Mikhail Rumiantcev

USER SERVICES FOR THE WEB-BASED TRADE PLATFORM

Bachelor’s Thesis
Technology, Communication and Transport

May 2015
### User Services for the Web-Based Trade Platform

**Abstract**

The purpose of this study is the implementation of the web application development using the object-oriented programming with the Java platform. The application which developed for this study performs the data manipulating to provide the user services such as accounting and communicating. In more detailed view the application allows users to register in the virtual system, create accounts with the personal data, modify this data, gain authorized access for the personal accounts, view other users and communicate with them by exchanging of the text messages.

The researches of the relevant data storing methodologies and main principles of the Java programming are presented in the theory part. The solid theoretical background helps a reader to get the main idea and the overview of the web-based software development field. All analyses and researches in the theory part of this study are based on the reliable literature sources. This way helps to consider different points of view for the studying subjects. When the developer looks wider and collects the information about huge number of methodologies it becomes easier to find tools and solutions which are the most suitable for the requirements. The practical part is concentrated on the implementation of the application development. It shows how theoretical knowledge can be used in practice.

The information in this study can be used by developers with different levels of skills. Experienced programmers can get some interesting information about user interface frameworks and some database programming features if they do not meet them yet. For the beginners it can be useful for understanding the web application and the data structures. However, this studying work explains detailed principles of the web-based Java programming.

**Subject headings, (keywords)**

Software development, Java, web application, user services

<table>
<thead>
<tr>
<th>Pages</th>
<th>Language</th>
<th>URN</th>
</tr>
</thead>
<tbody>
<tr>
<td>65</td>
<td>English</td>
<td></td>
</tr>
</tbody>
</table>

**Remarks, notes on appendices**

**Tutor**

Matti Koivisto

**Employer of the bachelor’s thesis**

Matti Koivisto
CONTENTS

ABSTRACT .................................................................................................................. 1
CONTENTS .................................................................................................................. II

1 INTRODUCTION ....................................................................................................... 1
1.1 BUSINESS ON THE INTERNET ........................................................................... 2
1.2 TRADE PLATFORM ............................................................................................... 2
1.3 AIMS OF THE STUDY ......................................................................................... 2
1.4 THESIS STRUCTURE ........................................................................................... 3

2 DATABASES ............................................................................................................. 4
2.1 DATABASE DEFINITION ..................................................................................... 4
2.1.1 DATABASE MANAGEMENT SYSTEM ......................................................... 4
2.1.2 DATABASE MODELS ..................................................................................... 6
2.1.3 DATABASE STRUCTURE ................................................................................ 9
2.2 MySQL .................................................................................................................. 10

3 WEB PROGRAMMING .......................................................................................... 11
3.1 JAVA ...................................................................................................................... 11
3.2 STRUCTURE OF THE JAVA PLATFORM .......................................................... 12
3.3 MAIN CONCEPTS OF JAVA ................................................................................. 13
3.3.1 OBJECTS ........................................................................................................ 13
3.3.2 CLASSES ........................................................................................................ 13
3.3.3 VARIABLES AND DATATYPES ...................................................................... 15
3.3.4 BASIC OPERATORS IN JAVA ....................................................................... 15
3.3.5 FLOW PROGRAM OPERATIONS ................................................................... 16
3.3.5 INHERITANCE ............................................................................................... 16
3.4 JAVA PLATFORM EDITIONS ............................................................................ 17
3.5 JAVA ENTERPRISE EDITION .......................................................................... 18
3.5.1 THE ROLE OF THE SERVER APPLICATION .......................................... 18
3.5.2 MODEL-VIEW-CONTROLLER STRUCTURE ........................................... 19
3.5.3 COMPONENTS OF THE JEE ....................................................................... 19
3.6 USER INTERFACE ............................................................................................... 22
3.6.1 JSF .................................................................................................................. 22
3.6.2 PRIME FACES ............................................................................................... 22
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.6.2</td>
<td>DAO METHOD</td>
<td>49</td>
</tr>
<tr>
<td>7.6.3</td>
<td>SESSION MANAGEMENT</td>
<td>49</td>
</tr>
<tr>
<td>7.6.4</td>
<td>CONTROLLER</td>
<td>49</td>
</tr>
<tr>
<td>7.6.5</td>
<td>LOGIN FILTER</td>
<td>49</td>
</tr>
<tr>
<td>7.6</td>
<td>PERSONAL ACCOUNT</td>
<td>50</td>
</tr>
<tr>
<td>7.6.1</td>
<td>USER ACCOUNT INTERFACE</td>
<td>50</td>
</tr>
<tr>
<td>7.7</td>
<td>USER VIEW AGGREGATION</td>
<td>52</td>
</tr>
<tr>
<td>7.7.1</td>
<td>USER INTERFACE</td>
<td>52</td>
</tr>
<tr>
<td>7.7.2</td>
<td>MODEL</td>
<td>53</td>
</tr>
<tr>
<td>7.7.3</td>
<td>CONTROLLER</td>
<td>54</td>
</tr>
<tr>
<td>7.7.4</td>
<td>DAO METHOD</td>
<td>54</td>
</tr>
<tr>
<td>7.7.5</td>
<td>USER AGGREGATION TEST</td>
<td>54</td>
</tr>
<tr>
<td>7.8</td>
<td>MESSAGING</td>
<td>55</td>
</tr>
<tr>
<td>7.8.1</td>
<td>USER INTERFACE</td>
<td>55</td>
</tr>
<tr>
<td>7.8.2</td>
<td>MODEL</td>
<td>55</td>
</tr>
<tr>
<td>7.8.3</td>
<td>CONTROLLER</td>
<td>56</td>
</tr>
<tr>
<td>7.8.4</td>
<td>DAO METHODS</td>
<td>56</td>
</tr>
<tr>
<td>7.8.5</td>
<td>MESSAGING TEST</td>
<td>56</td>
</tr>
<tr>
<td>7.9</td>
<td>PERSONAL INFORMATION EDITING</td>
<td>58</td>
</tr>
<tr>
<td>7.9.1</td>
<td>USER INTERFACE</td>
<td>58</td>
</tr>
<tr>
<td>7.9.2</td>
<td>FILE UPLOADER</td>
<td>59</td>
</tr>
<tr>
<td>7.9.3</td>
<td>RESULT</td>
<td>60</td>
</tr>
<tr>
<td>8</td>
<td>CONCLUSION</td>
<td>62</td>
</tr>
</tbody>
</table>
1 INTRODUCTION

Contemporary developed computer technologies change the life of humanity, and the world of computer Internet network is very popular around the world. Nowadays, when the rhythm of life is very busy, the necessity of visiting many places, also in different states and countries, communicating with huge number of business partners and purchasing different kinds of products during the day is usual for most people. They expect to get services extremely quickly and without additional overhead. In this situation offering products, making purchasing operations and establishing communication between suppliers and their customers using the Internet are very convenient ways to get efficient results. More and more information is put into information systems. Information technologies provide new methods to the organization of business and different services. They also require more advanced organization of data, which is necessary to prevent the loss and wrong allocation of information.

All services, which are related to transmitting information, can be done through the Internet. These are financial, consulting and booking services and many others such as communication processes like guest books, social networks, Internet telephony, video conferencing and email. The advantages of providing services through the Internet are the possibility of direct interaction with consumers, low price, and fast implementation.

The requirements of the software quality have also increased dramatically. The quality of the software includes reliability, scalability and performance. But, it is not limited by these key aspects, because convenient user interface and advanced capabilities, for example extensibility and possibility of efficient management of the data and settings of the application by customers, are also important.

There are many frameworks, which are implemented by independent parties, and by using them, we can establish the management and user interface of applications in way that is more efficient. On the other hand, it strongly binds developers to these frameworks and updating and changing applications to make them capable with the newest versions of platforms sometimes require additional work.
1.1 Business on the Internet

It is not a secret that the number of Internet users increases every moment and the Internet audience already reached huge sizes. Nowadays, the majority of people get news and information they require through the world wide network. This powerful tool interconnects humanity without geographical limitations. According to these facts we can suppose that advertisements and selling goods through the Internet increases the efficiency of business productivity. Bill Gates said: “If your business is not on the Internet, your business will be out of business” (Gates, 2008). In contemporary life this phrase takes an extremely important role, because doing business this way has already become a usual thing and most of businesses have systems for providing information and for purchasing their goods. If a company escapes using these technologies, it will be an outsider and loose competition and customers, because the goods will not be provided for the wide audience.

1.2 Trade platform

Internet trade platform is a very good example of a tool for establishing sales through the Internet. This type of system is expected to be used by both suppliers and customers. According to the roles in this kind of system users get opportunities to place all relevant information about the items for sale to the platform. After that it can be retrieved by potential consumers in convenient ways including brief and advanced searches. Users’ accounts can have different roles, depending on their requirements. For example they can be sellers or purchasers. Trade platform is an information system which supports deploying data about items, convenient the retrieving of this data, submitting feedbacks, establishing of bidirectional communication and order processes between business partners. Financial transactions should be made through other existent services.

1.3 Aims of the study

This study contains theoretical and practical or establishment software parts. The first target is creating a solid theory background for the further development. I will study methodologies of storing data in the web with databases and the Java object-oriented programming with the
web-optimized edition of the Java platform. In other words, I will describe information which is needed for establishing software for communication and business purposes.

In the practical part of this study I will concentrate on the implementation of the software of automated trade platform system. I will build server based application for the company which name is not mentioned here according to the non-disclosure rules. I will work with my college Anatoli Shakhov. My role in this project relates to the partial implementation of the application which will be responsible for establishing communication between users and the management of accounting data, in general: saving, storing and retrieving personal information in the system. Mr. Shakhov will work on the development part which relates to maintenance of the products data in the system. The working prototype should satisfy contemporary requirements of reliability, scalability and expendability.

