time of work.

Goal and objectives of the research

The goal of the research is creating an application to manage employee's information in the organization at the given level of access.

General objectives:

1) analysis of management of personnel affairs;
2) development of the use case diagram;
3) development of database structure;
4) choosing tools for software implementation;
5) design and implementation web-site to show information about employees;
6) functional testing of the developed web-site.

The List of Functions

The system features

The system can show all information about employees:

1) personal data of employee;
2) salaries of employees;
3) time and sick leave.

The system has an exporting-importing module.

The administrator features:

1) the administrator can add an account for the employee;
2) the administrator can delete the employee account;
3) the administrator can update data;
4) the administrator can import to or export from CIS.

The employee feature:

1) the employee can only view own personal data;
2) the employee cannot view data for another employee because all employees has password to access only to his\her own account.

1. DEVELOPMENT TOOLS

1.1. My SQL

MySQL is the most popular and widely used DBMS on the Internet (database management system). MySQL is not very good to work with a large flow of information (like in CIS), but it is ideal for working with small and large Internet sites [3,9].

The main difference of MySQL is fast work speed, reliability, elasticity. It is quite simple in operation and does not create great difficulties in its creation. MySQL server support is automatically included in the PHP package.

It is worth noting the fact that it is free. MySQL is distributed under the terms of the GNU General License (GPL, GNU Public License).

The task of MySQL is the long-term storage of information, which is often used in Web application programming: visitor counting, storing forum messages, remote management of information on the site, access to a private office and much more. Previously, long-term storage of information was carried out in files: they contained a number of lines, and then extracted them to continue working.

It should be noted that this way of working with files is very time consuming to use, in which there is a number of additional loads on the person: it is the need to place the necessary information in the files, such as: sorting and extracting, and at the same time not forgetting that all actions will take place on server host provider, where 99% is one of the variants of Unix - therefore, you should also not forget about the rights of access to files and their location. With this method of work, the amount of code increases, which entails a high probability of making a mistake and will lead to a malfunction of the program as a whole [11,18].

With the advent of MySQL, these problems were resolved. The creation and use of databases has simplified the work process and reduced the risk of errors to a minimum. Thanks to the developed MySQL algorithm, databases themselves ensure the security of the information stored in it and its sorting, which allows you to extract and post information using a single line. The command code for using the database is more compact, which simplified the tuning process and laboriousness.

Also worth emphasizing is the important fact of MySQL's work - it's speed. Extracting information from the database is much faster than retrieving information from files. Other advantages are reliability and ease of use MySQL.

1.2. PHP

PHP is a server-side scripting language designed for web development but also used as a general-purpose programming language. Originally created by Rasmus Lerdorf in 1994, the PHP reference implementation is now produced by The PHP Group. PHP originally stood for Personal Home Page, but it now stands for the recursive acronym PHP: Hypertext Preprocessor.

PHP was designed by Rasmus Lerdorf to display his resume online and to collect data from his visitors [6,12].

PHP code may be embedded into HTML code, or it can be used in combination with various web template systems, web content management systems, and web frameworks. PHP code is usually processed by a PHP interpreter implemented as a module in the web server or as a Common Gateway Interface (CGI) executable. The web server combines the results of the interpreted and executed PHP code, which may be any type of data, including images, with the generated web page. PHP code may also be executed with a command-line interface (CLI) and can be used to implement standalone graphical applications.

The standard PHP interpreter, powered by the Zend Engine, is free software released under the PHP License. PHP has been widely ported and can be deployed on most web servers on almost every operating system and platform, free of charge.

The PHP language evolved without a written formal specification or standard until 2014, leaving the canonical PHP interpreter as a de facto standard. Since 2014 work has gone on to create a formal PHP specification.

During the 2010s there have been increased efforts towards standardization and code sharing in PHP applications by projects such as PHP-FIG in the form of PSR-initiatives as well as Composer dependency manager and the Packages repository. PHP hosts a diverse array of web frameworks requiring framework-specific knowledge, with Laravel recently emerging as a popular option by incorporating ideas made popular from other competing non-PHP web frameworks, like Ruby on Rails [15].

Basically, PHP allows a static webpage to become dynamic. "PHP" is an acronym that stands for "PHP: Hypertext Preprocessor". The word "Preprocessor" means that PHP makes changes before the HTML page is created. This enables developers to create powerful applications that can publish a blog, remotely control hardware, or run a powerful website such as Wikipedia or Wikibooks. Of course, to accomplish something such as this, you need a database application such as My SQL.

PHP Admin is a free software tool written in PHP, intended to handle the administration of My SQL over the web. It supports a wide range of operations on.

My SQL and other DBMS. Frequently used operations (managing databases, tables, columns, relations, indexes, users, permissions, etc.) can be performed via the user interface, while the developer will have the ability to directly execute any SQL statement.

1.3. HTML

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript, it forms a triad of cornerstone technologies for the World Wide Web [17].

Web browsers receive HTML documents from a web server or from local storage and render them into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects, such as interactive forms, may be embedded into the rendered page. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by tags, written using angle brackets. Tags such as <img /> and <input /> introduce content into the page directly. Others such as <p>...</p> surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page.

HTML can embed programs written in a scripting language such as Java Script which affect the behavior and content of web pages. Inclusion of CSS defines the look and layout of content. The World Wide Web Consortium (W3C), maintainer of both the HTML and the CSS standards.

A markup language that is used to create documents on the world wide Web incorporating text, graphics, sound, video, and hyperlinks. HTML is used to create electronic documents (called pages) that are displayed on the World Wide Web. Each page contains a series of connections to other pages called hyperlinks. Every web page you see on the Internet is written using one version of HTML code or another. HTML code ensures the proper formatting of text and images so that your Internet browser may display them as they are intended to look. Without HTML, a browser would not know how to display text as elements or load images or other elements. HTML also provides a basic structure of the page, upon which Cascading Style Sheets are overlaid to change its appearance. Hypertext Markup Language (HTML) is the primary building block of creating a website. HTML is a very basic markup language and requires memorization of a few dozen HTML commands that structure the look and layout of a web page. Before writing any HTML code or designing your first web page, you must decide on an HTML editor or text editor, such as Notepad or WordPad. Once you have obtained an HTML editor and are ready to begin setting up your website.

1.4. CSS

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language [16].

Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document, including plain XML, SVG and XUL, and is applicable to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging webpages, user interfaces for web applications, and user interfaces for many mobile applications.

CSS is designed primarily to enable the separation of presentation and content, including aspects such as the layout, colors, and fonts.

This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple HTML pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

Separation of formatting and content makes it possible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or screen reader), and on Braille-based tactile devices. It can also display the web page differently depending on the screen size or viewing device. Readers can also specify a different style sheet, such as a CSS file stored on their own computer, to override the one the author specified.

Changes to the graphic design of a document (or hundreds of documents) can be applied quickly and easily, by editing a few lines in the CSS file they use, rather than by changing markup in the documents [16].

The CSS specification describes a priority scheme to determine which style rules apply if more than one rule matches against a particular element. In this so-called cascade, priorities (or weights) are calculated and assigned to rules, so that the results are predictable.

The CSS specifications are maintained by the World Wide Web Consortium (W3C). Internet media type (MIME type) text/css is registered for use with CSS by RFC 2318 (March 1998). The W3C operates a free CSS validation service for CSS documents [16].

1.5. Java Script

Java Script often abbreviated as JS, is a high-level, interpreted programming language. It is a language which is also characterized as dynamic, weakly typed, prototype-based and multi-paradigm [10].

Alongside HTML and CSS, Java Script is one of the three core technologies of the World Wide Web. Java Script enables interactive web pages and thus is an essential part of web applications. The vast majority of websites use it, and all major web browsers have a dedicated JavaScript engine to execute it.

As a multi-paradigm language, JavaScript supports event-driven, functional, and imperative (including object-oriented and prototype-based) programming styles. It has an API for working with text, arrays, dates, regular expressions, and basic manipulation of the DOM (Document Object Model), but the language itself does not include any I/O, such as networking, storage, or graphics facilities, relying for these upon the host environment in which it is embedded.

Initially only implemented client-side in web browsers, Java Script engines are now embedded in many other types of host software, including server-side in web servers and databases, and in non-web programs such as word processors and PDF software, and in runtime environments that make Java Script available for writing mobile and desktop applications, including desktop widgets.

Although there are strong outward similarities between Java Script and Java, including language name, syntax, and respective standard libraries, the two anguages are distinct and differ greatly in design; Java Script was influenced by programming languages such as self and schema.

2. DESING SOFTWARE

2.1. 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. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well.

A use case describes a sequence of actions that provide something of measurable value to an actor and is drawn as a horizontal ellipse [4].

An actor is a person, organization, or external system that plays a role in one or more interactions with your system. Actors are drawn as stick figures.

Associations between actors and use cases are indicated in use case diagrams by solid lines figure 1 shows the developed use case diagram.

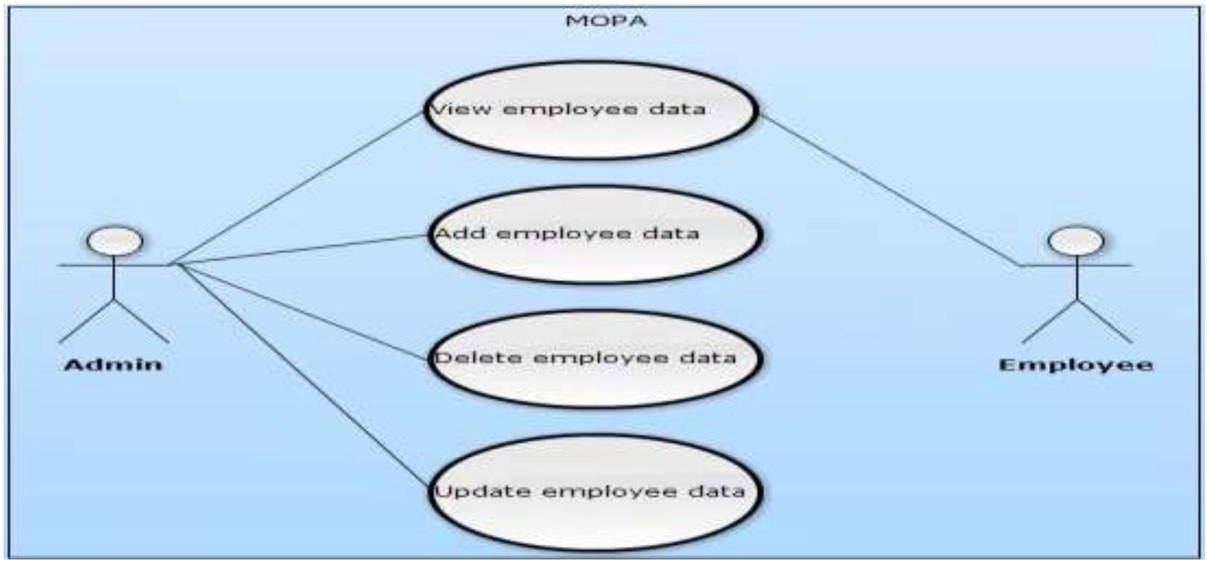


Fig. 1. Use Case Diagram

Use case diagram in my system (Management of Personnel Affairs).

