HAAGA-HELIA
University of Applied Sciences

Creating a management system for customer information for a small business CASE: Botisto

Timo Aho

Bachelor's Thesis
Degree Programme in Business Information Technology
2012

HAAGA-HELIA
University of Applied Sciences

Abstract

Business Information Technology

Author Timo Aho	**Year of entry** 2006
Title of thesis Creating a management system for customer information for a small business CASE: Botisto	**Number of pages and appendices** 24 + 4
Supervisor Juhani Välimäki	

The purpose of this thesis project was to find and create a better solution for handling customer information at Botisto.

Botisto is Spanish company selling mass customized men's footwear. Before this thesis project, the customer information was collected manually. This information consisted, for example, of names, preferred shoe sizes and address information.

During this project, it was decided that a customer information system will to be created from scratch. The system was developed with PHP programming language and MySQL database management system using Waterfall development model.

The project was successful and the software has been taken into use at the Botisto shop.

Keywords
Customer, Information, Handling, PHP, MySQL

Table of contents

1 Introduction

1.1 About Botisto

Botisto is a privately owned company which sells mass customized men's shoes. The difference between tailor made and mass customization is that in tailor made shoes the shoe is created individually for each customer. In mass customization there is pre-existing selection of shoe sizes and fits from which the customer selects the closest match to their feet.

At the time of writing there were two Botisto shops worldwide. The first shop was opened to Barcelona, Spain in August 2011 and the second one opened in Seoul, South-Korea in August 2012.

1.2 Development goal

Before this project the shop employees were keeping track of the customers manually with pen and paper. The collected information from customer included for example: email address, phone number and address information. There were several issues with this manual approach. The data collected varied from employee to another. The hand-writing was unclear when it was typed in a hurry. Searching for a certain customer was tedious. It was soon identified that in order to keep the daily work organized a new system was needed for handling Botisto customer information.

The goal of this thesis project was to find and create a better solution for customer information handling for Botisto. The solution will be used by the shop employees and Botisto management. The system should outperform the old one and be easy to use.

1.2.1 Included in scope

This project should deliver a working computerized customer information handling system.

The deliverables should include:

- Working customer information handling system
- Requirement documentation for the system
- Design documentation for the system
- Test plan document for the system

1.2.2 Out of scope

It is not going to be assessed whether the manual pen and paper method could be enhanced. It is going to be assumed that the computerized information handling is more efficient compared to manual bookkeeping. This assumption is not going to be proven or benchmarked. As there is no budget attached to this project, commercial approaches to reach the development goal are not going to be evaluated.

Due to the simplicity of the required software a user guide is not going to be created in this project. Instead a hands on training session will be arranged when the software is handed over to Botisto. In order to limit the scope of the project the software is not going to have user input validation.

1.3 Customer process in Botisto shop

When a new customer enters Botisto shop the customer is welcomed by a salesperson and the company and its products are introduced. If the customer is interested in buying a pair of Botisto shoes, the customer's feet are scanned with an optical foot scanner. The scanner produces a ticket which has the feet measurements of the customer.

After knowing the feet measurements salesperson can suggests shoe sizes and fits that the customer should try on. Depending on individual preferences, the customer selects the shoe size and fit that suit his taste. Besides the customers contact information, the selected size and fit are going to be saved for the future order by the salesperson.

The next step is that the customer chooses the model and the leather that he likes. After the selection the customer can place an order. The order is then sent to the Botisto factory where the shoes are going to be produced. When the shoes have been made, the shoes are delivered to the customer or alternatively the shoes can be collected from the Botisto shop.

2 Different solutions to the problem

When the project was started different approaches were considered to reach the development goal.

2.1 Requirements for the solution

The new solution should provide a unified way of storing the customer data. In other words the information stored should not fluctuate depending on the employee who stored it. The data should be easier to find than with the manual bookkeeping.

The system should speed up the daily work with the employees working with the customers. The customer information usability should be enhanced for both the employees working with customers and for the management managing the daily work.

2.2 Proprietary software

One solution would be to use commercial software for creating customer information handling system. It was decided that commercial software will not be used. This was mainly due to the licensing costs. Any additional costs wanted to be avoided and the cost of labour was not tracked within this project.

The commercial software's are typically big and complex. They have a lot of functionalities that can be used in variety of different businesses (TopTen Reviews 2012). With Botisto what was actually needed was a simple bookkeeping of the customer contact information.

2.3 Developing new software from scratch

As the software requirements for Botisto were quite limited, it was decided that the software would be created from scratch using open source tools. This way the software would have just the features and attributes Botisto needs. This approach also enables

customization in future releases for functions which may not exist in the commercial off the shelf software.

3 Development process model consideration

There are different models how the software development projects can be executed. These process models visualize the steps that are usually taken in software projects. The models also nominate the order of the different steps. What the models do not tell is the persons and the roles involved in software projects. (Sommerville 2011, 29.)

3.1 Waterfall

The waterfall process model is usually referred as the oldest and most well known process model. In the model there are different phases (Requirements, Design, Implementation, Testing and Maintenance). These phases are using feedback from the previous one and therefore the phases are executed one after another. (Braude & Bernstein 2011, 37-38.)

The most criticised pitfall of the waterfall model is that the testing phase of the software occurs near the end of the process model. If critical defects are found only in very late state of the project, it can cause serious delay to the project. (Braude & Bernstein 2011, 37-38.)

3.2 Agile

In Agile methodology overhead of traditional software development processes and documentation is tried to keep minimalized. The focus is on individuals and their interactions, software which works, customer involvement and reaction to change. (Braude & Bernstein 2011, 49-50.)

There are different methods to run Agile projects. For example extreme programming (XP) and Scrum have gained fame. Regardless what Agile method have been chosen key idea is to run fast incremental and iterative development where fast paced releases are reviewed with customer. (Sommerville 2011, 58-77.)

3.3 Why Waterfall model was chosen?

Waterfall development model was chosen for the thesis project. The approach was chosen as it is simple and well suited for this project. Although there are pitfalls with the traditional Waterfall model, the risk of ending up with software which is not meeting customer requirements was quite limited as the agreed use cases were simple and straightforward. (Braude & Bernstein 2011, 37-38.)

If there software would have been more complex and especially if there would have been more developers working on the project, the selection would have been harder. In that case Agile methods could have been stronger choice. (Braude & Bernstein 2011, 49-50.)

4 Development tools

4.1 Why PHP and MySQL combination was chosen?

There were numerous reasons why PHP and MySQL were chosen for the project. There are no licensing costs for these tools. The tools have wide community support and there are lot of examples and tutorials available even for complex application designs. The existing Botisto hosting service provider supported these tools. When the hosting providers were compared for the Botisto website, it was noticed that PHP and MySQL are widely supported globally with the hosting service providers. It also affected the decision that I had previous development experience with PHP and MySQL.

