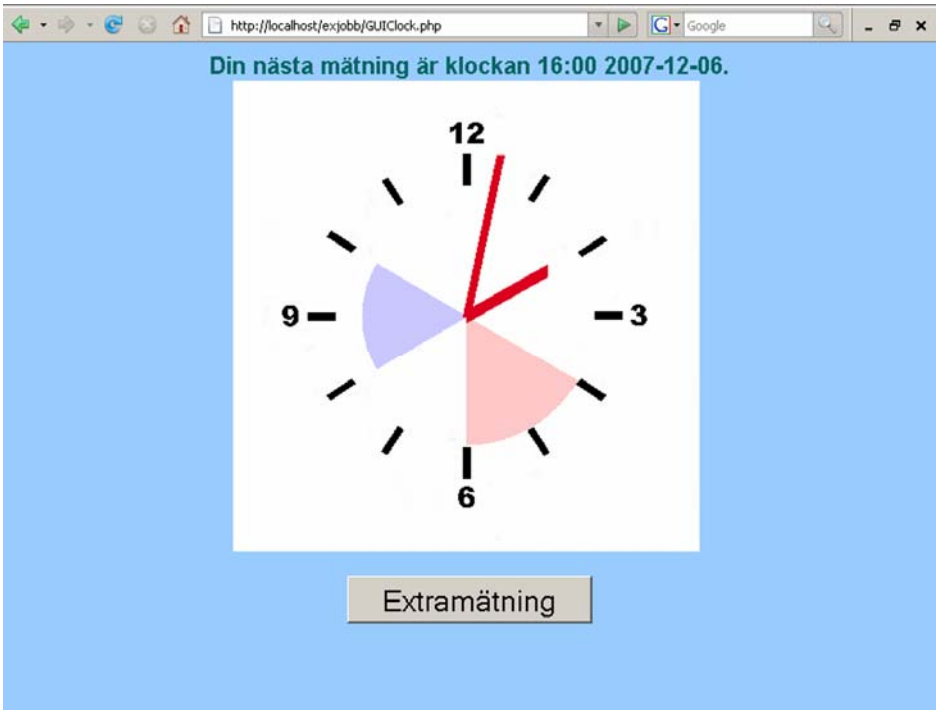
CHALMERS



Design and Implementation of a Web-Based User Interface for Home Care Applications

Master of Science Thesis

LINDA IVARSSON

Department of Signals and Systems
Division of Biomedical Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden, 2007
Report No. EX103/2007

Cover:
The clock page is used as a screensaver between the measurement sessions. For more information, see chapter 3.1, Graphical user interface.

Göteborg, Sweden 2007

Design and Implementation of a Web-Based User Interface for Home Care Applications

LINDA IVARSSON
Department of Signals and Systems
Chalmers University of Technology

Summary

With a population growing older for each year, the need for cost-efficient health care is growing as well. One way of facing these demands, as well as simplifying the everyday life for the chronic invalids, is to make it possible to perform measurements and monitoring at the patient's home.

The aim of this master thesis was to produce a web-based application based on an existing computer programme, making it possible for the patients to perform a set of scheduled measurements at home or anywhere where an Internet connection is available. The application was built using mainly PHP and HTML.

The report is written in English.

Keywords: e-health telemedicine, web, internet, homecare, heart failure.

Acknowledgements

A lot of people have been involved in this project. At first I would like to thank my tutor, Bengt Arne Sjöqvist, and of course Anna Gund, always helpful. Then there are all those who have helped me with PHP, HTML and other issues from the computing world: Lobo Olsson, Mattias Runge, Magnus Våge, and, most of all, Johan Haldén. Thank you for easing my frustrations!

Table of Contents

1. Introduction

1.1 Why eHealth?

As a result of better health care and a change of life style, our population grows older, but that also means that it becomes sicker. A decreasing part of the population must support the increasing part of the elderly, and this requires a new, more efficient health care.

One way of facing those demands is by introducing eHealth. eHealth provides, only to mention one prospect, the possibility of patient monitoring over distance, which is a great advantage for both the patient and the care giver, in many ways.

In this project, Care@Distance, daily scheduled measurement and the patient's subjective answers to a set of questions is intended to form the basis of improved treatment and good disease management by long-time monitoring of the patient's condition. It also provides the possibility of alarming when the results show too deviating values. Collecting large amounts of data on a patient enables tailored care, which is more efficient and leads to less frequent emergency situations. This is beneficial to the patient, who gets improved life quality, and to the care giver, who becomes more cost-efficient [1].

1.2 Heart Failure

Heart failure is also known as congestive heart failure (CHF) or congestive cardiac failure (CCF). It is a condition where the heart's ability to pump a sufficient amount of blood throughout the body is impaired [2]. There are many causes of heart failure, e.g. hypertension, myocardial infarction (heart attack), inflammation, cardiac valve disorders or genetic diseases [3].

Heart failure is a common illness. In Sweden, about 250 000 patients are diagnosed with the disease, equivalent to 2-3 percent of the total population. Every year, 30 000 new people develop heart disease. About two thirds of the patients are older than 75 years. The annual mortality is, even with the best treatment, 10 percent [4].

1.3 Purpose

The purpose of this master thesis was to provide a web service for people suffering from heart failure. It is based upon a master thesis written in 2004 by Robert Carlsson and Andreas Rudolfsson, where a system including a care giver's interface, a patient's interface and a database connecting them was constructed. The task was to convert the existing patient's interface to an internet solution [5].

The result is adapted to monitoring of heart failure patients, but it is designed to be easily changed to suit the use on other patient groups with chronic diseases, e.g. diabetics.

2. System Description

A system built for homecare patients, mostly elderly, needs to be easy and intuitive to handle. Other demands are compatibility with most browsers (at least Internet Explorer, Firefox and Opera), stability, swiftness and smoothness. Because of the usability aspects, with users that can be expected not to have high level of computer experience, the system can not be too complicated. This also gives us good conditions for a fast and robust application, since an application without many complex features can be constructed in a simpler way.

2.1 System Parts

The existing system consists of a database including patient data and data regarding measurements to be made, a program installed on the caretaker's client, and a web interface used by the caregiver to customize measurements and monitor patients [5].

This project was to be concentrated on the caretaker's part of the system, and to convert the existing program into a web based solution. The database had to be included in the work, although this was not intended from the beginning. Only the caregiver's interface is not covered in this report.

2.2 Choice of Languages

Because of the different parts of the system, different languages had to be used. The visual user interface is one part, the actual functionality another and the database system a third. The application was constructed used using HTML, completed as described in section 2.2.1, PHP and MS SQL Server.