It is consists of two actors.

The first one is Administrator. Iit is connect with four cases:

1) view employee data;
2) add employee data;
3) delete employee data;

4) update employee data.

The Second one is Employee. It is connect with one case:

1) view employee data.

2.2. User Interface of the System

The user interface (UI), in the industrial design field of human–computer interaction, is the space where interactions between humans and machines occur. The goal of this interaction is to allow effective operation and control of the machine from the human end, whilst the machine simultaneously feeds back information that aids the operators' decision-making process.

User interface design (UI) or user interface engineering is the design of user interfaces for machines and software, such as computers, home appliances, mobile devices, and other electronic devices, with the focus on maximizing usability and the user experience. The goal of user interface design is to make the user's interaction as simple and efficient as possible, in terms of accomplishing user goals (user-centered design) [5].

Generally, the goal of user interface design is to produce a user interface which makes it easy (self-explanatory), efficient, and enjoyable (user-friendly) to operate a machine in the way which produces the desired result. This generally means that the operator needs to provide minimal input to achieve the desired output, and also that the machine minimizes undesired outputs to the human figure 2 shows the user interface in my project.

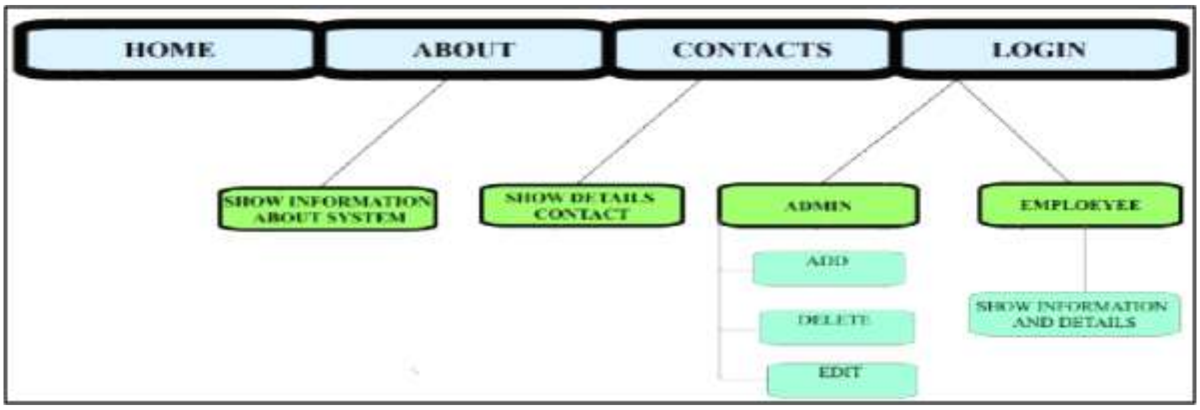


Fig. 2. User Interface

2.3. Description of Database

A database is an organized collection of data. A relational database, more restrictively, is a collection of schemas, tables, queries, reports, views, and other elements. Database designers typically organize the data to model aspects of reality in a way that supports processes requiring information.

A database-management system (DBMS) is a computer-software application that interacts with end-users, other applications, and the database itself to capture and analyze data. A general-purpose DBMS allows the definition, creation, querying, update, and administration of databases [3].

A database is not generally portable across different DBMSs, but different DBMSs can interoperate by using standards such as SQL and ODBC or JDBC to allow a single application to work with more than one DBMS.

Databases today are essential to every business. In essence, a database is a collection of information that exists over a long period of time, often many years. In common parlance, the term database refers to a collection of data that is managed by a DBMS.

The DBMS is expected to allow users to create new databases and specify their schemas (logical structure of the data), give users the ability to query and modify the data.

A DBMS is a powerful tool for creating and managing large amounts of data efficiently and allowing it to persist over long periods of time, safely. These systems are among the most complex types of software available [11].

2.4. Database Scheme

The database schema of a database system is its structure described in a formal language supported by the database management system (DBMS). The term "schema" refers to the organization of data as a blueprint of how the database is constructed (divided into database tables in the case of relational databases). The formal definition of a database schema is a set of formulas (sentences) called integrity constraints imposed on a database. These integrity constraints ensure compatibility between parts of the schema. All constraints are expressible in the same language. The states of a created conceptual schema are transformed into an explicit mapping, the database schema. This describes how real-world entities are modeled in the database.

A database schema specifies, based on the database administrator's knowledge of possible applications, the facts that can enter the database, or those of interest to the possible end-users. The notion of a database schema plays the same role as the notion of theory in predicate calculus. A model of this "theory" closely corresponds to a database, which can be seen at any instant of time as a mathematical object. Thus a schema can contain formulas representing integrity constraints specifically for an application and the constraints specifically for a type of database, all expressed in the same database language. In a relational database, the schema defines the tables, fields, relationships, views, indexes, packages, procedures, functions, queues, triggers, types, sequences, materialized views, synonyms, database links, directories, XML schemas, and other elements [3].

The scheme of the database in my project. It consists of 6 tables, described below figure 3 shows the database scheme in my project.

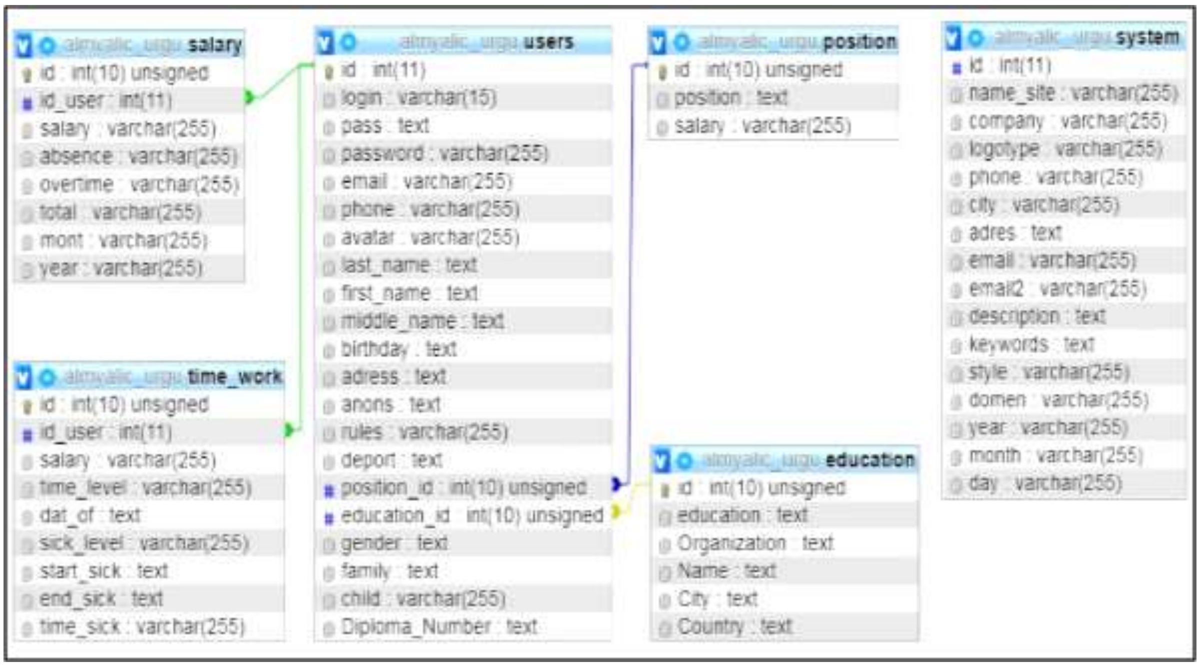
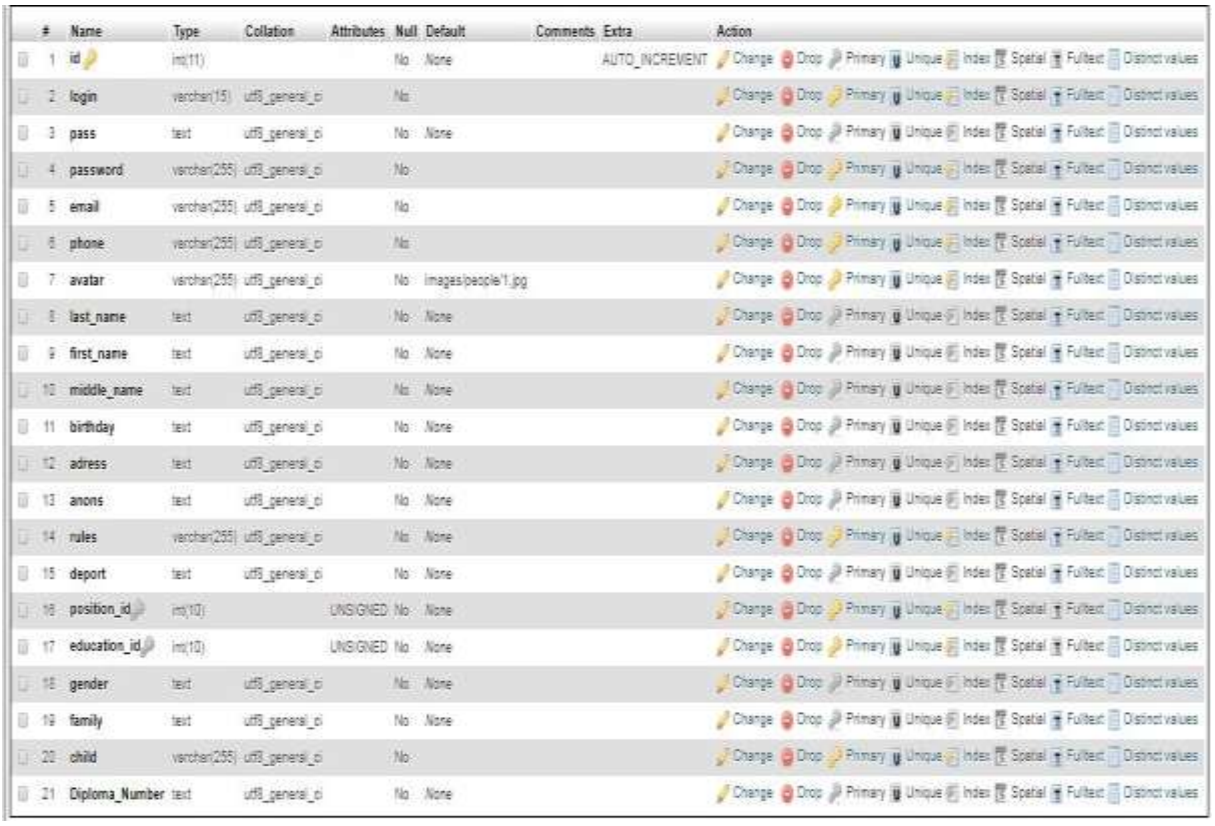


Fig. 3. Database Scheme

1. The table “Users”

The table “Users” contains information about employees. It consists of 21 fields as in the figure 4.