4.2 What is PHP?

PHP is a scripting language, which can be used to create dynamic websites. The PHP code is processed on a webserver, which is serving a web page which contains PHP. The code is executed on the server before the website is shown to the end user. This is why the end user is unable to see or drill down to the PHP code. (Lengstorf 2009, 4; MacIntryre, Danchilla & Gogala 2011, xvii.)

As the webserver is executing the code, there is no need to compile the code to machine code as with traditional programming languages. This can enable faster development process. On the other hand the PHP code cannot be tested on a computer, which doesn't have a webserver running on it. PHP supports various different operating system and web server combinations. (Lengstorf 2009, 4; Trachtenberg & Sklar 2006, xv.)

4.2.1 Brief history of PHP

PHP started its life in 1995 as a script collection which was authored by Rasmus Lerdorf. The acronym originally was spelled out as Personal Home Page. The first ver-

sions main functionality was the ability to handle HTML forms. Lerdorf made the script collection available so that anyone could download it and make use of it on their websites. (Gilmore 2006, 1-4; Trachtenberg & Sklar 2006, xv.)

Lerdorf continued developing PHP and the users of PHP also contributed in the PHP development. After several years and releases the PHP has matured from a script collection to a programming language. (Gilmore 2006, 1-4; MacIntryre et al. 2011, xvii.)

4.2.2 PHP Frameworks

Different PHP frameworks consist of ready-made software modules. Usually many different applications share the same basic functionalities e.g. authentication. As the frameworks have the building blocks for these basic functionalities, they can be used to speed up the development process. In order to utilize these building blocks the developer needs to follow up the framework development convention. (Bari & Syam 2008, 18; Abeysinghe 2009, 152)

As the frameworks are used widely by software developers, the code they contain is already well tested, compared to code one might develop for one's own use. The frameworks also enable coherent codebase if several developers follow the same framework coding convention when coding the software. (Bari & Syam 2008, 18; Abeysinghe 2009, 152)

Different PHP frameworks were considered in the beginning of the project. A small proof-of-concept was made using CakePHP framework. The framework was easy to set up for a simple test application. During the test it was soon noticed that learning the advanced usage and configuration of the framework would have required more time than what was available during this projects implementation phase.

4.3 What is MySQL?

MySQL is an open source Relational Database Management System (RDBMS). The database management system is the interface between actual database files and the application storing, using and deleting the data. The MySQL database can hold binary files, text and numeric data. As MySQL is a relational database, it means that the stored data may have relations within. (Ullman 2006, x; Tahaghoghi & Williams 2007, 3.)

4.3.1 Brief history of MySQL

Swedish company TcX developed their own database server software as they were disappointed with the speed of commercial products available. The employee who was responsible for creating the database software was Michael "Monty" Widenius. TcX released the database software to the public in 1996 with the name "MySQL".

After the initial launch in 1996, MySQL development continued and the popularity grew. In year 2001 a company MySQL AB was founded to provide services for the MySQL database users. (DuBois 2003, xiv; Gilmore 2006, 573; Vaswani 2004, 6.)

4.4 Other development approaches considered?

Stand alone software could have been utilized to achieve the development goal. It was turned down, as the information needed to be accessible from multiple computers. Utilizing an existing hosting provider also provided automatic backups and solid platform, which eliminates at least some amount of upkeep work compared to in-house hosted service.

Java programming language was considered for this project. Although the same software functionality could have been implemented also with Java, it was not supported by the existing Botisto hosting providers. Exchanging the service provider to other vendor and having that taken into this project would have required significant amount of time from the whole Botisto organization. The exchange could have also had undesired cost impact.

On database side MySQL and PostgreSQL were considered. Both database systems have lots of documentation and wide community support. Many comparisons exist where the differences of these database systems are evaluated. This project could have been executed with either one of them. In the end the previous experience of MySQL was why it was chosen over PostgreSQL. (WikiVs 2012.)

5 Project execution

In Haaga-Helia University of Applied Sciences the thesis work is always conducted as a project. When one begins the thesis work, a project plan is created. The project plan consisted of the following phases which were executed in sequence during the project. These phases were adopted from the Waterfall software development model.

5.1 Requirements phase

The Software Requirements Document (in Appendix A) was the deliverable created during the requirements phase. The document holds the software requirements information which was collected by interviewing the Botisto Barcelona shop manager over telephone.

The starting point for the system requirements was to find out what information should be stored using the system. The basis for the requirements was the information which was collected with pen and paper. On top of that it was agreed what information should be recorded to the system, which was not collected before the project.

After it was clear what information should be stored, it was time to decide the use cases which would be implemented in the first version of the software. As the idea was to create a simple bookkeeping system, it was quite straightforward to come up with create, modify, list and delete customer use cases. It was decided that the users could not delete the customers straight away. Instead the customers can be marked as to-be-deleted in the software. The database administrator will then periodically delete all customers which have been flagged for deletion.

5.2 Design phase

The aim of design phase was to create Software Design Document (in Appendix B). The database design is inherited from the requirements documentation. The software design is formed based on the use cases documented in the software requirements

documentation. Before writing the design document for the software, the basic idea of the software construction was drafted to paper.

The design is mainly visualised with tables describing the software. Besides Microsoft Office tools, Microsoft Visio was used to create the diagram found in the software design document. Database schema drawing was made using MySQL Workbench. Screenshots were added to the software design document after the layout was created in the software implementation phase.

5.3 Implementation phase

In the implementation phase the software was created. Software requirements and design documents were used as the backbone for the development. The documents had the needed information for the development and it was easy to go back to the documents and verify what was specified. Open source development tool NetBeans IDE PHP was used to create the HTML and PHP code. The database was implemented using MySQL Workbench.

In beginning of the implementation phase a development environment was set up. Ubuntu Linux was used as the operating system for the environment. Apache webserver and MySQL database were installed and configured to the environment. After this the software was developed with the tools mentioned above.

The visible parts of the software were coded at first. The next step was to create the modules which were needed for the database access and implement the database integration to the software.

5.4 Verification phase

During the software verification phase the test plan document (in Appendix C) was created. The main focus in designing the test plan was to make sure that the software would be tested for all of its use cases and that the software would be good enough for successful daily usage in the Botisto shop.

After the document was ready, the software was tested as described in the test plan. The software passed the tests and the usability and non-functional requirements specified in the software requirements document (in Appendix A). The software was reviewed and approved by Botisto steering group. After the approval the software was deployed and configured to the production environment.

6 Results of the project

After the verification phase, the software developed in this project was taken into use at the Botisto shop. After the software had been in use an interview (referenced in next chapter) was conducted.

Besides the working software the project delivered:

- Software Requirements Document (in Appendix A)
- Software Design Document (in Appendix B)
- Software Test Plan Document (in Appendix C)
- This Thesis Document

7 Evaluation of the development project

In this chapter the thesis project is evaluated.