2.2.1 Visual Web Interface

HyperText Markup Language (HTML) is a markup language, meaning a language containing both text and information about the text. It is the most widely spread for web pages, and it has built-in support for forms, a feature necessary in the project. It is also fast and easy to learn, and can be created using a simple text editor [6].

Cascading Style Sheets (CSS) is a stylesheet language used as an addition to HTML. It stores all style definitions such as background colour or font sizes in one single file and is used to describe the presentation of an HTML document. A CSS file allows the web programmer to apply a special style to a multiple set of web pages, and also simplifies changes [7].

To add more functionality to the web pages, JavaScript was used. JavaScript is the most commonly used Internet client side scripting language. The JavaScript is used together with HTML to make the web pages more dynamic and to enable extra – in some cases necessary - features, e.g. setting cookies, opening popup windows or reacting on events. Since there are possible browser compatibility problems using JavaScript, it should be used with consideration [8].

Ajax, Asynchronous JavaScript and XML (where XML stands for eXtensible Markup Language), is a name for a set of different technologies used to further increase the interactivity of a webpage. Function calls are still made in JavaScript. An object of special interest is the so called XmlHttpRequest object, allowing the webpage to contact the server without having the page reloaded. This method is a way of saving resources, since the processing can be made on the client computer instead of the web server. Ajax is known to be compatible with most browsers [9].

The Ajax functionality is implemented using Prototype, a JavaScript framework for simplifying web development. Prototype includes the Ajax library used in this application, and is browser independent [10].

2.2.2 Script Language

Finding the right script language appeared to be a choice between .NET Framework, Java and PHP: Hypertext Preprocessor, PHP. To gather opinions, different web forums and blogs (web based diaries) were read and a number of people questioned. Everybody had their own, subjective meaning, but finally, it was possible to make up an own picture.

.NET is Microsoft's managed code programming model, i.e. for building applications and user interfaces. Many pre-coded solutions are featured, making the programmer's job easier, and it also has the great advantage of making it possible to combine different languages. .NET is precompiled (in opposite to PHP), which makes it faster when the application is big. As a con, .NET is currently only available in its complete form on Windows platforms. Also, applications running in .NET tend to require more system resources. An installation requires 800 MB hard disk space [11].

Java is very common in online applications and has functions to communicate with SQL (Structured Query Language) databases. It is also a language taught at Chalmers, already known by the author, and is also precompiled. It is well suited for larger web constructions, and is independent of platform. Applications built in Java (as well as in .NET) tend to be well structured. A minimum of 98 MB free disk space is required. [12]

PHP is an HTML-embedded script language designed for creating dynamic web pages, and can access virtually any database available. It is suited for small or middle-sized web pages, but gets slower when the application grows bigger, because of the fact that it is interpreted and not compiled. It is also platform independent and is easy to learn. Another great advantage is that PHP is free of charge and that the PHP Group provides the complete source code for users to build, customize and extend for their own use (open source). This does have its cons, too, since this development style leads to shortcoming in quality assurance. Older versions of PHP did have security issues, but they have been corrected. Yet, it is considered stable, meaning that the server does not have to be rebooted often, and that the software does not change incompatibly between releases. PHP does not require much space, about 8 MB [13], [14].

Both Java and PHP were good alternatives (.NET was not considered because of the platform dependence, which was a too weighty reason) without any obvious cons, but finally, PHP was chosen. It is simply a neat and easy way of designing web applications this size. Regarding the short time available for learning also made the fact that it is easy to learn important, as well as the easy accessible support.

2.2.3 Database

One natural choice of database together with PHP is MySQL, a widely spread open source SQL database management system. This was not an option in this case, partly while the hospitals – our potential customers – are used to working in a Microsoft environment, but mainly since the existing database system was Microsoft SQL Server. The query language used is Transact-SQL, an implementation of ANSI/ISO standard SQL [15].

2.3 Preparation

The author's previous programming experience was limited to Java and some HTML, and the project had to start with a week of studying the languages to be used, concentrating on

PHP [16]. Further knowledge was gained throughout the project, mainly by information obtained through the Internet [14], [17], [18], [19].

3. Application

3.1 Graphical User Interface

The application is divided into four parts, presented in the following section. Information about scheduled measurements and other patient specific data is obtained from the database, where it is stored by the caregiver. For further technical details, see appendix A. The structure of the application is the same as in the preceding master thesis [5].

The application is optimized to fit a screen resolution of 1024 by 600 pixels, to fit the screens planned to be used in this project. The limiting factor is the height (600 pixels). Displaying the application is possible on any screen; the only difference will be the size. If displayed on a screen with lower resolution, the pages will be enlarged and the patient will have to scroll to see everything. If a screen with higher resolution is used, the pages will look smaller. If necessary, most adjustments can be done by changes in the CSS-file. All files are described in appendix A, and a flowchart of the most important ones is presented in appendix B.

There are three main sets of colour schemes: analogous, complimentary, and monochromatic. To obtain a soft, calm feeling, an analogous colour scheme was chosen, meaning that only colours from a certain range of adjacent colours of the colour wheel are used. The background colour was set to pleasant light blue, with text in a dark blue-green nuance to gain a good contrast [20].

The appearance of the application has been tested using Mozilla Firefox 2.0, Internet Explorer 6.0 and Opera 9.24.

3.1.1 Login and Clock Page

The first thing for the patient to do is logging in, by filling in a form. If a private computer is used, the possibility to save the login data is supplied by checking a box, meaning that cookies will be set. The login process is done using two files, one offering the form to fill in, and a processing file where the database comparison and cookie handling is made.

The clock page is more or less a screen saver, showing when the next measurement is to be performed. The page automatically switches over to the application start page when the time is due. It is also possible to perform an extra measurement outside the normal time slots by pressing a button on this page.

The clock page is refreshed every minute to give the impression of a real clock with moving hands. Each time the page is reloaded, a time check is performed to see if it is time to redirect to the start page. The clock is drawn on an existing image by a separate PHP file.

3.1.2 Start-up

The first page shown when a measurement session is started is a start page greeting the user. This start page includes a function that redirects back to the clock page if the end time of the session is exceeded; using a refresh every minute and a time check as on the clock page. To start a session, a start button has to be pushed. If this page has been reached by pushing the extra measurement button on the clock page, a button for regretting the extra measurement will be featured, giving the user the opportunity to go back to the clock page.