1.4 Thesis structure

I will begin my study with a data storage or databases overview. The second chapter examines database technologies and analyzes their working principles. The third chapter concentrates on the web programming and the logical management of web applications based on the Java object-oriented language including appropriate frameworks, platforms and integrated development environments. The detailed planning of the implementation is presented in Chapter 4. The requirements which are needed for the development are described in Chapter 5. The design of the developed application is explained in Chapter 6. After the theoretical researches I will build the required features for the web application according to my role in the project and explain all the steps of the development process in Chapter 7. The last chapter of my thesis includes the summary and the final conclusion for this study.
2 DATABASES

The main role of our case application is processing the data. Database is one of the most important parts of this project, because it is responsible for holding of the data. The next sections examine the key parts of databases in more detail: how they are structured and managed.

2.1 Database definition

Nowadays we use computer-based databases many times every day. Often we do not pay attention to them and sometimes do not even know about their existence. Nevertheless, they are met in all information technology systems which relate to providing services for people. For example, we face databases when making phone calls, checking email, paying in a shop by credit card and in many other situations. We cannot imagine human life in the 21st century without continuous interaction with databases.

In information and automation systems data means information sources which are structured in exact form. Storing and processing this information are required and the use of common data formats allows doing it in the efficient way. Contemporary digital database is a complex collection of information entries which are structured by table allocation with rows and columns inside. Records in the database are interconnected with each other and can be used by different applications and users according to permissions and purposes. (Nirupma 2008, 1.)

2.1.1 Database management system

Database management system (DBMS) is a set of data and software which takes responsibilities of manipulating this data. A DBMS allows administrators or users full central control over databases including creating them and the tables inside, saving, retrieving and editing data. All these capabilities and features can be done directly or through additional software such as user-friendly applications. The management systems of databases allow avoiding the needs of creating multiple copies of same files for different processes. According to this we can say that storage resources are not wasted. Also management software itself includes a big
number of technologies which bring convenient and efficient maintenance processes. (Nir-upma 2008, 1.)

DBMS software can vary depending on the exact type of the system. This complex system consists of subdivisions and there are basic common parts which perform the maintenance for databases. As Figure 1 shows each component is responsible for its own task. At first, DBMS software takes the role of the middle part between the operating system and the database to allow interaction between them. There are three different components for translating management data. The type of component depends on how data was processed. When it comes to the application, Data Manipulation Language (DML) preprocessor translates these programmed commands to the program object code. If information is received from queries, the query processor converts them to instructions. The Data Definition Language (DDL) compiler compiles the statements of database schema to management instructions. The database manager component is responsible for receiving instructions from the query processor and sending requests to the database according to them.

The database manager also consists of subdivisions. I will list and explain them: authorization control takes care of restrictions and permissions for access and manipulations for users. Command processor executes commands of authorized users. Integrity checker verifies the existence of operation capability with integrity restrictions when the database modification is occurred. Query optimizer defines the best way of query execution. Scheduler takes a switch role and performs the execution of processes in turn without collisions. Recovery manager saves the states of the database for future recovery when failure happens. Buffer manager holds information which should be transferred. (Connolly & Begg 2010, 77-79.)

FIGURE 1. Components of DBMS
Most of the management systems use Structured Query Language (SQL) for database maintenance. With this programming language the system is able to create and setup new databases, save and retrieve information from them and restrict access to provide security in general. In more detailed view, the sublanguage of SQL named Data Control Language (DCL) is responsible for controlling the information: how it should be stored and accessed. The manipulation of information is performed by executing commands of another sublanguage named Data Manipulation Language (DML). (Wilton & Colby 2005, 11.)

Database systems can be defined by two factors: how many users should use system and where the database is located. In different situations system access can be restricted for one user or for multiple users or groups. If the database is located in a special place and clients gained access to it from different sites, it is a centralized system. The distributed database management system consists of parts which are located on different sites of the network. For improving performance the parallel DBMS is used, when a fixed number of processors and storages are used by the database system at the same time. Different kind of software is run on personal computers and servers. The same situation can be met in the database management approach where a client-server DBMS allows executing one part of software on the client or front-end side and other part is located and executed on the server or back-end side. Clients are able to make requests for relevant services to the servers. (Pallaw & Krishna 2010, 11.)

2.1.2 Database models

The database model defines the exact way of data structuring inside it. In other words it means the form or exact kind of container for handling data. The database model describes organization of information and how it relates to other pieces of information.

The base or background kind of models is the entity-relationship (ER) model. It is based on objects or entities. All objects are represented so that each of them looks like an individual part. These entities can have their properties which describe their nature and identity. The relationship between entities describes exactly how they relate to each other. It can vary. For example, the relationship between two objects is known as one to one. Also, the entity can relate to many others or many entities can be associated to one. Sometimes we can meet group relationships, when many objects are associated with other groups. The mapping of relationships is also known as cardinality. This term identifies how many entities in one set relate to
entities in another set. As Figure 2 illustrates, entities from entity sets can have different types of relationships. (Pallaw & Krishna 2010, 11.)

![ER Diagram](image)

**FIGURE 2. Relations of ER DBMS**

According to Elmasri & Navathe (2011, 74) there is an enhanced entity relational model (EER). It includes all the features of the initial entity relationship model, and it also supports super and sub classes. A superclass can contain one or more subdivisions named subclasses. The entity of a subclass is also the same for superclass. For example, we can store information about a couple of employees. Each of them will be defined as a subclass, and the employee, which unites all of them, will be the superclass.

The relational DBMS is the most common type of management software for databases. In this model data is allocated in tables and the relationship is established between data through these tables. In normal allocation the columns define information specialized patterns of data, also known as attributes, which are used for all rows inside the same table. Columns should be named to identify logically which kind of information will be written, for example, identification numbers or personnel details such as “name”, “address” etc. These definitions are also named as properties or attributes. Also columns have to be set for exact data type to define which data is compatible to be stored inside the column. There are common standards for types of attributes in SQL. Let us list and describe them. To store a string of characters “CHARACTER (n)” or “VARCHAR(n)” types are used. The difference between them is just that first one has a definite length of characters defined by number (n), the length of the second can vary just limited by the maximum number. The same rule is used for binary strings:
“BINARY (n)”, “VARBINARY (n)”. Strings of numbers can be integers: “SMALLINT”, “INTEGER”, “BIGINT” differed by the size of lengths. Sometimes numerical strings have to consist of fractional numbers, for example: “DECIMAL (p, s)” has exact patterns for numerical value before decimal (p) and (s) after dot. For date and time there are “DATE”, “TIME”. Also, there is the “TIMESTAMP” attribute which holds date-time value up to seconds. In addition, in databases it is possible to store arrays, intervals and XMLs. Rows inside a database table are used to allocate information about special objects also known as entities. Each entity has its own row. This practice allows storing information of objects separately, and therefore, prevents chaotic mixing of entries. In addition, entities can have relationships which describe how entities communicate or interact in logical way with each other. Figure 3 shows the example of the relational model. (w3schools 2015.)

![Relational model](image)

**FIGURE 3. Relational model**

Although relational DBMS are the most popular today, there are some alternatives to it. The object-oriented model bases on the conceptual aspects of programming, particularly, on the encapsulation and inheritance of objects where data is stored in classes and specified with variables for description. The object-oriented database structure is illustrated in Figure 4. Also, a hybrid of the relational and object-oriented models exists which includes features from both models by establishing relations between objected databases. (Pallaw & Krishna 2010, 12.)
The hierarchical model consists of entries and parent-child relationships between them, the parent can have multiple children, or it is also known as one to many relationship (1: N), but the child can only have one parent. The hierarchical database model is shown in Figure 5. The network model involves link-based relationships between data entities. It removes limitations of (1: N) communication. (Pallaw & Krishna, 2010, 14.)

Contemporary database management systems are based on three-level architecture which meets database standard design definitions. Also, it is known as Standards Planning and Requirements Committee architecture defined by the American National Standards Institute (ANSI-SPARC). Internal level describes the physical storage of data in a database: how and where it is stored. Also it represents the interaction of the management system and data.
DBMS manages an operating system which controls the physical allocation and structuring of data on storages. It also includes compression and encryption itself. External level represents the user view of the databases and how users interact with them. At this level just that part of the database which is used or predicted for maintenance is shown to users. Views can vary from each other according to customized setups. This means that information can be shown in different ways depending on regional standards. The logical or conceptual level establishes communication or mapping between external and internal levels. It holds a logical representation or structuring of the data which includes all parts of the database, supports all customized views and points them to the internal level. (Singh, Shio 2011, 59-61.)

2.2 MySQL

In implementation of the project relating to this thesis I will use a MySQL server as the database management system for the maintenance of data and I will next describe this technology. According to Murphy and Cabral (2009, 3-4) MySQL is one of the most popular database management systems nowadays. A big number of web systems executing MySQL servers and the contemporary Internet world would look differently without MySQL.

MySQL databases have relational structure. The software of this database management system is free except for enterprise versions. People can use the MySQL software for free and also download its source code and make it more suitable for their purposes, because this software is open source. This DBMS is reliable, easy to use and it utilizes hardware efficiently. The management of MySQL is performed by injecting SQL scripts to the DBMS. Additionally, there is wide range of software with GUI which makes the management process extremely easy. MySQL Workbench is a good example of this kind of software. With Workbench we can create, modified databases and tables inside them as we want according to functional requirements. Modification of the databases includes complex process of modifying containers for the data, defining their allocation, types, behaviors and relations between each other. (MySQL 2015.)
3. WEB PROGRAMMING

Web software development is a complex system which involves many parts. Subdivisions of the web development are logically structured according to the purposes of developments. In general, web software should provide efficient functionality of the data processing and a convenient user interface. There are different technologies for establishing data processing. They are described by programming technologies. The code of the software management part can be represented and deployed by different ways depending on exact technology. There are technologies which were initially created for the web-based programming such as PHP. Its source code files are directly deployed to a server and provide management of the user interface content. Another way of web-based programming involves creating more complex applications which consist of many modules and are packaged to be deployed on servers for management purposes. Usually this technology is based on object-oriented programming including appropriate languages and frameworks. There is a wide range of technologies for the development of the user interface. They include markup languages such as HTML or XHTML and style configuration languages, for example CSS.