7.1 Evaluation of software requirement analysis phase

The software requirements phase was straightforward. The collaboration with Botisto Barcelona shop manager proceeded nicely. As they had previously already collected customer information, they had previous experience what information should be collected from the customers. This helped and speeded the requirements phase a lot, as there was already a solid point where to start from. The software requirements document (in Appendix A) worked out well.

7.2 Evaluation of software design phase

In the design phase the result was the software design document (in Appendix B). In Haaga-Helia courses the software documentation is focused on object oriented programs. As the software developed in this project was not object oriented, the documentation practises were applied. If there would have been more time for the design phase, the document could have been improved on the parts which describe the modules of the software.

7.3 Evaluation of software implementation phase

In the software implementation phase the software was created according to the software requirements and software design documentation. The software is fully functional as specified and it met the usability and non-functional requirements.

The software implementation took less time than originally planned. There is some repetition in the database access code, which in general should be avoided in programming best practises. On the other hand the software is modular. For example the database access routines can be changed by just changing one file. The modularity

makes the software easily maintainable, so the quality of the code can be considered quite high.

7.4 Evaluation of software verification phase

The software verification phase consisted of creating the test plan document (in Appendix C) and verifying the software. After the software was approved, it was taken into use.

Software verification was a success. The work spent on the requirements, design and implementation phases paid off as no defects were found in the verification phase. The basic functionality of the software was already verified in the implementation phase, but the overall functionality was verified in the verification phase.

7.5 Customer feedback on the project

In order to get feedback from the customer, the Botisto Barcelona shop manager was interviewed on 5.11.2012 (in Appendix D).

According to the interview the developed software has helped with the daily work load and it's stressed that the software is very easy to use. The employees prefer using the software over pen and paper, which was the method previously in use. It was said that using the software gives customers more professional image from Botisto. The management can now see customer information from the system without going to the shop.

There were few areas for improvement. The software should be developed so that it would not be dependant over network connection. In case the network connection is lost, they still have the old customer bookkeeping. It was also noted that a change log would be good, were the changes to the information could be tracked. If there's a mistake, this would enable them to go back to an earlier state.

7.6 Evaluation of development methods and tools

The Waterfall development method suited this project well. The possible risks of serious defects been found in the end of the development cycle did not realize, as anticipated. The software was approved by the steering group after it was verified. The well-executed requirements analysis phase definitely had impact on that.

The open source development tools used in this project were easy to use and had similar features seen on commercial development tools. The high quality of the tools was a surprise.

7.7 Evaluation of project management

I tried to keep the project management material as small and lean as possible. If there would have been more people participating into the project, more effort should have been put on to the project management. In this case, the small and lean approach worked out nicely.

7.8 Impediments during the project

The biggest constraint during the project was time. As I was working as a full time employee during the day, it proved to be a struggle to find enough time to complete the thesis project. In retrospect the biggest enabler for completion was when I took one week holiday from my day job to focus on writing the thesis. Also the lack of budget for this project nominated quite a lot which options could be considered in the project execution.

7.9 Evaluation of the learning during the project

The thesis project experience was definitely a learning process for me. So was running a software project end to end all by myself. Previously I had always participated in similar projects as a team member only focusing on my own area. Also the project management was something that I had not done in this extent before.

The most challenging parts for me were time management and the academic writing. My writing style has always been compact and to the point. Writing a whole thesis as a story has been a learning process for me.

8 Conclusions

In Botisto organization there was a need to improve the way they handled their customer information. The existing way limited the usage and accessibility of the information collected.

The improvement was executed during this thesis project by creating software documentation and tailor made customer information handling system to Botisto. The software development was run with Waterfall development model and the software was created using open source tools. In the process the requirements Botisto had and the existing IT environment was taken into consideration.

If there would have been more time, it would have enabled more thorough research for different options to reach the development goal. In order to keep the schedule previous experience in some technologies had weighted impact in the decision making process. Some approaches were eliminated due to lack of budget.

The project was a success and it reached its development goal. The created customer information handling system was taken into use. The customer feedback was positive, although there were two improvement suggestions for future. The software supports the Botisto business as it eases the daily work in the whole organization.

As the labour costs were not tracked during this project, the project was very cost efficient. Running similar project commercially would have been quite expensive for a small business. Most probably the completely tailored software would not have been a realistic option.

Journey from the beginning to the end was a learning experience to me. Running a thesis project besides working as a full time employee was a challenge. Looking back now it's rewarding to see that the project succeeded.

9 Future development of the software

In future the software should include customer order handling. It would be logical next step in order to keep the daily work organized in the shops. Adding the order handling would also enable other enhancements in the future. One enhancement could be integration with factory ERP (Enterprise Resource Planning) system. This way the staff could be updated about the shoe manufacturing process, without the need to consult the factory.

The software could also be expanded so that customers of Botisto could log in online and the system would find their foot scans. The customers would download the feet files and could view the image of their scanned feet.

More development ideas could be collected from shop employees and from shop manager. Making a questionnaire for them would reveal which enhancements and developments the users of the software would like to see.

At some point it could be worthwhile to go through customer processes and ways of working. During this analysis it would be good to go through the software as a part of the customer process and consider, if its support for the daily work could be enhanced. It should also be considered how the software could bring more value for the business.

Bibliography

Abeysinghe, S. 2009. PHP Team Development. Packt Publishing. United Kingdom.

Bari, A., Syam, A. 2008. CakePHP Application Development. Packt Publishing. United Kingdom.

Braude, E., Bernstein, M. 2011. Software engineering : modern approaches. John Wiley & Sons, Inc. United States of America.

DuBois, P. 2003. MySQL Second Edition. Sams Publishing. United States of America.

Gilmore, W. 2006. Beginning PHP and MySQL 5: From Novice to Professional, Second Edition. Apress. United States of America.

Lengstorf, J. 2009. PHP for Absolute Beginners. Apress. United States of America.

MacIntryre, B., Danchilla, B. & Gogala, M. 2011. Pro PHP Programming. Apress. United States of America.

Sommerville, I. 2011. Software Engineering, Ninth Edition. Pearson Education, Inc. United States of America.

Tahaghoghi, S., Williams, H. 2007. Learning MySQL. O'Reilly Media. United States of America.

TopTen Reviews 2012. 2013 Compare Best CRM Software. URL: http://crm-software-review.toptenreviews.com/. Accessed: 21 Nov 2012.

Trachtenberg, A., Sklar, D. 2006. PHP Cookbook, Second Edition. O'Reilly Media. United States of America.

Ullman, L. 2006. Visual Quickstart Guide MySQL, Second Edition. Peachpit Press. United States of America.

Vaswani, V. 2004. MySQL: The Complete Reference. McGraw-Hill Companies. United States of America.

WikiVs 2012. MySQL vs PostgreSQL. URL: http://www.wikivs.com/wiki/MySQL_vs_PostgreSQL. Accessed: 20 Nov 2012.

