Design and development of a Web-based Mentor matching system

Datateknik 15 hp

Halmstad 2018-06-19
Alexandra Johansson och Julian Sjödal
Abstract

Mentoring can be important, in certain cases necessary, for academic, professional or personal development. Successful mentoring results are likely to be achieved when the background and expectations of mentors and mentees match. This project, which was developed in collaboration with WiTEC (Women in Technology), proposes a web-based mentor matching system to improve the current company’s approach to form mentorship. This project report presents therefore the whole system design and development process, including a discussion about the different software development methodologies and web technologies that could be employed. The developed web-based mentor matching system incorporates an important non-functional requirement that is compliance with the main elements of the General Data Protection Regulation (GDPR).
## Contents

1 Introduction .................................................. 1
   1.1 Project background ........................................... 1
   1.2 Aim and objectives .......................................... 1
   1.3 System requirements ......................................... 2
       1.3.1 Project Outline ........................................ 2

2 Background ................................................... 3
   2.1 Related work ................................................ 3
   2.2 Software development methodologies .......................... 4
       2.2.1 Waterfall model ........................................ 4
       2.2.2 Spiral model ........................................... 5
       2.2.3 Prototyping model ....................................... 6
       2.2.4 Incremental model ....................................... 7
       2.2.5 Summary ................................................ 8
   2.3 Web-Based Systems ........................................... 9
       2.3.1 Web-Based System Fundamentals ......................... 9
       2.3.2 Client-side ............................................. 10
       2.3.3 Server-side ............................................. 11
   2.4 Integrity .................................................... 13
       2.4.1 General Data Protection Regulations .................... 14
       2.4.2 Authentication approaches .............................. 14
       2.4.3 Possible attacks on privacy .............................. 15

3 A Web-based mentor match system ................................ 17
   3.1 Requirement specification ..................................... 17
       3.1.1 Login .................................................. 18
       3.1.2 Register/Register new mentor ............................ 18
       3.1.3 Change profile/Edit other profiles ...................... 18
       3.1.4 View matching profiles ................................. 19
       3.1.5 Apply for mentee/mentor ................................ 19
       3.1.6 Approve mentees ........................................ 19
       3.1.7 View matches/Make final matching ....................... 20
   3.2 Data model .................................................. 20

4 Results ......................................................... 21
   4.1 Register/Register new mentor ................................. 21
   4.2 Login page ................................................... 23
   4.3 Change profiles/Edit other profiles .......................... 24
       4.3.1 Mentees and Mentors .................................... 24
       4.3.2 Administrators .......................................... 25
   4.4 Matching ..................................................... 25
   4.5 Approve users ............................................... 26
4.6 Database implementation ................................................. 27
  4.6.1 Members ................................................................. 27
  4.6.2 User ................................................................. 27
  4.6.3 Candidates ........................................................... 27
  4.6.4 Mentors ............................................................... 28
  4.6.5 Admins ................................................................. 28
  4.6.6 Matches ............................................................... 28
  4.6.7 Login attempts ....................................................... 28
  4.7 Testing ................................................................. 28

5 Discussion and conclusion .............................................. 29
  5.1 Did we fulfill requirements for the system? ......................... 29
  5.2 What could we have done for a better result? ....................... 29
  5.3 Related work ........................................................... 30
  5.4 The future for the project ............................................. 30
  5.5 Conclusions .......................................................... 30

A Website snapshots .......................................................... 35

B Timeplan ................................................................. 41
Chapter 1

Introduction

1.1 Project background

The idea for this project was proposed by WiTEC, which stands for Women in Technology. WiTEC is an organization that mainly aims at increasing the number of women working in technical or science fields [1]. To achieve this, WiTEC matches mentors with mentees who want guidance in their carrier. However, WiTEC wants to make the matching between mentees and mentors easier.

According to Ingrid Thuresson, President at WiTEC [1], from her previous experience is it important to match the right mentor with the right mentee otherwise, in a lot of cases, the mentees do not learn as much from the mentorship. At the current state, WiTEC only matches mentors and mentees manually through personal contacts. A mentee can ask the organization if there are any possible mentors for them and get presented with the most fitting and available mentor. WiTEC also arranges venues and other activities for mentees to find their mentor more easily and create mentorships that way as well. On their website, there is an option for mentors to register an interest in being a mentor but still, there is nobody who has signed up here.

Previously in 2015, a project conducted by students investigated an easier approach for the matching [2]. The students concluded that an alternative solution for this problem could be to create a web application for matching mentors and mentees. They argued that this solution would not only have benefits such as bringing mentors and mentees into one place but it would also be a solution that would be easy for WiTEC to maintain. In 2016, a follow up project, also conducted by students at Halmstad University, [3] resulted into a proof of concept for a match making web application. The result was a clean website, similar to WiTECs existing website were mentors and mentees could create profiles and answer questions that could establish which mentor would fit with which mentee.

1.2 Aim and objectives

The aim of this project is to deliver a web-based application of mentor matching that addresses the functional and non-functional requirements for the system. The system should be easy for the organization to test and integrate to their existing website.

---

[1] Ingrid Thuresson, President at WiTEC Sweden, 2017-12-13
1.3 System requirements

The following list contains the requirements that the system should fulfill at the end of this project.

- Administrators, mentors, and mentees should be able to have personal profiles.
- Administrators shall be able to manage mentors and mentees and create new mentors.
- A new registration of a mentee must be approved by an administrator before he/she can access the system.
- The system should generate recommendations of suitable mentors for mentees and vice-versa based on competencies and characteristics of their profiles.
- Administrators can see matching mentees and mentors and make the final matching.
- Mentors should be able to edit their profiles, see matching mentees and apply for mentees that they think would fit.
- Mentees should be able to register, update their profile, see matching mentors and also apply for mentors that they think would fit.
- The system should fulfill the requirements of the EU-regulations of data protection.

1.3.1 Project Outline

The remaining of this report is structured as follows. Section 2 explains the technical background of the project starting with software development methodologies, followed by web-based systems and ends with some security aspects. Section 3 explains the specification of the project and how one should analyze the results. Section 4 explains the preliminary results and the thesis ends with a discussion/conclusion in section 5.
Chapter 2

Background

This chapter will first present related works. The design and development of web-based systems, do not require only knowledge and skill about web technologies, but also about methods for develop software. The chapter will therefore presents an overview of common software development methodologies so the reader can understand the development approach adoption in this work. Moreover, an overview about traditional technologies to develop web application. This chapter also includes a section about data security and privacy, including the main elements of the General Data Protection Regulation (GDPR).