3.1 Java

According to Todd (2004, 1-3.) Java is the cross platform object-oriented programming language for the development of the client and server site applications. Its author is James Gosling. This technology has a lot of advantages such as high performance, convenient development process, automatic memory utilization, security and many others. The Java programming language was the product of the Sun Microsystems. This company was founded in 1982. It produced different kinds of software and hardware. In 2010 it was purchased by another IT development company called Oracle.

In our world development ideas do not appear without purposes for them and without developer’s motivation to create something. The same situation was with Java. Initially, the so-called Green project was created. The main purpose of that project was development software for the management of home devices. During the first steps the C++ programming language was selected for the development. But at that time this programming language was not completely suitable for the project, and in 1991, James Gosling began to develop a new
language. The name of the new programming language was Oak, like the name of the tree which grow near the office.

New technologies required suitable hardware, and the team which should develop that hardware joined with the Grean project. The hardware development project was named Star-seven. In 1992 the prototype for the hardware, a graphical user interface and the prototype for the device management system were released. The new programming language was renamed Java, brend one famous coffee trade mark. The name was changed, because Oak was the name of the existing programming language. After that Green project became a separate part of the Sun Microsystems, but project was finished two years later. In 1994 the Java compiler and Internet browser were developed. One year later the Java language was introduced officially. The Java Development Kit (JDK) is the environment for the development of Java applications. The first version of JDK was released in 1996. It had a lot of drawbacks. There was a small number of libraries and it was slow. However, for the first development it was great work. Since that time then the number of libraries increased rapidly, optimized versions of the Java platform appeared according to specific development requirements. The principle of reusing same methods and combining them for the development works well. Nowadays the Java API has a big number of ready classes in the libraries, and it brings great improvements to the development process. It becomes faster and more efficient. (Oracle 2015.)

3.2 Structure of the Java platform

The platform consists of two key components. The first one is the Java application programming interface (API) which consists of prebuilt libraries or classes. These classes are used as instructions for executing exact tasks, such as establishing communication, reading the output, injecting the input, transmitting and many others. This feature provides the use of ready-made methods by calling them instead of creating new ones from scratch. For example, in the real world we do not need to build a car again and again, when we need to make a trip. Required actions are starting the engine and driving. The second part of the platform is the Java Virtual Machine (JVM). It executes instructions which were written and provides the results of the application working process. Also the compiler is involved in the process for preparing a byte code for machines. (Todd 2004, 3-5.)
The general working principle of Java includes four main steps. At first a developer writes the source code which is considered as an instruction file with the “.java” extension. This document is understandable for programmers, but not yet for machines. After that the code is checked for the appearance of mistakes. Then, if the code is valid, the compiler generates the file with the “.class” extension which includes byte code for machines. In the end the Java Virtual Machine reads the byte code and provides the understandable code for relevant platforms, and program is run. (Sierra & Bates 2005, 3.)

The Java programming language has strict rules of typing. Integrated development environments according to a compiler provide notifications of written mistakes in the source code. These mistakes prevent the code from compiling. It helps to define mistakes, fix them and to avoid additional errors, to make the platform more robust. Also the development process becomes more efficient. (Todd 2004, 6.)

3.3 Main concepts of JAVA

Understanding the principles of programming Java-based applications is implemented for further development. Therefore, I next describe how programming with this language can be done. In this section I explain main concepts of the Java platform, their purposes and structures.

3.3.1 Objects

The real world consists of physical objects which have their properties and behaviors. There is the possibility to represent prototypes of these objects in object-oriented programming. An object-oriented programming language focuses on the properties or states and behaviors of objects. (Todd 2004, 4-5.) For example, now there is a cup in front of me and it has some properties, so it is ceramic, white and small. Let’s talk about behavior. It is possible to store liquids inside it and drink them.

3.3.2 Classes

A class represents the overview of an entity. Objects can be grouped into classes according to their properties. For example, a class which is known as “Cup” defines that it is a cup in general and nothing else. The object defines that it is a cup with exact shape, color, size etc.
A Java source code consists of classes. Each of them is a file with the “.java” extension. It takes less time to find the required class, if the classes are separated according to their functionalities. For convenient allocation and finding these parts they are organized into folders also known as packages. The name of the package shows the path to an exact folder and usually represents the purpose of use. As it was already defined, the class holds its members and methods. The code of the class should be placed between curly braces. At first, the java virtual machine accesses the class, and after that it begins to operate with methods which are inside this class. The application should have at least one main method to define the beginning point. The execution starts from this main method. Methods represent statements and define their behavior. In other words, the method is the exact instruction for the program. It tells the system what it has to do. Figure 6 illustrates the structure of the class in java. This class has method, which has variable definition, and also it has the instruction to show the string on the display. There are three types of classes. They can be public, private or protected. These types are defined by access modifications. If there is no modification, the methods of this class are available inside the same package by default. Other classes can access a public class, even if they are located in another package. If a class is declared as protected it means that only classes which inherit some properties from it can gain access to this class. A private class does not share anything with others. The content is available inside private class. Methods also have these modifications to restrict access to them in a more flexible way. (Sierra & Bates 2005, 7-13.) The simple code example of the class is shown in Figure 6.

```java
package com.example;

/**
 * @author Mikhail
 */

class PrintClass {
    public static void main(String[] args) {
        int i = 3;
        System.out.println("The value of our variable is: "+ i);
    }
}
```

**FIGURE 6. The class structure in Java**
3.3.3 Variables and datatypes

To keep the data Java programming language uses variables. The variable is the exact name of the memory reserved for allocating of the data. The variable should be declared to the datatype for which it is used. There are three types of variables in Java. The variable can be local. It means that it is declared locally in the method. Another type is instance. It refers to the declaration of the variable inside the class, but not inside the method. The static variable means that the single copy of it is shared for all the objects in the class. The data types there can be primitive and non-primitive. One of the primitive data types is a Boolean type which can hold just two different states of the variable. It can be true, but it is false by the default. The size of Boolean is one bit, because it keeps the value “0” or “1”. Another primitive type is numeric. It includes a character subtype and integral type. The character type is used to keep characters. Its default size is 2 bytes. The Unicode is used in the character variables and its default value is “\u0000”. The integral subtype contains integer types and floating-point types. The integer types are: byte, short, int, long. All of them are used to hold integers and differ from each other just in their sizes. The sizes of byte, short, int, and long are 1 byte, 2 bytes, 4 bytes and 8 bytes respectively. According to these sizes, variables can hold integers which are in the range of these sizes. The floating-point type is required to keep the numeric values which contain points. The float and data types have sizes 4 and 8 bytes respectively. (Javatpoint 2015.)

3.3.4 Basic operators in Java

Variable operations are used to process and manipulate variables in Java. One of basics operations is variable assignment. At first the name of the variable is declared to the variable with exact data type and then with the operator “=” the value is assigned to the variable. It is possible to assign parameters of one variable to another. In this case the variable reading operation is used. To implement this operation we just need to assign the name of the variable which tends to share its value with the “=” operator to the variable. Another type of operations is the arithmetic operation. It is used to perform calculations of variable values. This type of operations includes addition (is also used to concatenate strings), subtraction, multiplication and division. The variable can be also assigned to value which relates to some object. In this case we need to instantiate the object of the class. To implement an instantiation the key “new” is used. For example, if we need to instantiate object of the class “Teacher”, and assign it to the variable with the name “myTeacher”, we need to write the following code:
“Teacher myTeacher = new Teacher();”. The bitwise operators “AND”, “OR” and “XOR” are logically similar to multiplication and addition respectively. They are used to operate with bit values, but are also acceptable for other types of variables. Operator “AND” in Java has the “&” symbol, and if it is between two values, it returns multiplication of them. If there are Boolean values, the true output is possible when both of them are true. The operator “OR” has the symbol “|”, returns the addition of two values and if just one of Booleans is true, the result will also be true. The operator “XOR” is used with the symbol “^” and has the similar behavior as “OR”, but the result of Booleans will be true, if both values are similar. (Oracle 2015.)

3.3.5 Flow program operations

The “if statement” is used to define if the part of code should be executed according to the conditions which take place in the program process. There is also an optional part of this statement which defines an additional part of the code which has to be executed, if the declared condition is not met. There are other methodologies of the program flow maintenance, named loops. One of them is “for loop” and it allows the program to execute the part of the code repeatedly, if some conditions have occurred. In this case the interaction with the condition is performed inside the definition part of this condition. Another type of the loop is “while loop”. It also checks the condition, and if it is present, the part of the code is executed. The checking can be done in two ways. The code which is potentially predicted for executing can be entered to the loop before or after the occurring of the condition is checked. The normal “while loop” checks the condition before the execution process, and if the result is false, it will not run the code. The “do while loop” at first moves the code to loop and after that checks if the condition is present. In this situation executing is performed at least once. To choose between multiple parts of the code for executing the “switch statement” can be used. This statement has cases, and each case contains the code which is potentially should be executed. If the value of the case matches the value which was created to define condition, the code in this case is executed. There is also a default case. It is executed if no matches in listed cases are present. (Eclipse 2015.)

3.3.6 Inheritance

This feature allows using methods from one class by another. The class which shares methods is defined as a parent or superclass, the class which inherits these methods is known as the
child class. It is needed to simplify the code and avoid multiple creating the same methods. Sometimes classes need the same set of methods. It is very efficient way to keep these methods in one place and then share them for others. A child class should be marked in declaration with the word “extends” and with the name of the class from which the methods should be received. Also child classes are able to rewrite existing shared methods and have their own additional methods. The modification of inheritance, named implementation, is used very often. It is based on the creation of abstract methods and methods which implement them with their properties and functions. (Oracle 2015.)