Appendices

Appendix A: Software Requirements Document

Appendix B: Software Design Document

Appendix C: Software Test Plan Document

Appendix D: Customer Interview

Appendix A: Software Requirements Document

Timo Aho

Appendix A

2012

Table of contents

1 Introduction

Botisto is a Spanish company that sells men's shoes. The shoes are made exclusively for each customer according to their feet size and individual preferences.

When a customer enters Botisto store, the feet of the customer are scanned with a foot scanning device. After the measurement has been taken, the sales person is going to suggest shoes to be tried on. Customer may need to try next width or size of the shoe depending on the personal preferences. When the optimal shoe size has been found for a customer, the size is being marked down.

Now the customer can place an order, which is going to be processed and sent to Botisto's factory. When placing the order the customers can choose the model, the leather and the outsole of the shoe.

So far the customer information is written to paper by the employees in the store. In order to enhance the usage of the customer data and to speed up the daily work, a customer information handling system needs to be developed.

2 Environment

The customer information handling system will be implemented in PHP. MySQL database will be used for storing the customer information. The system will be used with a web browser and it is going to be deployed to a Botisto server.

This approach is chosen in order to reduce the time needed to take the software in use in new stores.

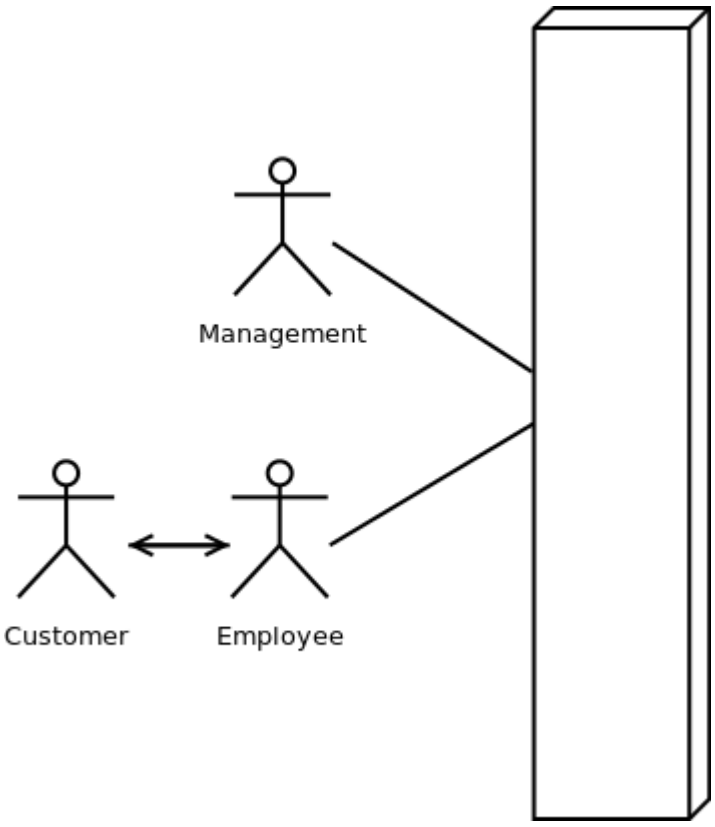
3 Actors using the system

Employee

The employees will use the system in their daily work. They will be usually the ones to register new customers to the system when the customer decides to measure their feet. Customers are not able to use the system. They can verify their information by discussing with the employee.

Management

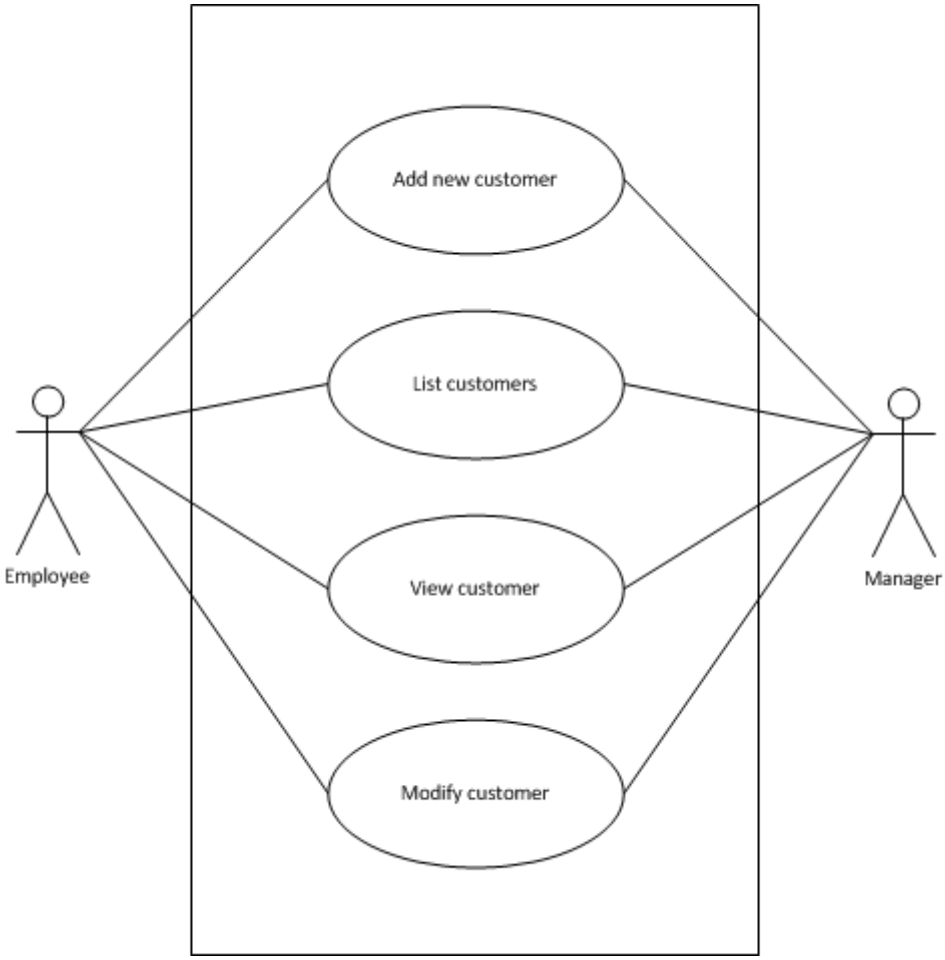
The Botisto management will be able to use this information when they process the customer shoe orders. If the customer approves, the same information can be used for marketing purposes also.



Picture 1: Actor diagram

4 Use Cases

The following use cases are going to be implemented during this project.



Picture 2: Use case diagram

4.1 Add new customer

Employees and managers can add new customers to the customer information system. The customer information will be collected after the customer has decided to scan his feet. The employee or manager will input and save the customer information.

4.1.1 Layout draft for new customer

Standard link list Add new customer (link) Customer list (link)	**New customer** (heading)
	Error messages (if any)
	Label: <text input field: blank> Label: <text input field: blank> Label: <text input field: blank> Label: <text input field: blank>
	<Button: Save> <Button: Cancel>

4.2 Customer list

Employees and managers can list all customers which have been saved to the system. On this page the customer will be selected for viewing and modification. Customers will be shown alphabetically ordered by their last names.