2.1 Related work

"Strategies for Mentor Matching: Lessons Learned" [2], Beth M. Hacker, Lalitha Subramanian and Lynn M. Schnapp, is an article that describe and tests different kinds of mentor matching systems. They state that The Institute of Translational Health Sciences (ITHS) received 34 inquiries between May 2010 and February 2012. The authors later determined the regional distribution of the inquiries and after establishing what the reason for contacting the ITHS the authors conclude that a central database would be a good solution to the problem. The project soon realized that this strategy did not work as well as earlier though due to the major differences in profession and specialization. They also tested to individually pair mentors and mentees using their personal network and were also difficult due to differences in profession. The solution to their problem was a tutorial or description to find the perfect mentor for the mentee. The solution also includes links to other databases that specializes matching in a particular areas [2]. The authors Diego R. Camacho, Christopher M. Schlachta, Oscar K. Serrano and Ninh T. Nguyen investigates in their article "Logistical considerations for establishing reliable surgical telementoring programs: a report of the SAGES Project 6 Logistics Working Group"[3] if a centralized database of mentors is an idea that mentees would appreciate in matching with a mentor. Their investigation includes the value of mentorship and education with practicing surgeons and trainees. Their results indicate an importance in the centralized computerized matching system of geographical distance between the two parts, the availability of the surgeon on the date of the case, compatibility of the information technology support and the type of procedure or technique that is going to be mentored. They also conclude that it is important to have a variety of mentors in, for example, the area of expertise. This lets the mentee to choose between a number of different mentors [3].
2.2 Software development methodologies

In a report about software models the authors wrote, "A Programming process model is an abstract representation to describe the process from a particular perspective." [4]. This means that a process model is an explanation of how to efficiently, in different scenarios, work towards a well-made program. There are many different, well-defined models and ways of implementing a program efficiently, for example, the waterfall method, the V-shaped method, and evolutionary development model. To choose the most fitted model for this project there is a need to go through a couple of methods. Four different models are going to be considered: the waterfall, spiral, prototyping and incremental.

2.2.1 Waterfall model

The waterfall model is a classical model in software engineering and also one of the oldest. The waterfall method often serves as the base of many other models. This model prioritizes work in documentation to create a good understanding of the process. The process is a series of non-overlapping steps which provide the users a good understanding of what and how the program should be implemented. The method has seven steps and starts with system requirements and follows with software requirements, architectural design, detailed design, coding, testing and ends with maintenance [5].

![Figure 2.1: The different steps of the waterfall model](image)

The following list explains the details of the steps as seen in figure 2.1. Parts of these steps are also used in other models explained later.

- System requirements: The components in the hardware and software tools are established in this step. The external software is also documented in, for example, databases and libraries.

- Software requirements: This step establishes functionalities of the system and also identifies the system requirements in the perspective of performance and user interface. There should also be an analysis which concludes the requirements of the database and other external applications.

- Architectural design: It determines the software framework of how to meet the requirements of each function. It defines the large components and the interaction between them. All of the components are not defined in this step.
- Detailed design: Examines the components defined in the previous stage and creates a specification of how each component should be implemented.

- Coding: Implements the specifications of the detailed design.

- Testing: Determines whether the program works as intentional and meets the requirements.

- Maintenance: Additional enhancements and bug fixes after the release of the system.

A review needs to take place between each stage to determine whether the project meets the requirements and can proceed to the next step. Since there is no overlapping in this method it is important to have all requirements and a full understanding of the project at the start. In the real world, this happens very rarely which triggers people to adjust this method to their project. This method contains immense planning and documentation which makes it work well in projects with quality as a concern. Due to this large amount of documentation, many developers think the model is too inflexible and excessive. [7].

### 2.2.2 Spiral model

The Spiral model was first introduced by Barry Boehm in 1986 [8]. The spiral model is a so called risk-driven process model which means that it is built to prevent major implementation errors. It was supposed to be an alternative to other software processes that had got a lot of criticism at that time because they were not flexible. Although, this model can integrate other process models, such as waterfall and prototyping, based on the risk patterns of a given project.

![Spiral model diagram](image)

**Figure 2.2: The spiral model includes four phases: Determine objectives, Risk Analysis, Development and Planning [9]**

As seen in figure 2.2 the spiral model has four big phases, a software project moves through these phases repeatedly in iterations, one iteration is called a spiral. The first step is to identify the objectives of the project. Secondly all of the risks needs to be assessed in order to reduce the key risks. In the next phase is the development and verification made,
any of the general development models can be chosen to develop the project. Finally all spirals end with an evaluation where the last spiral is reviewed and planning for the next spiral is made.

Barry stated in his report that the model incorporates many of the strengths of other models and resolves many of their difficulties [8]. Because of risk driven nature of this model a lot of issues are found in early stages which means that developers do not need to go back and change implementations in a later stage, which can be costly. Because the model iterate through the phases many times it is more adaptable than for example a waterfall model. The model works best with large and complex projects since the cost of identifying risks could be high. For smaller or low budget projects this model might be too expensive. Since there is no clear end of the spiral, the model can become quite complex and it can be hard to meet deadlines or budgets. It needs to be an experienced project manager to be able to create an efficient interpretation of this model since almost all spiral models can look different and incorporate with different models [4].

2.2.3 Prototyping model

This method is not a standalone methodology but rather an addition to larger, more established models, for example, waterfall and spiral model. This makes it more applicable to different scenarios. The idea is to split the project into smaller parts to create more, small goals during the project and produce prototypes faster. Prototypes are created with the mindset that it might be discarded. This involves the stakeholders during the iterative development process and the developers receive more feedback. It also gives the project a higher chance of acceptance in the perspective of requirements. The steps shown in the figure 2.3 are similar to the waterfall model. The difference is that after the requirements are gathered a more brief design of the product will be made than what would have been done with a waterfall model. After the first prototype is done the customer can evaluate it. After this more requirements can be added or changed. A new prototype can then be made and the project stays within these four steps until the customer is pleased with the product. The next step will then be to develop, test and maintain the product which is the same as the waterfall model. The prototyping development process is recommended to use in projects where requirements are vague or known to be added at a later stage. The requirements of the project might also be changed. It is also appropriate when the project have an experienced project manager and a stable composition of developers. Prototyping is not recommended in scenarios where the requirements are clear, and the understanding of the project is high. Also, it is considered worse if the design of the software implementation is weak [7].
2.2.4 Incremental model

The Incremental model works by dividing a project into smaller segments much like smaller waterfall models, then the models go through the waterfalls in iterations as seen in figure 2.4. This means that the product is designed, implemented and tested several times. In each iteration more functions can be added. This enables that additional requirements can be added, since the customer can test the product between each iteration. When an incremental model is used, you often focus on creating some sort of core product with only the basic requirements. This core product can then be used by the customer to evaluate the product. This enables an evaluation of the product and new requirements to better fit the customer can be added and another increment can start [7]. Unlike prototyping and other similar iterative models, the incremental model focuses on delivering an operational product with each increment. With this model, it will become less expensive to change the product during the development process. This makes the incremental project good to use when the requirements on a project is hard to understand or not well-defined. Another advantage of the model is that in every iteration different functions can be focused on by different people. The model is not good to use for very small projects because the model can become quite expensive since it is hard to anticipate how long the project is going to take.
2.2.5 Summary