The next page is presenting the measurements to be done to the patient. This is also where most of the database calls are made, using the username stored from the login process to get the right data.

3.1.3 Measurements

Two kinds of measurements are implemented in this first version: Text fields and radio buttons. Support for extending the measurement with text areas, checkboxes or dropdown menus is included, but not fully developed.

Text fields are used for measurements made with instruments, e.g. scales. Radio buttons are used for answering questions with discrete numeral answers. All measurement pages are generated from the same PHP-file and specified by data from the database. Every time the form page is submitted, its content is processed by a separate file. This file stores the value but also performs certain controls, e.g. that the question is actually answered, or that the answers not exceed specified and possibly personal limits. When the answer is safely stored, the user is sent to the next measurement page, until the array storing what measurements to be made is empty. Then, the last part of the session is entered.

3.1.4 Data Storing and Logout

All data is stored in a session variable and is not transferred to the database until the end of the session. The user is presented to the measurement answers, and can choose between going back to make changes, or confirming them. By clicking the "Skicka"-button, a certain file is called with the only assignment to send the collected data to the database and redirect to a page with information about the next measurement session.

To end the session, the user can choose between ending ("Avsluta") and logging out. If "Avsluta" is chosen, the user is just sent back to the clock page and can wait for the next measurement. If choosing to log out, the PHP session variables, corresponding to PHP's global variables, storing the user information are destroyed in a logout file and the user is sent to the login page. To make another measurement, it is necessary to log in again. Closing the browser will also generate a logout, since this too destroys the session variables.

3.2 Application Walkthrough

A complete measurement session will be demonstrated.

The first page to reach is GUILogin, see figure 1.

.

Figure 1. GUILogin.

The user enters his/her username and password (in this case "kaj" and "kaj") and presses "Logga in" to submit. If the user name or password is wrong, the user is sent back to the login page, displaying a message saying that the login did not succeed. Otherwise, the user is sent to GUIClock.

On GUILogin, there is also the possibility to check the box to set cookies and store username and password on the computer. Also, a help text is provided by pressing the "Hjälp"-button.

The clock page in figure 2 is the page shown between the measurements. The upcoming measurements are shown as pies, blue ones for mornings and pink for afternoons, and with different radiuses depending on when the measurement is to be performed. Morning pies are shown with a shorter radius than afternoon measurements.

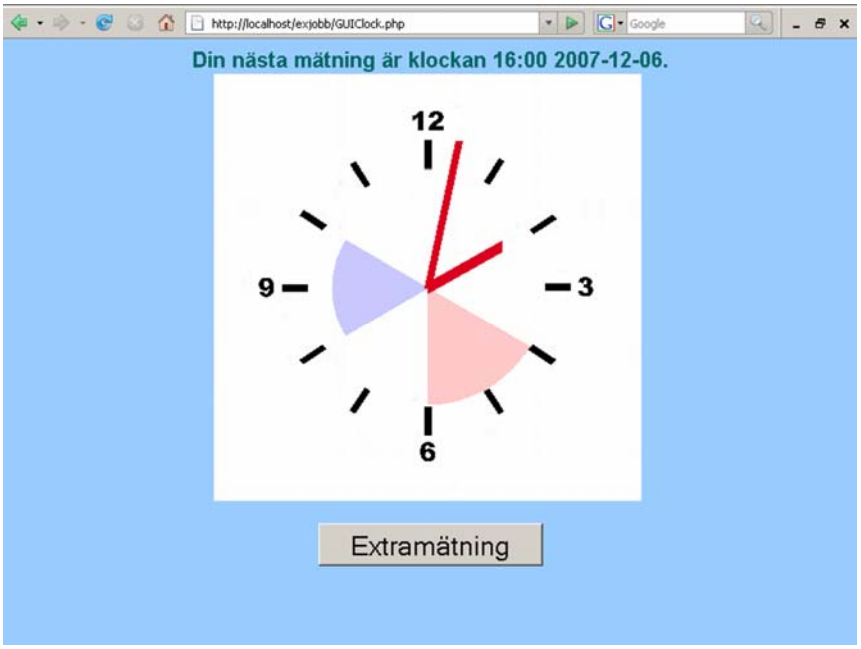


Figure 2. GUIClock.

The button saying "Extramätning" supplies the patient with the possibility to perform a measurement outside the scheduled times. Otherwise, the page automatically redirects the user to GUIStartPage (figure 3) when the time is due.

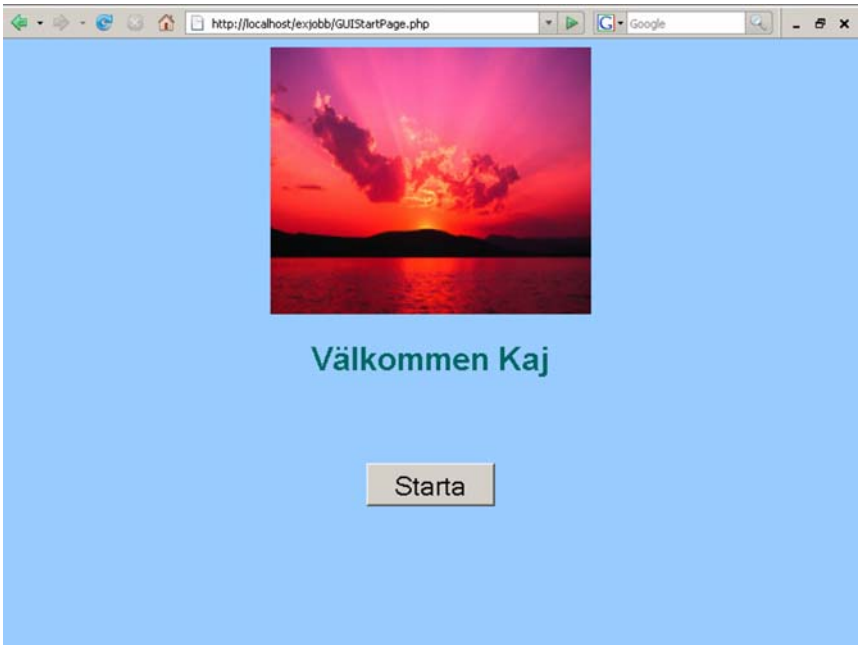


Figure 3. GUIStartPage

If this page is reached when performing an extra measurement, a regret button is shown, giving the user the choice of cancelling the measurement and returning to the clock page. By pressing "Starta", the session begins. If the button never is pressed, a function redirects back to the clock page when the end time for the session is exceeded. In that case, the database will be updated with a normal session id, a measurement id saying 0 (meaning missing measurement) and the answer null.