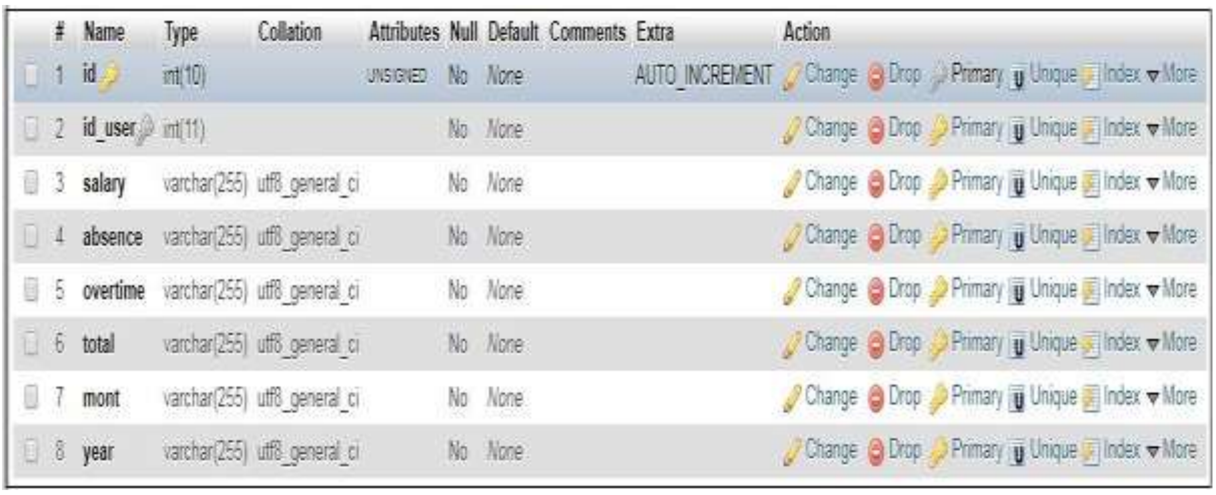


#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(11)			No	None		AUTO_INCREMENT	Change Drop Primary Unique Index Spatial Fulltext Distinct values
2	login	varchar(15)	utf8_general_ci		No				Change Drop Primary Unique Index Spatial Fulltext Distinct values
3	pass	text	utf8_general_ci		No	None			Change Drop Primary Unique Index Spatial Fulltext Distinct values
4	password	varchar(255)	utf8_general_ci		No				Change Drop Primary Unique Index Spatial Fulltext Distinct values
5	email	varchar(255)	utf8_general_ci		No				Change Drop Primary Unique Index Spatial Fulltext Distinct values
6	phone	varchar(255)	utf8_general_ci		No				Change Drop Primary Unique Index Spatial Fulltext Distinct values
7	avatar	varchar(255)	utf8_general_ci		No	images/people/1.jpg			Change Drop Primary Unique Index Spatial Fulltext Distinct values
8	last_name	text	utf8_general_ci		No	None			Change Drop Primary Unique Index Spatial Fulltext Distinct values
9	first_name	text	utf8_general_ci		No	None			Change Drop Primary Unique Index Spatial Fulltext Distinct values
10	middle_name	text	utf8_general_ci		No	None			Change Drop Primary Unique Index Spatial Fulltext Distinct values
11	birthday	text	utf8_general_ci		No	None			Change Drop Primary Unique Index Spatial Fulltext Distinct values
12	adress	text	utf8_general_ci		No	None			Change Drop Primary Unique Index Spatial Fulltext Distinct values
13	anons	text	utf8_general_ci		No	None			Change Drop Primary Unique Index Spatial Fulltext Distinct values
14	rules	varchar(255)	utf8_general_ci		No	None			Change Drop Primary Unique Index Spatial Fulltext Distinct values
15	deport	text	utf8_general_ci		No	None			Change Drop Primary Unique Index Spatial Fulltext Distinct values
16	position_id	int(10)		UNSIGNED	No	None			Change Drop Primary Unique Index Spatial Fulltext Distinct values
17	education_id	int(10)		UNSIGNED	No	None			Change Drop Primary Unique Index Spatial Fulltext Distinct values
18	gender	text	utf8_general_ci		No	None			Change Drop Primary Unique Index Spatial Fulltext Distinct values
19	family	text	utf8_general_ci		No	None			Change Drop Primary Unique Index Spatial Fulltext Distinct values
20	child	varchar(255)	utf8_general_ci		No				Change Drop Primary Unique Index Spatial Fulltext Distinct values
21	Diploma_Number	text	utf8_general_ci		No	None			Change Drop Primary Unique Index Spatial Fulltext Distinct values

Fig. 4. Structure of the table “User”

2. The table “Salary”

The table “Salary” contains information about employees. It consists of 8 fields as in the figure 5.

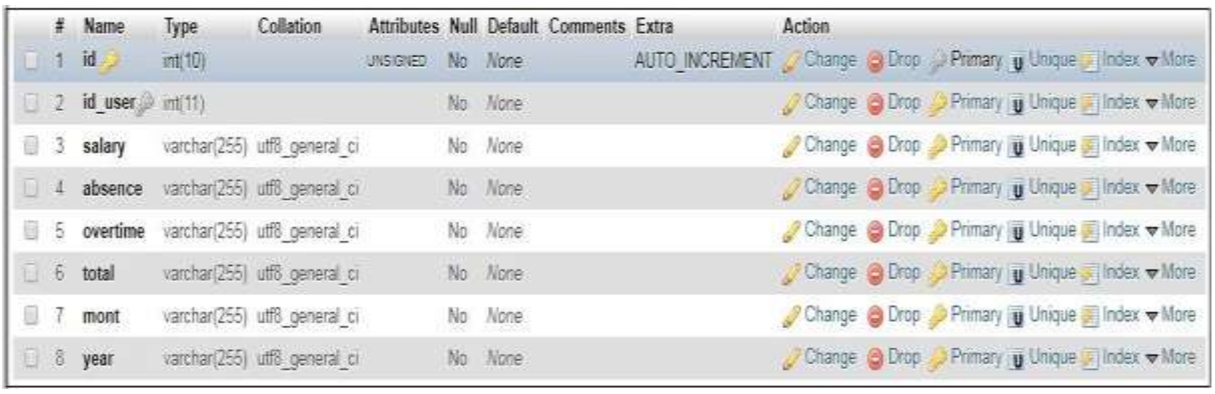


#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(10)		UNSIGNED	No	None		AUTO_INCREMENT	Change Drop Primary Unique Index More
2	id_user	int(11)			No	None			Change Drop Primary Unique Index More
3	salary	varchar(255)	utf8_general_ci		No	None			Change Drop Primary Unique Index More
4	absence	varchar(255)	utf8_general_ci		No	None			Change Drop Primary Unique Index More
5	overtime	varchar(255)	utf8_general_ci		No	None			Change Drop Primary Unique Index More
6	total	varchar(255)	utf8_general_ci		No	None			Change Drop Primary Unique Index More
7	mont	varchar(255)	utf8_general_ci		No	None			Change Drop Primary Unique Index More
8	year	varchar(255)	utf8_general_ci		No	None			Change Drop Primary Unique Index More

Fig. 5. Structure of the table “Salary”

3. The table “Time_work”

The table “Time_work” contains information about employees. It consists of 9 fields as in the figure 6.

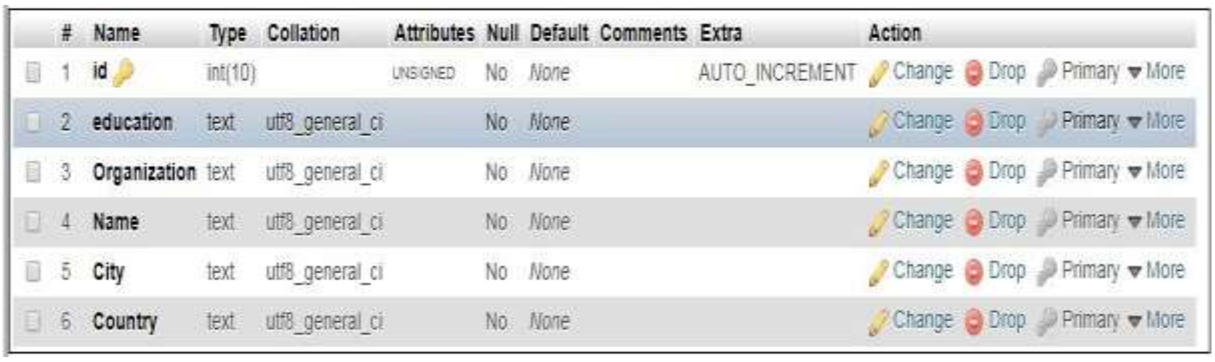


#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(10)		UNSIGNED	No	None		AUTO_INCREMENT	Change Drop Primary Unique Index More
2	id_user	int(11)			No	None			Change Drop Primary Unique Index More
3	salary	varchar(255)	utf8_general_ci		No	None			Change Drop Primary Unique Index More
4	absence	varchar(255)	utf8_general_ci		No	None			Change Drop Primary Unique Index More
5	overtime	varchar(255)	utf8_general_ci		No	None			Change Drop Primary Unique Index More
6	total	varchar(255)	utf8_general_ci		No	None			Change Drop Primary Unique Index More
7	mont	varchar(255)	utf8_general_ci		No	None			Change Drop Primary Unique Index More
8	year	varchar(255)	utf8_general_ci		No	None			Change Drop Primary Unique Index More

Fig. 6. Structure of the table “Time_work”

4. The table “Education”

The table “Education” contains information about employees. It consists of 6 fields as in the figure 7.

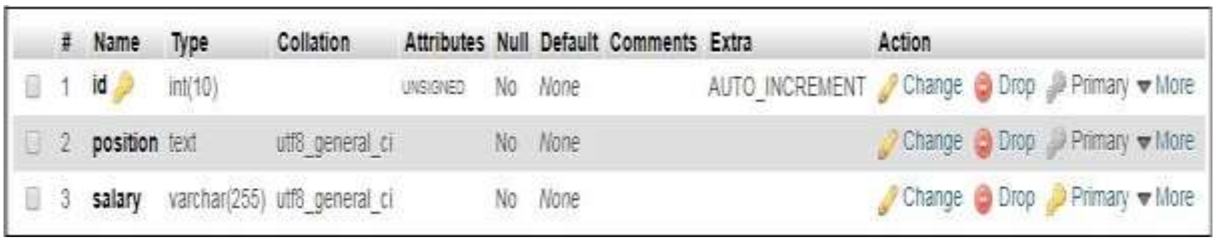


#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(10)		UNSIGNED	No	None		AUTO_INCREMENT	Change Drop Primary More
2	education	text	utf8_general_ci		No	None			Change Drop Primary More
3	Organization	text	utf8_general_ci		No	None			Change Drop Primary More
4	Name	text	utf8_general_ci		No	None			Change Drop Primary More
5	City	text	utf8_general_ci		No	None			Change Drop Primary More
6	Country	text	utf8_general_ci		No	None			Change Drop Primary More

Fig. 7. Structure of the table “Education”

5. The table “Position”

The table “Position” contains information about employees. It consists of 3 fields as in the figure 8.

