Självständigt arbete på avancerad nivå
Independent degree project – second cycle

Datavetenskap
Computer Science

IndUnd
The industrial machine maintenance program

Fredrik Nordmark, Sebastian Ryd
Abstract

This project aimed to do a study to see if it was possible to create a standardized system for maintenance of industrial machines. This project thus relied heavily on human computer interaction and security. A number of interviews and user tests were conducted to see if the goal was achievable. A prototype was created with PhoneGap to test on. A back-end with a Database and a REST API was also created for this purpose. In the prototype several security mechanics were also added to be tested, these were openSSL, login and encryption. The study showed that such a standard might not be possible as most companies want to have an internal system more customized for their own systems. The study also showed that if such a standardized solution were created with a lot more features and overall better security and performance than a customized system then several companies might consider using this instead. To achieve this, an extensive future development would be needed. Therefore a large chapter, containing future development, were added to give suggestions on how this system could be improved.
Acknowledgments

This project could not have been finished without the help of our supervisors Mikael Hasselmalm (Adjunct Lecturer at Mid Sweden University) and Mattias Dahlgren (Laboratory Assistant at Mid Sweden University). They always provided valuable feedback and support throughout the project.

We also want to thank Johan Deckmar (Supervisor at Dewire) who provided a lot of guidance and valuable insight in the development phase. Another thank go out to Jonas Bäckström for giving us a chance to do our thesis at Dewire.

A big thanks also go out to Mikael Söderlind at SSAB for giving a lot of valuable feedback during the interviews as well as being very helpful overall. We also want to thank Kent Ferneborg at SSAB for giving additional information.

We also want to thank all the interviewees and testers for their feedback and comments. Without these the project could never have been completed.

A final thank goes out to Mikael Gidlund for giving valuable feedback and challenging us to keep improving our work.
Content

Abstract ........................................................................................................................i
Acknowledgments ........................................................................................................ii
Terminology ..................................................................................................................5

1 Introduction .................................................................................................................7
  1.1 Background and problem motivation .................................................................7
  1.2 Related work ..........................................................................................................8
  1.3 Overall aim ............................................................................................................8
  1.4 Scope ......................................................................................................................8
  1.5 Project goals ..........................................................................................................8
  1.6 Problem definition ...............................................................................................9
  1.7 Planned Structure ...............................................................................................9

2 Theory .........................................................................................................................10
  2.1 Industrial systems ...............................................................................................10
    2.1.1 Industrial automation ................................................................................10
    2.1.2 Industrial control system ........................................................................10
    2.1.3 Automation communication ..................................................................11
    2.1.4 Communication protocols ..................................................................11
  2.2 IT security ...........................................................................................................13
  2.3 Human-Computer interaction .............................................................................13
    2.3.1 MockFlow .................................................................................................14
    2.3.2 Qualitative procedures ............................................................................14
    2.3.3 Prototype .................................................................................................14
  2.4 Application ...........................................................................................................14
    2.4.1 Responsive web design ............................................................................14
    2.4.2 PhoneGap .................................................................................................14
    2.4.3 JQuery .........................................................................................................15
  2.5 Back-end ................................................................................................................15
    2.5.1 REST API ..................................................................................................15
    2.5.2 MySQL Database .....................................................................................16
    2.5.2 Clonezilla ..................................................................................................16
  2.6 Security ................................................................................................................16
    2.6.1 OpenSSL ......................................................................................................16
    2.6.2 Hashing .......................................................................................................16

3 Methodology ............................................................................................................17
  3.1 Project management ..........................................................................................17
Terminology

Ajax (Asynchronous JavaScript and XML)

Android An operating system for smartphones developed by Google.

Apache (Apache HTTP Server) is a web server software.

API (Application programming interface)

Bitbucket A free versioning software.

CSS (Cascading Style Sheets)

HART (Highway Addressable Remote Transducer Protocol)

HTML (HyperText Markup Language)

HTTP (HyperText Transfer Protocol)

ICS (Industrial Control System)

Invisionapp Webpage for creating clickable areas to simulate a working application.

JavaScript Is a dynamic computer programming language.

JSON (JavaScript Object Notation) JavaScript object structure used for sending data.

Kanban Is a scheduling system for lean production.

LAMP (Linux, Apache, MySQL, PHP) A server configuration.

Linux Is an open source computer operating system.

MD5 (Message-Digest algorithm) A hash function.

Mendeley Mendeley is a free reference manager and academic social network.

Mock-up Is a model of a design or a device.

MockFlow A program for creating wire frames.

MySQL ((My) Structured Query Language)

OpenSSL Open source Secure Socket Layer.

OSI model (Open Systems Interconnection model)
PDF (Portable Document Format)
PDO (PHP Data Objects)
PhoneGap A framework for creating applications in HTML, JavaScript and CSS.
PHP (Hypertext PreProcessor)
PhpMyAdmin Is an open source MySQL database administration tool.
PROFIBUS (Process Field Bus)
RDBMS (Relational Database Management System)
RESTful API (Representational State Transfer Application Programming Interface)
RWD (Responsive web design)
Scrum A framework for agile software development
SHA1 (Secure Hash Algorithm)
URL (Uniform Resource Locator)
XML (Extensive Markup Language)
1 Introduction

Industry machine maintenance is today an extensive work task. In this project the aim is to make that work smaller and easier. Solutions in application-form are more and more common today, as well as this one. Applications are taking over because of their convenience and accessibility. This project is also moving a lot of information online to help create the paperless society. This is not only a convenience for workers, it is also has an environmental aspect in helping to reduce the amount of paper used during maintenance.

1.1 Background and problem motivation

Today there are a lot of industries and businesses which wants to move their information online. This is to save time while searching for information and to be able to have all that information directly in their phones. Thus being able to access that information anywhere at any time. There are several industries which develop their own solutions to this. This does, however, mean that there are different solutions in every company. This solution aims to make a standard for all these industries. Thus the primary users of this system will be maintenance technicians while conducting maintenance or inspections. There will also be administrators using the system to get an easier overview of the different machines and locations and also to assist them in understanding why and how different machines work. This will be made by including statistics and other helpful features which are more adapted to non-technicians. Thus all these companies do not need to spend large sums of money to develop individual solutions. It also works to make maintenance and other machine related tasks easier and faster for the technicians. This solution will be made primarily to assist industry maintenance. A similar project has been made earlier and several companies and industries expressed their interest in this idea. With a standardized solution, maintenance technicians moving between different industries and different locations will not need to learn a new system for every new industry or location. This system is not meant to replace any industrial automation systems, but instead enhance the system by making the information gathered available on a wireless device. Since there are a huge number of automation systems and automation communication systems on the market today, the interface in this project will only be integrated into the already existing systems. This project have been focused on process automation for geographical and familiarity reasons. The project was conducted in a town where several process automation industries exist. The project members were also involved in a project in process automation earlier.
1.2 Related work
The closest related work found was a paper concerning a decentralized resource monitoring system. The paper’s main focus was to reduce unnecessary resource consumption. By using a decentralized monitoring system a higher degree of reusability and adaptability could be achieved. [1]

1.3 Overall aim
The overall aim for this project is to conduct a study to see if a standardized solution for a large number of different industries is possible. This standard would consist of an application and a database located on the asked companies’ servers. With this standard system, maintenance technicians moving between industries would not have to learn a new system for every new location. Future development includes making all thinkable information about the machines available online. Also enabling technicians to run tests and searching for errors on machines without needing to be at the same location as the machine.

1.4 Scope
This project will be limited by the following statements;

- This project will not include any real time data from any industrial machines. These values will be simulated to demonstrate how it could be implemented in the future.
- PhoneGap development will be limited to Android phones.
- Information gathering will be limited to literature studies, interviews and a user test.
- The database will not include any real machine information from any industry.

1.5 Project goals
In this thesis a standard will be created with a main focus on the points listed below. Security and Human-Computer interaction are going to be the most important points.