Otherwise, the page coming after the start page is GUIToDoPage, see figure 4.

Figure 4. GUIToDoPage.

This is where the patient is presented to the upcoming measurements, and is also given the possibility to end the session by pressing "Avsluta". If pressed, a text asking the user if he/she really wants to quit is shown, as well as two buttons. If the user really ends the session, the database will be updated as in the previous example. Otherwise, the user enters the measurement pages.

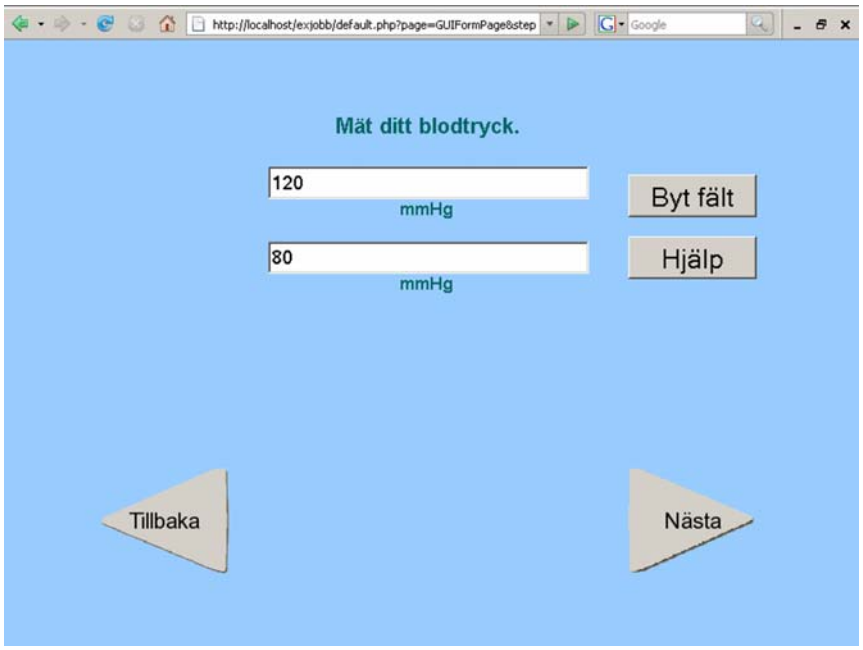


Figure 5. GUIFormPage.

A typical measurement is shown in figure 5. This page presents the possibility to get help, shown as a text beneath the text fields and an image to the left, by pressing the "Hjälp"-button, and button to change the text field focus. It is only possible to enter numbers, backspace, arrows or delete in the fields, except for when performing a weight measurement, which also allows commas. After submitting the form, the answer is checked

not to be null, not to exceed the specified limits for that specific measurement and patient and that other measurement specific requirements are fulfilled, i.e. that only one comma is allowed when performing a weight measurement. If any of the limitations are overstepped, the patient will be alerted by showing a text at the bottom of the measurement page. If the question is not answered, the patient is alerted in the same way, but is also given the choice to ignore the message and continue the measurement session by pressing an upcoming continue button.

A measurement can also be presented as a question with multiple choices, see figure 6. The answers can be chosen by either clicking them as usual, or by using the numbers on the keyboard. Pressing a letter key will not give any result, and pressing a number key not representing anything on the screen will give an error message shown at the bottom of the page.

Figure 6. GUIFormPage with radio buttons.

After performing all scheduled measurements, GUIConfirmPage is displayed.

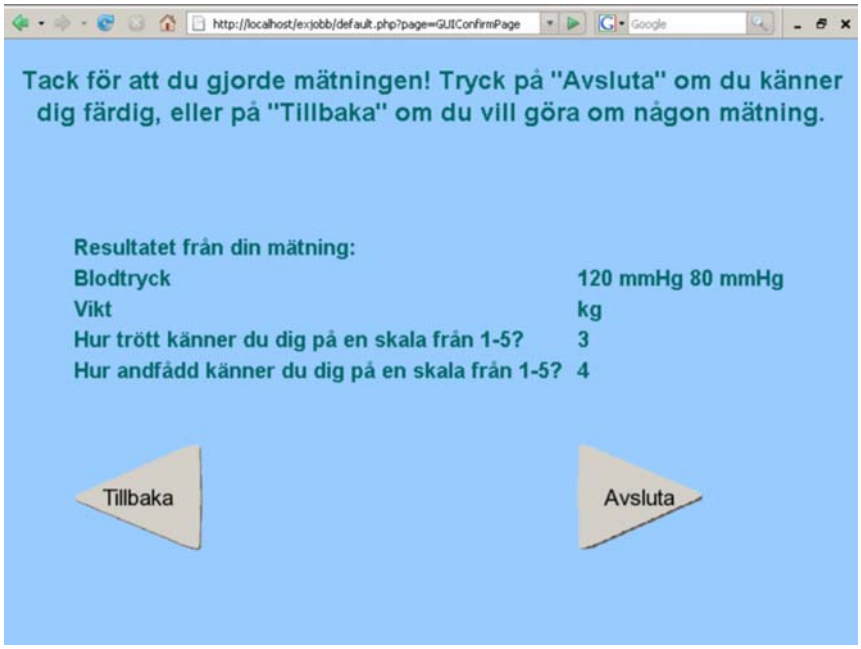


Figure 7. GUIConfirmPage.

As shown in figure 7, the patient has chosen not to perform the weight measurement. If the patient has had a change of mind, this is the opportunity to go back and complete the measurement. By pressing "Avsluta", the answers are stored in the database and GUIGoodbye is displayed, see figure 8.

Figure 8. GUIGoodbye.

By choosing "Avsluta", the user is sent back to the clock page, and by choosing "Logga ut", the user is logged out and login page is displayed.

3.3 Database

The database containing all information necessary did already exist, and was a part of the preceding master thesis mentioned in section 1.3. During the work, it became clear that

alterations had to be made. There was a need for new information storage, such as the patient's username and password, and some structural changes. No existing tables were deleted, but some were extended, and new tables were added and used in stead of the old ones.

The database structure is described in appendix C. It is important to keep in mind that this is a suggestion, not a complete database. Still, the application is built based on this suggestion.

3.3.1 dbo.Patients

To this table, two columns were added: username and password. This was a necessity, since this application, unlike the former, is not tied to a certain computer and a specific user, and thus we have to tell who the user is. The password is encrypted in md5, and there will be a need for an encryption function on the caregiver's side. A simple example of how this can be done is shown in encryption.php, see Appendix A for more information.