3.4 Java platform editions

There are different modifications of the Java platform, and each of them is optimized for particular kind of the development. To go to the exact platform consistently and explain the logic of the technology choice I need to analyze the alternatives of the Java platform. In this section I provide brief outlook on the Java editions.

Java SE
The Standard Edition of Java includes virtual machine, development tools and wide range of libraries in its API. The API of this edition provides basic functionalities for desktop application development. (Paololatella 2015)

Java EE
Java Enterprise Edition is optimized for the development of the crossed-platform web-based applications. It is based on top of Java SE and has many libraries from the standard edition API, but also it includes additional libraries. Java EE environment provides services for web applications deployment. (Java 2015)

Java ME
The Micro Edition of Java is optimized for mobile developments. It is built on top of the Java SE platform and utilizes some libraries from the standard API. This version includes tools which are more specified for the software development for custom mobile devices. (Oracle 2015.)
3.5 Java Enterprise Edition

I will develop the web-based application with Java EE platform. And to make solid educational background for further developments based on this platform, I go deeper exactly into this version of Java. In this section I introduce this edition with its main functional concepts.

3.5.1 The role of the server application

At first I want to describe purposes of the web-based application. It helps to understand how, where and for what the appropriate technology is used before going deeper to the Java EE specific properties. In this section I explain how the client-server communication is performed and which role the web application has in this process.

The servers have resource information. The server side role is holding this information and allowing users make requests for this information. According to permissions and allocation of resources, servers reply to users and share requested resources. Users and user applications, such an Internet browser, are located on the client side. These applications allow users to make requests to servers and get back results. (Siera&Bates 2008, 4-5.) The server web application takes care about providing data processing services, communicating with the user interface and with data storages. Figure 7 illustrates client-server communication, the user inputs search patterns to the application interface in the browser or submits web, then the server application receives the request, and process it.

---

**FIGURE 7. Client-Server communication**
3.5.2 Model-View-Controller structure

The Model-View-Controller (MVC) structure is the most popular structure in the object-oriented software development. This structure provides separated functional parts of the application. The data object model represents real objects in the application and holds all the data which relates to the properties of the objects. The view has presentational sense, this part is responsible for the user interface and how the data is shown to the users of the application. The controller element encapsulates the business and the management logic of the application. Figure 8 illustrates the structure of the MVC application model. In the simple imagination, the controller interacts with the view. According to these interactions the controller manages the model. The model receives inputs from the view, holds the objects data and updates the view with the relevant data. (McGovern, et al. 2003. 9-11)

FIGURE 8. MVC structure

3.5.3 Components of the JEE

There are three subdivisions in the Enterprise Edition of the Java platform. The first group is deployed and executed on the server. It includes general web components and Java classes for the logical management of the application. Also, there are classes available with extended functionalities in this specification of Java. They are named Enterprise Java Beans (EJB). The second part is also deployed on the server, but it is loaded and executed on the client machine. It consists of normal or extended Hyper Text Markup Language (HTML) pages which are
responsible for providing user interfaces of the application. The third subdivision includes JVM which compiles the code of the application. (Cole, et al. 2011, 6.)

**Resource adapters**
These components can be located outside the application, but they should be installed on the server and executed to provide interaction between Java and non-Java applications. Usually, resource adapters are used to make communication with data sources such as Customer Information Control Systems (CICS) or DBMSs. (Cole, McChesney & Raszka 2011, 9.)

** Servlets**
Servlets are small applications which are run on servers. The data of the web-based Java applications is processed by a dynamic way. It means that the structure of the application manages the dynamic content. The servlet is the web component which is interpreted as a Java class of the server application. The purpose of servlets is generating the data. In complex server-based applications each servlet has its own responsibilities, for example, operating with inputs from the client-side, calculations, interaction with data storages and the storage of the current data. Servlets provide communication with the client-side by exploiting Hypertext Transfer Protocol (HTTP). The main idea of servlets is providing functional services. (JCP 2015.)

**Enterprise Java Beans**
These are server-side components which provide concentration of the attention on business logic of the application. Services, such as transaction management, security, messaging, session management, obtaining of data sources and many others can be provided with EJB components instead of creating them manually. EJBs take similar responsibilities as basic Java classes, but development with EJBs brings more capabilities and makes the system more flexible. These components form the new level of the reusability of the code in Java. One component can be used in different places without changing its source code. All properties of the deploying can be setup inside XML configuration files, which describe how components should behave in different cases. All functionalities which are applied to the component are also reusable. By other words the component can be customized during the deploying on a server. (Reese 2011, 9-11) Figure 9 show the code example of the Enterprise Java Bean.
Containers

According to Cole, McChesney & Raszka (2011, 6) containers provide management and runtime support for components which are inside them, these components are isolated inside containers, but communication between them should be performed. Containers allow access to Java libraries which are called by components when it is required.

According to Pilgrim (2013, 12) Java EE has four main runtime environments they are also named containers, because their responsibilities are related to the providing lifecycle management of other components and keep these components deployed on the server. The first container takes care about enterprise beans of Java (EJB). Another container manages lifecycles of servlets and management beans, it is named the web container. The third container is responsible for client-side applications. Applets have their own container for the lifecycle management. Figure 10 illustrates the main containers of the Java EE platform.

![Java EE containers](image)

**FIGURE 10. Java EE containers**
3.6 User interface

One of the most important key aspects of the high qualitative software is the convenient and intelligent user interface. Usually the interpreting of the information is based on markup languages with additional stylizations and functionalities. With extendedness of markup languages for example with XHTML appears possibility to improve developments of the content presentation. There is a set of the frameworks which have optimized tools for the interface building. In this section I explain the most popular frameworks which I choose for the further developments.

3.6.1 JSF

Java Server Faces Is the Java-based framework which provides for developers a huge amount of tools for User Interface developments. Instead of the manual creating elements on the web pages developers can just insert ready components. These user interface components are supported by servers. It simplifies the development process because the system which is developed with JSF is based on reusable standard UI components. (Oracle, 2015.)

3.6.2 Prime Faces

This framework can be considered as the Java Server Faces modification. It has a huge number of Graphical User Interface components which brings the efficient utilization of the Extended Hypertext Markup language. All of Prime faces components are located in the central repository which makes easier their accessing. Prime Faces development team provides detailed documentation for all components with examples and code sources. It makes the development process more efficient and convenient. (Prime Faces, 2015.)
4. PLANNING THE IMPLEMENTATION

It is well-known fact that good planning of the working process makes it more efficient. Exact definition of goals and ways of achieving them are very important steps of any projects. Consistent implementation of tasks is critical part in software development projects. In this chapter I tell about implementation steps of the project and provide information about development methodologies.

4.1 Development methodologies

There is a wide range of ways of any product development process. Nowadays methodologies, which describe these ways, exist. In this section I will explain the most popular of them.

4.1.1 Waterfall

Waterfall model represents consistent way of the software development process. It begins from analyzing of requirements which include goals definitions of the project. When all targets are clearly defined the technical planning takes place. During the second step the structure planning of the project and defining the development solutions are defined. When the developer knows what should be done how the structure should look the technical development process begins. It includes finding solutions for achieving defined goals. After the achieving targeted goals, the developer verifies technical aspects of the created product. During this step the testing of the software and fixing problems should be performed. Some kinds of software need supporting during the exploiting process. Maintenance can include updates implementations and providing technical documentation. (Water-Fall model, 2015)

4.1.2 Scrum

This methodology describes the consistent way of the team product development. The whole development process is separated to small parts which are implemented in logical turn. The developers get tasks and have to implement them in defined period. When this time period is end they provide progress report of done work at the meeting. The project owner comments things which have done, gives recommendations of required modifications and discuss with the development team the next steps. This methodology brings efficiency of the implementa-
tion and saves the time because it is easier to change the part than the whole work. (Scrum, 2015.)

4.2 My methodology choice

In my software development I combine these two methodologies. At first, I analyze my goals and responsibilities. Then, I plan and prepare tools which are required for the development and plan the structure of the application. After that I begin consistent software development. But there are smaller subdivisions in my programming tasks and meetings are take place every week. They are needed to keep the contact with the project owner, to provide progress reports and to improve the development, when it is possible.

4.3 Steps of the project development

In this section I explain my development process step by step with all details included. It will describe the general image of the consistent software development process. The first step of the project development includes analyzing requirements. When I refer to the requirements I mean tasks which should be implemented and tools which are needed to perform these tasks.

My goals for the project and the work statement are provided by the project owner and clearly understandable. I have to implement user support services. In detailed view with these services users should be able to register in the system and login, also there should be possible to modify their personal information and exchange messages with each other. During the development process I use my knowledge background in programming and if it is needed I use the Internet to find solutions for completing tasks.

Then I prepare the environment and install all the required tools for further development. I need to install the database server for data allocation and management, the Integrated Development Environment (IDE) for writing my code and Java API with building tools which are grouped in JDK. To run web applications the local server is required. During the installation process the compatibility issue of products should be paid attention to.
During the second step I design the project including defining the structure of the application. At first, I plan what kind of data should be stored and processed for providing user services. Then I define how the functional part of the application should be created. The third step includes building the application, creating features, which relate to my development part, and testing the created modifications. As it was already defined that it is teamwork, the sharing of the source code has to take place between developers regularly. The implementation also includes meetings with the project manager every week. These meetings involve providing progress reports and discussing the ongoing work.

When all features according to my responsibilities are implemented the final presentation of my part of the project to the project owner takes place. Figure 11 illustrates the steps of my development process.

FIGURE 11. Steps of the implementation
5. REQUIREMENTS

The requirement analyses contain both requirements for the final product and for the tools and working environment. The requirements of the final product are defined by the product owner, and I shortly described them in the previous section. Here I concentrate on tool and environment requirements.