- Creating an industry standard for maintenance of industrial machines.
- High security consisting of encrypted connections and encrypted data.
- The system should be a package that can simply be installed on a company’s server.
- The application should be formed from the results of the interviews and the user tests.
- Application should be able to create graphs from stored data.
- Analyze and evaluate the results with regards to goals and methodology.
1.6 Problem definition

In such a large project there will be several problems which will have to be solved. The most severe problems have been listed below to make it easier to see how to solve as many problems simultaneously as possible.

- Create a standardized model for several different industries.
- To conduct interviews and user tests to gather information.
- Solve the connection between application and server.
- Create an intuitive application.
- Create a database to suit all the needs gathered in the interviews.
- Achieve high security while still being easily integrated.
- Create graphs from stored data.
- Create a package to install on servers to integrate with existing systems.
- Make the system able to detect what kind of security the industries has.
- Enable the system to decrypt those securities.
- Ensure point-to-point security.
- Gather enough reliable information for forming the system.

1.7 Planned Structure

Below is a mind map of the planned structure of the project. In every part the planned attributes are added.

![Figure 1: A mind map representing the planned structure of the project.](image)
2 Theory

This project will include a big variety of theories in a wide array of fields. These will be explained more thoroughly in this chapter.

2.1 Industrial systems

There are a large number of industrial systems used in today’s industries. These systems provide a number of features to an industry by integrating with the machines and the control systems. The theory behind these systems will be presented below.

2.1.1 Industrial automation

In modern industries most tasks are performed by automated machines. These machines keep operating without human control, thus reducing technician’s tasks to primarily installing, maintaining, repairing and dismantling these machines. For the automation to work without problems a system is needed to monitor and alter instructions in these machines. These systems are called industrial control systems (ICS). [2]

There are different types of industrial automation. This project has focused on applying the interface solution in process automation. The interface solution is not limited to only process automation however. The interface solution can also be applied in other types of industrial automation. The interface solution can also be integrated with discrete manufacturing without any problems. [2]

Process automation includes industries where continuous product production exists. For example pulp, paper, mining, steel, oil and gas industries are examples of process automation. All of these products are being produced continuously with as few production stops as possible. [2]

Discrete manufacturing includes industries where the products are produced in discrete steps. By discrete steps it means that the products are assembled in different sub-assemblies or that different components are produced separately. For instance the car, medical and food industry are discrete manufacturing industries. [2]

2.1.2 Industrial control system

Industrial control systems are used in almost all industries to control different machines in different ways. These systems are usually used to read of sensors in machines and send to a control room where a technician monitors it. Any information relevant from the machines is displayed through these systems. [3]
2.1.3 Automation communication

For any automation system to work there is a need for working communication between all the different parts of the automation chain. Communication in automation system can be seen as the following:

- Communication between different automation equipment spread out over a large area.
- Communication between dedicated real-time automation systems with operator work-spaces.
- Closed loop control, ranging from slow processes to fast processes.
- Interlocking and control.
- Monitoring and supervision.

Since there is such a large array of different communications and equipment in automation there is also a large array of different communication protocols. These protocols will be described more in the next chapter. [2]

2.1.4 Communication protocols

For the control systems to work communication protocols are needed between all the different devices and the control panels. In this project the interviews were mostly focused on people working in process automation industries. There is a large number of communication protocols available for industries today. There are both wired and wireless protocols in used which all have different advantages and disadvantages. For additional reading on the topic see Springer Handbook of Automation [4].

Two normal communications protocols used today are Process Field Bus (PROFIBUS) and Highway Addressable Remote Transducer Protocol (HART).

PROFIBUS is a communication protocol developed by Siemens. It is a wired protocol which uses twisted pair or fiber cables. PROFIBUS operates on three different layers in the Open Systems Interconnection (OSI) model. First in the application layer, then in the data-link layer and finally in the physical layer. [5]
Figure 2: Table describing the different layers used by the PROFIBUS protocol. [6]

The different parts of the layers will be described below.

- DPV0: Cyclic exchange of data and diagnosis.
- DPV1: Acyclic data exchange and alarm handling.
- DPV2: Isochronous mode and data exchange broadcast.
- FDL (Field bus Data Link): This is the security layer.
- EIA-485: Electrical transmission using twisted pair cables.
- Optical: Optical transmission via fiber cables.
- MBP (Manchester Bus Powered): MBP transmission technology sending data and field bus power through the same cable.

For more information on the topic see the official PROFIBUS website. [5]

HART is a communication protocol governed by HART communication foundation. HART is applicable on both wired and wireless connections and is used in process automation. HART operates on 5 different layers in the OSI model. It operates on the physical layer, data link layer, network layer, transport layer and the application layer. [7][8]

Figure 3: Table describing the different layers used by the HART protocol. [8][9]

For more information in the topic see the official HART website. [7]
2.2 IT security

IT security consists of eight security objectives which will be explained in this chapter.

Confidentiality: The purpose of the confidentiality objective is to prevent the disclosure of information to unauthorized people or systems. In automation systems this is important to keep the confidential data out of the wrong hands. [10]

Integrity: The integrity objective should take care of detecting changes made by unauthorized people or entities. This includes but not limited to changes in data, notes or reports. [10]

Availability: Ensuring availability means to deny unauthorized entities keeping authorized users from using the service. This applies to all different aspects in the system. Threats such as denial-of-service are included in this scope. [10]

Authentication: Authentication is tasked with determining the real identity of a user and connecting that user with if it is authorized or not. [10]

Authorization: The objective of authorization is to control the permission of entities to access the service, by denying the unauthorized from accessing the service. [10]

Auditability: The auditability objective is tasked with being able to reconstruct the history of the system to be able to recreate malfunctions. This would help with discovering why something went wrong. [10]

Non-reputability: This objective should fulfill a role that can provide indisputable proof to a third-party of who did something in the system or service. This provides accountability and liability to the user that performs actions. This objective does not add any security but it provides the possibility of finding out who did what. [10]

Third party protection: After a successful attack on the system there is a possibility of a third party using the system to perform tasks it is not supposed to do such as being a part of a denial-of-service. This objective is tasked with preventing such a thing to happen. [10]

2.3 Human-Computer interaction

In this chapter the different theories behind human-computer interaction will be explained.
2.3.1 MockFlow
The mock-ups used in this project have been created in a program called MockFlow. “MockFlow Design Cloud provides online services to Plan, Build and Share work for designers.” [11]

2.3.2 Qualitative procedures
Qualitative procedures for information gathering yields more exact information about the specified field of inquiry. Qualitative procedures include methods such as interviews and observed user testing. [12]

2.3.3 Prototype
A prototype is an early version of the solution. It is created to be tested to conclude if the functions and features implemented so far are good or even relevant. After a test on a prototype the noticed flaws or suggested improvements are evaluated.

2.4 Application
An application is a program which is installed on for example a smartphone. Usually these applications are programmed in the native language of the operating system on the smartphone. There are other alternatives however.

2.4.1 Responsive web design
Responsive web design is an approach to web design which aims at providing an optimal viewing experience. This includes minimizing resizing, panning and scrolling over a large number of different devices. This is done by adapting the layout to the viewing environment by using proportion-based grids and flexible images. The responsive web design (RWD) is implemented in cascading style sheets (CSS) code. [13]

2.4.2 PhoneGap
PhoneGap is a method to show websites in the form of an application on a smartphone [14]. PhoneGap uses standards-based-based web technology to bridge the gap between web applications and phone applications. PhoneGap is 100% open source, under the Apache Cordova project. PhoneGap works in the way of that it uses the phones native web browser and maximizes it to have no borders. PhoneGap then renders the hypertext markup language (HTML) and CSS based user interface, this makes it possible to keep it platform independent, see figure 4.
2.4.3 JQuery

“jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use application programming interface (API) that works across a multitude of browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript.”[16]

2.5 Back-end

In this chapter the different theories behind the different back-end parts will be presented.