Right now, the passwords are the same as the usernames, but stored encrypted.

3.3.2 dbo.LindaMeasurements

In the existing database, there are two tables to describe the measurements: dbo.ValueTypes and dbo.ValueTableNames. Instead, a new table called dbo.LindaMeasurements were created. This table contains id numbers to identify the measurements, and all information necessary to draw the web pages. Also, questions are thought of as measurements, and the old table dbo.Questions is no longer needed.

Brief column description:

- measurementId – connects this table with the others, and identifies a certain measurement.
- measurementName – tells what measurement it is.
- textString – the text that will be presented to the patient during a measurement session.
- inputType – contains a id number connected to the table dbo.LindaInput. Tells how the answers are to be given.
- nrAlternatives – tells how many text fields or radio buttons that are to be drawn.
- unit – is written out on the measurement pages.
- helpText – the text that is shown if the patient chooses to push the Help-button.
- image – a text string with the address to the image connected with the measurement.

3.3.3 dbo.LindaInput

dbo.LindaInput is the table where information about the different input types can be found.

Brief column description:

- inputType – an id number, specific for each type of input.
- description – the input type in plain language.

3.3.4 dbo.LindaLimits

dbo.LindaLimits collects the limit values for the measurements and is used by processAnswer.php to check if the answers are reasonable. The limits can only be used

together with text fields. Since there are different limits for different measurements, no default values can be set automatically. This table still needs a unique identifier.

Brief column description:

- patientId – to know which patient is performing a measurement.
- measurementId – there are different limits for different measurements.
- lowerLimit – is set by the caregiver.
- upperLimit – is set by the caregiver.

3.3.5 dbo.LindaQuestionsInForm

This table has the same structure as the old dbo.QuestionsInForm, but the column questionId is replaced with measurementId, since the proposed database structure does not make any difference between measurements and questions. New id numbers are filled in to match the new measurements.

3.3.6 dbo.LindaAnswer

In the old database, the answers to the measurements and questions were collected in several different tables. To simplify, all answers are now gathered in only one table.

In this table, it is possible to see what kind of measurement that has been performed. An _e at the end of the session id indicates an extra measurement, and if the measurement id is 0, it shows that a measurement has been missed or that the patient has chosen to end the measurement on GUIToDoPage.

This might not be the optimal solution. When more than one answer is given, for example when performing a blood pressure measurement, there is still only one column available for data storing. While waiting for a complete rebuilding of the database, the problem is solved by underscore separation of the values when inserting them. When selecting answers from the database, one must be aware of this, as well as the fact that the data is stored as text and not as integers or any other kind of numeric values.

Brief column description:

- ID – a column which contains unique identifiers to the table.
- sessionId – the same sessionId as before. Is used to identify user and a certain measurement session.
- measurementId – the id of the performed measurements.
- answerText – the result of the measurement.

4. Known Problems

During the project, larger or minor problems arose. Having trouble solving specific programming tasks is an inevitable part of programming, and those problems are not presented here.

4.1 Database Structure

The database used was, as mentioned before, already constructed, and the instructions were not to alter it, but a partly reconstruction of the database turned out to be necessary. There was a need for new information storage, such as the patient's username and password, and some structural changes. The author took part in preparing the reconstruction, but will not perform the actual work.

4.2 JavaScript

JavaScript was used to manage events, such as pressed buttons or keys. It is a usable way of increasing the interactivity of a webpage, but there are also risks. JavaScript is not browser independent, and the more JavaScript functions included, the more vulnerable the system. Most of all, the popup windows used to present information to the user turned out to be a problem. Adding new JavaScript functions became tricky, since every new function could affect older ones.

Because of this, as many functions as possible were made using Ajax and Prototype Framework, see section 2.2.1, instead. This is a way of making sure that the JavaScripts used are browser independent, and by using Ajax, it is also possible to make the same functions in PHP, which normally is a much safer and easier way. One remaining problem is, however, that Prototype Framework needs Internet Explorer 6.0 or newer.

4.3 Drawing Pies and Time Aspects

In GUIClock, the time slots for future measurements are shown as colored pies on the clock face. These are drawn in drawClock.php, a file that is calculating angles according to the time, and then using them as arguments for drawing pies and hands. A rounding problem sometimes arises when drawing time slots shorter than two minutes. The angle values for the start and end times turn out to be the same, and the pie covers all of the clock face. Since no time slots this short are likely to be used, the problem is considered less important, and is left without action taken.

Another thing that can lead to problems is if two measurement sessions are scheduled with an overlap, i.e. if the end time of the first measurement is later than the start time of the next. The result is that GUIClockPage will redirect to GUIStartPage all the time.

4.4 Radio Buttons

GUIFormPage is constructed so that it is free to choose the number of radio buttons or text fields to show by storing it in the database. The number is however limited by the size of the page. It is most obvious when drawing radio buttons. The HTML table has a fixed width, well adjusted to a number of five or six radio buttons. If the number of alternatives is increased, there may be a risk of bad displaying of the page. This can be solved by manually altering the sizes of tables and radio buttons.

4.5 Extra Measurements

The redirect function on GUIStartPage and GUIGoodbye is not implemented when performing an extra measurement. Firstly, one can expect the user to finish the

measurement if an extra measurement is deliberately chosen, and secondly, the extra measurement does not supply the application with any time limits. If a time check still is wanted, this can be arranged with some kind of checking the start time for the upcoming measurement and a reloading of GUIStartPage without the specific extra measurement features.

If an extra measurement is started before a scheduled measurement but ended after the start time, the user will be immediately redirected to an ordinary measurement. When getting the scheduled measurements, an extra measurement can never replace an ordinary.

4.6 Multiple Logins

It is possible for the user to be logged in more than once. This can lead to more than one result of the same measurement. For example, if the user is logged in on two browsers at the same time but only performing the measurement using one of them, the database would be updated once with the results from the measurement, and once with a zero indicating a missed measurement (from GUIStartPage).

4.7 Opera

When using an Opera browser, the CSS file is not able to control the size of checkboxes and radio buttons. They each have an own class in the CSS file ("kryssruta" and "radio"), but they do not work out in Opera. Still, the radio buttons and the only checkbox used (on the login page) are usable, but small.

4.8 Cookies

To be able to use the function to store username and password on the computer, the browser has to be set to accept cookies. Otherwise this information cannot be used.

