Requirements Document

Version 1.3

Client: A software toolbox for small retail shops

Dan Plămădeală, S3436624
Abel Nissen, S3724786
Ruben Biskupec, S4235762
Florian de Jager, S3775038
Arjan Dekker, S3726169

university of groningen

Faculty of Science and Engineering

Lecturer: Mohamed Soliman,
Andrea Capiluppi
Teaching Assistant: Hichem Bouakaz
Last updated: Thursday 11th June, 2020

Contents

Introduction

Several hundreds, if not thousands, of small supermarkets in The Netherlands are using archaic software systems to run their Points of Sales (cash desks) which are not very friendly in terms of extracting management information. Simple queries like "what is being sold in my shop at which time-of-day", "how many of item X are being sold per week", "Should I order more or less of item Y" are difficult to answer with hard data. Hence most shop owners rely on their "gut-feeling" which is not a smart thing to do.

The "Dorpswinkel" in Sauwerd (a village 10km north of Groningen) is a good example. The shop is owned and run by the people living in the village. As it is a "voluteers project" there are many people doing work in the shop and there is not a single "set of brains" in the shop that knows all. So relying on gut feeling is not an option - we need hard data on sales to be able to optimize the shop.

As a team our objective is to build a parallel MySQL database that will work as a better basis to retrieve management information. We also want to build a smartphone app that is able to scan a barcode and popup the recent sales of this product. The final goals are to make information on the stock and sales of goods accessible in and to create a generic database platform that allows queries to be performed alongside apps that contain these queries.

Actors

- manager: The manager of the shop. Keeps track of the financial situation of the shop and is responsible for it.
- employee: A standard employee of the shop that does various jobs in the shop. One example of such job is comparing and updating price labels.
- moderator: Is also an employee of the shop, but has extra privileges. The moderator is able to view purchasing information and issue queries.

User stories

This chapter focuses on actions users should be able to perform in the finished product.

Critical Functional Requirements

[c1] As an employee, I want there to be a website that is used as primary interface to the shop's data, so that I can work with the data in a common-day manner.

- Backbone of accessing the database.

[c2] As an employee, I want to be able to retrieve sales numbers, of a product of my choosing, over the last week, month, quarter and year, via a website, so that I have concrete data to base the ordering of new stock on.

[c3] As a manager, I want the parser to automatically upload new sales data that is created by Casman, the current software the shop uses, in the mysql-database, so that I don't need to manually upload them.

- The system automatically opens a new zip-file that is created by Casman every day, parses it and stores the data in the database.

[c4] As a manager, I want the website access to the shop's data in the mysql-database to be secure (protected by an SSL-certificate), so that I don't need to worry about data safety.

- Accessing data safely.

Important Functional Requirements

[i1] As a manager, I want to be able to access the website off premise, so that I can keep track of sales while at home.

- Website should be accessible off premise.

[i2] As an employee, I want to be able to scan the barcodes of products using a smartphone ('s camera), so that I don't need to manually type the name or EAN code every time when I want to access a specific product's data via the website.

- Should be able use any smartphone's camera to read barcodes of products.

[i3] As a manager, I want to be able to view 'Koppelverkoop', i.e. pairs of products that are sold together, via the website, so that I have more insight in the combinations.

- (For instance, is the pair of beer and potato chips often bought together?)

[i4] As a manager, I want to be able to retrieve the shop's revenue (cash withdrawals should be excluded from this), over a period of my choosing, via a website, shown by an interval of my choosing* in a graph, so that I can identify positive or negative growth.

- Useful to decide if the shop needs to be open at certain times.

[i5] As a manager, I want to be able to view the number of customers in the shop, over a period of my choosing, via a website, shown by an interval of my choosing* in a graph, so that I see which days (hours, etc.) are more busy than others.

- Useful for deciding the number of employees that need to be present in the shop at a given time.

Useful Functional Requirements

[u1] As a manager, I want the system to calculate the margins of a product by subtracting the buying price from the selling price, so that I can have more insight in which products to promote.

- Useful for computing profit/loss.

[u2] As a manager, I want to be able to retrieve sales numbers, of a product of my choosing, over a period of my choosing, via a website, shown by an interval of my choosing* in a graph, so that I have full insight in the data of this product and can base my opportunities on it.

- Useful for deciding on deals, predicting sales increase/decrease due to the season, etc.

*: Intervals should be predefined, and especially include half-hours.