2.5.1 REST API

A representational state transfer (REST) API is a software architecture style for communication between different machines. It exists to create scalable web services. REST APIs usually communicate over the hypertext transfer protocol (HTTP) with a number of different functions. These functions can include GET, PUT, POST and DELETE. These are used to retrieve, update, add and remove information for, for instance, a database. Using a REST API is a good way to make sure that no user on the front-end can access the database directly. With a REST API as a middle layer with all traffic passing through it, it is easier to control which data can be accessed, altered, added or deleted in the database. [17]
2.5.2 MySQL Database

A database is a method for storing data in a server in an organized way. A MySQL database is constructed in a language called MySQL. It is a very widely used open source relational database management system (RDBMS). MySQL is often used in creations of web applications. It is also a central component of any LAMP server. MySQL also works very well together with phpMyAdmin. [18]

2.5.2 Clonezilla

Clonezilla is a program used for partitioning and cloning disks, it is a useful tool for backing up disks and restoring. Which can be used to restore, for example, a server on another machine. [19]

2.6 Security

In this chapter the theories behind all the different security aspects of the project will be explained.

2.6.1 OpenSSL

OpenSSL is an open source toolkit for implementing SSL and transport layer security protocols. It is managed by a worldwide community. OpenSSL is based on the SSLeay library. The toolkit is also licensed under an Apache-style license, so it is free to use for commercial and non-commercial purposes. [20]

2.6.2 Hashing

Message-digest algorithm version 5 (MD5) is a cryptographic hash function which produces a 128-bit hash value. This hash value is usually expressed in text format as a 32 digit long hexadecimal number. [21]

Secure hash algorithm version 1 (SHA1) is another cryptographic hash function, but this function produces a 160-bit hash value instead. And it is expressed in text format but as a 40 digit long hexadecimal number. [22]
3 Methodology

This project includes a large number of parts and thus there will be a lot of different methods of working and problem solving.

3.1 Project management

The project management method which was used was an agile method called Scrumban. Scrumban is a mix of Scrum and Kanban with sprints made up of one week each [23] [24]. This includes meetings with supervisors every week. A Kanban board from Kanbanize.com was used to keep a visual overview of the project [25]. An agile method was chosen because there is limited time to this project, and the resources are also locked. Thus the result had to be the part which would be modified if necessary.

During the project meetings were held at the end of every sprint to keep track of the progress in the project. Additionally three more meetings were held every week. These additional meetings were also held to keep track of the progress, however they were held with the supervisors. Below in appendix A there is a document containing all the sprint meeting protocols.

3.1.1 Work division

During the project the work was divided between the two project members. There was continuous cooperation throughout the project. However, the responsibility for the different parts of the project was given to one person at a time. The main division was security and design for Sebastian Ryd and the human-computer interaction for Fredrik Nordmark. The other parts of the project was divided into smaller parts and distributed evenly. Below are diagrams to represent the division of the implementation of the system in the project.
Figure 5: A diagram representing the work conducted on the system by Fredrik Nordmark.

Figure 6: A diagram representing the work conducted on the system by Sebastian Ryd.

The human computer interaction parts of the project have also been divided between the two project members. Below is a simple diagram representing the work division in this part of the project.
3.1.2 Versioning

To have easier control over the different versions of the developed software a program called Bitbucket will be used. A private repository will be shared between the project members to assist in the development. This method was used to keep track of the different versions of the application. Bitbucket [26]

3.1.3 References

To get references from portable document format (PDF) files and books a software called Mendeley was used. It exists both as a website and as a desktop application. Mendeley allows users to search for names of different literatures and easily get the reference from that document or literature. It is also possible to drag and drop PDF files to the desktop application and the program will find all available information about that particular file which is needed for a reference. Mendeley was chosen to reduce the time spent on adding references, and focusing the work on the implementation instead. [27]

3.2 Human-Computer interaction

The method used for information gathering about the standardization of the application was mainly done by qualitative procedures such as interviews and mock-up demonstrations. There was also user testing on the prototype at a later stage of the project. The user tests were also used for usability tests to see if the application was intuitive. These tests were very important for the development of the project. Because of that human-computer interaction was given a significant amount of time and was prioritized highly.
3.2.1 Mock-up

For creating the mock-up MockFlow was used. MockFlow allowed the use of tools that simplified the creation of a mock-up with the availability of predefined pictures, icons and frameworks. MockFlow also supplies the user with the ability to upload their own design, which was used to fit the vision of the application better.

For making the mock-up feel more alive Invisionapp was used, this helped greatly when showing how the vision for the application would be. This is achieved by adding clickable areas on the existing mock-up pictures and linking that area to a target picture desired.

3.2.2 Interviews

The interviews in this project were conducted mainly by phone. If, however, any opportunity for a live interview presented itself then it would have been conducted live. This opportunity was never given however. The reason for focusing on phone interviews was because workers in industries do not always have time for extra meetings. The interviews were held with people in different fields on several companies. This was to ensure that no area of the information necessary to make a standard was left out. It was also to get different perspectives from within the same company. These perspectives were considered to evaluate how different users would use the system. The interviews followed a predefined list of questions to get answers to some specific questions which were important for the development of the project. Interviews were chosen because it gave more personalized answers with insight in how the maintenance procedures work in that specific company or industry. The drawback was that there was less answers, although quality was evaluated more than quantity in this case.

3.2.3 User testing

In this project a user test based of Hallway testing was used [28]. A traditional Hallway test includes allowing random people somewhere test a number of prototypes. Besides the testers being chosen randomly the user testing followed a standard user testing procedure. Four main methods which can be used in user tests can be chosen to be used. These are listed below. [29]

- Concurrent Think Aloud: Tester is talking while doing the tasks to explain what the tester is doing.
- Retrospective Think Aloud: Tester retraces their steps after the session to explain what they did and why.
- Concurrent Probing: When a tester does something unexpected the test leader asks questions about it right away.
- Retrospective Probing: Saves the questions about unexpected behaviour until after the session is completed.
In this project retrospective think aloud and retrospective probing was used. These were chosen to not disturb the testers while the tasks were conducted. All the questions were saved until after the tests had been completed. This way more reliable times could be given from the tests.

3.3 Development
This project was developed by cooperative programming. The development was divided into several phases. These phases are explained more thoroughly below.

3.3.1 Planning
The development of the project was first structured in a mind map and in a UML diagram to get an overview of the entire structure. From this the work was easier to divide among the project members. It also simplified the order of which the different parts were implemented.

During the planning and the implementation the Kanban board was used extensively. Every part of the development was added as a task on the board. There every task was taken by one of the project members, this way there were no simultaneous work at any same part and thus decreased confusion and duplicates. This method was chosen from these above mentioned reasons.

3.3.2 Application
PhoneGap was chosen for the application because one of the goals of this project was to make a general and flexible application that would fit all kinds of industries. PhoneGap is platform independent that is why it was chosen for this. That means that there is only need for one set of code and that PhoneGap takes care of building the application for the different platforms.

The PhoneGap version that was used in the application was the command line for Linux. This made it easy to build and installing the application on a phone that is connected to the computer with a USB-cable. To build and run the application only a simple run command is needed to accomplish building and installing the application on the target device, PhoneGap takes care of the steps in between.

3.3.3 Server setup
The server which was used in this project was made in Linux. The server used is called a LAMP server. This server setup was used because it is easy to use, easily implemented and open source. LAMP includes the use of Apache, MySQL and hypertext preprocessor (PHP) on a Linux server. Apache is the web server which housed the REST API. MySQL is the database in where all the information was stored. And the REST API which was used to manage the connections between front-end and the database was created in PHP.
To manage the database on the server phpMyAdmin was used. It is an open source database management tool which is very easy to use because of its graphical interface. Thus there was no need to write the whole database in MySQL code but instead just add it in the interface. This saved a lot of time whilst creating and managing the database. PhpMyAdmin can also give a graph with the overview of the database structure to simplify the understanding of the database structure. So this was chosen because of its simplicity and usability.