5. Future Possibilities

The program is built to suit patients suffering from CHF, but the questions asked and measurements made can easily be changed to suit other patient groups as well. Any changes are to be made in the database only, to make the program easy to customize. One possible disease to develop this project towards is diabetes, a common chronic disease striking patients all ages. A combination of CHF and Type 2 diabetes is not unusual, which too makes this area interesting.

Another way of developing is to add more technical features, both to the measurement instruments and to the field of usage. More measurements can be included, which demands more and perhaps more advanced and well-adjusted instruments using new measurement methods, e.g. bio impedance. A future implementation of Bluetooth communication between the instruments and the computer is a possible way to go, making the measurement results show up on the computer screen without the patient having to type anything themselves. A cell phone version of this application would provide the patients with the opportunity of living a mobile life.

As mentioned before, the code is designed to be easily extended to fit other usages than the original.

6. Conclusions

It is clear that eHealth is an important way of making the health care more efficient. The development of new tools and new fields of application must be supported.

Building this web application was a new experience to the author, and an excellent opportunity to explore a new area of knowledge. One of the main things learnt is the importance of being aware of the fact that different browsers behave very differently, and also that JavaScript gives the possibility to make web pages very dynamic, but also, in some cases, unreliable. Ajax was a completely new acquaintance, and a very useful one. The technology is believed to be widely spread over the Internet, the new way of making web pages.

Getting to work with databases, even though just a little, was also interesting and a good experience, creating understanding for how it works.

7. References

[1] Gund, A. ; Sjöqvist, B. A. ; Lindecrantz, K. et al. Övervakning av hjärtsviktspatienter i hemmet, IT för sjukvård i hemmet – Projektkatalog. VINNOVA Information VI 2007:05, 2007.

[2] Schenck-Gustafsson K, Kanter T, Mårtensson J. Hjärtsvikt. 2005/2006 [Downloaded 2007-07-25]. Available: http://www.apoteket.se/content/1/c4/78/27/Hjarta_2.pdf

[3] Pfizer. Hjärtsvikt är vanligt och ökar med stigande ålder. 2006 [Downloaded 2007-07-25]. Available: http://www.pfizer.se/default____5302.aspx

[4] Wikipedia. Heart failure – Wikipedia, the free encyclopedia. 2007 [Downloaded 2007-07-25] Available: http://en.wikipedia.org/wiki/Heart_failure

[5] Carlsson R, Rudolfsson A. Care@Home. Göteborg, Chalmers University of Technology, 2004.

[6] W3Schools. HTML Tutorial. 1999-2007 [Downloaded 2007-07-16]. Available: http://w3schools.com/html/html_intro.asp

[7] W3Schools. CSS Introduction. 1999-2007 [Downloaded 2007-07-16]. Available: http://w3schools.com/css/css_intro.asp

[8] Wikipedia. JavaScript – Wikipedia, the free encyclopedia. 2007 [Downloaded 2007-07-16]. Available: http://en.wikipedia.org/wiki/JavaScript

[9] Denis Sureau. Ajax and XMLHttpRequest Tutorial. 2006-2007 [Downloaded 2007-10-02]. Available: http://www.xul.fr/en-xml-ajax.html

[10] Prototype Core Team. Prototype JavaScript framework: Easy Ajax and DOM manipulation for dynamic web applications. 2006-2007 [Downloaded 2007-10-02]. Available: http://www.prototypejs.org/

[11] Microsoft Corporation. .NET Framework Developer Center. 2007 [Downloaded 2007-07-16]. Available: http://msdn2.microsoft.com/en-us/netframework/default.aspx

[12] Sun Microsystems. java.com: Hot Games, Cool Apps. [Downloaded 2007-07-16] Available: http://www.java.com/en/

[13] The PHP Group. PHP: Hypertext Preprocessor. 2001-2007 [Downloaded 2007-07-16]. Available: www.php.net

[14] Converse T, Park J. PHP 4 Bible. USA, IDG Books Worldwide, Inc., 2000.

[15] Microsoft Corporation. Microsoft SQL Server 2005 Home. 2007 [Downloaded 2007-07-18]. Available: www.microsoft.com/sql/

[16] Glass M, Le Scouarnec Y, Naramore E, Mailer G, Stolz J, Gerner J. Beginning PHP, Apache, MySQL® Web Development. Indianapolis, Wiley Publishing, Inc., 2004.

[17] Watt A. Microsoft® SQL Server™ 2005 for Dummies®. Indianapolis, Wiley Publishing, Inc., 2006.

[18] Webdesignskolan. Webdesignskolan. 2007 [Downloaded continuously throughout the project]. Available: www.webdesignskolan.se

[19] W3Schools. W3Schools Online Web Tutorials. 1999-2007 [Downloaded continuously throughout the project]. Available: http://w3schools.com/

[20] Beaird J. Color for Coders - Color and Design for the Non-Designer. 2004-12-15 [Downloaded 2007-10-23]. Available: http://www.sitepoint.com/article/color-for-coders

The driver for PHP was downloaded from http://se.php.net/get/php-5.2.4-win32-installer.msi/from/a/mirror.

Appendix A – List of Files

Login and Clock Files

These files take care of the login process and the clock page.

GUILogin.php

This is the page shown when the patient is not already logged in, for example when not displaying the pages on their own computer, or after the browser has been closed down. The user name and password can be stored on the computer as cookies by selecting the checkbox to the right. This is only to be recommended when using the home computer, since otherwise anyone will be able to access the patient's measurement account.

loginHelpWindow.php

This file is the one showing the help text when the "Hjälp"-button is pressed. The file is called in GUILogin.php and is using Ajax technology.

cookieWarningWindow.php

A file used in the same way as loginHelpWindow.php, warning when the patient is checking the box to set cookies to remember username and password.

processGUILogin.php

This is a file used for processing the form data in GUILogin.php. At first, the user name and password is compared to the database, and if the login succeeds, the user is sent on to GUIClock.php. If the user has chosen to be remembered, the cookies are set. If the user name and/or the password are incorrect, the user will be sent back to GUILogin.php, displaying an error message. The file also gets the patient id from the database and stores it in a session variable so that it can be used by all files.

conn.inc.php

Including a file means that the file called is inserted in the calling file, row by row. An include file does often have a certain assignment, as conn.inc.php, which, connects to the server and chooses database. It is included in default.php and all other files communicating with the database.

auth_user.php

An include file as conn.inc.php, making sure that pages cannot be accessed without the user being logged in. If shown, it redirects the user to GUILogin.php. It is included in all pages except for the login page.