From the facts established in table 2.1, we have come to the conclusion that a mix of prototyping and the waterfall method is the best for this project. The prototyping method is good in this scenario due to the fact that WiTEC wants to see results in order to give us more response. Because this is a rather small project the waterfall model would be appropriate to integrate as well. It complements well with prototyping to reach the result in a shorter amount of time. Although this is a very small project and therefore the project could look a lot like only the waterfall since it is not as hard to go back and change things in a later stage but the goal is still to create a prototype that the company can review.
Table 2.1: A table presenting different development methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Strengths:</th>
<th>Weaknesses:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Waterfall</td>
<td>Ideal for less experienced developers. Progress of system development is measurable</td>
<td>Need a clear view of the requirements System performance cannot be tested until system is almost fully implemented</td>
</tr>
<tr>
<td>Spiral</td>
<td>Decreases possible risks Flexible, can use multiple methodologies.</td>
<td>Need experienced developers Do not have a clear start or end point</td>
</tr>
<tr>
<td>Prototyping</td>
<td>Fast results Useful in projects with unclear requirements</td>
<td>Can lead to poorly designed systems Can lead to non-usable prototypes</td>
</tr>
<tr>
<td>Incremental</td>
<td>Gives concrete evidence of project status Gradual implementation provides the ability to see effects of new changes</td>
<td>Well defined interfaces is required Complex implementations tends to be pushed to later stages.</td>
</tr>
</tbody>
</table>

2.3 Web-Based Systems

In order to develop a web based application there is a need to know about a few basic web technologies. The subsections below establishes how the web works and which technologies will be used.

2.3.1 Web-Based System Fundamentals

The web can sometimes be referred to as a client-server model of communications [12]. There are two parts of this model: clients and servers as seen in figure 2.5.

Figure 2.5: An illustration of the client-server model.
2.3.2 Client-side

A client can be a desktop, laptop, smartphone, etc. The most important characteristics of a client are that it makes requests to a server. The client-side is where you write the type of code that is executed or interpreted by the web browser. The design is implemented here and the code is often visible for any user. The requests sent by the client are then being processed by the server. To make it easy for WiTEC to use this website with their current website we will choose to develop the client-side with the same languages as their website, which is HTML, CSS, and JavaScript.

2.3.2.1 HTML

HTML is the building block of all web pages. A large part of the massive growth and success of the web is thanks to the simplicity of this language. HTML is short for HyperText Markup Language. A markup language is a system of annotating a document in a way that makes the annotations distinguishable from the text being annotated. Markup languages like HTML are used to create documents that define how the structural and visual elements should be laid out and displayed. The latest version of HTML that most web browsers support is HTML5. According to the book "fundamentals of web development" are there three main aims of HTML5[12].

- Specify how the browsers should deal with invalid markup.
- Provide an open programming framework, via JavaScript, for creating web applications.
- Be compatible with the rest of the web.

HTML documents are built out of textual- and HTML elements. An element is defined in the document by tags. A tag consists of the name of the element within angle brackets. There needs to be both an opening tag and a closing tag. Between the tags can we write the text we want to display on our site. HTML can also label pieces of content such as heading, body, paragraph and so on. Browsers will not display any of the HTML tags, they will only use it to render the content of the page. This means that with only these tags and labels can we structure and display a different kind of titles, subtitles, tables, and a lot of other things on our site. There are more things we need to consider to create a complete site but these are the basics of HTML.

2.3.2.2 CSS

In this project will a language called CSS be used to describe the style of our HTML document. CSS is short for Cascading Style Sheets. It is a tool for web developers to modify the visual representation of web pages, more exactly is it a tool for describing the appearance of HTML elements. With the help of CSS developers can define colors, sizes, borders, images, and positions of elements. CSS can be added directly to an HTML element or in a separate text file that only contains CSS.

Most of these things can also be done in HTML so why is CSS a better way of describing appearances? There are several reasons, first of all are there a significantly more formatting control in CSS than HTML. Since all formatting can be centralized in one CSS file is it easier to maintain the web site because you will only need to change code in that one file to make changes cite-wide. It will also be faster to download a cite that is built on one centralized CSS since each individual HTML file will contain less information and markup [12].

CSS can both be used embedded in an HTML document or be stored in an external CSS file. If an external CSS document is used can there not be any HTML code inside the CSS file. If CSS is used inside an HTML document the CSS code must be surrounded by tags called style. A CSS document or a CSS style section consists of one or more style rules. A
rule consists of something called a selector that identifies the HTML element that will be affected. For example, if we wrote a text within an h1 tag inside an HTML file would h1 then be our selector inside our CSS file. The selector is then followed by a declaration block where you, for example, define color, position or fonts for the selector.

2.3.2.3 Javascript

In this project will we use JavaScript to do the client-side scripting. We will use JavaScript since it works with most browsers and is widely used by other developers to script the client-side. JavaScript enables interactive web pages and which makes it an essential part of web applications. JavaScript can animate, move, transition, hide and show parts of a website. This can be done without having to reload the entire page from the server which naturally reduces the load on the server. There will also be a lot faster response to the user then what a call to the server ever could have taken. Something that needs to be considered when using JavaScript is that not everyone has enabled scripting on their browser. These users will not receive some or any parts of what the JavaScript is responsible to show or do on the site.

Larry Ullman define JavaScript in his book *Modern JavaScript: Develop and design* [13] as an object-oriented, weakly typed, scripting language. An object-oriented language means that almost every variable in the language is an object. Variables are objects in a way that they can have its own subvariables, called properties, and functions called methods. Unlike many other object-oriented languages such as Java and C# are functions and arrays also objects. Another difference is that in most other object-oriented languages would you create classes and then create objects as instances to those classes but that is not how JavaScript works. This is because JavaScript is prototype-based rather than class-based [12]. The second part of the description of JavaScript was that it is a weakly typed language, also called dynamically typed. This means that variables can be easily converted from one datatype to another. In for example java is a variable strongly typed, also called statically typed, which means that the variable is defined by the programmer and enforced by the compiler. In JavaScript is the type assigned and can also be changed during runtime.

2.3.3 Server-side

The server-side is essential for this model, the web server can host web applications, store information about users and perform security tasks. The server needs its own code to interpreted requests from the client. This code will usually not be displayed to the general user and is often not even available to them.

2.3.3.1 Development language

Three popular free programming languages that are used for web development are PHP, ASP.NET and Python [14]. Atul Mishra compares ASP.NET with PHP in his report *Critical comparison of PHP and ASP.NET for web development* [15]. Both ASP.NET and PHP can be written in an object-oriented fashion but only PHP can be used in a procedural perspective. When it gets to the interactivity with databases Mishra states that both of the mentioned languages have well developed APIs. Therefore, both languages are designed to work closely with databases and both can handle for example Oracle and MySQL. The report also mentions that due to the framework the ASP.NET is most suited to larger web development projects and PHP is more suited in the smaller applications. Mishra also mentions that both languages handles the most common security risks well and have rich functions and libraries to prevent this. When Mishra compares syntax he states that both is similar to C and Java. But due to the smaller size for this project we have decided to use PHP before ASP.NET.
Klauss Purer compared PHP, Python and Ruby in his report *PHP vs. Python vs. Ruby – The web scripting language shootout* [16]. Among other things the author compared the popularity, availability, security, syntax and performance in a web environment. He concluded that it is hard to say which of the languages was the best to use PHP and Python were the same in performance, although PHP had the best availability, meaning that it supported most websites and was also the most popular language based on how many businesses that was using it for server-side development. Python on the other hand had a better security and more functional features. Purer also concluded that the syntax of PHP is very similar to known programming languages such as Java and C. The syntax of Python could, therefore, be a little bit harder to learn since it does not have this similarity. The authors of this project have no experience with Python or PHP but have both programmed in both C and Java and will therefore choose to use PHP for this project also based on the popularity and use of the language which means that there are a lot of documentation and help on the internet to make use of [15].