The REST API used to connect the application to the database was constructed with PHP data objects (PDO) which simplified the connections to the server. PDO is object based and very easy to use when accessing servers of different kinds as it acts as an abstraction layer. So it was chosen to reduce repetition of code in the case companies had different types of servers. [30]

3.3.4 Application functionality

JQuery is a library that was used to simplify the coding for the functionality behind the design in the application, as well as add useful help functions. This method was chosen because of the simplification of the handling of objects while programming.

JsRender is a library for rendering html code with information taken from a JavaScript object notation (JSON) object. JsRender was chosen because of the simple framework that made it possible to generate objects in a simple and intuitive manner.

For touch navigation and manipulation of the application an API called Hammer was used. “Hammer is an open-source library that can recognize gestures made by touch, mouse and pointerEvents”. [31]

3.3.5 Rendering statistics

In the application there was statistics displayed in the form of graphs. To display graphs in the application an open source API named Google Charts created by Google was used [32]. It is an AJAX API with HTML code and JavaScript. This API was used to save time during implementation, and Google’s APIs usually have high quality.
3.3.6 Data transfer method

All the data sent from the database to the application was sent using JSON. JSON is an open standard format used mostly to transfer data between a server and a web application. JSON is originally derived from JavaScript but is now language independent. JSON is formatted in such a way that it is easy for humans to read and write them, and also so that it is easy for machines to parse and generate them. JSON was chosen in this project because it is very lightweight and it has a very simple syntax. Thus it results in less overhead text in the formatted strings. It is also much easier to use compared to other alternatives when using JavaScript on the front-end. [33]

3.3.7 Responsive web design

The application was made using responsive web design to make the application change the design depending on screen dimensions for example if a phone is in landscape or portrait mode.

3.3.8 Testing

During the implementation of the application and all other programming related parts of the project a continuous testing was used. The testing was conducted according to two testing methods, unit testing and integration testing. These include testing for each function, or module, which have been added to determine if it worked properly. The unit testing includes first a test which is created by defining what the function or module should be able to accomplish. These tests was not automated in this project however, the time needed to create all the automated tests was spent on implementing more of the project instead. After that the function or module was created and tested, when it did what the test demanded it was considered finished. The integration test then later follows the same procedure as unit testing but was conducted every time the function or module was integrated with the other parts of the program. [34][35]

These methods were chosen to reduce time spent on finding bugs in the code. With those methods the bugs was found immediately and could thus be solved before more functions and modules were integrated. This greatly reduced time needed while integrating the different parts of the program since there were no unknown errors in unknown locations. It also made the documentation easier. [36][37]
3.3.9 **Graph and Diagram creation in report**

During the project several graphs and diagrams have had to be created to represent different kinds of data and information in the report. These graphs and diagrams have been created in two different programs. The diagrams to represent the database were created in Dia [38]. All other diagrams and graphs were created in Google Drive [39]. The diagrams were created in Google Drawings and the graphs in Google Sheets, these are two applications in Google Drive [40].

3.4 **Security**

Several kinds of security methods were used in this project. These are presented below.

3.4.1 **SSL certificate**

The security for the connections in the project used an SSL certificate to prevent eavesdropping and other security risks. The SSL certificate which was used in this project is a free OpenSSL downloaded from StartSSL [41]. This included a certificate with RSA encryption to keep information safe. This was chosen because it is free, easy to implement and since security was important. Thus implementing this it was clear that it would be possible to implement a stronger certificate in the future.

3.4.2 **Hashing**

In this project two kinds of hashing were used. These two methods were chosen to demonstrate that it is possible to hash data in the database and still be able to use it in requests. The reason to choose these two methods specifically was because they are very easy to implement in PHP code. These two methods will be described below.

3.4.3 **One time use URL**

For security reasons it was chosen to send the PDF from the database to the application using one time use uniform resource locator (URL). This was no external user could access the PDFs and no extra authentication procedures would be needed when sending the files. This was chosen because of its simplicity and because it was not possible to send authentication data with the PDF requests from the application.
4 Implementation

In this chapter all the implementation during the project will be presented.

4.1 Technical specification

Here a technical specification with regards to the project goals and problem definition will be presented.

These are the goals for this project which needs to be completed before the prototype is considered finished.

- Conduct a study to see whether a standardized solution is possible for the proposed industries.
- Conduct interviews with relative companies to gather information about the proposed system. Both design and to gather information for the standardization study.
- Conduct user tests on a prototype at a later stage of the development to gather additional information needed for forming the system.
- A functioning server with a specialized database and REST API.
- An application formed from the results of the interviews and user tests.
- Reliable and secure communications between the server and the application.
- Encrypted data in the database to increase security.
- Display relative statistics in form of graphs in the application.
- Create a package with all the back-end parts included to simplify installation on other servers.

4.2 Back-end

In this chapter the result of the different parts of the back-end will be presented.

4.2.1 Database

The database will be an important part of the project so there was extensive planning on the structure of the database. The first version of the structure was a very simplified structure. It is presented below.
After several reworks and extensions of the database because of some discussions and additions to include statistics easier, a new structure was constructed. This new structure is presented below.

After the planning and structuring a first version of the database was implemented on the server using phpMyAdmin. This version was not the final version however. During the implementation of the database several additions were added to suit the needs and to make statistics even easier to retrieve.
The final version of the database used for this project became as follows.

All information adding to and retrieving from the database is handled through the REST API. This is very limited in this prototype however, so if for instance a new machine is to be added in the database then this will have to be done manually through phpMyAdmin.

4.2.2 REST API

The REST API for the connections between the database and the application were created in PHP and implemented PDO to make the connection more secure and to simplify the connections. The API was structured into several cases for different requests. Also an authentication function was created to restrict the access to data if the user is not authorized. The code for the authentication function can be seen in figure 9. The code in the API was also made in such a way that SQL injections are not possible. All the data retrieved from the database were sent as JSON formatted text strings to the application. All the data sent from the application to be added in the database is also sent as JSON strings. Thus all data sent between the application and the REST API is sent as JSON strings.
The structure of the REST API is as follows. Depending on the address in the http requests sent from the application there are several cases in the API which have different functions. For instance if the request sent wants a list of all machines available for a specified user then the request needs to contain the username and password for that user and also it needs to access the case for machines. The REST API then makes a request to the database with this information and returns all the machines which are located in locations that user has access to. Thus it prevents users from viewing any information they are not authorized to view. The code for this case is presented below.

```php
function authenticate($conn, $un, $pw) { // Check so username and password are correct
    if ($un == NULL) {
        echo "Please enter a username!
    } else {
        $query = $conn->prepare("SELECT * FROM User WHERE username = "'".$un.'";"");
        $query->execute();
        $value = array();
        $arr = array();
        while($row = $query->fetch(PDO::FETCH_ASSOC)) { // Read the records one row at a time
            $value[] = $row;
        }
        if ($value[0]["password"] == $pw) {
            $arr = $value[0]; // return all user information
            return $arr;
        } else {
            $arr = "You are not authorized for this content";
            return $arr;
        }
    }
}
```

Figure 11: The authentication code from the REST API where ‘Saun’ is the username put in and ‘Sapw’ is the password put in.

```sql
use 'machines'; // list all machines
check = authenticate($conn, $un, $pw);
if ($check("total") != 0) {
    $query = $conn->prepare("SELECT * FROM Machine, location, UserLocation WHERE UserLocation.username = "'".$un.'";
        AND UserLocation.locationID = Location.locationID AND Location.locationID = Machine.locationID ");
    $query->execute();
    $arr = array();
    while($row = $query->fetch(PDO::FETCH_ASSOC)) { // Read the records one row at a time
        $arr[] = $row;
    }
    echo "Not authorized to view these machines!";
    echo;
}
```

Figure 12: An example of a case structure in the REST API.
In the prototype for this system there are a number of cases implemented in the REST API. These cases are as follows. Returning lists of all machines, parts, spare parts, reports, manuals, notes and answers to the notes which are available to the user logged in. The REST API also has cases to return information about a specific machine, part, spare part or note, and also returning specific reports or manuals. It can also return some limited statistics. The REST API also has cases to add new notes and the answers to these notes. It is also possible to update a note from unsolved to solved via the REST API. Every case uses the authentication function to make sure the user is allowed to view the information. Every case also sends the username in the requests to the database so it will only return the data connected to that user.