GUIClock.php

GUIClock.php is the page shown as soon as the user has been logged in, and is shown between the measurements as long as the browser is open. It is displaying an analogue clock with the two upcoming measurement marked as a light blue or pink pies, light blue if morning and pink if afternoon. The date and time of the next measurement is also written on top of the page. The page is refreshed every minute to simulate moving hands.

When the time is due for a measurement, the page automatically redirects to GUIStartPage.php. It is also possible for the user to make an extra measurement outside scheduled time, by pressing the button saying "Extramätning", also pointing to GUIStartPage.php.

drawClock.php

The clock in GUIClock.php is actually an image drawn by drawClock.php. The file is receiving a time from GUIClock.php, and uses it to calculate the angles used for the pies representing upcoming measurements. An angle calculation is also necessary when drawing the hands, representing the present time. Those angles are obtained from the server's clock. The hands and pies are drawn onto an existing image, *mindreKlocka.gif*.

Measurement Session

These are the pages where the measurements are made.

GUIStartPage.php

This page greets the patient, shows a nice picture (*Solnedgang.jpg* at the moment) and gets session schedule id from the database. By pressing the "Starta"-button, the user is sent to GUIToDoPage. If the user has reached the page by pressing the "Extramätning"-button on the clock page, this is also displayed, and a button saying "Ångra" is visible to make it possible for the patient to regret their choice.

If the start button is not pressed during the measurement time slot, the page will automatically redirect back to the clock page, and the measurement id 0 with the value NULL will be stored in the database, signalling that the measurement was missed. To be able to perform the time check, the page refreshes every minute.

default.php

GUIStartPage redirects to GUIToDoPage.php by using the specific PHP $_GET variable and default.php. default.php is a default file containing information used by many pages. To specify what page to display, a file is included, as described in previous sections, at the end of default.php. To know what file to include, the address field of the browser is used. GUIStartPage.php redirects to default.php?page=GUIToDoPage, meaning that this text will be written in the address field. At first, default.php is opened, and at the beginning of the file, $_GET is used to receive the information sent through the address field. For instance: $_GET['var'] = file if the address says default.php?var=file. In our case, $_GET['page'] = GUIToDoPage, meaning that GUIToDoPage is the file to includ in default.php. The suffix .php is added in the include statement in default.php.

GUIToDoPage.php

This file is included in default.php. In this file, almost all requests to the database regarding the measurements are done. A session id is set and the patient is presented to the scheduled measurements. drawButton.php (see below) is called on both onMouseUp and onMouseDown, but with different pictures, to make it look like the buttons are actually pressed.

By pressing the "Nästa"-button (on this page, the button is actually an image link), the user is sent to the first measurement page by the command default.php?page=GUIFormPage&step=0, unless there are no measurements in schedule. Then the user will be directly sent to GUIConfirmPage, see next section.

exitDialogue.php

This file is using Ajax technology and is called in GUIToDoPage.php when pressing the "Avsluta"-button. The file is asking the user if they really want to quit the session, and displaying two buttons saying "Avsluta" and "Fortsätt". By pressing "Avsluta", the user is sent to exitReadToDatabse.php, see next section.

exitReadToDatabase.php

If the user really wants to exit the measurement session, the database is updated with the same information as if a measurement is missed, see GUIStartPage.php. Then, the user is redirected to GUIClock.php.

drawButton.php

The buttons shown on GUIToDoPage are, as all arrow-shaped buttons in the application, drawn by a separate file, drawButton.php. It also gives the possibility to complete the button with dynamic text strings. The file is called with three parameters, for example: drawButton.php?text=Nästa&image=bilder/klickadKnapphTrans.gif&fontsize=22. The parameters are text (what to write on the image), image (what image to show) and font size (of the text to be written). The parameters are stored by using $_GET. The different image files used when calling drawButton are *knappvMindre.gif*, *knapphMindre.gif, klickadKnappvMindre.gif* and *klickadKnapphMindre.gif*.

GUIFormPage.php

This file is drawing all measurement pages. To know what to display, a request is sent to the database asking for information for that specific measurement (see 3.2.2 dbo.LindaMeasurements), using the measurement id. When the question is answered, the user presses the "Nästa"-button to continue. It is also possible to go back to previous measurements. If they are answered, the old answer is shown when displaying the page.

GUIFormPage.php is displaying error messages set by processAnswer.php if more than one comma is written or, if the blood pressure is measured, if the diastolic pressure is higher than the systolic. Also, the limits specified for each measurement and patient are checked. Having text fields as input type, it is only possible to write numbers, backspace, arrow signs, delete and, if it is a weight measurement, commas. If the user is trying to press a letter key, a message telling the user what signs are allowed is shown (see below). When the input type is radio buttons, it is possible to check buttons using the keyboard. Pressing keys representing numbers that are not alternatives will result in showing an error message. All error messages are displayed together with an OK-button.

An error message, set by processAnswer.php, is shown if no answers are given, still giving the user the choice to ignore the question by pressing a continue button shown together with the text, sending the user on to the next page.

It is also possible to get information about the measurement by pressing the "Hjälp"-button (see helpWindow.php and showImage.php). This will show a short text and an image, both stored in the database.

numbersOnlyWindow.php

This file displays a simple error message telling the user that only numbers or commas is allowed if the input type is a text filed. The file is called by the Javascript function *numeralsOnlyAndNoSubmit* in default.php and it uses Ajax technology.

radioButtonWindow.php

This file tells the user that it is only possible to use number keys representing the radio buttons shown, that is, if there are only three alternatives, it is no possible to press number 6. The file is called by the Javascript function *check* in default.php and uses Ajax technology.

helpWindow.php

This is a file showing a help text when the "Hjälp"-button on GUIFormPage is pressed. The text is obtained from the database, and the file is using Ajax technology.

showImage.php

This is a very small file that simply draws the image, which path is stored in the database.

processAnswer.php

This is an invisible file, only used to process the answers from GUIFormPage.php, check the values, and if okay, saving them in a session array. If no answer is given, if there are more than one comma, if the limit values are exceeded or, in case of a blood pressure measurement, if the diastolic pressure is higher than the systolic, processAnswer.php redirects back to GUIFormPage.php and an error message is displayed. Otherwise, the results are stored in a session array. If the input type is text fields, possible commas are replaces with dots to suit the database and the file redirects to the next measurement or, if there are not any left, to GUIConfirmPage.php.

processAnswer.php is also called when the user is going back between the pages (GUIFormPage.php). This means that answers given also are stored, no matter if the user is going backwards or forwards. If the "Tillbaka"-button is pressed, it is allowed not to answer the questions.