To perform any engineering task in a professional way all the required tools and environment should be prepared beforehand. Software development is not an exception to the rule. Defining the equipment which is predicted to be used is important. This is needed to maximize the efficiency of the work and to avoid the wasting time during the development process.

When I talk about the development environment for web-based Java application I mean tools which are needed to create and modify its source code and the software which provides the execution of my application. The services which have to be provided by the application are related to the data processing. For the data storage purpose I install the MySQL server. To develop the Java base application I need to install the Java Development Kit (JDK) which requires Java libraries for the development and running the application on the Java platform. To write the code I need the Integrated Development Environment (IDE). I and my college agreed to use NetBeans IDE for the development. The web application should be deployed and run on a server. For this purpose the Glassfish web server is chosen. The creating of the empty application also can be considered as the preparation step, because immediately after creation it does not have the source code and can be compared with a white piece of paper for further writing. In the following sections I explain step by step how I create the development environment and the project.

5.1 Installation of the development environment

At first, I download and install the MySQL server. There is a version of the MySQL server which is assembled into one installer package. The installer is 32 bit, but it contains both 32 and 64 binaries, which makes it compatible with both versions of the CPU. Figure 12 illustrates the MySQL downloading page.
When we run the downloaded file we need to accept the license terms. After that the installation process will start. There are different types of the setup. By default the developer type is chosen. We keep it and go to the next step. The developer type includes products for the MySQL development purpose. With MySQL server database connectors are provided in this case. The graphic user interface software is also installed for the convenient management of the server and databases, It is named MySQL Workbench.

MySQL uses the 3306 TCP/IP port number for running. The checkbox “Open Firewall for network access” should be checked to avoid blocking MySQL by the Firewall. Figure 13 shows the network configuration of the MySQL server.
The next step is setting a security password for the accessing management of the server. By default the root password is used. It is enough for management, but there is the possibility of multiple customized users in the system. After that, we need to confirm our configurations and finalize the installation process.

Finally, I start the Workbench to test the connection with the MySQL server. During connection and server starting steps I need to provide the password to gain access. It was created during the installation. Figure 14 illustrates Connection dialog.
Figure 15 shows the management page of the MySQL server in the Workbench. The message log shows the status of the server.

![MySQL Management Page](image)

**FIGURE 15. Administration page of MySQL**

Now I need to download and install JDK. Its latest version is available on the official site of the Oracle. During this step the choosing of the operation system and the 32 or 64 version should be paid attention to according to the CPU version. In my case the Windows 64 version is suitable. After confirming the license agreement the downloading is available.

Figure 16 illustrates the download page of the JDK.

![JDK Download Page](image)

**FIGURE 16. JDK downloading page**
After installation I should be sure that the installed component is available. So I need to check the path to the Java component in the operation system settings. I follow the way System properties – Advanced – Environment Variables and in the System Variables area check if the JAVA_HOME path entry is present. This entry should be created automatically but if it is not so I should create it manually with the path to the folder where my installed JDK is located. Figure 17 shows the JAVA_HOME path entry in system variables settings.

![FIGURE 17. JDK path settings](image1.jpg)

Then the general system path to the JAVA_HOME system variable should be checked for the appearance of the name of created variable. Figure 18 shows the configuration of the Path entry.

![FIGURE 18. General path variable entry](image2.jpg)

Now I need to install IDE and web server. I download NetBeans IDE 8.0.2 Java EE version for the Windows platform. Currently, it is the latest version. It includes web server bundles, I do not need to install them separately and installation process is simplified in this case. Figure 19 illustrates the downloading page of the NetBeans IDE.
FIGURE 19. Downloading page of the NetBeans IDE

After the completing of the downloading process I run the installer of the IDE. During this step license terms have to be confirmed, and after that I should pay attention for checking the server’s bundles check boxes to install web servers additionally. As it was already mentioned I chose the Glassfish server but if it is required the Tomcat server can be installed. Figure 20 shows the step of installation where web servers are included.

FIGURE 20. Server defining installation page
5.2 Creating the project

Initially, the basic Java web application was created by my college Mr. Shakhov. But later we decided to use the Maven building tool for the development. In this section I describe reasons of using Maven tool and explain how to create the application and test it.

Maven is optimized for Java projects and simplifies the building process. Support JAR files are represented as dependencies. Using Maven we do not need to add each JAR file to the library folder, instead of this it is possible to define them in the single file which is named project object model. Maven building tool takes care about downloading defined JARs. According to their properties they can be stored locally or leased from the central repository. Using of the central repository decreases the application weight.

To create the simple application from scratch I perform the following steps:

1) Start the NetBeans IDE and click “File-New project”,
2) Chose “Maven web application” and click “Next”.
3) Set the name and the group Id of the application and press “Next”.
4) Choose the server and the version of the Java EE platform and press “Finish”
5) Run the application

Now the application is started. It has not the source code just one web page with simple text inside it. The purpose of running is getting sure that creating process was successful. Figure 21 illustrates the output of the created application.
6 PROJECT DESIGN

Before the development of functional part I describe the general structure or the design of the application. In this section I list the data structure, parts of the application and explain their responsibilities.

6.1 Data

Normally user services are based on the personal information. This information can be divided into three groups. They are access control data, general information and communication data.

The first part is required to allow authorized access to the system. This part is based on login and password. They are created by users during registration step and used to gain access at the entering to the system step.

The second part is general information about a person which represents the person as the virtual entity in the system. It includes the name of the person, photo, and contact information. This information can be used for creating business contracts. Additionally the personal bank account data is needed for contracts, it should be private and used during generation of contracts steps.

The third part represents messages which are expected to be exchanged between users. As in the real world, message should contain information which is required to be sent and the information about sender and receiver. For this purpose the separated table in the database should be created.

6.2 Application structure

When the data which is needed for further processing by the application according to functional requirements is defined exactly I can plan how the management part of the application should be created. The application is a complex system which includes set of smaller modules. They should have the exact structure to work in a team. Each of modules should have exact functional responsibilities. It is very convenient to allocate modules with similar respon-
sibilities in the same place to separate them from others. Assigning names to them is also im-
portant aspect, names should represent modules purposes. It helps to avoid the situation when
parts are allocated chaotically and it would hard to find quickly needed parts.

6.2.1 Web pages

Web pages are required for providing graphical user interface (GUI) of the application. They
include XHTML pages and style configuration (CSS) files. With these pages the application is
able to get input from the client side and show the output.

6.2.2 Java source packages

Java classes and EJBs are grouped into packages according to their functionalities. These
packages are located under the appropriate domain name of the company for which I make the
application. I explain in details how they are structured to be able to provide functionalities of
the application according to my role in the project.

The package with named “beans” includes Java classes which directly interact with GUI and
manage the data. These classes receive inputs from XHTML pages, call methods to send the
information further and generate dynamic content which is expected to be shown as output on
web pages.

The model of the application data is represented by classes which are located in the Data
Transfer Object (DTO) package. They take care of holding the data and make it accessible for
the processing.

The Data Access Object (DAO) package is responsible for interactions with the database. The
purpose of it is establishing the communication with databases and exchange data with them.
Classes inside this package have methods which provide input/output of the data in the data-
base by injecting SQL scripts into DBMS.

Additional services of the application such as file management and common database conne-
tion are included into the “service” package. The “servlets” package provides support of the
working process of some services.
The content and access to the application should be filtered in some cases and classes inside the “filters” source package should be implemented. Figure 22 illustrates Java source packages of the application.

![Source Packages](image)

**FIGURE 22. Java source packages**

### 6.2.3 APIs and plugins

To provide functionalities of the application developers need not to create everything manually and ready-made APIs and plugins are used. They are represented with Java archive (JAR) files and are allocated in separate folders. According to the type of our project these files are named dependencies. Figure 23 shows the list of application dependences.

![Dependencies](image)

**FIGURE 23. Dependences**
Let me first to review again things which should be done. I am responsible to develop features of the application which are related to user services. These services include the user account representation in the virtual system, allowing authorized access to accounts and possibility of communication between users.

In this chapter I explain the detailed process of the application development. The main idea of this part is telling about the professional way of solving programming tasks. From one side I try to show how I use my knowledge in this area on practice, from another side I explain what problems I meet, what solutions can be found and what discoveries I make during this software development.

I decide to document the implementation step by step as it is appeared on practice. This way is logical, convenient and understandable for a reader. Also it makes clearer the whole picture of the work which is explained here.

7.1 Database development

In this section I explain how the databases can be created and modified. Developments which are described in this part are implemented with GUI of the Workbench. Names of the database with included tables and fields inside tables are given just for example, because according to non-disclosure rules I cannot provide any confidential inform about the application data. It is important to define which data is required for serving users of the system and how it should be stored.

7.1.1 Database creation

When the MySQL server is started I can begin the database development process. I make it with GUI and software injects the SQL script to the management system. The SQL script which can be used for manual database creation is shown in Figure 24.

```
CREATE SCHEMA `business` ;
```

FIGURE 24. Create database SQL script
7.1.2 Table creation

After that I add a new table to the database with their relevant fields and datatypes. The identification number of the entity which is described by storing data should be unique. It is defined as primary key. So, this value should not contain a null value. Also, it has to be auto-incremented when new records are created. The SQL script of this step is shown in Figure 25.

![SQL script of the table creating](image)

**FIGURE 25. SQL script of the table creating**

Additional tables which are needed for the development process can be created and modified in the same ways. I want to emphasize that a separated table is required for the messaging data maintenance, in which at least four columns should be present. They are the unique id of the message, identification of the sender and receiver and the content of the message.