4.2.1 Layout draft for customer list

Standard link list (same as in 4.1.1) Add new customer (link) Customer list (link)	**Customer list** (heading)
	Error messages (if any)
	Customer 1: Last Name, First Name - View (link) Modify (link) Customer 2: Last Name, First Name - View (link) Modify (link) Customer 3: Last Name, First Name - View (link) Modify (link) Customer 4: Last Name, First Name - View (link) Modify (link)

4.3 View customer

After the customer has been selected from the customer list, the employees and managers can view all information stored for the customer.

4.3.1 Layout draft for view customer

Standard link list (same as in 4.1.1) Add new customer (link) Customer list (link)	**View customer** (heading)
	Error messages (if any)
	Label: Attribute Label: Attribute Label: Attribute Label: Attribute

4.4 Modify customer

After the customer has been selected from the customer list, the employees and managers can modify information stored for the customer.

4.4.1 Layout draft for modify customer

Standard link list (same as in 4.1.1) Add new customer (link) Customer list (link)	**Modify customer** (heading)
	Error messages (if any)
	Label: <text input field: showing current values> Label: <text input field: showing current values> Label: <text input field: showing current values> Label: <text input field: showing current values>
	<Button: Save> <Button: Cancel>

5 Information collected from customers

The following information is currently being collected from the customers.

Data name	Data description	Data type	Mandatory data	Maximum length
Sizing code	Unique number used to identify feet measurement	Integer	Yes	20
First Name	First name	Text	Yes	50
Last Name 1	First last name	Text	Yes	50
Last Name 2	Second last name	Text	No	50
Address	Customer street address	Text	Yes	128
City	City where customer lives in	Text	Yes	30
State	State (if in use)	Text	No	30
Postal code	Postal code of the customer	Text	Yes	12
Country code	Code for the country where customer lives in	Text	Yes	2
Telephone	Telephone number	Text	No	20
Email	Email address	Text	Yes	60
Password	Password	Text	No	40
Language code	Language code of the language the customer speaks	Text	Yes	2
Direct marketing	Does customer allow direct marketing	Boolean	Yes	1
Newletter	Should customer receive newsletter	Boolean	Yes	1
LLLA_FIT	Left leg last set A fit	Integer	Yes	10
LLLA_SIZE	Left leg last set A size	Integer	Yes	10
RLLA_FIT	Right leg last set A fit	Integer	Yes	10
RLLA_SIZE	Right leg last set A size	Integer	Yes	10
LLLB_FIT	Left leg last set B fit	Integer	Yes	10
LLLB_SIZE	Left leg last set B size	Integer	Yes	10

RLLB_FIT	Right leg last set B fit	Integer	Yes	10
RLLB_SIZE	Right leg last set B size	Integer	Yes	10
Additional notes	Additional customer specific notes	Text	No	200
To be deleted	Indicator if customer should be deleted in the next deletion round	Boolean	No	1

5.1 Possible fits and sizes

The shoe sizes are indicated with two numbers. Shoe fits are indicated with one letter. There also needs to be a selection for both shoe sizes and fits that there is no optimum size or fit available for customer.

6 Actors access rights to the stored information

Both employees and managers are able to create, read and modify the information stored. Neither employees or managers are able to delete anything. Customer information which should be deleted are going to be flagged to "Additional notes" field. Database administrator will delete the flagged records periodically.

7 Non-functional requirements for the software

7.1 Usability

The software should be easy to use, as no user guide is going to be created during this project. The system should be usable by multiple users at the same time.

7.2 Performance

The software should simple enough so that the performance is not going to be an issue. Performance target is that actions should not take more than four seconds to complete. This target should be reachable from the store's network.

7.3 Software language

The software user interface will be implemented in English language as all Botisto employees have the needed English skills.

7.4 Information sensitivity

No sensitive information (social security numbers, credit card information) should be stored into the system. All customers need to accept that their information will be stored to our system at the time of the purchase.

Information should be stored in such manner that it can not be accessed by unauthorized users.

7.5 User interface style

Style of the user interface should follow the Botisto website styling and corporate appearance. The style will be adapted from Botisto website at the time the software is ready to be delivered.