Storing Data and Logout

This part is where the answers are sent to the database and the user is either logged out or just sent back to the clock page.

GUIConfirmPage.php

On this page, the user is presented to the answers given, and has the opportunity to go back and alter them if they are wrong. If the user is satisfied, the "Avsluta"-button is pressed and the file called is readToDatabase.php.

readToDatabase.php

This page is invisible and is storing the measurement results in the database. If more than one answer is connected to a measurement id (for example when measuring blood pressure), the answers are stored separated by an underscore. Finally, the file is redirecting to GUIGoodbye.php.

GUIGoodbye.php

The user is told when the next scheduled measurement is to be performed, and is given the choice between logout and ending ("Logga ut" or "Avsluta"). The difference is that if the button saying "Logga ut" is pressed, the file logout.php is called and the user needs to log in again to be able to reach the pages. This is recommended if another computer than the user's own is used. By pressing "Avsluta", the patient is simply redirected to GUIClock.php.

logout.php

This is a tiny file, simply emptying the session variables storing the user name and password, giving the result that the file auth_user.php will not allow the user to see the pages. logout.php is then redirecting to GUILogin.php.

Other Files

Except for the files really making the application, some other files are used.

care.css

This file is a Cascading Style Sheet file responsible of the graphics. There are separate definitions of headers and normal text, as well as classes that can be used to enlarge text or specify the look of an input.

Solnedgang.jpg

The image shown on GUIStartPage. Can be replaced with any picture wanted.

mindreKlocka.gif

This is the background image on the clock page. If the size of this image is altered, corresponding changes have to be made in drawClock.php to make sure that hands and pies are correctly drawn.

knappvMindre.gif, knapphMindre.gif, klickadKnappvMindre.gif, klickadKnapphMindre.gif

These image files are the ones displaying the "Nästa"- and "Avsluta"-buttons, and the corresponding buttons if pressed.

Help Images

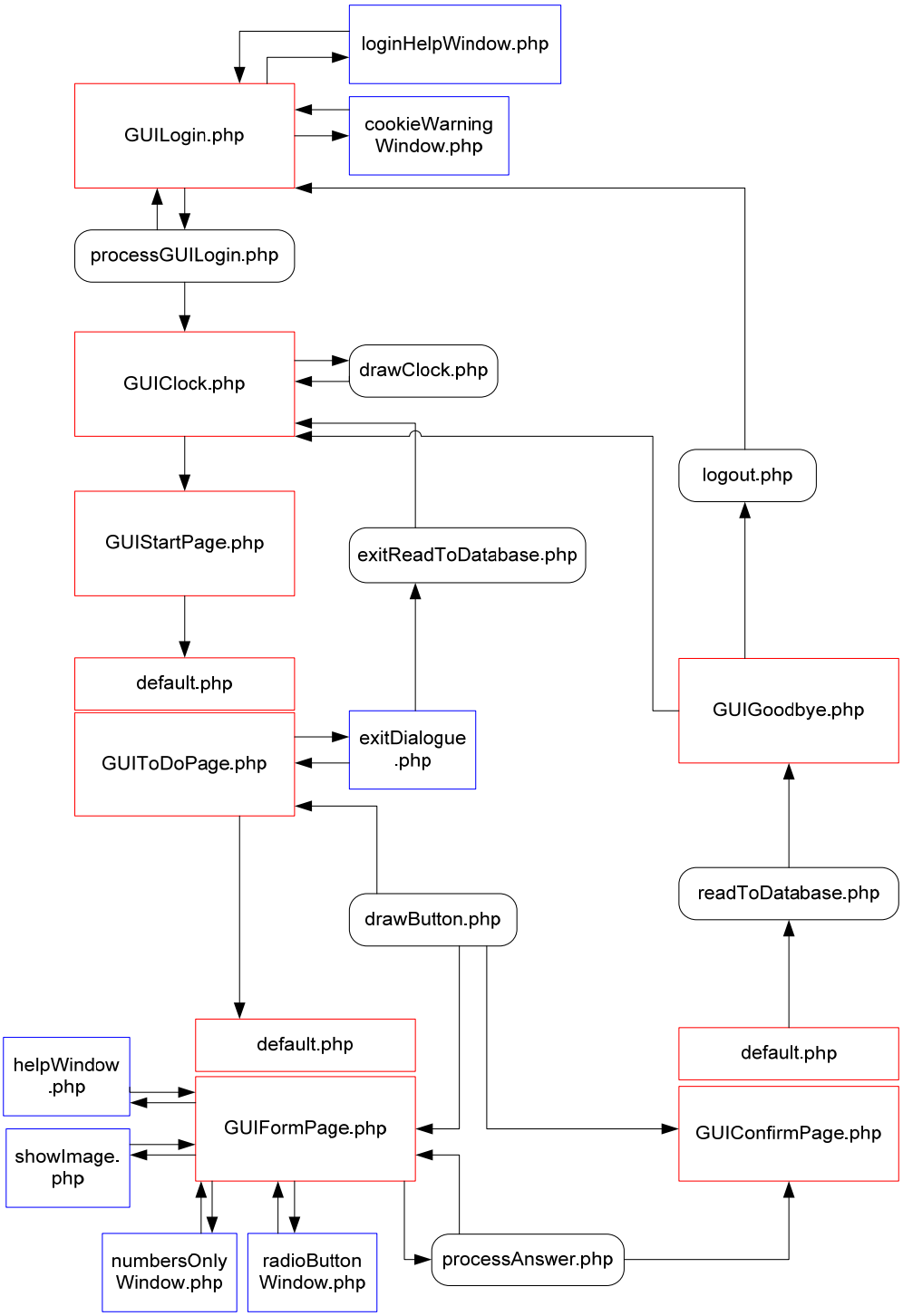
There is a possibility to include images to be displayed when the help button on the measurement pages is pushed. These files have to be stored on the server, and only their paths are stored in the database.

encryption.php

This in an example file to show how the users' passwords can be stored encrypted in the database. The application's login function is assuming encryption.

Appendix B - File Flowchart

Not showing image files, or the files conn.inc.php (included in processGUILogin.php, GUIClock, GUIStartPage, exitReadToDatabase, default.php, processAnswer.php and readToDatabase.php) and auth_user.inc.php (included in GUIClock.php, GUIStartPage.php, default.php and GUIGoodbye.php).



Appendix C - Database Diagram