PHP stood for personal home page, although it is now a recursive acronym that means PHP: Hypertext Preprocessor. PHP script is the most commonly used web development technology and is used in for example Wordpress and Facebook. It is well established, open source and easy to learn due to the variety of different websites that provide good documentation. PHP is free to use and can be used on all major operating systems, Linux, Microsoft Windows and MAC OS, etc [17] [12].

PHP is a dynamically typed language that can be embedded within the HTML code and is executed on the web-server. It supports most common object-oriented features, such as classes and inheritance, but can also be programmed in a procedural way. PHP is capable of generating dynamic page contents as well as managing cookies. It can also encrypt data and send orders to the database. PHP code is executed on the server and is then returned as plain HTML. A PHP file can contain text, HTML, CSS, JavaScript, and PHP. We are going to use this language for the implementation of the major functions of the website. For example sending queries to the database or sending the user to a new page. We will also use this when deciding what to show the user and which access the user has [12].

How do you use PHP? PHP is arguably very similar to the programming language C++, which means that the syntax reminds of C++ syntax. As mentioned above, it can be embedded within the HTML code as tags. To use this you simply surround the PHP script with the HTML tags and this lets you write server-side scripts in the HTML files [12].

2.3.3.2 Database

A database is a collection of information that is organized so it is easy to access, manage and update. We will need to use a database in this project considering the amount of data that will be accumulated over time. The database also brings aspects of security due to the fact that is a separate system which makes it easier to protect from hackers etc.

When implementing a database one can use a relational database and a non-relational database and in the article “A Survey and Comparison of Relational and Non-Relational database” [18] the authors Nishtha Jatana, Sahil Puri, Mehak Ahuja, Ishita Kathuria and Dishant Gosain compare the different approaches of database implementation. The article explains that the non-relational database is not using a table structure as the relational database. This grants the non-relational databases to be preferred in databases where the data does not have to be highly structured, the data does not have to be placed at a specific spot within the storage. The article also states that the non relational databases is highly scalable and but does compromise in consistency. The non-relational database only uses columns which returns an inefficient searching capabilities especially with multiple criterias since the storage lacks in structure. The article also mentions that the non relational database is more suitable for storing larger amount of data if no specific structure is needed. The article also explains that relational databases that it uses both columns and rows with a tabular structure. This method provides a good structure for the storage and will therefore
be easier to both search and extract specific information. The article also states that the structure, on the contrary to non-relational databases, provides a prevention towards data duplication which prevents inconsistent data to occupying the storage. With this information this project will therefore use a relational database [18].

The previously mentioned article, [18] also states differences between MySQL and Oracle as a database. Some of these differences are that MySQL is free unlike, oracle is very flexible in its syntax which MySQL lacks somewhat in, Oracle is more compatible in usages of larger amount of data and supports for example Active Data Guard and Data Mining. MySQL and Oracle handles their temporary tables differently where MySQL drop the table whenever the sessions ends while Oracle needs to actively dropped. Oracle is most often used in for example banking and finance companies where a high amount of data needs to be processed while MySQL is more used in smaller businesses and projects. With this information this project will therefore use MySQL as a database [18].

The database will be implemented with a language called SQL (Structured Query Language). SQL is the standard language used when implementing a relational database. SQL lets you access and manipulate databases. The language uses self-explanatory statements which contain clauses, expressions, and predicates. The clauses explain what the predicates should do and the predicates use expressions to know where or what to change. There are different versions of the SQL language but all of them must, in order to be compliant with the ANSI standards, at least support major commands such as SELECT, UPDATE, DELETE, INSERT etc [12].

2.3.3.3 Web-Server

To test our website and database we need a web-server. The primary function of a web-server is to store, process and deliver web pages to clients. This means that the web-server stores one or a compilation of files containing code that describes the functionality and design of a website. A web-server is a computer that interprets and responds to hypertext requests from the client-side. It is where the functionality of the website is implemented. Some other important tasks that the web-server is responsible for are encryption/decryption, data compressing, managing permissions, files, and security.

2.3.3.4 Solution Stack

A solution stack is a set of different programs or application software that are bundled together in order to produce the desired result. In this project, we are creating a web application which means that we need to define the solution stack as the target operating system, web server, database, and programming language. Considering the choices made above we can use a solution stack that is called WAMP - Windows, apache, MySQL and PHP. This stack provides, as needed, a web server called Apache HTTP server that can run on a personal computer, a database management system, MySQL and the web development language PHP. WAMP is an open source and free program that gives developers an easy way of implementing web-based projects. The AMP stack also exists using the operating systems Linux and Macintosh. Those are known as LAMP respectively MAMP [19]. Since the computer used in this project already had windows installed beforehand WAMP will be used. Since most actual web server deployments use the same components as AMP, it makes transitioning from a local test server to a live server easy as well.

2.4 Integrity

Users on the web are often exposed to security and privacy risks when using web applications. The vulnerabilities in many websites can result in compromised private information. Information such as username and password can be used to profile the user which is a major
privacy concern. To tackle these possible vulnerabilities one need to take these into account and implement the application in a way to prevent possible attacks. Web application security is a process of securing confidential data stored online from unauthorized access and modification. There are three major security requirements to consider when creating and running a web application [20].

- Confidentiality- no sensitive data stored in the application should under any circumstances be exposed.
- Integrity- data in the application should be consistent and no unauthorized user should be able to modify it.
- Availability- granting access to different parts of the application depending on an authenticated user’s entitlements.

2.4.1 General Data Protection Regulations

"The EU General Data Protection Regulation (GDPR) replacing the Data Protection Directive 95/46/EC and was designed to harmonize data privacy laws across Europe, to protect and empower all EU citizens data privacy and to reshape the way organizations across the region approach data privacy" (https://www.eugdpr.org). GDPR will take effect from 25 May 2018.

One of the biggest changes to the current regulations (1995) is the increased territorial scope. The regulations will now apply to data subjects in the EU where their personal data is being processed. It also states that it applies to data subjects that is in the European Union and not where the controller or the process is being held. Previously, the territorial applicability only refers to data process “in context of an establishment”.

The conditions for consent has also strengthened and the companies will no longer be able to use the “terms and condition” method. You now need to give them your consent for them to use your information for a specific purpose. The consent must be withdrawn easily and the conditions and the usages of the subjects' data must be easily retrievable and readable.

Now the subjects have to be notified if there is a breach. If a breach has occurred the companies, handling citizens of the European Union personal data, have a maximum of 72 hours to notify the users about the breach. The customers and the controllers have to be notified “without undue delay”. Data subjects now have the ability to delete their personal data from the data storage. This means that if a data subject, for example, withdraws their consent the personal data have to be erased in the database. The subjects should also have the ability to change their personal data easily.