Appendix B: Software Design Document

Timo Aho

Appendix B

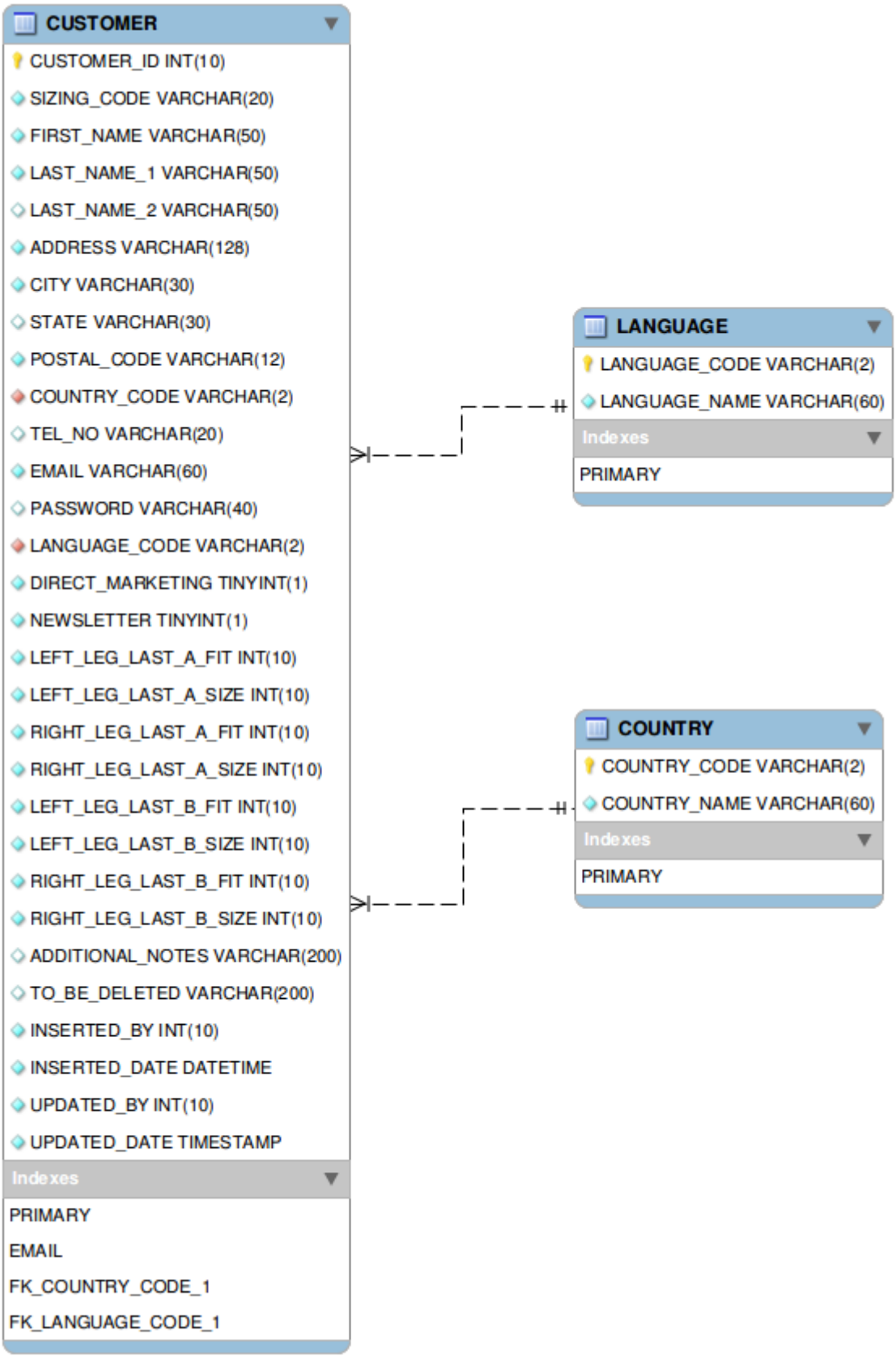
2012

Table of contents

1 Database design

Database design originates from the requirements specified in the software requirements document. Database which is chosen for the project is MySQL. The database was chosen as it is widely supported by different hosting providers. Other major reason was to avoid licensing costs. Database was designed using MySQL Workbench.

1.1 Database schema



1.2 Table descriptions

Name	Description	Initial data	Maximum	Yearly increase / decrease rate
Customer	Contact information for Botisto customers	2 test customers	100 000	+ 1000 -10
Country	Countries used for address information	247	300	+/- 10
Language	Pre-defined languages for customer communication	3	50	+/- 10

1.3 Column descriptions

1.3.1 Customer table

Column	Data type	Length	Can be null?	Remarks
CUSTOMER_ID	INT	10	NO	Primary key, auto increment from 10000 onwards
SIZING_CODE	VARCHAR	20	NO	
FIRST_NAME	VARCHAR	50	NO	
LAST_NAME_1	VARCHAR	50	NO	
LAST_NAME_2	VARCHAR	50	YES	
ADDRESS	VARCHAR	128	NO	
CITY	VARCHAR	30	NO	
STATE	VARCHAR	30	YES	
POSTAL_CODE	VARCHAR	12	NO	
COUNTRY_CODE	VARCHAR	2	NO	Foreign key
TEL_NO	VARCHAR	20	YES	
EMAIL	VARCHAR	60	NO	Unique key

PASSWORD	VARCHAR	40	YES	
LANGUAGE_CODE	VARCHAR	2	NO	Foreign key
DIRECT_MARKETING	TINYINT	1	NO	
NEWSLETTER	TINYINT	1	NO	
LEFT_LEG_LAST_A_FIT	INT	10	NO	
LEFT_LEG_LAST_A_SIZE	INT	10	NO	
RIGHT_LEG_LAST_A_FIT	INT	10	NO	
RIGHT_LEG_LAST_A_SIZE	INT	10	NO	
LEFT_LEG_LAST_B_FIT	INT	10	NO	
LEFT_LEG_LAST_B_SIZE	INT	10	NO	
RIGHT_LEG_LAST_B_FIT	INT	10	NO	
RIGHT_LEG_LAST_B_SIZE	INT	10	NO	
ADDITIONAL_NOTES	VARCHAR	200	YES	
TO_BE_DELETED	VARCHAR	200	YES	
INSERTED_BY	INT	10	NO	
INSERTED_DATE	DATETIME	-	NO	
UPDATED_BY	INT	10	NO	
UPDATED_DATE	TIMESTAMP	-	NO	

1.3.2 Country table

Column	**Data type**	**Length**	**Can be null?**	**Remarks**
COUNTRY_CODE	VARCHAR	2	NO	Primary key
COUNTRY_NAME	VARCHAR	60	NO	

1.3.3 Language table

Column	Data type	Length	Can be null?	Remarks
LANGUAGE_CODE	VARCHAR	2	NO	Primary key
LANGUAGE_NAME	VARCHAR	60	NO	