7.2 Model

The model is required to copy and store one or many objects and redirect them from one place of the system to another. This class should implement the Serializable class. Serializable properties allow converting the object data to the bit stream and vice versa. In this class I create private variables to store the relevant user data. I can say that these variables represent fields in the database tables with one exception. This exception is the variable which defines files for uploading to the system. For example, photos are stored on the hard drive and the database stores their addresses. Also, to make this data accessible I need to create getters and setters for them. In NetBeans IDE the creation process of setters and getters can be made automatically. To do this I press “Alt+insert”, choose “Getter and Setter” and in the appearing window the check checkboxes of variables for which I need to make getters and setters. The data object model example is shown in Figure 26.
7.3 Data manipulation

To be considered as a high quality web application my application has to utilize the database and manipulate the data in an efficient way. Basically the interaction with the database includes inserting, retrieving and updating the data. It is better to create an engine which will allow these interactions beforehand and to describe its small modifications during further creation of the features of the application. In this section I explain how the interaction with the database can be implemented in a Java web application.

7.3.1 Basic variant

This type combines the connection to the database and SQL script injections in one class. Methods in this class can be declared as public void methods, because they do not need to return values and should be accessible for other processes of the application.

At first, I need to add the dependency to the Project Object Model (POM) configuration file. This dependency defines the database driver. In my case it should be the MySQL driver. Its newest version is “5.1.34”. The injection of the driver dependency is illustrated in Figure 27.
I create the method which is responsible for adding information to the database. This method can get the data from objects such as models for example I use the “ModelObject” name. To make it possible, the class of the model has to be imported and declared. To refer to objects in this class I assign the name for it. The “PreparedStatement” object should be imported to the class and defined in the method because it is used to prepare the data and compile it to the database statement. This object is responsible not only for compiling SQL queries but also for their executing. The “Connection” also should be imported and declared because it is needed to open the database connection. The implementation of this method is shown in Figure 28.

```java
public void addInfoToDb(ModelObject mObject) {
    PreparedStatement ps = null;
    Connection con = null;
    try {
        Class.forName("com.mysql.jdbc.Driver");
        String url = "jdbc:mysql://localhost:3306/database";
        con = DriverManager.getConnection(url, "root", "qwer1234");
        if (con != null) {
            String sql = "INSERT INTO user_account(login, password) VALUES(?,?)";
            ps = con.prepareStatement(sql);
            ps.setString(1, mObject.getLogin());
            ps.setString(2, mObject.getPassword());
            ps.executeUpdate();
            System.out.println("User added successfully");
        }
    } catch (Exception e) {
        System.out.println(e);
    } finally {
        try {
            con.close();
            ps.close();
        } catch (Exception c) {
            e.printStackTrace();
        }
    }
}
```

FIGURE 28. DAO method

This way of the data manipulation is acceptable and it works well. But imagine that we have complicated requirements of the application and need to implement for example 20 methods in the similar way. It would be a big headache to repeat the same code in each method. Also the possibility of errors increases rapidly because the same establishment of the database con-
nection is repeated many times. Each time we have to pay attention for the closing of the connection and precompiled statements when the interaction is done. To avoid some problems related to this issue, I implement the connection establishing with the database in separate class and program methods in this case should refer to it to access the database. But this separated class simplifies partly the code. Precompiled statements of the database are still needed and have to be created manually before compilation. From one side implementing of this method looks as unnecessary work if we pay attention to all listed disadvantages of this method and existence of other possible ways. But from another point of view this methodology has more pedagogical sense than practical. It helps to understand better the working principle of the database programming in Java web application. I explain it here because it can be used during early developments of the application to verify that the system works and just one DAO method is implemented.

7.3.2 Connection template

When we understand how the application interacts with the database we can improve this functional part. To avoid of the repeating the same code in different methods is required because the rise of their number is expected in the future. Java Database Connection (JDBC) template is the enterprise solution for the database programming because it put connection responsibilities to the system and allows developers to focus on the business logic of the application. It can be considered as an efficient technology because it utilizes innovative principle of the Java EE platform. This principle is based on the rule “Compile once and use everywhere”. JDBC takes care of compiling database queries. This advantage simplifies the documentation of the development. In the further developments I can just customize basic methods which I explain in this section.

Dependencies

I am going to implement the JDBC template which is the Spring component. To make components from this framework accessible I need to update the POM configuration file which already has MySQL connector declaration and add new dependencies. These dependencies are shown in Figure 29.
Data source

The declaration of the data source can be declared in the configuration XML file. The declaration includes defining of the driver, the path to the MySQL database and access control data for the database server. It makes this source accessible for all parts of the application. The data source bean can be implemented with the following code shown in Figure 30.

```xml
<bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource">
  <property name="driverClassName" value="com.mysql.jdbc.Driver"/>
  <property name="url" value="jdbc:mysql://localhost:3306/database"/>
  <property name="username" value="root"/>
  <property name="password" value="qwer1234"/>
</bean>
```

FIGURE 30. Data source declaration

Row mapper

I create a row mapper component to bind data from the database to the objects. A method of the abstract “RowMapper” class generates the data based on the row number in the database and makes it suitable to inject to Java objects. This method can be customized to map the data from exact database to exact objects. By this way it is possible to get entries from the database and set them to objects. This method can be implemented by the code shown in Figure 31.

```java
public class UserRowMapper implements RowMapper {
  public Object mapRow(ResultSet rs, int rowNum) throws SQLException {
    User user = new User();
    user.setId(rs.getInt("id"));
    user.setName(rs.getString("name"));
    user.setFamilyName(rs.getString("family_name"));
    return user;
  }
}
```

FIGURE 31. Row mapper
Data Access Object

By using the object oriented programming principle I create abstract methods for the database interactions in the “DAO” class and their implementations in the separated “DAOImpl” class. Creating abstract methods is quite simple task and I focus on the methods which implement the abstraction.

In the implementation class the “DataSource” refers to the data source configuration which was made in the XML configuration file. The “JdbcTemplate” object is the component of the Spring framework which is responsible for SQL queries. Both of them should be imported, declared and assigned in the implementation class.

The updating and inserting processes are logically similar. To show the example I create the method which receives data from the object and inserts it to the database. In this method the JDBC template is set by configurations which are defined in the data source. Then it performs updating the table according to values which are received from the object. The example of the code is shown in Figure 32.

```java
public void insert(User user){
    String sql = "INSERT INTO USER " + "(id, name, password) VALUES (?, ?, ?)";
    JdbcTemplate = new JdbcTemplate(dataSource);
    jdbcTemplate.update(sql, new Object[] {user.getId(),
    user.getName(), user.getPassword()})
    }
}
```

**FIGURE 32. Inserting to DB method**

Retrieving data from the database logically has opposite sense to inserting process. I create the method which retrieves the data from the database table and sends it to the object. This method uses the Row Mapper to bind data automatically to the object. Example of the code is shown in Figure 33.

```java
public User findByName(String name){
    String sql = "SELECT * FROM USER WHERE name = ?";
    JdbcTemplate = new JdbcTemplate(dataSource);
    User user = (User) jdbcTemplate.queryForObject( 
    sql, new Object[] { name }, new UserRowMapper());
    return user;
}
```

**FIGURE 33. Retrieving data method**
At this stage my application does not have any user interfaces, they will be created later. To test interaction with the database at this step I create a simple class with a main method which contains manually created data of the object and calls database interaction methods. At first it should insert the row with exact id, name and password values. Then it should retrieve this data back and print on the display. Figure 34 illustrates the test of created methods.

```java
User userPut = new User(5, "Matti", "secret");
userDAO.insert(userPut);
User userRetrieve = userDAO.findByName("Matti");
System.out.println(userRetrieve);
```

![Output](image)

**FIGURE 34. Test implementation**

### 7.4 Navigation manager

Redirection feature is responsible for transferring users from one web page to another. This feature can be implemented two ways. The first one is defining the path to the redirection page with attribute “action” of the button of the user interface. This is the simplest way of redirection. Usually, it is used when the user interface component has to establish only navigation. A menu bar is a good example of this feature. Another type of the redirection is implementing it in the Java class. The second type can be used to make the possibility of calling redirection by methods in the application. I create the Java class and declare it as management bean. The management declaration is shown in Figure 35. The name should be assigned to refer to this class.

```java
@ManagedBean(name = "navigationBean")
```

**FIGURE 35. Managed bean declaration.**

This class should contain objects with redirection instructions. These instructions can be put to string objects as it is illustrated in Figure 36. When other web pages will be created redirectors to them can be implemented by the same way.
FIGURE 36. Redirecting example.

7.5 Sign up

Users begin to use the system from signing up process. During this step they need to input and save the personal information into database. Initially, I create the user interface and then develop the functional part.

7.5.1 User interface

XHTML page

User interface components should be located on the XHTML pages. To create a new page I right click on the “Web pages” folder and chose “New-other”. There is a filtering field in the appeared page creation window, I input “XHTML” and press “Next”. To include presentation elements I chose the “Transitional” declaration of the web page and press “Finish”.

For the document type defining and using frameworks with their components they should be declared in the document and have to be assigned with variables. Extensible Markup Language (XML) namespaces represent URI addresses of the components in remote repositories. Figure 37 illustrates the example of the namespace declaration.

```html
<html xmlns="http://www.w3.org/1999/xhtml"
     xmlns:h="http://java.sun.com/jsf/html"
     xmlns:f="http://java.sun.com/jsf/core"
     xmlns:p="http://primefaces.org/ui">
```

FIGURE 37. Namespace declaration

To refer to components of these namespaces, the variable names can be used. For example the header of the page is considered as JSF component and can be implemented with “<h:head>” tag name, because the JSF namespace is declared with “xmlns:h” name.
Panel
This JSF component can be considered as the place for allocation other user interface components. I use two types of panels. They are Panel Group and Panel Grid. The first one represents the panel which contains many panels. The second type has more flexible inside allocation configurations and contains components directly.