![Diagram representing the different states of the REST API.](image)

In figure 13 all the different stages of the REST API is shown. GET retrieves data from the database, POST adds new data to the database and PUT updates an existing row in the database. All the different requests will be explained more below.
GET:

- User; returns a list of all users in the database.
- Login; only validates the current user.
- Machines; returns a list of all machines available for the logged in user.
- Notes; returns a list of notes on a specific machine.
- Note; returns all information and answers to a specific note.
- Manuals; returns a list of all manuals on a specific machine.
- Manual; returns one specific manual for reading.
- Reports; returns a list of all reports on a specific machine.
- Report; returns one specific report for reading.
- Parts; returns a list of all parts on a specific machine.
- Sparepart; returns a list of possible spare parts on a specific part.

POST:

- Note; posting a new note to a specific part.
- Answer; posting a new answer to a specific note.

PUT:

- Note; update a note to solved state.

All the above functions are available to any logged in user in the application. Any other data will have to be added directly into the database in the current prototype.

4.2.3 PDF

Methods for sending PDF files from the server to the application were implemented to simplify the viewing of reports and manuals in the application. It was also added to simplify adding new reports and manuals in the database.

4.2.4 Server package

The back-end parts of the project were planned to be put into one single file for installing on other servers. The package would be made with the program Clonezilla.

4.3 Front-end

In this chapter the implementation from the front-end will be covered.

4.3.1 Structure

The structure of the project is the PhoneGap application running and sending requests over an SSL encrypted connection to the database to retrieve and render the information for the user of the PhoneGap application.
In figure 14 the simplified structure of the working project is shown. With a focus on how the PhoneGap application structure works and communicates with itself as well as the server.

The application flow diagram is shown in figure 15, as it is shown you can see that there is multiple ways to add a note to a part. This is made so that it would be more intuitive to the user operating the application.
4.3.2 User interface and functionality

The user interface layout is made using HTML5, CSS3 and JavaScript. The html code is divided into two separate files, one file dedicated to the authentication of the user and the second file for all the functionalities. The CSS code is in a separate file for easier management, JavaScript is also kept separate where possible for a better overview of the functions.

The communication is handled with Ajax requests sent over a secure communications with an SSL certificate.

4.3.4 Login

When logging into the application, a login check is made where the username and password is stored in a session-storage. After that the username and password is checked against the database if the user exists and the password matches, then the user level is set to “1” which signifies that user is authorized to use the application, see figure 16.

```javascript
function logincheck()
{
    sessionStorage.setItem("username", $('#username').val());
    sessionStorage.setItem("password", $('#password').val());
    url = "https://www.projectmimer.eu/api/login"
    backend.get();
    sessionStorage.setItem("UserData", JSON.stringify(backend.container));
    backend.UserData = JSON.parse(sessionStorage.getItem("UserData"));
    if(backend.UserData.level == "1") {
        url = "https://www.projectmimer.eu/api/machines"
        backend.get();
        //Toggles machine choice overlay
        $('#LoginMachineOverlay').toggle();
        //Render the machine choice object
        var template = $('#LoginMachineTemplate');
        var htmlOutput = template.render(backend.container);
        $('#MachineTableDiv').html(htmlOutput);
    } else {
        alert("Wrong credentials");
    }
}
```

Figure 16: The authentication function in the application side.

If the user is authorized then an overlay is toggled to visible and a list of machine objects is generated by using JavaScript templates with JsRender.
4.3.5 Swipe

By using the Hammer.js library to detect a swiping event in the body, one for left and one for right. The applications different tabs are index in an array to keep track of which tab to change to when a swipe is detected. When a swipe is detected the corresponding function is called for changing the visible tab in the application. See figure 17.

```javascript
var myElement = document.getElementById('body');

var hamertime = new Hammer(myElement);
hamertime.on('swipeleft', function()
    navigate_right();
});

hamertime.on('swiperight', function()
    navigate_left();
});

function navigate_right()
    var current = current_tab();
    var indexNumber = pagelIndex.indexOf(current);
    if (indexNumber > 0)
        indexNumber = 0;
    change_tab[pagelIndex[indexNumber]];

function navigate_left()
    var current = current_tab();
    var indexNumber = pagelIndex.indexOf(current);
    if (indexNumber < 4)
        indexNumber = 4;
    change_tab[pagelIndex[indexNumber]];

Figure 17: The swipe function in the application.

4.3.6 Ajax request

For the Ajax requests to fetch the data from the server to display in the application, a get function was implemented with Ajax. The request was synchronous which means that the application would wait for the server to respond. This was chosen to keep the security high. Since every request was accompanied with the username and password there would be no chance to use the application without the right authentication, see figure 18.

```javascript
get function(){
    var xhr = new XMLHttpRequest();
    xhr.open('GET', 'http://example.com/data', true);
    xhr.setRequestHeader('Authorization', 'Basic ' + btoa(unescape(encodeURIComponent(username) + ':' + unescape(encodeURIComponent(password)))));
    xhr.onload = function(){
        if (xhr.status == 200)
            console.log(xhr.responseText);
        else
            console.log('error: ' + xhr.status);
    }
    xhr.send();
    success: function(data)
        console.log(data);
}

Figure 18: An example of an Ajax request in the application.
4.3.7 Notes

To add a new note to a machine first of all a part must be chosen so that it is clear what kind of part it concerns for the user reading the note later on. There is a form at the bottom of the note overlay to add a comment and marking the note as solved, which would indicate that no further work is needed to fix the issue.

4.4 Security

The security in this project included several parts. Security have been a very important part of the project since the companies which would potentially use the system do not want any outside sources or users be able to access their information. The results of these security methods will be presented below.

4.4.1 Connection

In this project openSSL was used to increase the security of the system. To set up the SSL certificate a web domain was needed. So a domain was purchased and linked to the database in order to include the SSL certificate. Security have been an important part of the project so this was a must have in this project.

4.4.2 Login

The login function in this system was implemented by adding a table in the database where user information was added. This information included usernames and passwords which were used to authenticate users. When accessing the database through the API from the application the user cannot access any information unless the username and password were correct, and the specified user had access to that particular machine or location.

4.4.3 Password Hashing

Since every user of the system will need a password and username to access any information, these usernames and passwords are rather important for the security aspects of the project. Therefore extra security was added in the form of hashing of the passwords. Thus if somehow any external user gets access to the database there still will not be any password information available to take.

All passwords have been hashed using SHA1 and MD5 in the following way. First the password is hashed with SHA1 to create a salt. This salt is then added together with the original password and then hashed with MD5. This way it should be close to impossible to know what the original password was if the only information about the password is the hash value.
5 Result

In this chapter the results for the different parts of the project will be presented. This chapter will focus on the Human-computer interaction parts of the project. Other parts will also be presented however.

5.1 Interviews

The interviews conducted in this project focused mainly on large industries in Sweden. Industries in different fields were chosen to give better views on how to create a standard. During the interviews 16 different companies were contacted. The interviews were conducted early in the project to get information for the forming of the system as early as possible. All interviews were held over phone. For results of the interviews see the result chapter.

Out of the 16 companies interviewed eight initially expressed interests in the project. Three companies were not interested and the remaining five never gave any definitive answer. Since the time spent on the interviews had to be limited the interviews focused on the eight companies who expressed interest.