If these regulations are not followed the company can be facing a penalty. These penalties vary but can stretch up to 4% of the annual global turnover or 20 million euros, whichever is greater [21] [22].

2.4.2 Authentication approaches

Username and password is one of the most used ways to identify a user trying to log in to an application. The username is to establish which user trying to authenticate itself to the application and the password is used in the identification of the user, the person trying to log is the same as the user. The password is a key for accessing information that is connected to a user. This key is sent to a web-server and later checked with the stored key to grant access or not. But there are different ways of implementing a secure web-application, using for example a physical key [23].

Physical keys are an addition to the security in different identification situations. For example, when opening a door one often only needs a key to unlock. In a web application there are often a password that is needed. If one would combine these two elements, put a
lock on the door with a key hole and a password and only be able to write a password when you got the key. The security of this door will become higher. It is because of the new level in the identification process. Banks often use this type of technology with their bank card readers and the PIN-codes. [23].

### 2.4.3 Possible attacks on privacy

An attack to a system is an approach to compromise or modify data. One way to gain access to a system or application is to gain knowledge of the password for a user. There are also techniques or strategies to infiltrate applications using other kinds of information such as cookies. Cookies are values that confirms access to a website. It can also be used as information of a user in different implementations such as shopping carts etc.

Timing attacks is a time check technique which measures the time a function spends on calculating the password’s correctness. If an “if-statement” with a “==” is used as a verification function, the more correct password will provide a larger calculation time. This is because of the functionality of the “==” checks one character at a time. If a character is wrong the function terminates and a “false” statement is sent to the “if-statement” which implies shorter time spent in this function [24].

XSS, Cross-Site Scripting, attacks is another way of compromising valuable data. These attacks are usually put in two different groups, user input scripts (Stored XSS) or sending malicious code to the server scripts (Reflected XSS). Stored XSS is a piece of code that is sent to, for example, a comment section where it can be hidden as a picture. Whenever this picture is loaded in to the site it can execute the code and gather information for the attacker. Reflected XSS is, on the other hand, code directly sent to the web server. When this code is executed the user is sent to a copy of the website where cookies can be extracted and the user can then be sent back to the correct domain. These attacks will allow the attacker to use those cookies to gain access to the target site with a browser plug-in that allows cookie modification [12].

SQL injection is another sort of attack. This is where SQL code is sent to the server and later executed on the database. This allows the attacker to extract or modify information from outside database [24] [12].
Chapter 3

A Web-based mentor match system

This chapter will specify the requirements of the system and describe how all the requirements will be tested.

3.1 Requirement specification

Figure 3.1 is a use-case diagram that shows what different profiles are able to do on the website. The application will need to have three different kind of users to fulfill the requirements. These users are mentees, mentors and administrators, they will have different use-cases described in the diagram.

Figure 3.1: A use case diagram of the web-application
3.1.1 Login

All of the users need to be authenticated with a password to use the system. To authenticate a user one need to implement a login page. A login page is used in the majority of today’s systems where authentication is vital. The page usually consists of two text rows where the user will write their user-name in one and the password in the other. The user-name indicates which user wants to authenticate itself and the password verifies whether access should be granted or not. The login function will also identify which of the three users, mentees, mentors and administrators, is trying to login. The three users will have different starting pages. There will also need to be a logout option on the website that deletes the session and logout the user.

3.1.1.1 Testing

The login page will be verified by controlling that when the right username and password is entered the user will be taken to the correct page. Mentees should be taken to their personal profile page, the same is for mentors. Administrators on the other hand should be taken to their homepage. When a wrong username and/or password is entered the user should not grant access on the website. We also need to verify that if the user has entered wrong username and/or password more then 5 times within a hour period he/she should be blocked from logging in. There also need to be a verification that when a mentee that has not been approved by an administrator is trying to login they will not grant access to the site. Finally we will need to control that when a user logout the session is deleted and nobody else can use the session.

3.1.2 Register/Register new mentor

To gain access to the application a user needs to authenticate itself to the system through the login page. But to be able to do that he/she needs to be registered in the database. The users will do this by a registration page where new users leave their information to be used in future logins. This way the system can verify the authenticity of a user. We will require a username and a password from the user. The username is the name the user claims to be and together with the password can we authenticate a user to the name they chose. To be able to match profiles we will also need to require more personal information.

The questions will be similar to what mentees need to answer. One important part in order to fulfill some of the parts in GDPR is to consider all of the information we require so that no unnecessary information is stored and all data is motivated for the user. Since it is important for administrator to know which mentors they offer for mentees can mentors only be created by a administrator.

3.1.2.1 Testing

The registration page will be verified by controlling the that all the information that the user entered is stored in the correct table in the database. The password stored in the database should be hashed and not visible by anyone. This will be controlled by entering all of the required fields in the registration form and then control the database of what information has been stored. If required information is not filled in the registration should not be completed and a message should appear.

3.1.3 Change profile/Edit other profiles

According to GDPR all users on a website must have the right to view data and demand that all data about them will be deleted. Therefore should mentees and mentors be able to update their personal profile if something has changed from when they first registered. The administrator should also be able to change information about a registered profile if a user
need help. Administrators should also be able to delete a profile in case a user do not want to keep their data on the site.

3.1.3.1 Testing

This function will be tested by editing a profile as a mentor or a mentee and controlling the database that all the correct information has been changed for the correct profile. Secondly we need to edit a mentee and mentor as an administrator and controlling the database that the updated information is stored for the correct profile. Finally we also need to delete a user and confirm that no information about that user exists in the database.

3.1.4 View matching profiles

A matching between mentors and mentees will be done based on the attributes the profiles has entered through the registration. A mentee should be able to view which mentors has matching attributes with their profile and vice versa. They should also be able to view the full information about the matched profiles. The matching should contain an attribute that describes what the mentee/mentor want from the mentorship respective can contribute. A geographical aspect should also be considered.

3.1.4.1 Testing

This will be tested by creating test profiles, both mentees and mentors, one pair should have matching attributes and another pair that do not. Then we can verify that the correct matches is shown under the matching section in our website and in the database.

3.1.5 Apply for mentee/mentor

If a mentee believes one or more of the matched mentors would fit with them they should be able to 'apply' for this mentor. The admins should be able to see this and by that know whether someone have special wishes. The same function should be available for mentors.

3.1.5.1 Testing

This will be tested by applying for a mentor as a mentee. Then we can control that the administrator can see this in their view. There should also be an attribute within the mentee that indicates which mentor he/she applied for. This also needs to be tested for a mentor applying for a mentee.

3.1.6 Approve mentees

If a mentee is going to use this application it needs to be both registered as well as approved by an administrator. After a mentee has registered via registration page he/she should not be able to login until the he/she has been approved by an administrator.

3.1.6.1 Testing

This will be tested by register a new mentee and make sure this mentee is not able to login. The database should show that a administrator has not approved this mentee. We should also verify that when a administrator approve a mentee that user is able to login and also confirm that the database now states that the mentee is approved.
3.1.7 View matches/Make final matching

The administrators should be able to view all of the matches that exists among the mentees and mentors. The administrators should also be able to view matches that mentees and mentors apply for. When an administrator makes a final matching will the matched mentee and mentor not be able to match with somebody else.