Field
There are components in the Prime Faces framework for creating fields. In the sign up implementation I use <p:inputText/> and <p:inputPassword/> tags. With the first one I create fields for inputs of the data which has no high level of confidentiality and can be shown during typing. The second type is optimized for the password submitting, it shows dots instead of characters during typing and it is cleared automatically if some actions occur without submitting. A field has the “value” attribute, which contains the link to the object where the input data should be sent.

To prevent submitting empty field the “required” attribute should be implemented with “true” value. Notification about the empty field is provided by the “requiredMessage” attribute.
Fields can have unique “id” values. It is used to make references to the field. For example if we have field with the id=”name”, we can easily make a hint word inside it with “<p:watermark for="name" value="Name" />” component.

The input data should be entered in the correct format. For that purpose the client side validation can be used. It can be implemented with JSF core component for validation. Each type of the data has its own format. For example, personal names should begin from capital letters and dates have to contain numbers with dot separations in specific order. For input control specific patterns can be defined in attributes of validation component. The validation of the email address is shown in Figure 38.

```
<p:inputText id="email" value="#{class.object.variable}"
    required="true" requiredMessage="Please enter your email address"
    validatorMessage="Value does not match pattern"
    styleClass="loginInput">
    <f:validateRegex
        pattern="^([A-Za-z0-9-]+@[A-Za-z0-9-]+(\.[A-Za-z0-9-]+)+)$" />
</p:inputText>
```

FIGURE 38 Email validation
Button
On the registration page I implement two Prime Faces buttons components. The first calls the method which is responsible for the input data processing and the second just redirects to the login page. The method and the redirection page which should be defined in attributes of the buttons are not yet exist. Their names can be given as examples and modified later. The implementation of buttons is shown in Figure 39.

```xml
<ui:commandButton value="Create account" action="#{class.object.variable()}">
<ui:commandButton value="Sign In" action="page?faces-redirect=true"/>
```

FIGURE 39. Buttons

Figure 40 illustrates the user interface of the signup page.

FIGURE 40. Sign up page

7.5.2 DAO method

The model receives the data from the user interface and then it should be inserted to the database. For this purpose I create the abstract method in the “DAO” class and implement it in the “DAOImpl”. The implementation can be done by the same way as it was described in the inserting to database example in Section 7.3.2.
7.5.3 Model

For registration I create a separated model class, which contains fewer variables, because not all data should be provided during registration. The purpose of additional object is logical allocation of the model because before sign in it is out of the system and some additional resolving can be required.

7.5.4 Controller

The controller method calls the DAO method and performs redirection to the login page if the execution of the database interaction is successful. It calls the method from the navigation class for redirection. The controller execution is started by pressing the button on the web page where the controller is defined in the attribute. During the development process I meet the problem, because during performing sign up the error appears. That is because the data does not exists in the objects of the model and according to this it should be prepared and come to the object in the right format. To resolve new incoming data the “ELContext” component should be used. The controller can be implemented with the following code which is shown in Figure 41.

```java
public String addNewUser() {
    ELContext elContext = FacesContext.getCurrentInstance().getELContext();
    ViewBean viewBean = (ViewBean) FacesContext.getCurrentInstance().getApplication().
        getELResolver().getValue(elContext, null, "model");
    UserDao ud = new UserDao();
    ud.addUserToDb(viewBean.getUser());
    return navigationBean.redirectToLogin();
}
```

FIGURE 41. Sign up

7.5.5 Registration test

Now I test the registration feature. For this I fill the fields on the sign up web page and press the submit button. To be sure that this operation is successful I need to check the database table. In the Workbench I can see the new row with the data which I just submitted. Figure 42 shows the updated table in the database.
7.6 Sign in

When the application is able to register users and save their information to the DB the function to enter data to the system should be implemented. In this section I explain how the signing in feature can be implemented, which components are needed and how they should work.

7.6.1 User interface

I implement the submitting of the access control information for users on a separate web page. This page can be created the similar way as the registration page. The implementation methodology is repeatable, and I do not see any reasons to explain it in detail. Instead, I list what the “sign in” web page should include. It should contain two fields: “login” and “password”. The submitting button should call the methods of the controller which select the data from the database and process it, if the login is performed successfully. In my application the login is a separated value, but sometimes email is used for this purpose, and in that case the validation for the relevant field can be implemented. It is needed to avoid unsuccessful request to the server, if the email is mistaking caused by a wrong format. For example, sometimes a user does not pay attention to the language of the keyboard layout, and types another character instead of “@” in the email editing field. The interface of the login web page is illustrated in Figure 43.
7.6.2 DAO method

As a blueprint I use method which was described in Section 7.3.2. This method exists in the “DAO” and implemented in “DAOImpl”. It is responsible for selecting rows from the database. I create a new similar method, but here it selects the row from the database table and sends the data to the model where the “login” and the “password” values are matched with user interface inputs.

7.6.3 Session management

The logged-in user should be held by the application during the whole session. The data about the current user should be stored in the object and should be accessible during the session. I create a class with the object “currentUser” for this purpose.

7.6.4 Controller

I create the new method with a logically relevant name. At first, I declare the Boolean value which should hold the state of the sign in the establishment. The name of this variable can be, for example, “isLoggedIn”. The controller method of this feature calls the relevant “DAO” method described in Section 7.2.2. If the inputted values of the “login” and “password” from the interface are matched with the values in the database, the selection of the data from the database is established and the variable “isLoggedIn” is assigned with “TRUE” value. After that, it updates the “currentUser” object with the data which was generated and appeared in the model.

7.6.5 Login filter

During the security development I place all web pages which logically need to be secured to a separated folder and name it “secured”. The access to these pages will be controlled by the method in the Login filter class. The main purpose of this feature is preventing unauthorized access to the web pages which should be secured. Its working principle is based on checking the Boolean value declared in the controller class. According to checking the results it allows or denies access to the secured web pages. The filter can be implemented by the method which is shown in Figure 44.
7.6 Personal account

When the sign up and the sign in features are implemented the personal account page should be developed. This page has only presentational sense because the personal account data retrieving from the database is performed during the sign in process. It should contain general information and the menu bar. In this section I explain components of the user interface which can be used in the user account page.

7.6.1 User account interface

XHTML page

I create empty XHTML page of the user account. As it was already mentioned this page is located in the “secured” folder and should be accessible if the sign in is performed successfully. Declarations of the page type and required frameworks can be done by the same way as it was made during sign in and sign up pages creations. The panel for allocating all components is also similar to previous interface developments. All data which is needed to be displayed on the personal page is peaked from the “currentUser” object.

Menu bar

The menu bar is required not only on the personal page but also on other web pages. Otherwise the navigation possibility will disappear if the user will transfer to pages which do not contain the menu. The menu is the same for all pages in my application. To avoid repeating the code I create one XHTML page with menu and use it as template in other parts. The menu is implemented as the list of links. The code of the menu template is shown in Figure 45.
Text
Text represents values which are stored in the “currentUserId” object. To display these values I just need to type the reference to exact objects. For example, when I need to display the name of the current user I implement the link to the variable in the relevant object which represents the name of the user.

Dialog
The dialog component appears above other elements to draw attention to it. It can be called pressing a button or appears automatically, if some action occurs. On the personnel page I implement the dialog to show the bank data of the user. To implement the dialog I use the Prime Faces component which can be declared with the tag <p:dialog/>. The dialog is defined by the unique value of the “widgetVar” attribute. Calling the dialog can be established with defining its unique number in the “onclick” attribute of the button. Figure 46 shows the dialog view with the bank data. The modal dialog shadows background web page to concentrate attention on the data inside the dialog.

![Figure 46. Dialog view](image)

Figure 47 shows the personal account interface.
7.7 User view aggregation

When user is able to view and edit personal account page it is time to add interaction between users in the system. In more detailed view interaction means possibility of viewing other users in the system. In this section I describe the implementation of the users view organization in my web application.

7.7.1 User interface.

For the users view aggregation I need two different interfaces, the first one should show the list of users and another interface shows detailed information of the selected user. I consider that user should see the own page and pages of other users in the similar interpretation.

XHTML pages
I need the web page which illustrates the detailed information of the selected user. It can be implemented in the same way as it was described in Section 7.6.1 where I explained interface of the personal account. But here the page provides data of another person, and the data editing feature should not be included on this page. Another page is required for printing out the list of users. In this section I concentrate on components of the web page of the users list.
**Search template**

To show the list of users I implement the search engine in my application. I create the text field in the menu template for the keywords inputs and the search button, which should call methods from the controller to establish searches.

**Table**

The list of users can be allocated in the table. There is a table component in the Prime Faces framework, it can be implemented with the `<p:table>` tag. The attribute “value=” defines which objects should be displayed in the table, there I place the link to array list which holds users. I want to limit the number of users on the single page, because no sense to show all of them. With the attribute “rows=” I can set the number of rows. Other users will be shown on other pages, to make convenient listing these pages I include attribute “paginator=”. To allow redirection to the web page with detailed information of the user by clicking it in the list, I allocate all content of rows in the list to the “<link/>” tag. A row should contain an identification number of a user. It is needed to define the exact user in the list and to perform selection during the redirecting to the detailed view. Also I decide to show images and names of users in the list. Figure 48 illustrates the possible implementation code of the table.

```xml
<p:dataTable id="searchResults" styleClass="borderless" value="#{sessionBean.users}" var="selectedUser"
    paginator="true" rows="10"
    paginatorPosition="bottom" paginatorAlwaysVisible="false">
    <p:column>
        <h:link outcome="user-view">
          <f:facet name="user-id">#{selectedUser.userId}</f:facet>
        </h:link>
        <f:facet name="left">
          <p:column styleClass="link" style="width: 24%">
            <img src="#{resource.contextPath}/image/#{selectedUser.imageURL}" width="100%" height="auto"/>
          </p:column>
          <p:column>
            <h2>#{selectedUser.userName}</h2>
            <h2>#{selectedUser.familyName}</h2>
            <hr style="margin: 1em 0"/>
          </p:column>
        </f:facet>
        <f:facet name="right">
          <p:column>
            <f:setPropertyActionListener value="selectedUser" target="#{ajax:target}" property=""/>
          </p:column>
        </f:facet>
    </p:column>
</p:dataTable>
```

**FIGURE 48. List of users interface implementation**

### 7.7.2 Model

The application has to store the users and generate the list of them to send to the interface which was created at the previous step. I create the array list object in the model class to allo-
cate the list of users. I want to emphasize that getter and setter have to be also added to make this list accessible.