The eight companies which initially expressed interest received a copy of the mock-up as well as some more information in an email. This information and mock-up were then sent to other workers in the same company by the recipient. Additional phone interviews were conducted at a later stage to give the workers, at the companies, time to look over the material.

These companies were later contacted again to get their feedback. A number of these companies later declared, after looking over the material more, that it would not be optimal or suitable for their working environments, alternatively that they already had a similar project ongoing. Ultimately one company continued to express interest and provide information necessary for the project. Below is a pie graph showing the percentage of companies interested in the project.
There have been several companies which have had something similar or ongoing project with the same idea as the one being developed in this project. And for obvious reasons companies with their own version of the idea which is customized for their own use will rather use that instead of changing to a standard program developed to several different industries.

Some companies which said no but still expressed some interest in the idea still provided some feedback on the idea. This feedback will also be taken into consideration when the program is developed. During the interviews only one company provided enough answers to get any information about the different systems used in the industries. The only communication protocol which that particular company used was PROFIBUS. For more information on communication protocols and PROFIBUS see the theory chapter. All the feedback will be presented below and in the index.

During the last week of the interviews a large amount of time was put to getting answers from the three companies that expressed interest. It was, however, not easy to reach all the workers in the companies.

The results of the interviews were put together into a single document. This document will be added in appendix B. These results were taken into consideration when developing the application and the database.
5.2 Design

In this chapter the structure and design of the application will be presented and explained. It also includes a mock-up and the architecture of the application.

5.2.1 Mock-up

The mock-up was created using the Mockflow webpage, described in methodology. The Mockflow webpage provided many tools for creating pictures for the mock-up. The mock-ups purpose was to peak interest from companies, which would provide more data to customize the application.

When all the different pages were created they could simply be downloaded and used for further development of the mock-up to make it a more finished experience.

Here in figure 20 it is show how an android layout of the main page of the application was envisioned to look like. This was done by using a grid tool and adding images to represent buttons for the different categories.

At the top of the mock-up picture there is shown tabs of the envisioned pages, this was achieved by using a tab tool with this tool it was also possible to change how curved the tabs would be in the corners and also how closely they would be placed with respect to the neighboring tabs.

5.2.2 Mock-up presentation

For presenting the mock-up to potential companies for information gathering, first was a sitemap created to show how the flow of the application was envisioned to be like, see figure 21.

![Figure 20: The starting page of the mock-up created for the interviews.](image)
After the sitemap of the application was created, a representation of the application was put together in Google Slides [29]. This resulted in a presentation of the application with clickable areas linked to the corresponding picture, to allow the user to navigate the envisioned application.

To make this experience feel more real and skip the usage of a computer or take extra steps to make run the presentation. The Invisionapp webpage was used to together with its tools to essentially do the same things as in Google Slides but with one big difference. The Invisionapp webpage provides a service that produces a URL to download a shortcut for a smartphone. This shortcut is a web view which shows the mock-up you have made and with the function of just clicking through. This URL was sent with all the mails to potential companies for increase their interest to cooperate with us and provide information.

5.3 Back-end

In this chapter the result of the different parts of the back-end will be presented.

5.3.1 Database

The database worked as expected without complications.
5.3.2 REST API
The REST API worked as expected without complications.

5.3.3 PDF
During the implementation of the application some complications with the PDF reading were encountered. When trying to read a PDF from the database through the REST API it worked as intended with no authentication were involved. However when the authentication were included then the PDF were only sent as a byte stream in the application.

This problem was however identified and a solution to this was found. The solution found was to use a one-time use URL for sending the PDF. This way the PDF cannot be accessed externally, and the security have been assured.

5.3.4 Server package
This part of the project was researched but not implemented because of time restrictions. This project was performed with an agile method, this resulted in that it was deemed to be put in the future development category.

5.4 Front-end
In this chapter the results from the front-end will be covered.

5.4.1 User interface and functionality
The user interface and functionality of the application is implemented according to project specifications.

5.4.2 Communication and security
The communication between the application and server is secure and stable.

5.4.3 Login
Login function works as intended.

5.4.4 Swipe
Swipe function works when swiping over the body of the application.

5.4.5 Ajax request
The Ajax request returns the information that is expected without issues.

5.4.6 Outlay
The prototype for the project is following the framework of the mock-up with a few alterations that was decided to be more suitable when implementing. As seen in figure 22 the layout of the main page follows the broad strokes that the mock-up set up to inspire in the beginning of the project.
The color blue was chosen because it is a pleasant color for the eye, the color choice is beneficial to avoid unnecessary strain on the eyes. The reason for the buttons being as large as they are is because the results of the interviews implied that usability should be the focus.

5.5 Security
In this chapter the results of the implemented security will be presented.

5.5.1 Connection
The openSSL certificate was implemented on a purchased domain as it only accepted top domains. Then the certificate had been implemented it worked as intended without any complications.

5.5.2 Login
The login function in the REST API worked as intended and was implemented without problems.

5.5.3 Password Encryption
The encryption worked smoothly and no unexpected complications were encountered. Both SHA1 and MD5 were implemented easily directly in the REST API.

5.6 User testing
In this chapter all the results from the conducted user tests will be presented.
The user testing was included by allowing a group of people (not involved in the development) to test a prototype of the system. The feedback these tests yielded was taken into account for future development of the system. The user testing was mostly focused on layout and usability, however, some functionality was also included in the tests. Hallway testing was chosen since it would give a valuable insight into how users without prior knowledge of the system would use it. The testers were taken from different backgrounds, different knowledge in computer science and different genders and ages. Statistics were saved for every user test conducted to assist in improving the system for future development.

The tasks requested of the testers were as follows:

1. Log in
2. Enter one machine
3. Enter a specific part
4. Open the list of spare parts
5. Open the list of notes
6. Open a specific note
7. Add a new note
8. Add a new answer
9. Set note as solved
10. Open the list of reports
11. Open a specific report
12. Open the list of manuals
13. Open a specific manual
14. Open the statistics page
The first question was completed instantly so no special data could be collected there. Also the fourth question could not be completed with reliable time data since the function were not completed. However the question was kept so the users could be asked to provide feedback on the topic anyway. During the tests both retrospective think aloud and retrospective probing. The exact results of the user tests can be seen in appendix C.

5.7 Complications
In this chapter all the complications encountered during the project will be presented.

5.7.1 Security system detection
A problem that was expected was that to detect and decrypt different security systems in different industries. During the interviews it was revealed that no such information would be given to utilize in the project from any parties that were contacted. Therefor no such test or researched could be completed in this project.
6 Discussion

In this chapter all the results from the project will be presented and discussed.

6.1 Human computer interaction

In this subchapter all results regarding the human computer interactions will be discussed.

6.1.1 Mock-up

The mock-up was created to show the potential companies how the application would work. The mock-up helped a great deal in assisting the interviewees to understand the scope of the project. The interviewees’ responses indicated that it made the decision of if it was an idea that they were interested in. This can be considered a success for the project which saved the time of both the project and the potential companies’ employees.

6.1.2 Interviews

The interviews were conducted in an attempt to see if it would be possible to create a standardized program for machine maintenance for different companies in different industries. Judging from the results it would be possible to create a standardized program for this, however this potential program needs to be much more advanced and include more functions than the prototype created in this project. Since half the companies interviewed believed the idea was a good one but declined for different reasons it could be possible to create a standard with a larger team and a more ambitious program.

Another conclusion from this would be that since half the companies liked the idea it would mean that the industries interviewed are interested in this kind of programs. But as it would seem they are more interested in a program more specialized for their own company instead of having a standardized one.

The interviews still were of great help when designing and implementing the system. Every feedback which was given to during the interviews was taken into account when developing the system. Even feedback from companies which did not express interest or answered that a similar project was already being conducted internally were taken into account. This way the system could be formed into something as close as possible to a standard.
6.1.3 User testing
From the user testing several conclusions could be made. First to view the list of parts on a machine were implemented to appear when a user taps the image of the machine on the machine view. This was however not intuitive enough and a better solution will be needed in the future. It was also concluded that the function to add a new note on a specific part needs to be clearer as several users had trouble finding it quickly.