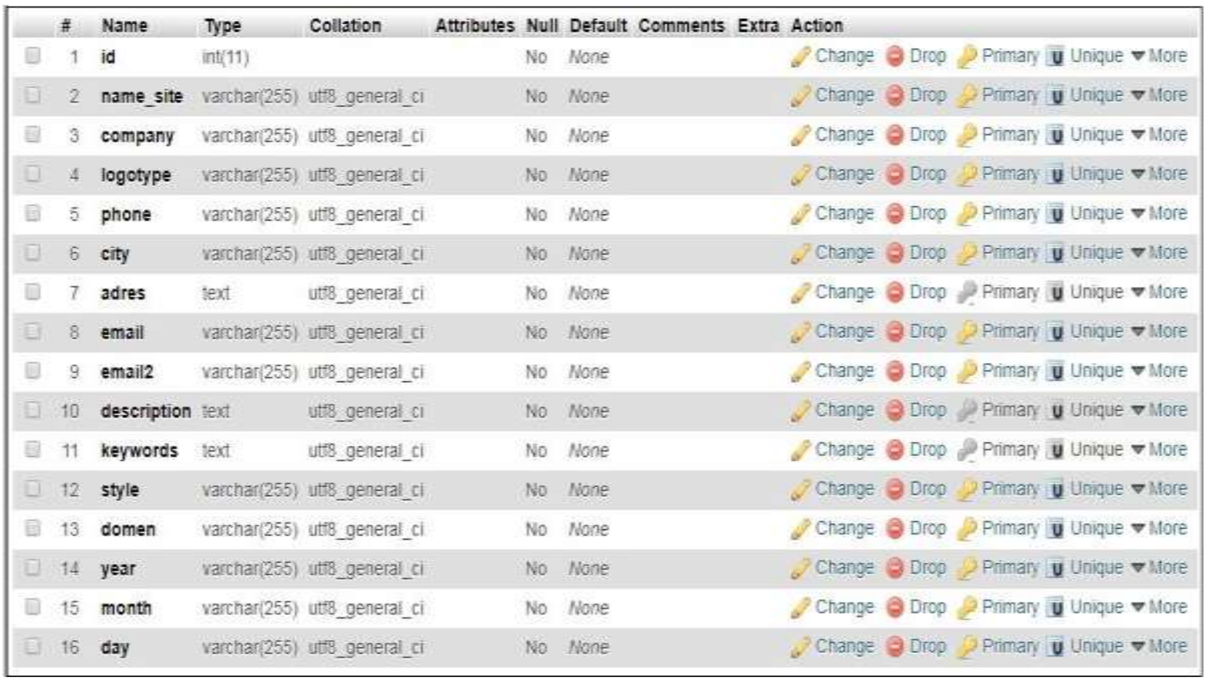


#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(10)		UNSIGNED	No	None		AUTO_INCREMENT	Change Drop Primary More
2	position	text	utf8_general_ci		No	None			Change Drop Primary More
3	salary	varchar(255)	utf8_general_ci		No	None			Change Drop Primary More

Fig. 8. Structure of the table “Position”

6. The table “system”

The table “System” contains information about employees. It consists of 16 fields as in the figure 9.



	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
	1	**id**	int(11)			No	*None*			Change Drop Primary Unique More
	2	**name_site**	varchar(255)	utf8_general_ci		No	*None*			Change Drop Primary Unique More
	3	**company**	varchar(255)	utf8_general_ci		No	*None*			Change Drop Primary Unique More
	4	**logotype**	varchar(255)	utf8_general_ci		No	*None*			Change Drop Primary Unique More
	5	**phone**	varchar(255)	utf8_general_ci		No	*None*			Change Drop Primary Unique More
	6	**city**	varchar(255)	utf8_general_ci		No	*None*			Change Drop Primary Unique More
	7	**adres**	text	utf8_general_ci		No	*None*			Change Drop Primary Unique More
	8	**email**	varchar(255)	utf8_general_ci		No	*None*			Change Drop Primary Unique More
	9	**email2**	varchar(255)	utf8_general_ci		No	*None*			Change Drop Primary Unique More
	10	**description**	text	utf8_general_ci		No	*None*			Change Drop Primary Unique More
	11	**keywords**	text	utf8_general_ci		No	*None*			Change Drop Primary Unique More
	12	**style**	varchar(255)	utf8_general_ci		No	*None*			Change Drop Primary Unique More
	13	**domen**	varchar(255)	utf8_general_ci		No	*None*			Change Drop Primary Unique More
	14	**year**	varchar(255)	utf8_general_ci		No	*None*			Change Drop Primary Unique More
	15	**month**	varchar(255)	utf8_general_ci		No	*None*			Change Drop Primary Unique More
	16	**day**	varchar(255)	utf8_general_ci		No	*None*			Change Drop Primary Unique More

Fig. 9. Structure of the table “System”

2.5. Development of the interface

The user interface, also known as Human Machine Interface (HMI) or Man-Machine Interface (MMI), is the aggregate of means by which people-users interact with the system – a particular machine, device, computer program or other complex tool. The part of an interactive computer program sends messages to and receives instructions from a terminal user [5].

User Interface Design and Ergonomics deals with analysis, design, implementation and evaluation of user interface design**.**

We will implement future views of an application.

2.6. Implementation of the web interface

User interface design is the design of websites and software applications with the focus on the user's experience and interaction.

The goal of user interface design is to make the user's interaction as simple and efficient as possible, in terms of accomplishing user goals. The home page is he login page such as it is shown in figure 10.

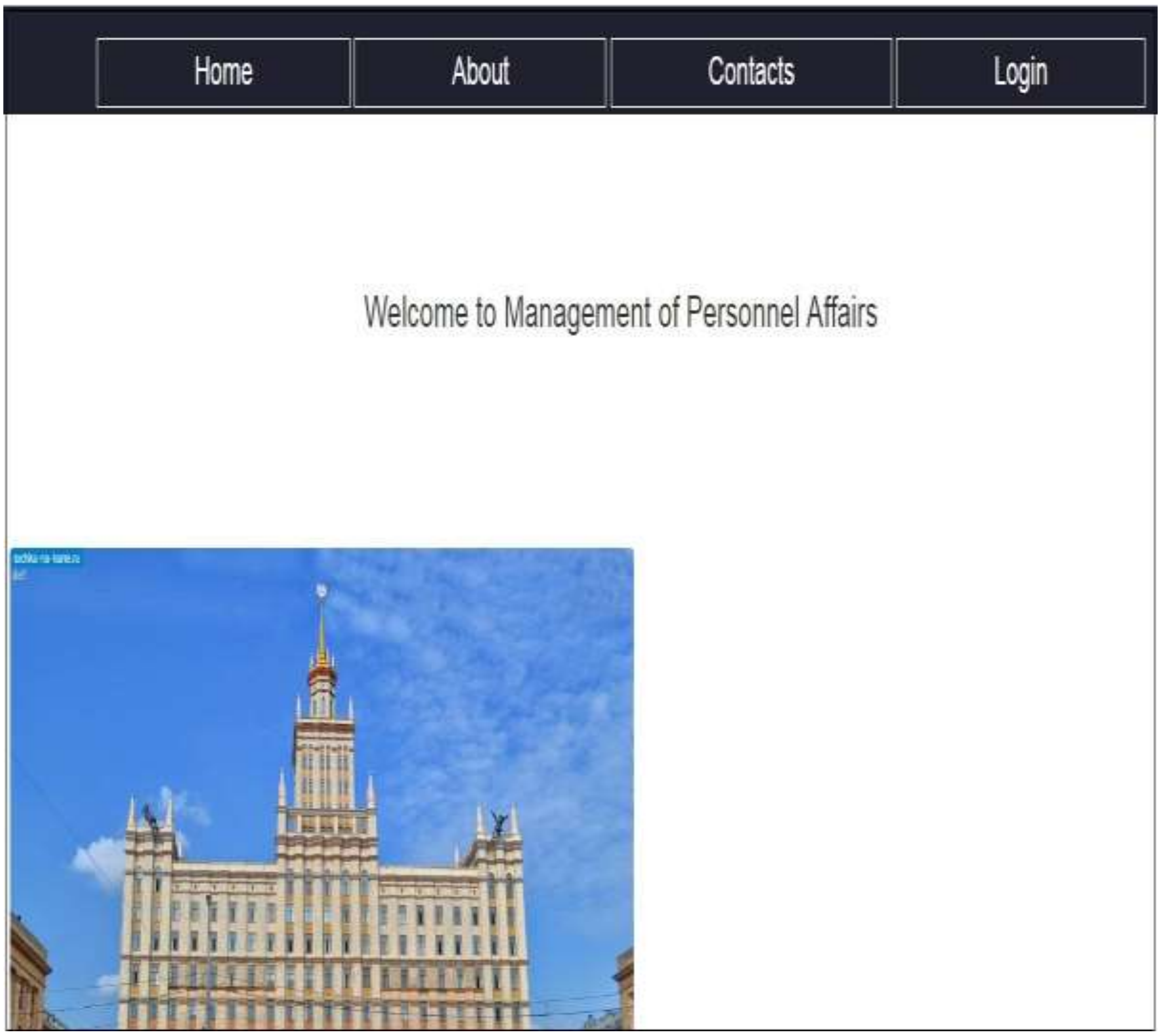
The main interface includes the following commands: HOME, ABOUT, CONTACTS, LOGIN.

Home page.

Home: when pressing this button it will return us to the main interface from any place.

Admin: this command enables the manager to login to the program. When pressing this button it will display another window called: "Admin Login Page" which includes the following fields and buttons: User, Password, Login and Back.

Figure 10 shows the main page for my website Management of Personnel Affairs.



Welcome to Management of Personnel Affairs

Fig. 10. Implementation of the web interface

Login for administrator

After typing the "Username" and "Password" and then pressing "Login", the manager will be able to login to the program. See figure 11.