3.1.7.1 Testing

This will be test by confirming that all the matches made on the website is stored inside the database. We also need to confirm that the matches in the database is also listed in the admin view.

3.2 Data model

The goal is to create a database that is easy to use and understand since the company wants to continue working on this implementation at a later stage. First we should ensure that only related data is stored in the same table. Data redundancy should be minimized, with other words should we not store the same data into several tables. One example of this is that there should be a specified table where both mentees and mentors can store information when attributes is the same, for example first name and last name. Although there should be a dedicated table for mentees and mentors for information that is role specific, for example if we have different questions in the registration form. To minimize the database calls it should be implemented so that information related to the same function should be considered to be in the same table, this would simplify and increase the functionality. The database should also include a table containing the matches that has been established during runtime. Due to security aspects the information regarding authentication processes should be contained within an own table. Figure 3.2 shows a data model that shows how the data should be organized.

Figure 3.2: A data model of the database
Chapter 4

Results

This chapter presents and discusses the implementation results.

4.1 Register/Register new mentor

In order to use and gain access to the application the user must be registered and have their information stored in the database. The registration form contains a couple of different questions that is of importance in either matching, login or general information purposes. Later when the user has filled the form and presses the "Apply now" button on the bottom of the registration form the information starts the registration process, see figure 4.1. This starts with a verification process of the password, the password and the confirmation password is checked to confirm that it is the same, has 6 or more characters and at least one uppercase letter, one lowercase letter and one number. The password is also hashed to make sure that if a breach occurs the password will not be available.

When the password is verified the information gathered through the registration form is sent to a function called register. In the first rows of this function is the username checked to make sure it is not already in use by somebody else. After that is the role specific information sent to a table called "mentee" inside the database and the shared information is sent to a table called "user".

The registration process for a mentor is very similar to the registration process of a mentee. The major difference is that it is the administrator that registers mentors. The inputs is also modified to fit the attributes that is necessary.
Figure 4.1: This is a snapshot of a the registration page.
4.2 Login page

The start up page of our application is a login page. This page contains two textareas and a registration button, see figure 4.2. The two textareas is where the user enter his or her username and password and later, if correct, sent to their respective user pages. The login process is containing a couple of different functions to make sure that the process works as intended. Due to security reasons and the verification password inside the database the password entered by the user is encrypted with an encryption algorithm called sha512 [25]. In the first part of the code the system checks if there is a username that is the same as the username filled in by the user. The encrypted password will then be verified against the encrypted password inside the database with an built in function called password\_verif(), the parameters are the passwords that needs to be verified. This function controls that both passwords is the same and also prevent timing attacks by having a fixed calculation time [26]. Timing attacks is, as mentioned previously, when someone measures the time spent calculating the password’s correctness in order to figure out someone else’s password. The login process also include a function to prevent bruteforces. This function keeps track of failed login attempts by users. It stores timestamps in the database to count and prevent more than five failed logins in a row for one user.

![Login Form](image_url)

Figure 4.2: This is a snapshot of the login page.

Later the system needs to know which type of role this user have. This is solved with function called roleCheck(). The paramteters are which role to check, the ID of the user and the link towards the database.
4.3 Change profiles/Edit other profiles

The application has three different kind of roles, mentees, mentors and administrators. All of the mentioned roles have different kinds of profile views.

4.3.1 Mentees and Mentors

Mentees and mentors both have the ability to change their own personal data via their respective profile page. We have implemented this through another form that shows the current attributes that is stored in the database which the user then can change. The changes is sent to a function which takes the changed attributes and replaces the old stored data in the database. After these functions the profile has been updated and the user can see the changed result on the personal profile, see figure 4.3.

![Figure 4.3: This is a snapshot of a mentees profile page.](imageurl)
4.3.2 Administrators

Administrators should have the ability to change another user’s profile. We have solved this with a page where the admin can read and navigate through all profiles. Through this page the admin can later go to a specific profile and edit their information or delete the user from the database. The function prints out the information about each user in a two dimensional list that the admin can navigate through and look at all mentees respectively mentors separately, see figure 4.4.

![Alla profiler](image)

**Figure 4.4:** This is a snapshot of what an administrator sees when looking at all the profiles.

When the administrators clicks on a profile he/she enters the same looking page as mentioned above where the mentees and mentors edit their own profile. The system then looks at the database for the ID that the administrator clicked on and writes out their information to edit.

4.4 Matching

The matching is applied via the personal profile of each user. It is based on the attributes learn/teach, postal number and the distance attribute. The matching starts with teach respectively learn attributes. If a mentee is using the system it will check the database for a mentor with a teach attribute that is the same as the mentees learn attribute. After that it iterates through the result sent back from the database and is getting the attribute postnr and distance from the mentee and postnr from the mentors. With these results we can now retrieve the coordinates for each postal number through a web-based API in which coordinates for most Swedish postal numbers are stored [27]. When we have the coordinates we can calculate the distance between the points with the help of the haversine formula [28]. The result then is compared to the stored distance from the mentee and if the distance is larger this is a match that will be stored in the database. The mentee can now see the resulting mentors on the matching page, see figure 4.5.
4.5 Approve users

This requirement is solved in two separate ways for mentees and mentors. As mentioned in registration the mentors is created through an administrator. Because of this the mentor is already approved by admin and will not have an attribute controlling this. The mentees on the other hand needs to be verified since anyone can register on the application. We have solved this through the admins first page where we list the unapproved mentees, see figure 4.6. In that list we have put check boxes for each unapproved mentees so the admin simply can check the box and approve more mentees at once. The system then tells the database that new users has been approved and where changes has been made.

Figure 4.6: This is a snapshot of an administrator’s point of view when approving mentees.
4.6 Database implementation

The database is containing seven tables that are connected through relational attributes. The tables are members, user, mentees, mentors, admins, matches and login_attempts, see figure 4.7.

![Database Diagram]

Figure 4.7: This figure represents the design of the database used in this application.

4.6.1 Members

The members table is containing three different kinds of rows, ID, username and password. ID is the primary key of this table and also the number that the user is going to be recognized with during the usage of the application. The username is unique for each user and is also the name that the user have chosen to authenticate itself with during the login process. The password is encrypted in the database and is also chosen by the user to verify the authenticity of a user trying to login with the username. Both the username and the password attributes is only used during the login process.

4.6.2 User

The user table is containing nine different attributes describing each user, both mentees and mentors. The ID is a foreign key attribute and refers to the primary key, ID, in members. It, as mentioned above, identifies each user uniquely. Firstname, lastname, dob(date of birth), and gender is meant to describe the user as the names suggests. The email attribute is the way of contacting the user, other(other information) and posabout (positive about) is storing other information that might be relevant and a positive attribute of the user, all of this is register process inputs. The distance attribute is also filled in by the user and is used in the matching process.