2 Database implementation

2.1 SQL script to create tables

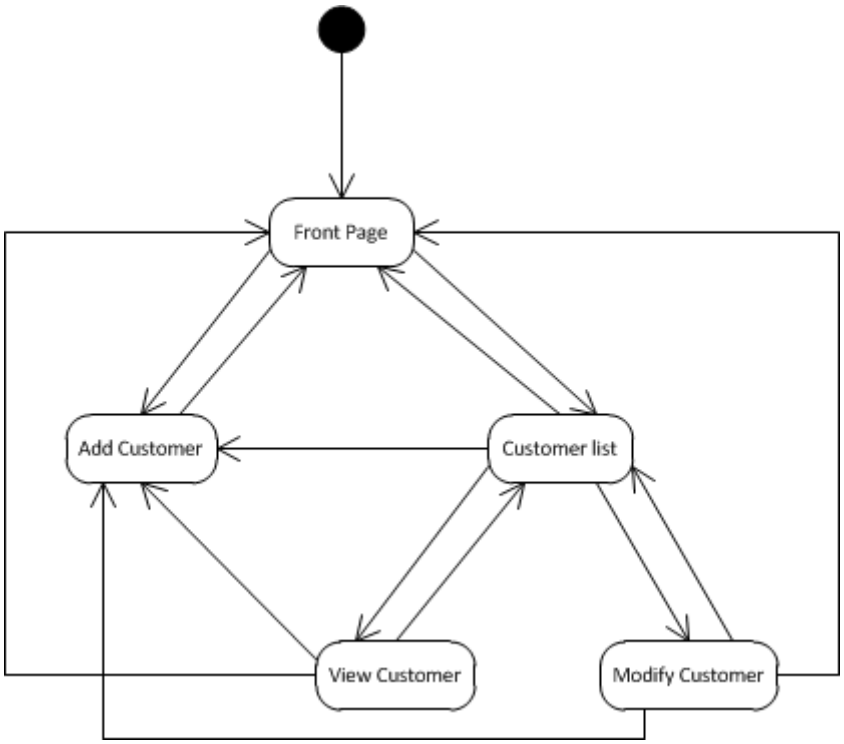
```
CREATE TABLE `COUNTRY` (
`COUNTRY_CODE` VARCHAR(2) NOT NULL,
`COUNTRY_NAME` VARCHAR(60) NOT NULL,
PRIMARY KEY (`COUNTRY_CODE`)
) ENGINE=INNODB DEFAULT CHARSET=UTF8;

CREATE TABLE `LANGUAGE` (
`LANGUAGE_CODE` VARCHAR(2) NOT NULL,
`LANGUAGE_NAME` VARCHAR(60) NOT NULL,
PRIMARY KEY (`LANGUAGE_CODE`)
) ENGINE=INNODB DEFAULT CHARSET=UTF8;

CREATE TABLE `CUSTOMER` (
`CUSTOMER_ID` INT(10) NOT NULL AUTO_INCREMENT COMMENT
'SYSTEM GENERATED ID (10000+1)',
`SIZING_CODE` VARCHAR(20) NOT NULL,
`FIRST_NAME` VARCHAR(50) NOT NULL,
`LAST_NAME_1` VARCHAR(50) NOT NULL,
`LAST_NAME_2` VARCHAR(50),

`ADDRESS` VARCHAR(128) NOT NULL,
`CITY` VARCHAR(30) NOT NULL,
`STATE` VARCHAR(30),
```

```
`POSTAL_CODE` VARCHAR(12) NOT NULL,
`COUNTRY_CODE` VARCHAR(2) NOT NULL,

`TEL_NO` VARCHAR(20),
`EMAIL` VARCHAR(60) NOT NULL,
`PASSWORD` VARCHAR(40),
`LANGUAGE_CODE` VARCHAR(2) NOT NULL,
`DIRECT_MARKETING` TINYINT(1) NOT NULL,
`NEWSLETTER` TINYINT(1) NOT NULL,

`LEFT_LEG_LAST_A_FIT` INT(10) NOT NULL,
`LEFT_LEG_LAST_A_SIZE` INT(10) NOT NULL,
`RIGHT_LEG_LAST_A_FIT` INT(10) NOT NULL,
`RIGHT_LEG_LAST_A_SIZE` INT(10) NOT NULL,
`LEFT_LEG_LAST_B_FIT` INT(10) NOT NULL,
`LEFT_LEG_LAST_B_SIZE` INT(10) NOT NULL,
`RIGHT_LEG_LAST_B_FIT` INT(10) NOT NULL,
`RIGHT_LEG_LAST_B_SIZE` INT(10) NOT NULL,

`ADDITIONAL_NOTES` VARCHAR(200),
`TO_BE_DELETED` VARCHAR(200),

`INSERTED_BY` INT(10) NOT NULL,
`INSERTED_DATE` DATETIME NOT NULL,
`UPDATED_BY` INT(10) NOT NULL,
`UPDATED_DATE` TIMESTAMP NOT NULL DEFAULT CUR-
RENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
PRIMARY KEY (`CUSTOMER_ID`),
UNIQUE KEY `EMAIL` (`EMAIL`),
CONSTRAINT `FK_COUNTRY_CODE_1` FOREIGN KEY (`COUN-
TRY_CODE`) REFERENCES `COUNTRY` (`COUNTRY_CODE`),
```

```
CONSTRAINT `FK_LANGUAGE_CODE_1` FOREIGN KEY (`LAN-
GUAGE_CODE`) REFERENCES `LANGUAGE` (`LANGUAGE_CODE`)
) ENGINE=INNODB DEFAULT CHARSET=UTF8;
ALTER TABLE `CUSTOMER` AUTO_INCREMENT=10000;
```

3 Navigation diagram

User needs to log in to the system on the front page. After a successful login user can navigate from every page to front page, customer registration page and list customer page. View customer and modify customer pages can be accessed only from list customers page.



4 Source code file descriptions

File	Visibility to user	Description
db_connection.php	Non-visible	File containing database connection information.
db_sql.php	Non-visible	File containing functions to access database.
header.php	Visible: Included by content file	Beginning of the page html tags. Contains also DB initialization.
footer.php	Visible: Included by content file	Contains the end tags for the html-page.
menu.php	Visible: Included by content file	Link list on the left.

index.php	Visible: content file	First page to be accessed. Contains authentication.
customer_list.php	Visible: content file	Page for viewing list of all customers.
new_customer.php	Visible: content file	Page for creating a new customer.
view_customer.php	Visible: content file	Page for viewing all stored customer information.

5 Page structure

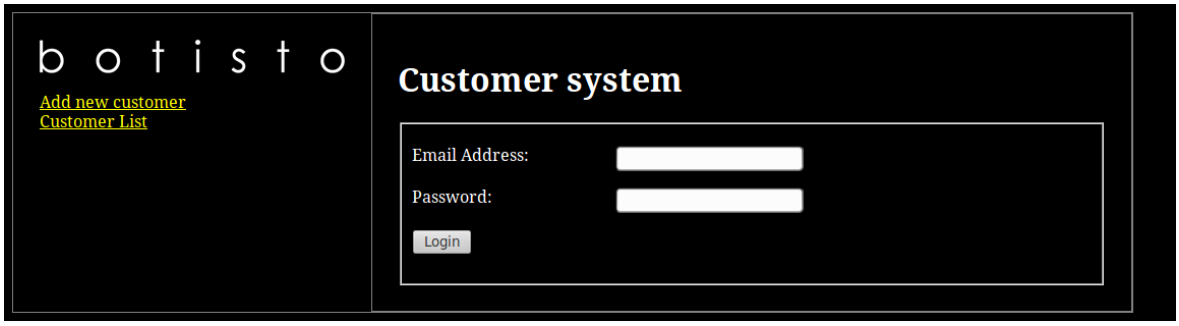
The page the customer is requesting is going to do include for header, menu, footer and db_sql files. This way the software is modular and easier to maintain.

header.php	
menu.php	<< Content PHP file >>
footer.php	

6 User Interface design

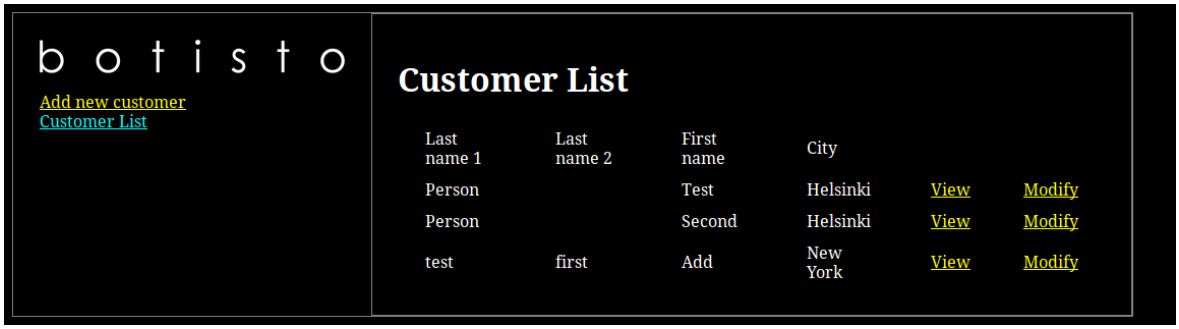
User interface must fit into the Botisto coloring scheme and corporate image.