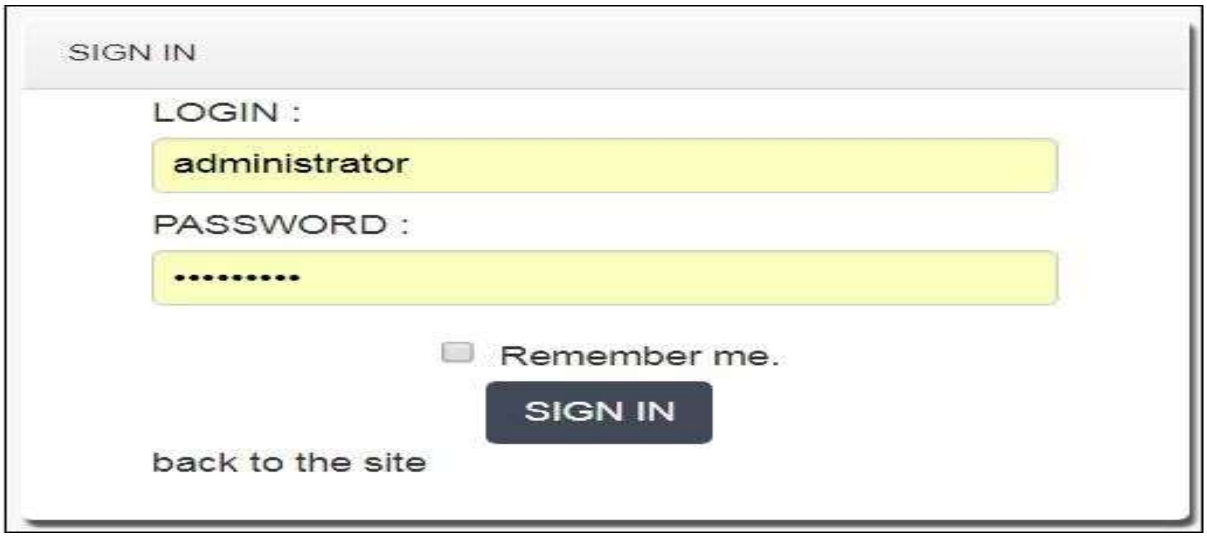
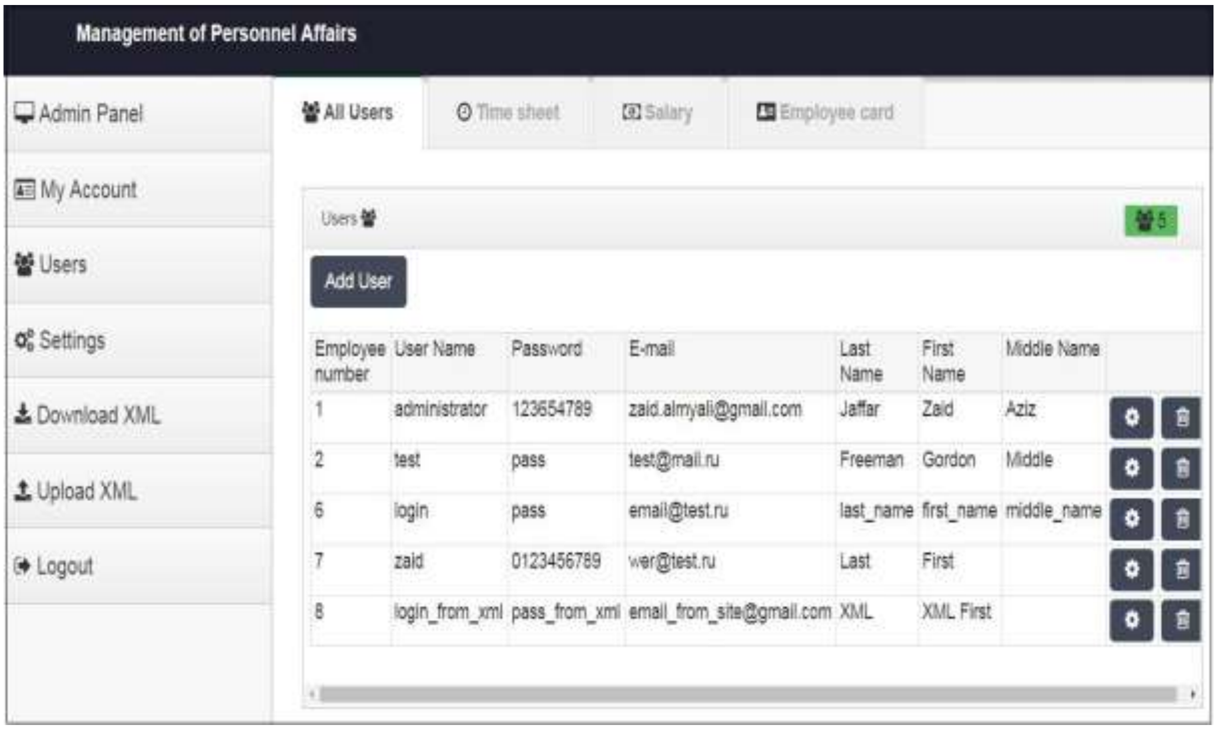


Fig. 11. Login for administrator

Control panel for administrator

After pressing "Login" in the previous interface another window will be displayed. It contains the following options: All Users, Time sheet, Salary, Employee card, see figure 12.



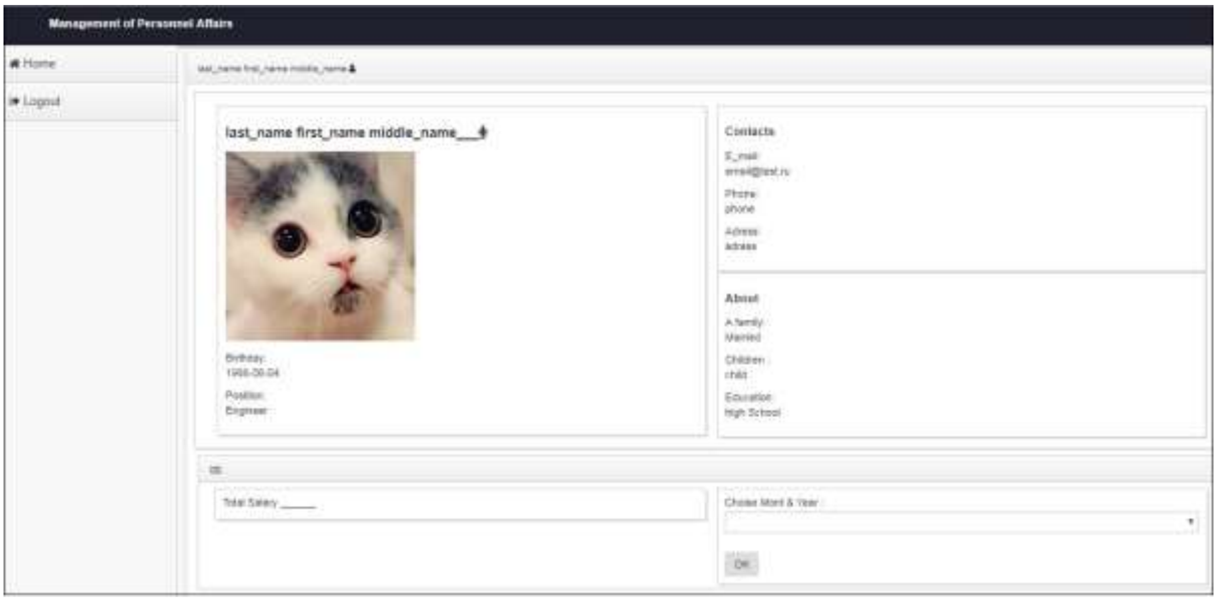
Employee number	User Name	Password	E-mail	Last Name	First Name	Middle Name
1	administrator	123654789	zaid.almyali@gmail.com	Jaffar	Zaid	Aziz
2	test	pass	test@mail.ru	Freeman	Gordon	Middle
6	login	pass	email@test.ru	last_name	first_name	middle_name
7	zaid	0123456789	wer@test.ru	Last	First	
8	login_from_xml	pass_from_xml	email_from_site@gmail.com	XML	XML First	

Fig. 12. Control panel for administrator

Login for employee

After typing the "Username" and "Password" and then pressing "Login", the employee will be able to login to the program. See figure 11.

After pressing "Login" in the previous interface another window will be displayed. It contains the following. See figure 13.



last_name first_name middle_name

last_name first_name middle_name___

Birthday:
1986-08-04

Position:
Engineer

Contacts

E_mail:
email@test.ru

Phone:
phone

Adress:
adress

About

A family:
Married

Children:
child

Education:
High School

Total Salary ______

Chose Mont & Year :

Fig. 13. Login for employee

3. IMPLEMENTING THE BASIC FUNCTIONALITY ON THE WEB-SITE

3.1. Page “Home”

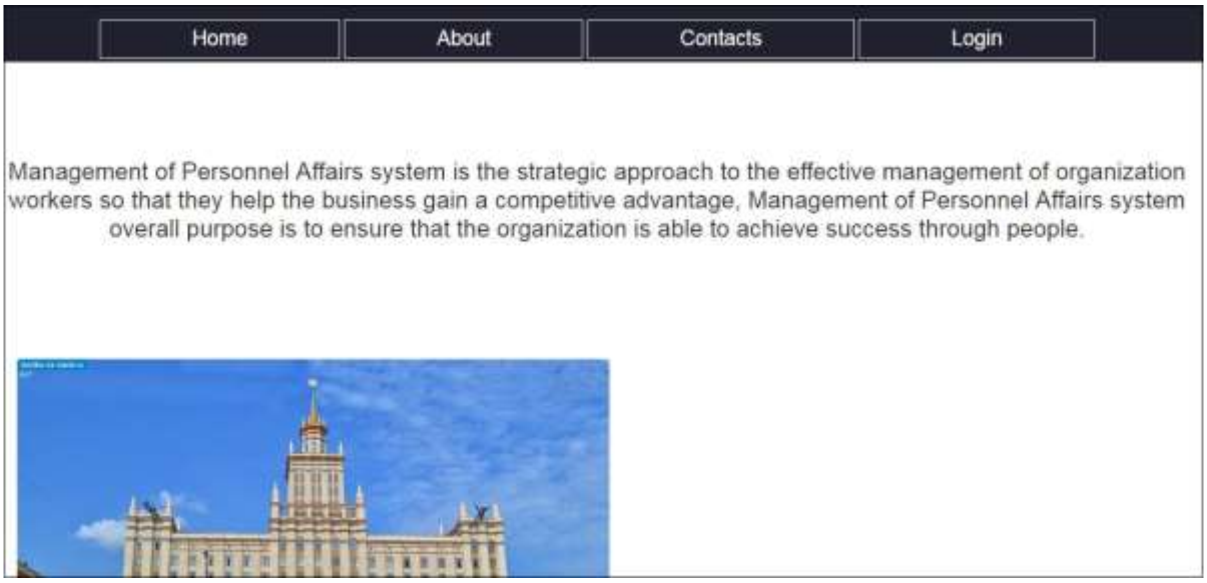
As it was mentioned above the home page is a login page and its appearance is presented on figure 10. It has a menu with such items like Home, About, Contacts and Login which are made as hyper-references to corresponding php-files.

Listing 1. Codes for page "Home"

```
<?php
include ("admin/inc/config.php");
?>
<!DOCTYPE html>
<html>
<head><meta http-equiv="Content-Type" content="text/html;
charset=ansi_x3.110-1983">
<title>Management of Personnel Affairs</title>
<link href="css/style.css" rel="stylesheet" type="text/css">
<link rel="shortcut icon" href="favicon.png" type="image/x-icon" />
<head>
<body>
<div class="row">
<header>
<ul id="menu">
<li><a href="/">Home</a></li>
<li><a href="about.php">About</a></li>
<li><a href="contacts.php">Contacts</a></li>
<li><a href="admin/index.php">Login</a></li>
</ul>
</header>
<div class="content">
<div class="container">
<div  class="wel">
<h1>Welcome to Management of Personnel Affairs</h1>
<p><img src="img/bg1.jpg" width="50%" height="auto"
 alt="university"></p>
</div>
</div>
<div>
</div>
</div>
<? include 'admin/inc/footer.php';?>
</body>
</html>
```

3.2. Page “About”

Page “About” has appearance similar to page “Home” (see figure 10), that is why their codes are alike; the difference between them is page "ABOUT" contain information about my system, it makes the system easier, and more understandable for users, and provides users abstract about management of personnel affairs. See figure 14 page “About”.