7.7.3 Controller

I create the method in the controller class which calls the DAO method for retrieving users from the database. In more detailed view, the controller sends the input value from the search field to the DAO method, then updates the model object with received values from the database and redirects the user to the web page where the list of users is shown.

Another controller method is needed to perform redirecting to the exact user page. It receives the user identification value from the row in the list which is clicked, then sends it to the DAO method, which retrieves the selected user from the database.

7.7.4 DAO methods

I create the method which retrieves the list of the users from the database according to the keyword which is received from the controller. If there is no keyword, then this method retrieves all entries from the database. After that this method sends retrieved values for the further processing.

7.7.5 User aggregation test

At first I press the search button with empty search field and see all users which currently are registered in my application. The result of the listing all users is illustrated in Figure 49.
Now I test my search engine. I enter the name “Gordiy” to the search field and press the search button. Now I can see just one user which name matches with the searched keyword. Figure 50 illustrates the result of the search.

![User Search Result](image)

**FIGURE 50. The result of the user searching**

### 7.8 Messaging

Messaging feature allows users communicate with each other by sending text messages. In this section I explain how this feature can be implemented.

#### 7.8.1 User Interface

User Interface of the messaging feature should provide entering of the sending messages for further processing and showing received messages with identification of the users, who sent them. Received messages are represented as conversations, for this purpose I implement two tables. The first one shows users with whom the current user has conversations. The interface of this page can be implemented in the same way as the list of users web interface described in Section 7.8.1. But now a link which locates in a row should call the dialog with messages list. Also I add the button to the web page with detailed information about users. This button should call dialog with the message field and submitting button.

#### 7.8.2 Model

To store messages in the application I create object in the model class which describes details of the message such as identifications of receivers and senders, content of messages and the date and time of the sending.
7.8.3 Controller

The controller of the messaging feature is responsible for the calling DAO methods which save and retrieve message data in the database. The conversation list is generated dynamically according to identification numbers of the receiver and the sender. Example of the implementation the sending message is shown in Figure 51.

```
public void addNewMessage(Integer currentUserID, 
  Integer selectedUserID, String usMessage) {
    Message message = new Message();
    message.setTitleName(titleName);
    message.setSenderID(currentUserID);
    message.setReceiverID(selectedUserID);
    message.setMessageContent(usMessage);
    MessageDao metaData = new MessageDao();
    metaData.addNewMessage(message);
}
```

**FIGURE 51. Controller of the sender method example.**

7.8.4 DAO methods

I create two DAO methods in the similar way described in Section 7.3.2, and modified SQL strings for new purposes. The first method receives the message data from the application and inserts it into the database. Another method retrieves rows from the message table in the database where identification numbers of the current and selected user are matched, the retrieved data is sent to the array list in the model.

7.8.5 Messaging test

When all parts of the messaging feature are implemented it is time to test how it works. At first, I visit the informational page of the person and try to send the message. I press the button on the personal page of that user for whom I want to send the message. Then I type the message example and press the submitting button. Figure 52 illustrates the sending message step.
After the establishing the test conversation I visit the web page which shows my conversations. By clicking on the user I see the dialog with the list of our messages with this user. Figure 53 illustrates the messaging history.
7.9 Personal information editing

The editing of the personal information is a very important feature of the application. It allows users to modify their information in the system. In this section I explain the principles of implementing this feature.

7.9.1 User interface

XHTML page

I build the User Interface (UI) of the editing page in the similar way as it was done for the personal account page. But here I create fields instead of normal displaying of the data and include submit button which calls methods for updating the model and the database with new values. To autofill fields with current values of the personal information I define current objects inside the “value” attribute.

Field

Fields on the personal information editing page can be implemented in the same way as it was described in Section 7.5.1. To avoid the submitting empty fields I choose the method which fills fields automatically with data received from the object of the current user. For this I just need to place the link of the relevant variable in the “currentUserName” object to the “value” attribute. Example of the code implementing field is shown in Figure 54.

\[
\text{FIGURE 54. Information editing field.}
\]

Image updating view

The image displaying can be implementing on the page by similar way, as was described in Section 7.6.1. But for making the interface more convenient to view I decided to concentrate users attention on the image by displaying it in the dialog during the updating process. I put the image to the “h:outputLink” tag which calls the dialog where the image is shown by the same way. Figure 55 illustrates the implementation of this feature.

\[
\text{FIGURE 55. Image as a link}
\]
In the dialog I allocate the picture, uploading and updating buttons. The uploading button calls the file uploader methods. The Update button calls DAO methods to perform database updating with value of the received image name from the file uploader. The code of the dialog is shown in Figure 56.

```
<dialog header="Photo" widgetVar="dlg" modal="true" showEffect="clip" hideEffect="drop">
  <h:panelGroup>
    <h:form enctype="multipart/form-data">
      <p messages id="messages" />
      <p fileUpload id="file" value="${sessionBean.current_user.userImage}" mode="simple" />
      <p:roll id="productMessage" showDetail="true" redisplay="false" />
      <p:commandButton value="Add photo" actionListener="${user.updateUserPhoto()}">
        update="userMessage, addUserPanel" ajax="false"
        style="margin-bottom:10px; margin-top:10px" />
      <br />
      <img src="${request.contextPath}/image/${sessionBean.current_user.image_url}" width="400px" height="auto" />
    </h:form>
  </h:panelGroup>
</dialog>
```

**FIGURE 56. XHTML dialog component**

### 7.9.2 File uploader

To perform updating images in the system I use the file uploader component. It contains methods which define the path to the file allocation, get the file name and preform uploading. Let me describe these three parts in detail.

At first, I create a filename generation method. It reads the upload file name and extension as separated parts. Then it combines these values with “.” Separation into a single value and puts it to the object.

Then I create the uploading method. It contains input and output objects. The first one receives a file from outside. The second provides it for saving. If the uploading is performed successfully, the address value of the file object is updated. If this process fails, the method returns the filename value “undefined_image.png” which represents the default image. Finally, when the process is performed, the interaction should be closed. The file uploading method can be implemented with the code shown in Figure 57.
Finally I test how the personal data can be modified in my application. I go to the settings page. Here I can see my current personal data. Figure 58 illustrates settings web page.

After that I click on the image, the dialog appears. In this dialog I add my new personal photo and press the submitting button. The photo editors dialog is shown in Figure 59.
Now I see that my photo is already updated. At the next step I modify some of my personal data and press the “Update” button. Figure 60 illustrates the data modification.

Finally I am redirected to the personal account web page. The updated personal data appears on this page. Figure 61 illustrates the personal account page with updated data.
8. CONCLUSION

The first aim of this study was building of a solid theoretical background for data storing methodologies and web-based software development with. The alternatives of different database management systems, the main concepts of the Java programming and the principles of the web-based software development were described in the theory part which based on existing educational sources. The second aim was showing how the knowledge in the area can be applied to practical software development. I took part in a project which relates to the development of the web-based trade platform with Java EE.

According to my responsibilities I developed the user services of the application. These services allow users to register in the virtual system, to share personal information and the data for further business deals, to modify the data and to communicate with each other through the messaging. All the required features were created, and each goal was achieved. The whole development process was covered in the practical part of this study. I tried to show the professional way of solving the engineering tasks. These tasks involve the programming problems during the development process and the finding of their solutions.

This project was the most significant development experience in my life. But the created application is the prototype of the full version and has limitations. First of all, I want to emphasize that the interaction between the application and the data storage is represented as a plain text messages. The data management process in this case is unsecured. Further, the encryption of the data should be implemented to avoid losses and possible threats. The registration in the system can be improved with additional authentication development to make the system more secured. There are services in the Internet which provide the bank account verification, this kind of support costs money, but it is not expensive and can be also included in the commercial version of the application to avoid the fake bank data registration. According to ethical rules the personal data sharing should be capable of customization in a more flexible way. For example sometimes users do not want to show their places of residence, or ladies prefer to hide their birth year. This kind of the data should be stored in the system, because it is required for business, but should be privatized partly. The GUI style of the prototype is implemented with the minimalistic principles. It looks good but I would pay more attention to it in the commercial version of the application because the interface should encourage users to return back to the website. The audience of that kind of software usually varies in ages, coun-
ties, social group etc. The usability of the system was paid attention to during the development, but not tested properly by different audience groups. It is a necessary step in the commercial version development. And finally, different human preferences should be analyzed to optimize the system for all kind of users.
BIBLIOGRAPHY

Books:


Web sources:

Eclipse, Specifying model operations. The company’s WWW pages.
Referred 25.03.2015
JavaTPoint, Variables and datatypes in Java. WWW pages.
http://www.javatpoint.com/variable-datatypes. Referred 24.03.2015

MySQL. Documentation. The company’s WWW pages.
http://dev.mysql.com/doc/refman/5.0/en/what-is-mysql.html. Referred 05.02.2015

Oracle. Java Documentation. The company’s WWW pages.
http://docs.oracle.com/javase/tutorial/java/concepts/. Referred 20.03.2015

Paololatella, Java. WWW pages.
https://sites.google.com/site/ingpaololatella/java/java-ee-java-se-and-java-me
Referred 02.04.2015


https://www.scrum.org/resources/what-is-scrum. Referred 10.04.2015

Water-Fall model development methodology. WWW pages.

W3schools. SQL General Data Types. The company’s WWW pages.
http://www.w3schools.com/sql/sql_datatypes_general.asp. Referred 25.01.2015