The tests also revealed that the use of tabs for the different parts of the application were very intuitive and had a high usability. This concept will thus be used more in future research to make the application as intuitive as possible.

6.2 Back-end
Here the results for the back-end part of the project will be discussed.

6.2.1 Database
The database in this project have mostly been based on what the developers believed would be necessary and on the feedback from the one company that continued to express interest. It is thus difficult to claim that the database is optimal as a standard for several different industries.

6.2.2 REST API
The API created for this project was simple and worked without problems throughout the project. The main reason for simplicity and success in the API was the use of cases as it made it incredibly easy to structure the API. The authentication function could have been more sophisticated however. Though it worked well and made sure no unauthorized users could access the information in the database.

6.2.3 PDF
The reason why the PDF transfer between the database and the application did not work as intended in the beginning was because no authenticating data could be sent from the application to the API. Thus the API could not send the PDF as a file but only as a text string.

The solution to this was to use one time use URLs when sending the PDF. This proved to be a secure enough method for sending the PDF as no external users could get access to it.
6.3 Front-end

In this chapter the front-end of this project will be discussed.

6.3.1 Structure

The structure of the front-end is based on what was believed the end user would need and the feedback from the interviews. The structure is aimed towards being functional as a priority while still keeping it intuitive. The flow of the applications feels intuitive.

6.3.2 User interface

The user interface of the application could have been made more intuitive, the buttons in the application did not look like buttons enough according to the user tests.

6.3.3 Functionality

The functionality works in an acceptable level. There is always room for improvement such as lower load time which could be achieved having a faster and more reliable connection to the server.

6.4 Security

The different security aspects of the project will be presented in this chapter. The security in this project is rather straightforward. It will still be discussed however.

The login function as well as the password encryption was implemented without problem and worked as intended throughout the project. No complications or extra work had to be put into these to be completed.

During the implementation of the SSL certificate a domain had to be purchased in order to set up the certificate properly. A free domain was not enough as the SSL certificate demanded to be set up on a top domain, thus no subdomains were valid.

The eight security objectives three of them were implemented in the project; confidentiality, authentication and authorization. The rest of the eight security objectives were not implemented in this project since the time constriction on this project was not adequate for everything to be implemented.

6.5 Project aims and goals

In this project there were several aims and goals which were hoped to be achieved. These aims and goals will be discussed and evaluated in this chapter.
6.5.1 Overall aim

The overall aim of this project was to conduct a study to see if it was possible to create a standard solution for several different industries. The results from the interviews would suggest that such a standard is not particularly attractive to most companies. The results indicated that most industries are more interested in having their own internal systems which are more specific to that particular industry. The idea behind the system is however very sought after.

6.5.2 Project goals

First goal was to create a standardized solution for several companies. This goal could not be met in this project. As only one of the interviewed companies was interested in this while most other companies already had their own internal systems or were developing it at the time the interviews were conducted. It is however possible to develop in a better standard in the future since several companies would have been interested if the product had more features and where the system had been worked on more.

Second goal was to have high security with encrypted data and encrypted traffic. This goal was reached with an implemented SSL certificate and encrypted data in the database. The encrypted data was only implemented on the password in the database, but using the same method it is possible to encrypt several other parts of the database. This goal is thus considered are finished.

Third goal was to create a package of the server side of the project which could be installed on another server. This way the work would not have to be repeated every time a new company installed the system on their own servers. This goal was partially completed. Because of lack of time this package could not be created and tested, a method for creating a package was found however. This method is explained more in the theory chapter, and the actual way to use it is explained more in the future development chapter.

Fourth goal was to form the application from the results of the interviews and user tests. This goal was completed by considering all the interview answers and the results or the user tests when forming the application. However, since only one company provided enough feedback on the mock-up during the interviews; the application was formed mostly to suit that particular company.

Fifth goal was to create graphs in the application from data stored in the database. This goal was also completed as an API from google was implemented in the application. The API from google can be used to create several different kinds of graphs. To achieve this, the data in the database was stored in such a way that creating valuable statistics could be done easily.
Sixth goal was to evaluate the results with regards to the goals and methodology. This is what has been done above in this chapter. So this goal is also considered complete.

6.6 Ethics
There are several ethical aspects of this project which will have to be discussed. These will be discussed in this chapter.

6.6.1 Possible negative effects
One important aspect would be that if this system becomes very efficient in increasing the productivity of the technicians in a company, then a consequence could be that some of them would lose their jobs at that company.

6.6.2 Possible positive effects
A positive aspect of this system could be to decrease the use of paper reports and manuals. And thus helping to create a better environment with less trees being cut down and decreased paper industries.

Another positive aspect could be to make technicians more efficient and thus companies could expand and have more machines active per technician. Thus increasing productivity and increasing the income.

6.6.3 Human computer interaction aspects
In the human computer interaction parts of the project every tester and company has been kept anonymous so no information about any individual person or company can be found. There is however one exception of a few people and a company being mentioned in the report as they provided very valuable information and helped the project along significantly.
7 Future Development

In this chapter the possible future development will be presented.

7.1 Human computer interaction

In this chapter the possible developments to the human computer interaction aspects will be presented.

7.1.1 User testing

In this project there was a user test on a prototype of the application to examine how well users with no prior knowledge to the system would use it. This was a rather small test with only four testers and only a handful of tasks to complete. So in the future there will be several more and more extensive user tests to get even more reliable data on how to improve the application and underlying functions. In the tests retrospective think aloud and retrospective probing were used to get more feedback on parts which were considered unclear to the testers. These methods worked well in this project as it gave more reliable time data from the tests and also provided feedback on the parts needed without altering the times needed to perform the different tasks.

7.2 Back-end

There are several possible developments to be made on the back-end part of this project. These will be presented below.

7.2.1 Database

In the database the tables which contain the data could be created in such a way that it is more open to more kinds of different types of machines. There could also be more relations between the tables in such a way to make requests from the application easier.

7.2.2 REST API

In the REST API the request from the application are very a few rather open requests to data. This could be further developed into many more requests and to more specific requests to ease the load on the application side with less data that is needed to be processed. Also the data traffic would be reduced with these kinds of requests.

7.2.3 Server package

Using Clonezilla live with a USB drive would make it possible to install a predefined image of the server.
7.3 Security
Since security was a very important part of the project there are several possible developments of the security worth presenting. So in this chapter those developments will be presented.

7.3.1 Location based login
In the future there could be implemented the ability to log in to an industrial area depending on the location of the user. This might prove make it even easier for the user to operate the application but it would result in less security.

7.3.2 Security objectives
In the future the rest of the security objectives could be implemented to increase the security further and have a more complete security plan for the service.

7.4 Features
There are also a number of possible features in this project which will be presented in this chapter.

7.4.1 Statistics
As of now the statistics available in the application is very limited. In the future a lot statistics should be available in the application to make maintenance easier, and also to increase productivity. There should be statistics aimed for both executives and technicians to help in several fields.