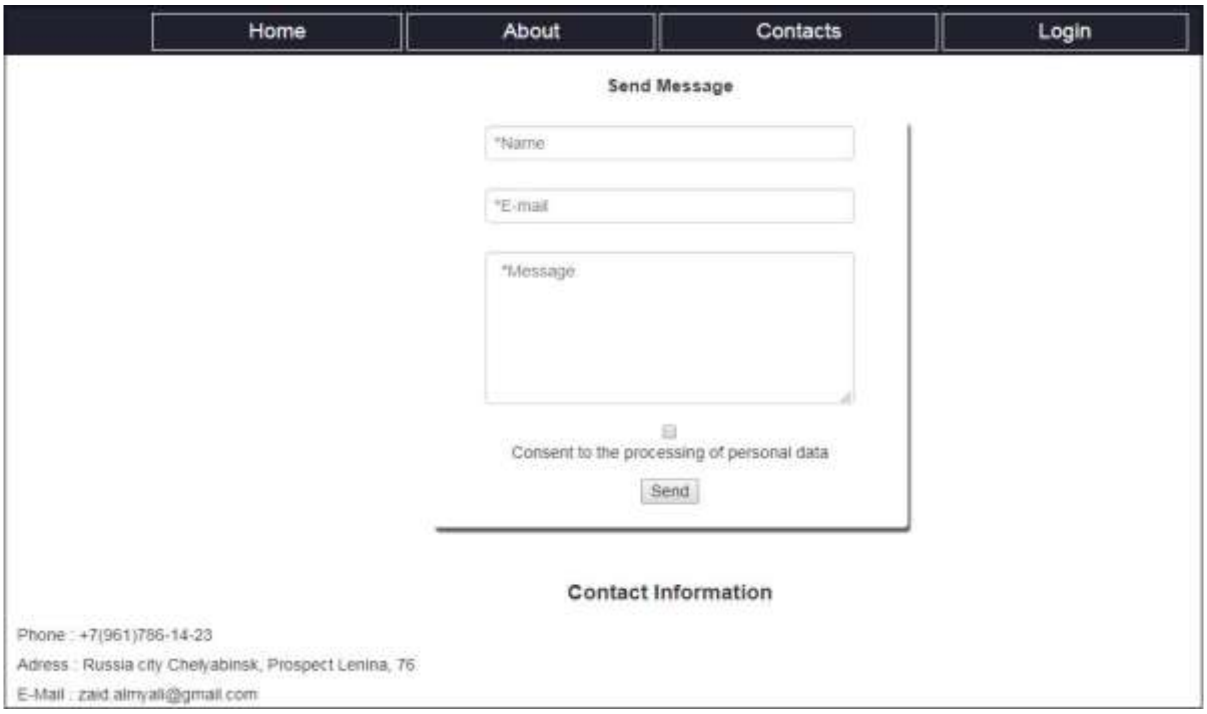


Management of Personnel Affairs system is the strategic approach to the effective management of organization workers so that they help the business gain a competitive advantage, Management of Personnel Affairs system overall purpose is to ensure that the organization is able to achieve success through people.

Fig. 14. Page “About”

3.3. Page "Contacts"

Php-code for page “Contacts” retrieves company’s data from MySQL database and provides access for sending feedback messages. See figure 15 page "Contacts".



Consent to the processing of personal data

Contact Information

Phone : +7(961)786-14-23

Adress : Russia city Chelyabinsk, Prospect Lenina, 76

E-Mail : zaid.almyali@gmail.com

Fig. 15. Page "Contacts"

Listing 2. Codes for page "Contacts"

```
<?php
include ("admin/inc/config.php");
?>
<html>
<head>
<title>CONTACTS</title>
<link href="../css/style.css" rel="stylesheet" type="text/css">
<link rel="shortcut icon" href="favicon.png" type="image/x-icon" />
<head>
<body>
<div class="row">
<header>
<ul id="menu">
<li><a href="/">Home</a></li>
<li><a href="about.php">About</a></li>
<li><a href="contacts.php">Contacts</a></li>
<li><a href="admin/index.php">Login</a></li>
</ul>
</header>
<div class="content">
<div class="container">
<div  class="wel">
<div class="contact-form centered">
<h3>Send Message</h3>
<?
$sql = mysql_query("SELECT * FROM `system`");//taking data from MySQL
while ($result = mysql_fetch_array($sql)) {
if (isset ($_POST['messageF'])) {
mail ("".$result['email']."", //taking e-mail from system table
"For YOU message ".$_SERVER['HTTP_REFERER'],
"Username::   ".$_POST['nameF']."
// putting name, contacts and messag
E-mail    ".$_POST['contactF']."
// from web-form
Message text:
".$_POST['messageF']);
}
}
echo'
<div class="dialog">
<div class="panel panel-default">
<div class="panel-body">
<form method="POST" id="feedback-form">
<div class="form-group">
<input type="text" class="form-control span6" name="nameF" required
id="name" placeholder="*Name" x-autocompletetype="name">
</div>
<div class="form-group">
<input type="text" class="form-control span6" name="contactF"
required id="email" placeholder="*E-mail" x-autocompletetype="name">
</div>
<div class="form-group">
<textarea type="text" class="form-control span6" name="messageF"
id="comment" required placeholder="*Message" x-
autocompletetype="name" rows="6"></textarea>
</div>
<div class="form-group">
<input type="checkbox" required name="politicF"><br /><span
style="font-size:14px;">Consent to the processing of personal
data</span>
</div>
```

```
<div class="form-group">
<button type="submit" class="message-btn">Send</button>
</div>
</form>
</div>
</div>
</div>';
$sql = mysql_query("SELECT * FROM `system`");
while ($result = mysql_fetch_array($sql))
echo '<br/><br/>
<h2>Contact Information</h2>
<p>  Phone : '.$result['phone'].'</p>
<p>  Adress : '.$result['city'].', '.$result['adres'].'</p>
<p>  E-Mail : '.$result['email'].'</p>';
?>
</div>
</div>
</div>
<div>
</div>
</div>
<?
include 'admin/inc/footer.php';
?>
</body>
</html>
```

3.4. Page "Login"

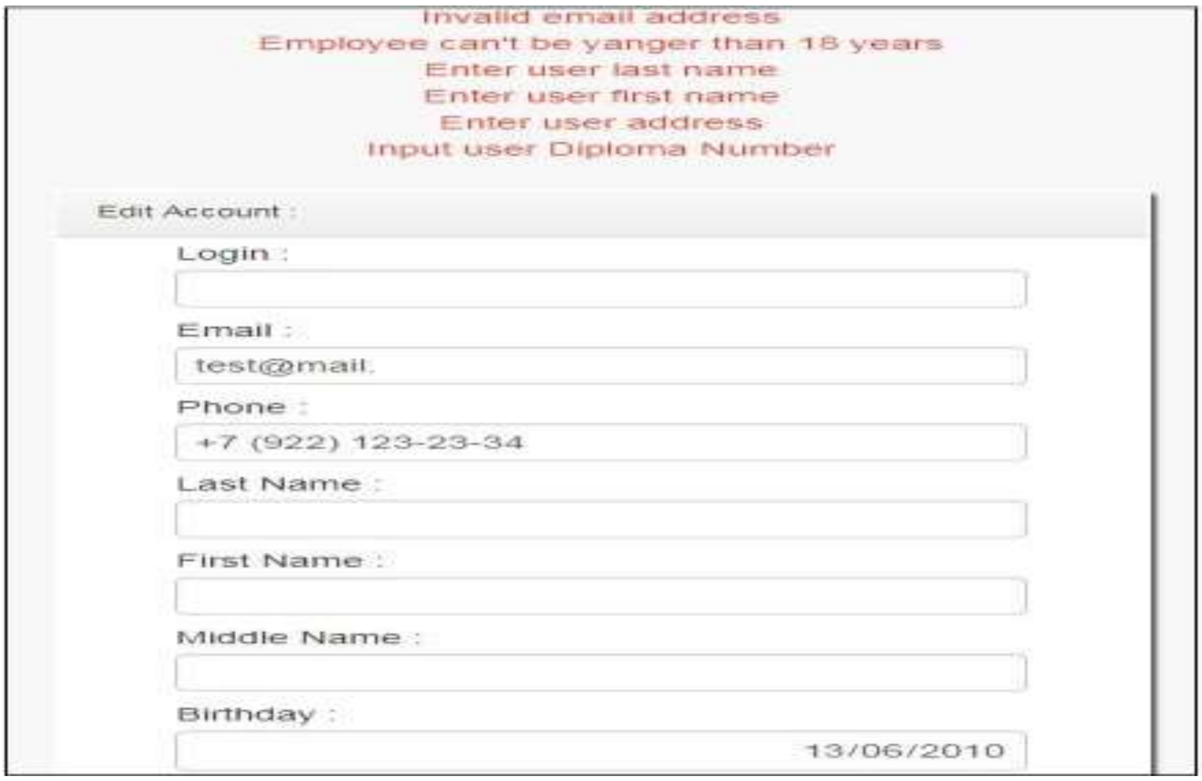
Php-code for page “Login” makes a query for login data from MySQL database and provides access for sending feedback messages (see figure 11).

Listing 3. Codes for page "Login"

```
<?php
include ("inc/config.php");
include ("inc/index_hed.php");
echo '
';
if (!empty($_SESSION['login']) and !empty($_SESSION['password']))
{
$login = $_SESSION['login'];
$password = $_SESSION['password'];
$result = mysql_query("SELECT * FROM users WHERE login='$login' AND
password='$password'",$db);
$myrow = mysql_fetch_array($result);
}
if (!isset($myrow['pass']) or $myrow['pass']=='')
{
print <<<HERE
<div class="dialog">
<div class="panel panel-default">
<p class="panel-heading no-collapse">SIGN IN</p>
<div class="panel-body">
<form action="testreg.php" method="post">
<div class="form-group">
<label> LOGIN :</label>
<input type="text" name="login" class="form-control span12">
HERE;
print <<<HERE<div class="form-group"><label> PASSWORD :</label><input
name="password" type="password" class="form-control span12 form-
control">
HERE;
print <<<HERE
<center>
<label class="remember-me"><input name="save" type="checkbox"
value='1'> Remember me.</label></center>
<center>
<input type="submit" name="submit" class="btn btn-primary"
value="SIGN IN">
<br>
<p><b><a href='/'>back to the site</a></b></p>
</center>
</form>
</div>
</div>
</div>
HERE;}else{
print <<<HERE
<div class='dialog'>
<div class='panel panel-default'>
<p class='panel-heading no-collapse'>LOGIN</p>
<div class='panel-body justy-fly-center'>
<div class="form-group" align='center'>
<ul id="login">
<li><a href='$_SESSION[rules].php?id=$_SESSION[id]'><h4><i class="fa
fa-sign-in" aria-hidden="true"></i> Sign in</h4></a></li>
<li><a href='exit.php'><h4><i class="fa fa-sign-out" aria-
hidden="true"></i> Logout</h4></a></li>
</ul>
<img alt='$_SESSION[login]' src='$myrow[avatar]' width='30%'
height='auto'></div>
<!-- Between operator
"print <<<HERE" output html code with the necessary variables from
php
You are logged in to the control panel as an administrator-->
<!--<b>$_SESSION[login]</b>
<br>-->
<!-- above link to logout -->
<center>
<p><b><a href='/'>back to the site</a></b></p>
</center>
</div>
</div>
</div>
</div>
HERE;
/}
include "inc/footer.php";
echo "</div>";
?>
</div>
<style type="text/css">
Footer
{
position: fixed;
left: 0;
bottom: 0;
width: 100%;
height: 80px;}
</style>
</body>
</html>
```

3.5. Checking correctness of e-mail and birth date

Figure 16 shows how e-mail address and birth date is checked in the form for inputting and editing user’s data.