Non-Functional Requirements

1. **Security**
   - ☐ To secure communication, https should be used.
   - ☐ The website should only be accessible when the employee is on the local network of the shop.
2. **Performance**
   - ☐ The parser script should be able to process a receipt in less than 10 milliseconds so as not to get a backlog of receipts in the system.
   - ☐ Queries should not put a big strain on the server. At least 10 queries need to be able to be performed at the same time without noticing any delay.
   - ☐ API calls should take less than a second. Again to prevent a backlog.
   - ☐ Reading barcodes should be close to instantaneous.
3. **User friendliness**
   - ☐ The language of the user-interface should be in Dutch.
4. **Usability**
   - ☐ The entire website should be easy to understand and use. A single 20 minute training session should be enough for a basic employee to know how to operate it.
   - ☐ Experienced employees should be able to use all the system functions after using it for 2 shifts. After this, the average number of errors made by experienced users should not exceed two per day.
5. **Portability**
   - ☐ The website and all its functions (barcode scanner, graphs, etc.) should work properly on modern browsers including browsers on phones.
   - ☐ The client app should work on both iOS and Android platforms.
   - ☐ Browsers such as Chrome, Firefox, Safari.
6. **Maintainability**
   - ☐ The code should be clear and readable. [For potential new employees to be able to tweak and update it.]
   - ☐ There should be an English documentation of the system.

Won't Do

☐ -

Global Overview

A rough overview of the features we will be focussing on. For details see the design document.

☐ Store the shop's (sales) data in a mysql database accessible via a website.

☐ Allow for extensive queries to be run via that website.
Think of queries such as:

- How many products are being sold between 17:30 and 18:00
- How do the sales of a product change during the year? (Show number of sales over 12 months)

☐ Access product's info in the database by scanning its barcode with a smartphone camera.

☐ View products frequently sold together in pairs.

☐ View product margins and overall shop revenue.

Customer Meetings

When	What
26/02/2020	- Introduction meeting, discussed initial requirements. - Decided to prepare a requirements document for next time.
09/03/2020	- Confirmed requirements, discussed coding approach. - Decided to prepare a short demo presentation for our next meeting.
24/03/2020	- Short demo presentation. - General discussion about the project.
20/04/2020	- Demo presentation. - Gathered more updated requirements.
27/04/2020	- Showed finished modules. - Discussed implementation in the shop.
06/05/2020	- Discussed actual deployment in the shop. - Updated client on our progress.
12/05/2020	- Hands-on mobile phone demo. - Made preparations for testing.
19/05/2020	- Discussed bugs and improvements. - Gathered hands-on feedback.
26/05/2020	- Prepared for final presentation.
10/06/2020	- Discussed employee test results. - Final meeting.

Change log

Who	When	Which section	What	Time
Dan	27.02.20	The document	Created the document, added features.	20 min
Dan	27.02.20	The document	Added log table and user stories template. Made some changes to the layout.	10 min
Abel	28.02.20	The userstories document	Added 6 user stories.	1h
Abel	06.03.20	The userstories document	Added all provided requirements, rewrote and sorted them.	40min
Abel	06.03.20	The userstories document	Updated requirements (non and functional ones) with all available data so far.	30min
Florian	06.03.20	The userstories and users document	Extended the non-functional requirements and added basic info to user.	1h
Ruben	08.03.20	Introduction	Added the Introduction.	1h
Abel	10.03.20	The userstories document	Split complex requirements into multiple more clear ones. Added loyalty card user stories.	1h
Abel	10.03.20	The customer meetings document	Updated document with meeting had so far and decisions made.	10min
Ruben	10.03.20	User stories	Split compound user stories and other changes.	20min
Dan	13.03.20	Front Page	Improved the layout.	5 min
Dan	13.03.20	non-functional requirements	improved the non-functional requirements.	20 min
Abel	13.03.20	non-functional requirements	finished non-functional requirements. 1/3	1h
Arjan	13.03.20	non-functional requirements	finished non-functional requirements. 2/3	1h

Who	When	Which section	What	Time
Florian	13.03.20	non-functional requirements	finished non-functional requirements 3/3	1h
Abel	13.03.20	Introduction	edited the introduction.	15 min
Ruben	16.04.20	Functional Requirements	Prioritized user stories.	15 min
Abel	20.04.20	Functional Requirements	Updated due to new meeting with client.	15 min
Abel	27.04.20	Global Overview	Added important functions in a quick overview.	30 min
Ruben	11.05.20	User stories	Made user stories more specific.	45 min
Abel	16.05.20	Introduction & Global Overview	Compacted the intro, detailed the overview.	30 min
Abel	19.05.20	User meetings	Updated relevant information.	20 min
Abel	24.05.20	Functional and Non-Functional Requirements	Edited requirements to be testable, removed duplicates, included new and more up-to-date ones.	1,5h
Abel	11.06.20	User meetings & Functional Requirements	Updated meeting information, added slight detail to requirements.	15 min