4.6.3 Candidates

The candidates table is containing the information about mentees which has the following attributes, ID, candidate, approved, learn, postnr and whymen. The ID attribute is the foreign key and refers to the primary key in the Members table. The candidate attribute is a unique id to identify the mentee in the matches table, more on that later. The attribute approved tells us if the mentee should be able to log into the system or not. The learn attribute is filled in from the register process and is meant to describe what the mentee
wants to learn during the mentorship. Whyment (why mentor) is also required during the registration process and is describing why the mentee wants/needs a mentor. This attribute was required from WiTEC for them to be able to validate the sincerity of a mentee. Postnr (postal number), as the name suggests, is the postal number of the user.

4.6.4 Mentors

The mentors table is containing four attributes, ID, mentor, teach and postnr. The ID attribute is the foreign key and refers to the primary key in the Members table. Same as above is mentor a unique id to identify the mentor in the matches table. The teach attribute is added in the registration process and is describing what the mentor can bring to the mentorship. Postnr (postal number), as the name suggests, describes the postal number of the user.

4.6.5 Admins

The administrators table only contains one attribute and that is ID. It is a foreign key to the primary key of Members, ID, and is used to indicate which user is an administrators.

4.6.6 Matches

The table called matches is, as the name suggests, the table that saves the matches that has been made on the system. When a mentor or mentee views the matches these will be stored in this table. It is containing 3 rows, Mentor, Candidate and Approved. The Mentor is a unique identification number, ID, of the respective mentor in the match and the candidate attribute is the unique ID of the mentee. The Approved attribute is the component the admin provides which acts as a confirmation to the match.

4.6.7 Login_attempts

Login_attempts is the table that collects data and allows the system to prevent brute force attacks. In the table there are 2 rows and they are ID and a timestamp. The ID indicates which user who tried to login and the timestamp concludes when the attempt occurred.

4.7 Testing

Most of the tests where successful but there are a few that did not pass. These tests will be mentioned in this section and it will also state why the test did not pass.

The login function and the registration for both mentors, mentees and administrators passed all the tests. The function edit other profiles, did we did not have the time to implement a function that deletes a user and therefore we could not test if all of the data in the database is deleted when deleting a profile. The other tests created for these functions did, on the other hand pass the test. View matching profiles passed all of the tests. We did not implement the function, Apply for mentee/mentor, again because of not having time and therefore did these test not pass. Approve mentees passed all of the tests. View matches/Make final matches did not pass the tests since we did not have time to implement these functions.
Chapter 5

Discussion and conclusion

This chapter summarizes the project and also present potential directions.

5.1 Did we fulfill requirements for the system?

While reviewing the requirements in section 1.3 System requirements we could establish that we have fulfilled the majority of the requirements but there are small parts missing. We have a personal profile for each type of role including administrators, mentees and mentors. Administrators does have the ability to change any mentors or mentees personal data while also being able to create new mentors. Before the mentee is able to login and use the system he/she must be approved by an administrator. Both mentees and mentors is able to see and edit their own personal data. They can match with the other role but they can not apply/wish to match with a specific profile. Due to lack of time when creating the administrators, they are not able to see matches and make the final matching. While learning more about EU-regulations we quickly concluded that we as developers can not guarantee that the regulations will be fulfilled. We can not say that the organization will not be using that data for something that we have not established in the documentation that is required in order to be within the regulations. This is therefore something that the organization itself needs to address. We have implemented security to make the system safer and but since the technology is always advancing we can not assure that this system is safe in the future.

5.2 What could we have done for a better result?

As mentioned above we did not fulfilled all of the requirements and unfortunately we feel that there is a lot of things that we could have done to deliver a better result. The biggest problem we had in the end was the lack of time, the project was a lot harder than what we had planned from the beginning and therefore took a lot more time. Afterwards we realized that there are a few big things we could have done to save time. The most significant part is that we should have started the project from the other side. We created the database first and the client-side second we have now after the project is done realized that it would have been a lot easier to do the opposite way. It is a lot easier to realize what is needed to get a good user experience when looking at the client-side than just looking at the server-side. Since we did not have a clear picture of the website beforehand we needed to do a lot of time-consuming changes to the database after we had created the lay-out of the website and also could get feedback from the organization that we worked with. Another important part is that we should have narrowed the project down and focused on maybe just a few things. If somebody with more experience had done the project they could absolutely had made it
in time but since we had so little experience about website development from the beginning the project became really hard to both plan and develop.

5.3 Related work

In the previously mentioned article [2] was different strategies for Mentor Matching was presented. They concluded that a centralized database is not a good solution since there are so many different careers to choose from and therefore really hard to find both a mentor and a mentee that wants the same if the database is not specialized into something specific. They also concluded that a personalized matching through their own network was not a good solution either because of how much time it took.

What we have developed is exactly a centralized database. Then why would the database we have created work if they concluded that it will not? We believe our solution is different since we have developed a database that is not relaying to much on the database to do the matching. The task for the database is rather to categorize the users that are registered so that the administrator can choose from fewer people while they make the final matching. With this solution, we reduce the problem of manual matching taking to much time.

Although, a risk with our solution considering the article could be that WiTEC is trying to match people from too many areas. Therefore there might not be enough mentors in a particular area that knows about WiTECs website. This could make potential mentees goes to another matching site that is more focused in their specific area. Therefore we suggest that WiTEC needs a very clear picture in what areas they want to do the matching, it would be a good idea to start small to get a good reputation and many mentors and then grow and do the matching in more areas when more people know about the site.

The second article about telementoring within surgery [3] has a very similar idea to our own project. They want the system to narrow down the potential matches but then instead of making the administrators make the final matching the mentees can choose which mentor they want. This could make the matching process faster since the users do not have to be matched by an administrator before knowing what mentor they will get but it could also create less beneficial mentorships. The article does not evaluate the method in practice but it does mean that more projects propose similar solutions to ours. Although in this article they are focusing on smaller areas of profession than WiTECs idea is which highlights their need to narrow down the area of focus.

5.4 The future for the project

The future for this project looks bright as we have spoken to the board of the organization and they are planning to continue the development in the matter of design as well as functionally. The plan is to create a potential product in which might be either saleable and also an organizational product. Since WiTEC is a European wide organization it has potential to get a larger user base. We have also stated to WiTEC that we will be available through future development and by that simplify the learning process for different student groups when for example adding more design. We think this system has a lot of potential and simplifies the procedure and can help people to get to where ever they want to be in their career or life in general.