Invalid email address
Employee can't be yanger than 18 years
Enter user last name
Enter user first name
Enter user address
Input user Diploma Number

Edit Account :

Login :

Email :
test@mail.

Phone :
+7 (922) 123-23-34

Last Name :

First Name :

Middle Name :

Birthday :
13/06/2010

Fig. 16. Message that user’s data are invalid

Listing 4 contains code in PHP for checking validity of e-mail.

Listing 4. Check e-mail

```
/*Check email */
$email = $user['email'];
if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
$errors[] = "Invalid email address";
}
/** ~~~ Check email ~~~ */
```

Listing 5 contains code in PHP for checking validity of birthdate: user can’t be older than 90 years and younger than 18 years.

Listing 5. Check of birthdate

```
/** check birthday */
$birthday = $user['birthday'];
$date = strtotime( $birthday );
if ($date < strtotime("-90 YEAR"))
{ $errors[] = "Employee can't be older than 90 years";
}
if ($date > strtotime('-18 YEAR'))
{ $errors[] = "Employee can't be yanger than 18 years"; }
/** ~~~ check birthday ~~~ */
```

3.6. Importing and exporting procedures with XML and XSD

Usually data for such a web-site are taken from external database of human resource department. That is why the application should have an importing and exporting procedure. It is common to use XML files as data transfer in corporate database. XSD is used to check validity of XML-files.

Downloading users to XML file

Figure 17 presents the interface for downloading user's data from an XML file to the web-site.

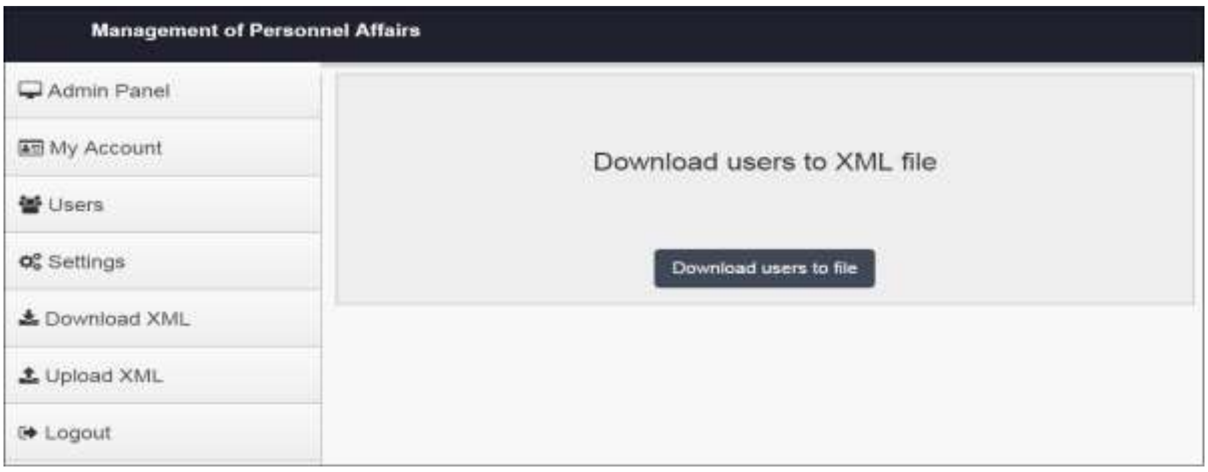


Fig. 17. Downloading users from an XML file

Program import.php implements this importing procedure (see listing 6).

Listing 6. Codes of import.php

```
<?php
include ("../inc/config.php");
include_once ("../inc/user.php");
$errors = [];
$success = [];
if (isset($_POST['submit']))
{
if (!is_file($_FILES['file']['tmp_name'])) // Check uploaded file
{ $errors[] = 'Please, choose file to upload'; // if file no upload,
add error
}
if (!count($errors))
{
$document = new \DOMDocument(); // Create new instance DOMDocument
$document->loadXML( file_get_contents( $_FILES['file']['tmp_name'] )
);
// LOAD XML FROM USER
$check = $document->schemaValidate('../../xsd/user.xsd');
// check XML FROM XSD Scheme
if (!$check)
{
```

```
$errors[] = 'XML is not valid';
// add error
}
}
if (!count($errors))
{
$users = $document->getElementsByTagName('user');
// GET ALL elements named "user" FROM XML file
/** @var \DOMElement $row */
foreach ($users as $row)
// iterate by all user-elements
{
$user = User::getNew(); // get empty user for save in database
foreach ($row->childNodes as $node) // iterate childs user elements,
which named as columns in table users
{
$user[$node->nodeName] = $node->nodeValue; // fill in user object
}
// make password column
$password = $user['pass'];
$password = md5($password);
$password = strrev($password);
$password = $password . "b3p6f";
$user['password'] = $password; // set password to user
if (User::save( $user )) // save user and check result for success
{
$success[] = 'IMPORT SUCCESSFUL <strong>' . $user['login'] .
'</strong> <small>'.$user['email'].'</small>'; // save result to
success array
}
else
{ $success[] = 'IMPORT ERROR <strong>' . $user['login'] . '</strong>
<small>'.$user['email'].'</small>'; // save result to success
array }
}
}
}
include ("../inc/header.php");
?>
<div class="content-sys">
<div class="main-content">
<? if (count($errors)):
// if we get some errors display it ?>
<div class="errors" style="color: red;"><?= implode('<br />',
$errors)
?>
</div>
<? endif;
?>
<? if (count($success)):
// if we get some success, display it ?>
<div class="success"><?= implode('<br />', $success) ?></div>
<?
endif; ?>
<div class="div-center">
<form method="post" enctype="multipart/form-data">
<label>Upload users from XML file</label>
<input type="file" name="file" />
<button class="btn btn-primary" type="submit" name="submit">Import
users from file</button></form>
</div>
</div>
</div>
```

Downloading users to XML file

Figure 18 presents the interface for uploading user's data from the web-site to an XML-file.

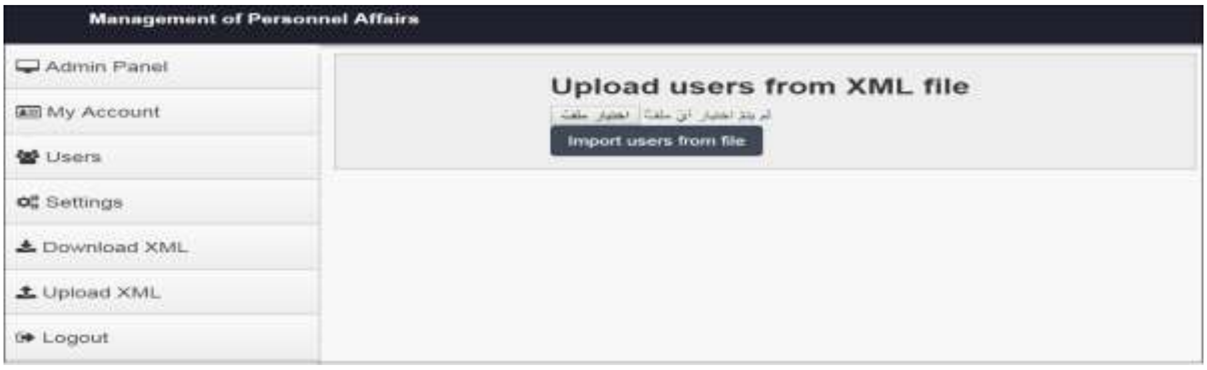


Fig. 18. Uploading users to an XML file

Program export.php implements this importing procedure (see listing 7).

Listing 7. Codes of export.php

```
<?php
include ("../inc/config.php");
if (isset($_POST['submit']))
{ $document = new \DOMDocument(); // get instanse of class
DOMDocument
$root = $document->createElement('root'); // create root XML element
$document->appendChild($root); // ADD root to document
$xml_users = new \DOMElement('users'); // CREATE users element
$root->appendChild($xml_users); // users element add to root-element
$res = mysql_query("SELECT * FROM `users`"); // get all users from
mysql-database
while ($row = mysql_fetch_assoc( $res )) // iterate users from
database
{
$user = new \DOMElement('user'); // create new element named "user"
$xml_users->appendChild($user);
foreach ($row as $name => $value) // iterate columns of user from
datase
{
if (in_array($name, ['password', 'anons'])) // skip it
continue;
$elem = new \DOMElement($name); // create element with name from
database table users
$elem->nodeValue = $value; // put value from database
$user->appendChild( $elem );}} // append child to user
$check = $document->schemaValidate('../../xsd/user.xsd'); // if XML
is valid by XSD
if ($check)
{
header('Content-type: text/xml'); // TYPE OF FILE TO DOWNLOAD
header('Content-Disposition: attachment; filename=users.xml'); //
NAME OF FILE TO DOWNLOAD
echo $document->saveXML(); // echo XML file to browser
exit;
}
}
include ("../inc/header.php");
?>
<div class="content-sys">
<div class="main-content">
<form method="post" style="text-align: center;">
<h1>Download users to XML file</h1>
<button class="btn btn-primary" type="submit" name="submit">Download
users to file</button>
</form>
</div>
</div>
```

Below an example of XML-file which can be created by the site or corporate information system is given (see listing 8).

Listing 8. Example of users.xml

```
<?xml version="1.0"?>
<root> <!-- root element, can be only one, complex type, means that
it contain other elements -->
<users> <!-- element users, contain list of users from/to site,
conmplex type -->
<user> <!-- element user, complex type, contain data of the user -->
<id>1</id> <!-- identificator of the user in database, type int -->
<login>administrator</login> <!-- login of the user, type string -->
<pass>123654789</pass> <!-- password, type string -->
<email>zaid.almyali@gmail.com</email> <!-- e-mail, type string -->
<phone>+7(961)786-14-23</phone> <!-- phone, type string -->
<avatar>images/people/1526569457_hard.png</avatar> <!-- avatar, type
string -->
<last_name>Jaffar</last_name> <!-- type string -->
<first_name>Zaid</first_name> <!-- type string -->
<middle_name>Aziz</middle_name> <!-- type string -->
<birthday>1984-01-15</birthday> <!-- type date, format yyyy-mm-dd -->
<adress>Chelyabinsk, Lenina, 76</adress> <!-- -->
<rules>admin</rules> <!-- access level of the user -->
<deport>Marketing</deport> <!-- type string -->
<position_id>1</position_id> <!-- type int, contain ID of position in
database -->
<education_id>5</education_id> <!-- type int, contain ID of education
in database -->
<gender>male</gender> <!-- type string -->
<family>Married</family> <!-- type string -->
<child>6</child> <!-- type string -->
<Diploma_Number>SU-123</Diploma_Number> <!-- type string -->
</users>
</root>
```

In order to check validity of XML-files an XSD file was developed (see listing 9).