7.4.2 Admin controls
In this current prototype all new machines, parts, spare parts and so on are added directly in the database. In the future however an admin tab will be added to the system to allow adding of new table content without having to go into the actual database. The reasons for this is both security and to simplify adding and removing data. This way, users will need no prior knowledge of databases to be able to alter the content in the database.
7.4.3 Real time data

A very popular possible future feature to this system is real time data from machines in the application. This would not be any easy feature to implement however. First a device would be needed which has to be connected to the machines or to the control systems in the location where the machines are located. This device is required to send the real time data to the database continuously. Thus a reliable connection to the database is needed. Then the application will need to get this information from the database continuous as well. Considering this there will need to be a reliable connection to the application as well. Thus this application will always need a working internet connection to function properly. This feature could help to significantly increase the productivity and reduce the time needed for maintenance.
References


https://www.openssl.org/

[21] MD5. "RFC 1321 - The MD5 Message-Digest Algorithm"
http://www.faqs.org/rfcs/rfc1321.html

[22] SHA1. "RFC 3174 - US Secure Hash Algorithm 1 (SHA1)"
http://www.faqs.org/rfcs/rfc3174.html

[23] Scrum. Create software that delivers more value.
https://www.scrum.org/

https://www.atlassian.com/agile/kanban

https://kanbanize.com/

https://bitbucket.org/

[27] Mendeley. Your research, anywhere. Mendeley is a free reference
manager and academic social network.
http://www.mendeley.com/

http://www.digitalgov.gov/2014/02/19/10-tips-for-better-hallway-usability-testing/

[29] Usability. "Running a Usability Test"


[38] Dia Diagram Editor. "Dia is a program to draw structured diagrams", http://dia-installer.de/ (retrieved: April 2015), 2014.


Appendix A: Spring Meeting protocols

Finished week 7 (-February 15)
Mock-up
UML diagram
Project goals
Background
Problem definition

Finished week 8 (February 16 - February 22)
Interview Questions
Possible Companies

Notes
The interview questions and finding possible companies went good. Node.js has created some trouble on the development side however. The work had to be slightly limited this week because of an exam in another subject.

Finished week 9 (February 23 - March 1)
Setup PhoneGap

Notes
Continued Interviewing with several companies. Some answers were received. Android SDK created some problems but were fixed and PhoneGap was set up successfully.

Finished week 10 (March 2 - March 8)

Notes
Continued Interviewing with several companies. More answers were received. Several large companies have expressed interest. Continued programming on the application, tabs and swipe function implemented. Research about responsive web design started.

Finished week 11 (March 9 - March 15)
Set up server
Note
Continued interviewing and information gathering. Started writing on the result part about the interviews and human-computer interaction. Continued coding on the design of the application.

Finished week 12 (March 16 - March 22)
Main page design

Note
Continued interviewing and information gathering. Additional information added in the result chapter about the interviews. Completed the design of the main page of the application.

Finished week 13 (March 23 - March 29)
ER diagram

Note
Continued interviewing and information gathering. Additional information added in the result chapter about the interviews. A first ER diagram for the database was completed.

Finished week 14 (March 30 - April 5)
Set up a basic structure for the database

Note
Continued interviewing and information gathering. Additional information added in the result chapter about the interviews. A first version of the database was created.

Finished week 15 (April 6 - April 12)
Added test data to the database

Note
Additional information added in the result chapter about the database. Further work on the database. Further development of the graphical interface on the application.

Finished week 16 (April 13 - April 19)
Interviews and compiling all interviews into a single document

Note
Additional information added in the result chapter about the database. Further work on the database. Further development of the graphical interface on the application. Overlay created in some parts of the application.

**Finished week 17 (April 20 - April 26)**

- 

Note
Additional information added in the result chapter about the database. The rest API have been created and worked on. Further development of the graphical interface on the application. Overlay created in some parts of the application. Began preparing the connection to the REST API.

**Finished week 18 (April 27 - May 3)**

REST API creation

Note
Additional information added in the result chapter about the database. The rest API have been worked more on. Further development of the graphical interface on the application. Overlay created in some parts of the application. More preparations for the connection to the REST API.

**Finished week 19 (May 4 - May 10)**

- 

Note
Additional information added in the result chapter about the REST API. The REST API has been worked more on. Further development of the graphical interface on the application. Overlay created in some parts of the application. More preparations for the connection to the REST API.

**Finished week 20 (May 11 - May 17)**

Coding shell to application
Connecting app to back-end

Note
Additional information added in the result chapter about the REST API. The REST API has been worked more on. Further development of the graphical interface on the application.

**Finished week 21 (May 18 - May 24)**
Note
This week was completely focused on the completion of the report.
Appendix B: Interview results

Interview questions

1. Are you interested in an application that can keep all relevant information about your machines in your phone?
2. What would be the most important kind of information in an application like this?
3. Would topics like; spare parts, logs, manuals and so on be relevant?
4. How would you want to identify machines with the application? Serial number, wireless or other?
5. Would real-time sensor data be relevant in the application?
6. Would graphs from the stored data be relevant in the application?
7. Would an interactive model of the machine be good to find certain parts during maintenance?
8. Would you prioritize functionality or outlay?
9. Is the outlay in the mock-up a good solution? If not, what could be changed?
10. How would you like documents to be presented? Pdf, web based or other?
11. What kind of server do you have?
12. What kind of security does those servers have?
13. What kind of systems do you have for retrieving sensor data from machines?
14. What kind of communication protocol is used for the sensor data? For future development (not in this scope).
15. Would you want to be able to add reports and notes directly in the application?
16. Would you be interested in a standard web browser page for similar use?
17. Is there any possibility of meeting in person to try a prototype of the application?
18. Is it also possible for us to come and observe the work process?
19. Is it ok to mention you and/or the company in the report?

Interview feedback

Mikael Söderlind (SSAB)

1. Yes
2. Security, spare parts, work orders
3. Yes
4. Scan ID with camera and type in manually (no chip)
5. Definitely, it sounds good.
6. Yes
7. No, list and accessibility to blueprints is more necessary
8. User friendly
9. Yes, it looks good  
10. Don’t know, (pdf)  
15. Yes, that would be good  
16. No, it already exists. (They want the things they already have in their computers to the phone)  
17. Possible  
18. Possible  
19. Yes  

Kent Ferneborg (SSAB)  
11. 2008 R2 64bit windows  
12. Process firewall (virtual)  
13. IBA, Argus (mostly IBA)  
14. Siemens, ABB (about everything) (Siemens profibus DP usually) 

Interviewee 3  
1. Don’t know, but it sounds interesting  

Interviewee 4  
1. It could be interesting  

Interviewee 5  
1. It could be interesting  

Interviewee 6  
1. Yes, it sounds interesting  

Interviewee 7  
1. Yes, it can be interesting  
2. Simpler reporting  
3. Yes  

Interviewee 8  
1. Yes, it sounds interesting  

Interviewee 9  
1. Interesting, but we already have a similar system  

Interviewee 10  
1. No. We already have a similar system but to iPad instead
Appendix C: Usertest results

1. Log in
2. Enter one machine
3. Enter a specific part
4. Open the list of spare parts
5. Open the list of notes
6. Open a specific note
7. Add a new note
8. Add a new answer
9. Set note as solved
10. Open the list of reports
11. Open a specific report
12. Open the list of manuals
13. Open a specific manual
14. Open the statistics page

<table>
<thead>
<tr>
<th></th>
<th>Tester 1</th>
<th>Tester 2</th>
<th>Tester 3</th>
<th>Tester 4</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0:35</td>
<td>0:14</td>
<td>0:14</td>
<td>0:18</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1:30</td>
<td>0:27</td>
<td>0:56</td>
<td>0:58</td>
<td>not intuitive</td>
</tr>
<tr>
<td>4</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0:06</td>
<td>0:14</td>
<td>0:02</td>
<td>0:04</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0:03</td>
<td>0:05</td>
<td>0:28</td>
<td>0:04</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>1:01</td>
<td>0:41</td>
<td>0:12</td>
<td>0:22</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0:25</td>
<td>0:19</td>
<td>0:11</td>
<td>0:12</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>0:24</td>
<td>0:18</td>
<td>0:05</td>
<td>0:16</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0:03</td>
<td>0:04</td>
<td>0:05</td>
<td>0:02</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>0:01</td>
<td>0:01</td>
<td>0:01</td>
<td>0:01</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>0:06</td>
<td>0:02</td>
<td>0:02</td>
<td>0:02</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>0:01</td>
<td>0:01</td>
<td>0:01</td>
<td>0:01</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>0:01</td>
<td>0:02</td>
<td>0:01</td>
<td>0:01</td>
<td></td>
</tr>
</tbody>
</table>