5.5 Conclusions

Due to the knowledge position we had beforehand, perspective of time and approach we still believe that these results is okay. We have learned a lot during the process of this project as development, presentation and work experience, what an organization or company expects from the developer. We have learned how to implement a website including
four new programming languages, using a solution stack and increased our knowledge in database implementation. We now know that following a methodology used as a plan is more comfortable to work with rather than none. We have acknowledged our mistakes and learned from them while also handling them during the development period.
Bibliography


Appendix A

Website snapshots

```javascript
function loginFormhash(form, password){ //Handles the hashing of the password
    var p = document.createElement("input");
    form.appendChild(p);
    p.name = "p";
    p.type = "hidden";
    p.value = hex_sha512(password.value);
    password.value="";
    form.submit();
}
```

Figure A.1: This code represents the function in which the password is hashed.

```javascript
if($stmt = mysqli_prepare($link, "SELECT ID, PW FROM members WHERE username= ?")){
    mysqli_stmt_bind_param($stmt, "s", $username);
    mysqli_stmt_execute($stmt);
    $result = mysqli_stmt_store_result($stmt);
    mysqli_stmt_bind_result($stmt, $ID, $db_password);
    mysqli_stmt_fetch($stmt);
    if($stmt->num_rows == 1){ //Checks if there is a user
        if(checkBrute($ID, $link) ==true){ //checks if it is a bruteforce attempt
            echo "brute";
            return false;
        }else{
            if(password_verify($password, $db_password)) { //checks if the passwords match
```

Figure A.2: This code represents the passwords verification function.
Figure A.3: This code represents prevention function of brute force attacks.

```javascript
function checkBrute($ID, $link) {
    $timestamp = time();
    $valid_attempt = $timestamp - (60*60);

    if ($stmt = mysqli_prepare($link, "SELECT time FROM login_attempts WHERE ID = ? AND time > '$valid_attempt')") {
        mysqli_stmt_bind_param($stmt, "i", $ID);
        mysqli_stmt_execute($stmt);
        mysqli_stmt_store_result($stmt);
        if ($stmt->num_rows > 0) {
            return true;
        } else {
            return false;
        }
    }
}
```

Figure A.4: This code represents the function where the passwords length and content is checked while also being hashed.

```javascript
function regForHash(form, username, password, conf) {
    if (username.value == '') {
        password.value = ''; 
        conf.value = '';
        alert('You must provide all the requested details. Please try again');
        return false;
    }
    if (password.value.length < 6) {
        alert('Passwords must be at least 6 characters long. Please try again');
        form.password.focus();
        return false;
    }
    var re = /^[a-zA-Z0-9!@#$%^&*()~`_+=\|;:,.<>/?{\[\],\|\-\_\=\+\?\]}/
    if (re.test(password.value)) {
        alert('Passwords must contain at least one number, one lowercase and one uppercase letter. Please try again');
        return false;
    }
    if (password.value != conf.value) {
        alert('Your password and confirmation do not match. Please try again');
        form.password.focus();
        return false;
    }
    var p = document.createElement("input");
    p.name = "p";
    p.type = "hidden";
    p.value = b64_sha12(password.value);
    password.value = "";
    conf.value = "";
    form.submit();
    return true;
}
```
function register($username, $pw, $firstname, $lastname, $gender, $spoturi, $spoons, $skills, $other, $email, $dob) { 
    mysql_prepare($link, "SELECT ID FROM members WHERE username = ?", true); 
    mysql_stmt_prepare($stmt); 
    mysql_stmt_store_result($stmt); 
    mysql_stmt_bind_result($stmt, $ID); 
    if ($stmt->num_rows == 0) { 
        // Check if it is an available email 
        mysql_prepare($link, "INSERT INTO members(username, PW) VALUES (?, ?)"); 
        mysql_stmt_execute($stmt); 
        mysql_stmt_store_result($stmt); 
        mysql_stmt_bind_result($stmt, $ID); 
        mysql_stmt_fetch($stmt); 
        $link = mysql_fetch_row($stmt); 
        $sql = "UPDATE users SET approved = 1 WHERE ID = $ID"; 
        $stmt = mysql_query($sql); 
        if ($stmt) { 
            return true; 
        } 
    } 
    return false; 
}

$sql = "SELECT ID, approved FROM candidates"; 
if ($result = $link->query($sql)) { 
    if ($result->num_rows == 0) { 
        $sql = "UPDATE candidates SET approved = 1 WHERE ID = $ID"; 
        $stmt = mysql_prepare($link); 
        $sql = "UPDATE candidates SET approved = 1 WHERE ID = $ID"; 
        $stmt = mysql_query($sql); 
        if ($stmt) { 
            return true; 
        } 
    } 
    echo "<tr class="odd-md-4" >"; 
    echo "<label class="checkbox-inline" >"; 
    $id = $row['ID']; 
    echo "<input type='checkbox' name='id[$id]' value='true' >"; 
    echo "<input type='submit' value='Approve' >"; 
    echo "</label>"; 
    echo "</td>"; 
    echo "</tr>"; 
} 

Figure A.5: This code represents how and what information is sent to the database.

Figure A.6: This code represents how the administrators is able to approve new mentees.
Figure A.7: This code represents the function and calculations of the matching process.

```php
function roleCheck($role, $ID, $link) {
    if ($link != mysql_prepare($link, "SELECT ID FROM role WHERE ID = "$ID")) {
        mysql_stmt_execute($stmt);
        $result = mysql_stmt_store_result($stmt);
        mysql_stmt_bind_result($stmt, $ID);
        mysql_stmt_fetch($stmt);
        if ($stmt->num_rows == 1) {
            if ($role == "candidates") {
                return true;
            } else {
                if ($stmt = mysql_prepare($link, "SELECT ID FROM candidates WHERE ID = "$ID AND approved = 1")]) {
                    mysql_stmt_execute($stmt);
                    $result = mysql_stmt_store_result($stmt);
                    mysql_stmt_bind_result($stmt, $ID);
                    mysql_stmt_fetch($stmt);
                    if ($stmt->num_rows == 1) {
                        return true;
                    }
                }
            }
        }
    }
    return false;
}
```

Figure A.8: This code represents the function in which the role of the user is established.
Figure A.9: This code represents a part of the profile change functionality.

```php
if(isset($_POST['Firstname'])){
    $ID = $_SESSION('user_id');
    $firstname = $_POST['Firstname'];
    $link = query("UPDATE 'user' SET Firstname = '$firstname' WHERE ID = '$ID');
}
```

Figure A.10: This code represents a part of the function sending the information to the database.

```sql
SELECT ID, whynam, learn, postnr FROM candidates;
if($result = $link->query($sql)){
    while($row = $result->fetch_assoc()) {};
}
```

Figure A.11: This code represents how we list the mentees when showing the user all profiles.
Figure A.12: This code represents how we list the mentors when showing the user all profiles.
Appendix B

Timeplan

<table>
<thead>
<tr>
<th>Activity</th>
<th>Responsible</th>
<th>Week</th>
</tr>
</thead>
<tbody>
<tr>
<td>Research</td>
<td>AJ, JS</td>
<td>6 7 8 9 10 11 12 13 14 15 16 17 18 19</td>
</tr>
<tr>
<td>Client- part</td>
<td>AJ</td>
<td></td>
</tr>
<tr>
<td>Server- part</td>
<td>JS</td>
<td></td>
</tr>
<tr>
<td>Database- part</td>
<td>AJ, JS</td>
<td></td>
</tr>
<tr>
<td>Half-time report</td>
<td>AJ, JS</td>
<td></td>
</tr>
<tr>
<td>Security implementatation</td>
<td>AJ, JS</td>
<td></td>
</tr>
<tr>
<td>Testing</td>
<td>AJ, JS</td>
<td></td>
</tr>
<tr>
<td>End report</td>
<td>AJ, JS</td>
<td></td>
</tr>
<tr>
<td>First Prototype Done</td>
<td>AJ, JS</td>
<td></td>
</tr>
</tbody>
</table>
Alexandra Johansson

Julian Sjödal