Listing 9. File user.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">
<xs:element name="root"> <!-- root element in target XML -->
<xs:complexType><!-- Type of root element, complex means, that root
include other elements --><xs:sequence><!-- specifies that the child
```

```
elements must appear in a sequence. -->
<xs:element name="users">
<!-- One or more elements users -->
<xs:complexType><!-- type of element users, means, that users include
other elements -->
<xs:sequence>
<!-- specifies that the child elements must appear in a sequence. -->
<xs:element name="user" maxOccurs="unbounded"><!-- XML element user -
->
<xs:complexType><!-- type complex -->
<xs:sequence>
<xs:element name="id" type="xs:int"
nillable="true"></xs:element><!-- element id, type integer, can be
nillable -->
<xs:element name="login" type="xs:string"></xs:element><!-- element
login, type string, login of the user -->
<xs:element name="pass" type="xs:string"></xs:element><!-- element
pass, type string, password of the user -->
<xs:element name="email" type="xs:string"></xs:element><!-- element
email, type string, contain E-mail address -->
<xs:element name="phone" type="xs:string"></xs:element><!-- element
phone, type string, contain phone number of the user -->
<xs:element name="avatar" type="xs:string"></xs:element><!-- element
avatar, type string, contain path_to image file -->
<xs:element name="last_name" type="xs:string">
</xs:element>
<!-- element last_name, type string -->
<xs:element name="first_name" type="xs:string"></xs:element><!—
first name, type string -->
<xs:element name="middle_name" type="xs:string"></xs:element><!—
middle name, type string -->
<xs:element name="birthday" type="xs:date"></xs:element><!—
birthday, type date. Must contain date in format yyyy-mm-dd -->
<xs:element name="adress" type="xs:string"></xs:element>
<!-- adress, type string, contain home address of user -->
<xs:element name="rules" type="xs:string"></xs:element>
<!-- rules, type string, contain access level of the user -->
<xs:element name="deport" type="xs:string"></xs:element>
<!-- deport, type string -->
<xs:element name="position_id" type="xs:int"></xs:element>
<!--position_id, type integer, contain id of position from position
table -->
<xs:element name="education_id" type="xs:int"></xs:element><!—
education_id, type integer, contain id of education row -->
<xs:element name="gender" type="xs:string">
</xs:element><!-- gender, string -->
<xs:element name="family" type="xs:string">
</xs:element><!-- family, contain marital status -->
<xs:element name="child" type="xs:string">
</xs:element><!-- count of childrens -->
<xs:element name="Diploma_Number"></xs:element>
<!-- number of diploma -->
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:schema>
```

4. TESTING

Functional system tests should be based around coverage of the functionality described in the requirements, but it is common for the design document to be used as the baseline for testing because the requirements cannot be related to the product.

Each test of my system contains input and output information. Therefore, we compare the actual results and the expected results [14].

4.1. The main page testing

After I finished the to design and implementation my website, I must be checking from the functions for main page, it must be working correctly. Table 1 shows the necessary steps to do that.

Table 1. The main page testing

No	Test case	Test steps	Expected Result	Actual results
1	To show "Main page".	1. The user opens the internet explorer. 2. The user writes local host/almyali.	Any employee can only watch this With the list of the main sections.	The function works correctly.
2	To give all employees the permission to see the page "Contacts".	The user press "Contacts" button.	Any employee can See the page "Contacts".	The function works correctly.
3	To give all employees the permission to see the page "About".	Any employee can See the page "About".	Any employee can See the page "About".	The function works correctly.

4.2. Admin interface testing

After I finished check the main page, I must checking from the functions for Admin interface testing, it must be working correctly. Table 2 shows the necessary steps to do that.

Table 2. Admin interface testing

No	Test case	Test steps	Expected Result	Actual results
1	When the admin enters the user name and Password properly.	1. Enter "Admin" in name. 2. Enter “Admin1984” in password.	Open the admin page.	The function works correctly.
2	When the admin Enters username and password improperly.	1. Enter "Admin" in name. 2. Enter incorrect password.	System should prompt the user to enter valid values.	The function works correctly.
3	If the admin enters invalid username and correct password.	1. Enter incorrect name. 2. Enter “Admin1984” in password.	Should don’t open the administrator Page.	The function works correctly.
4	If the admin enters valid username and incorrect password.	1. Enter a correct name. 2. Enter “4321” password.	Should don’t open the administrator Page.	The function works correctly.

End of table 2

No	Test case	Test steps	Expected Result	Actual results
5	Add a new employee in the database.	The admin can insert name, job, e-mail, etc. and register them in database.	The system saves the employee information.	The function works correctly.
6	The admin wants to delete some employee.	1. The admin selects employee name.	The system will delete employee information.	The function works correctly.
7	The admin to wants edit some employee.	1. The admin selects employee name. The admin press “Edit” button.	The system will edit employee information.	The function works correctly.

4.3. Employee interface testing

After I finished check the main page and Admin interface testing, I must checking from the functions for employee interface testing, it must be working correctly. Table 3 shows the necessary steps to do that.

Table 3. The employee interface testing

No	Test case	Test steps	Expected Result	Actual results
1	If the user enters valid username and valid password.	1. Enter "Zaid Almyali" in name. 2. Enter “15011984”In password.	Should open the page for Zaid's employee.	The function works correctly.

End of table 3

No	Test case	Test steps	Expected Result	Actual results
2	If the user enters Incorrect user name and password.	1. Enter incorrect name. 2. Enter “5555” in password.	Should don’t open the Page for employee.	The function works correctly.
3	If the user enters invalid username and correct password.	1. Enter incorrect name 2. Enter “15011984” in password.	Should don’t open the Page for employee.	The function works correctly.
4	If the user enters valid username and incorrect password.	1. Enter correct name. 2. Enter “0000” password.	Should don’t open the Page for employee.	The function works correctly.
5	If the user wants to see his/her own personnel information, for example the employee (Zaid Almyali) wants see his own information.	1. Enter "Zaid Almyali" in name. 2. Enter "15011984" in password.	He will see his information.	The function works correctly.

CONCLUSION

A serious need to create corporate websites arose with the realization of the need to create a single information space which allows employees of the company to receive reliable and timely information about the wages in real time.

Management of Personnel Affairs is an integrated technological system which stores, and analyses data relating to an organization's human resources. It efficiency of personnel management because it provides valuable information about employees and managers can use this system to track staff.

No doubt that management of personnel affairs can help both employer and employee to do their job. It can make easier to an organization to go smoothly in using information technologies. Organizations can improve their management system from traditional approach to a modern approach using a modern technology base. In addition, systems like mine can be a competitive advantage, because it increases effectiveness of personnel management (more profits) and satisfaction of staff (more attractive in labor market).

There are some benefits of implementing the system for management of personnel affairs:

1. Standardization. Management of personnel affairs provides uniformity through templates and predetermined procedures for uploading data and downloading reports. It also means that data retrieved and viewed is in a format that is easily identifiable and user friendly.

2. Knowledge management. Knowledge management is an important element in successful management of personnel affairs. Management of personnel affairs become a house of important information on the various aspects of an employee's history within the company.

Lastly, implementing this project was a good experience for me because it made me understand the peculiarities of human resources information system and I can use this knowledge in my future in personnel management of information technology department.

REFERENCES

1. Academy of Management Journal.amj.aom.org. [Electronic resource] URL: https://journals.aom.org/journal/amj/ (date of access: 18.03.2018).

2. Apache.org. URL [for electronic resource]: http: // org / ABOUT_ APACHE.html/ (Date of Arrival: 13.11.2017).

3. Database Management System (DBMS). [E-mail] URL: https: // www.techopedia.com/definition/24361/database-manageme. Time dbms/ (date of arrival: 15.11.2017).

4. Dennis I., Wixom I., Analysis and Design Systems Tegarden I.: a Object oriented approach with UML. - Fifth Edition - USA, New Jersey: Wiley, 2016. - 507 p.

5. Karson I. Great UI designs. URL [for electronic resource]: http: //www.creativebloq.com / web-design / examples-ui-design-7133429/ (Date of Arrival: 15.12.2017).

6. Larry Ullman. PHP for the World Wide Web, Second Edition. – Wily Publishing, 2004. – 480 p.

7. Lucidchart. [Email] URL: https://www.lucidchart.com/ pages / uml-deployment-diagram/ (date of arrival: 17.02.2018).

8. Molina H.G., Ullman J.D., Widom J. database systems. The Complete Book. 2nd ed. – USA: Pearson Education Inc.,2009. – 1241 p.

9. Mysql.com [e-mail] URL: http://www.mysql.com/why-mysql/ (date accses: 14.04.2018).

10. New in JavaScript 1.7. [Electronic recourse] URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/ (date of access: 16.04.2018).

11. Official site of MySQL Server. [Electronic Resource] URL:http://dev.mysql.com/doc/refman/4.1/en/what-is-mysql.html/ (date of access: 16.02.2018).

12. Official site of phpMyAdmin. [Electronic Resource] URL: http: //

www.phpmyadmin.net/home_page/index.php (date of access: 10.03.2018).

13. Technologies Should every web developer be able to explain? [Email] URL: http://blog.differential.com/14-technologies-every-web developer-should-be-to-explain / (date of access: 15.01.2018).

14. Test Functional Test: What is, Process, Types, & Examples. [Email resource] URL: http://www.guru99.com/functional-testing.html/ (Date of arrival: 05.05.2018).

15. Upton D. CodeIgniter for Rapid PHP Application Development. – UK: Packt Publishing Ltd, 2007. – 257 p.

16. W3C: Cascading Style Sheets. [Electronic recourse] URL: https://www.w3.org/Style/CSS/Overview.en.html/ (date of access: 16.03.2018).

17. W3C: HTML. [Electronic recourse] URL: http://www.w3.org/html/ (date of access: 16.02.2018).

18. Web site of MySQL Server. [Electronic resource] URL: http://dev.mysql.com/doc/refman/4.1/en/what-is-mysql. (date of access: 13.02.2018).

19. Web Technologies - Part 2. URL [e-resource]: http: //webigg.com / blog / web-technologies-part-2 (date of arrival: 13.10.2017).

20. Why is news important? URL [for e-resource]: http: // schoolvideonews.com/Broadcast-Journalism/Why-is-News (Arrival Date: 2017/10/13).

21. World Information Technology Report 2015 - 2015. - P. XI.