6.1 Front page screenshot

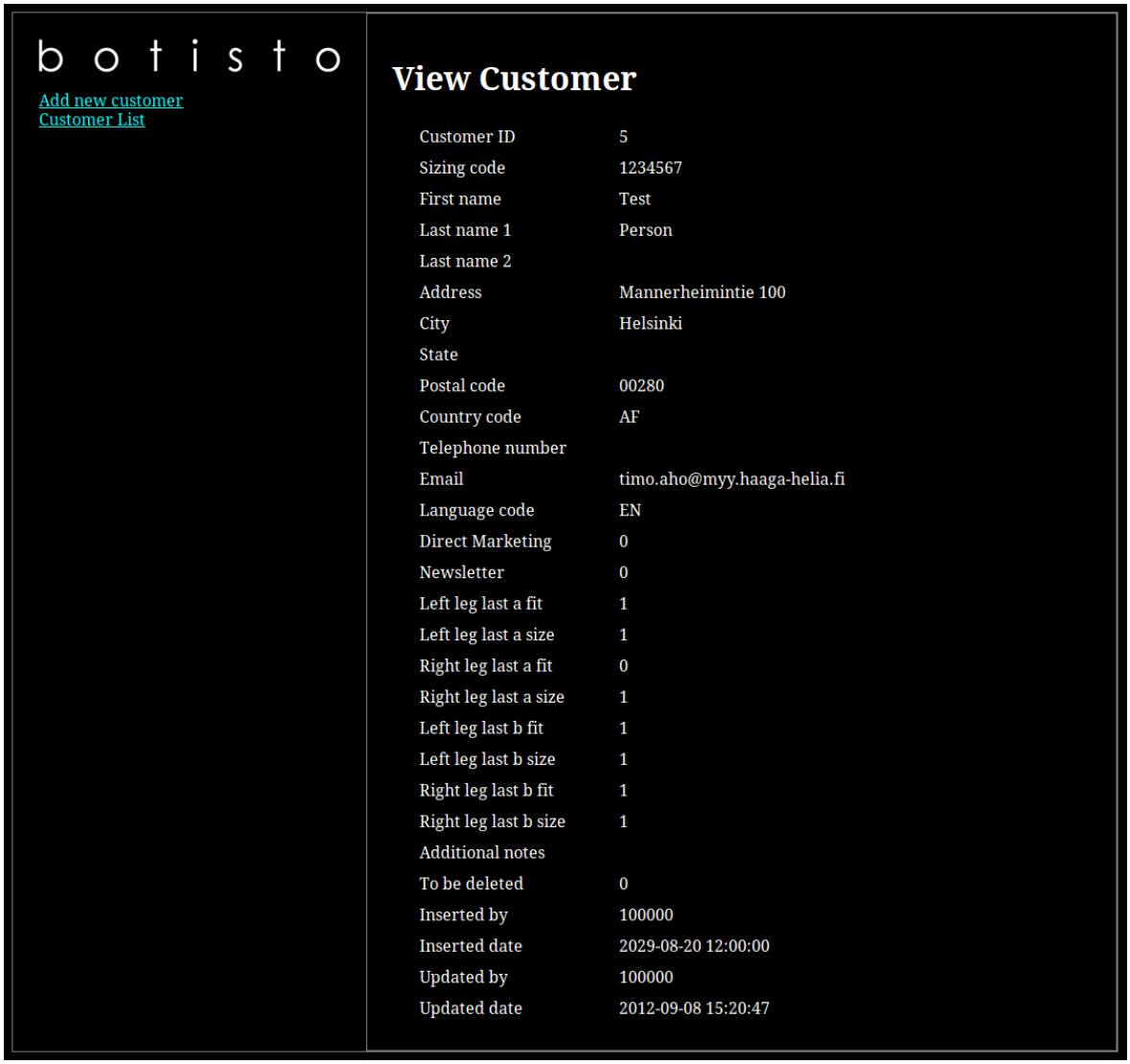


6.2 New customer screenshot

6.3 Customer list screenshot



6.4 View customer screenshot



6.5 Modify customer screenshot

Appendix C: Software Test Plan Document

Timo Aho

Appendix C

2012

Table of contents

1 Background

The purpose of the testing is to ensure that there are no defects in the software. Testing should also verify that the software works logically as planned.

2 Scope

In software testing it is quite rare to have all possible scenarios and functionalities tested in an application. In most cases the amount of combinations are virtually infinite, so in reality testing everything is not even possible.

The test scope needs to be set on a level which should be good enough for the applications intended usage. Also the time and budget usable for the project needs to be taken into consideration.

The Botisto customer information handling system is not business critical in that sense that in case it is not working it doesn't interrupt business. It will cause unwanted delay and inconvenience though.

3 Test plan

The following use cases should be tested:

- Creating a customer
- Listing existing customers
- Modifying an existing customer
    - Marking an existing customer as to be deleted

As the software doesn't have user input validation in the first release, use cases which are known to fail are exceluded from the testing. This means that when creating a new customer, all the mandatory information is going to be filled in.

4 Test execution

The software should be tested by following the next steps. Mark down all defects/issues found. Successful test run should not take more than one hour to complete.

4.1 Test sequence

1. Log in to the software.
2. Create five new customers each with different set of data. Fill in all mandatory information.
3. List existing customers.
4. Create five new customers with only mandatory information entered. The information should include special characters in order to verify that the UTF-8 character set handling works.
5. List existing customers.
6. Test modify existing customer use case. For the first five modify only some attributes. On the other five, update all possible fields (including mark customer as deleted).

HAAGA-HELIA
University of Applied Sciences

Appendix D: Customer Interview

Timo Aho

Appendix D

2012

Question: How has the software changed the daily work in the shop (front office/back office)?
Answer: Implementation of the functionality has eased our daily workload. Before having the functionality we had to manage our customer information manually and there was always a risk of problems arising due to human errors. Obviously, we still have to do our work with care but overall it's lot easier now. The functionality to manage customer information is easy to use. Updating our customer database is also easy. Employees also like it because they don't have to deal with paper so much anymore. It also gives a more professional image of our company to our customers when we use software instead of paper and pen in the style of "Mom and Pop Store".

Management's life has been eased as well because before it was necessary to go to the shop to check some customer information but nowadays we just log in to our website to update customer information.

Question: How would you evaluate the software as a whole?
Answer: Very easy to use. It's very logical and it takes only about 5-10 minutes to understand how it works. It's very easy to teach our employees to use it as well. There are now seven people using the software and no one has had any problem with the software.

Question: What are the pros and cons working with the software?
Answer: It's just easier and makes your life easier. Managing customer information is easier. Because its web based software it's also convenient that you have access to your information anywhere. This also has its negative sides, though: one time I needed some information from one of our clients but I couldn't get the information because my internet connection was not working properly.

We still keep our old system as a backup system in the store. It's always possible that when we are adding customer information to the software that the software won't work because the internet connection is not working. So it's not as reliable as our old system (paper and pen).

Question: What should be the next areas of development with the software?

- **Answer:** Improving the reliability of the software in case of a lost internet connection.
- Being able to see past changes made in the customer information database
    - In case there's a mistake, we could easily restore the data