INVENTORY CONTROL AT LAFRESCO

Software Requirements Specification

Version 1.0.0

Date 25-01-2019

Team Members
Ravi Maurya
Vinayak Mohite
Deepak P.V.R
Prudhvi Kiran Katuri

Prepared for

CS 258 Software Engineering

Spring 2019

Revision History

Date	Description	Author	Comments

Document Approval

The following Software Requirements Specification has been accepted and approved by the following:

Signature	Printed Name	Title	Date

Table of Contents

REVISION HISTORY **2**

DOCUMENT APPROVAL **2**

1. INTRODUCTION **5**

1.1. PURPOSE OF THE SYSTEM 5
1.2. SCOPE OF THE SYSTEM 5
1.3. DEFINITIONS, ACRONYMS, AND ABBREVIATIONS 5
1.4. REFERENECES 6
1.5. OVERVIEW 6

2. GENERAL PERSPECTIVE **6**

2.1. PRODUCT PERSPECTIVE 6
2.2. PRODUCT FUNCTIONS 6
2.3. USER CHARACTERISTICS 7
2.4. GENERAL CONSTRAINTS 7
2.5. ASSUMPTIONS AND DEPENDENCIES 7

3. SPECIFIC REQUIREMENTS **7**

3.1. EXTERNAL INTERFACE REQUIREMENTS 7
3.1.1 User Interfaces 8
3.1.2 Hardware Interfaces 8
3.1.3 Software Interfaces 8
3.1.4 Communication Interfaces 8
3.2 FUNCTIONAL REQUIREMENTS 9
3.2.1 Billing 9
3.2.2 Add Customer and User Account 9
3.2.3 Account and Stock Management 9
3.3 CLASSES/OBJECTS 9
3.4 NON-FUNCTIONAL REQUIREMENTS 9
3.4.1 Performance 9
3.4.2 Reliability 10
3.4.3 Availability 10
3.4.4 Security 10

3.4.5 Maintainability 10
3.4.6 Portability 10
3.5 INVERSE REQUIREMENTS 10
3.6 DESIGN CONSTRAINTS 10
3.7 LOGICAL DATABASE REQUIREMENTS 10
3.8 OTHER REQUIREMENTS 11

1. Introduction

1.1 Purpose of the system

A case study at "La-fresco" (on-site general store management) cited issues regarding a basic resources requirement list that has to be maintained manually by the staff. To keep track of their inventory levels they have to calculate a list of the groceries utilized during a course of time, calculate and analyze the requirements for the future, and place their next order to the vendors if needed. This process takes up a lot of time and human effort and is also prone to human error. This poses a problem of a situation that the staff at "La-fresco " as well as many other stores faces. It takes up a lot of time to manually keep track of sales and place correct orders to vendors, wasting useful labor in trivial works. A product which would assist in tackling the above-mentioned problems would prove to be fruitful to clients such as "La-fresco" inventory manager and similar enterprises as this product would help convert the unproductive time to something more useful, by removing the unnecessary error prone complications and efforts.

1.2 Scope of the system

The project aims at providing an efficient interface to the store for managing their grocery inventory based on each item sold. The basic idea involved here is that each item is linked to its atomic ingredients which are stored in a database. At the end of each day, the system analyzes the total sale of menu items and proportionately deducts the appropriate amount from the resource database.
The Inventory manager proposed a new feature of "Wallet Balance". In which, every time a customer buys some items respective amount will be deducted from his/her wallet. The product also aims to keep track of the shelf life of resources.

1.3 Definitions, acronyms, and abbreviations

Manager: The manager implies the manager of the store who handles all the administrative works.
Item: This is the entity that the customer buys.
Vendor: This is the store that provides the items with the required items.
Order: Order is the list of items and the quantities that is or is to be requested from the vendor.

1.4 References

1.5 Overview

We propose to develop software that keeps track of the inventory of the store and updates it according to daily sales. Each item is linked to respective stock and as each item is sold the stock balance gets updated. These changes in inventory are kept track of through utilizing a database. We are also intended to create another database for balance records of the customers which are directly linked to the stock database. Every item which the customer buys is reflected in a bill. There is a certain feature 'membership' which carries certain advantages such as discount etc.

2. General Description

2.1 Product Perspective

- ✓ Platform
- ✓ Admin allowing user accounts to be created
- ✓ Number of users being supported by system
- ✓ Search
- ✓ Billing system
- ✓ Stock management

2.2 Product Functions

- ✓ Add customer
- ✓ Edit customer info
- ✓ Add user
- ✓ Edit user
- ✓ New sales person
- ✓ Generate an older bill or transaction histories
- ✓ View/report product, stock, customer, customer transaction, customer account
- ✓ Daily in and out of products
- ✓ Daily out and daily in
- ✓ Available product of stock

- ✓ Collection summary
- ✓ Login transactions logs of logins
- ✓ Product with barcode
- ✓ Bill precancellation list
- ✓ Sales product details
- ✓ Bill precancellation details

2.3 User Characteristics

- ✓ Not an occasional user
- ✓ Has experience in this kind of accounts for customers.
- ✓ The admin has access to everything. The admin creates the customer accounts. The admin can change the wallet balance and stock details. The admin gets the overall analysis of accounts.

2.4 General Constraints

2.5 Assumptions and Dependencies

The following are the requirements from the server:

- ✓ Windows based system(preferred)
- ✓ Node JS installed(preferred)
- ✓ MySQL database engine installed

It is assumed that the operating system and other underlying pieces of the software on the user-side are free from any error which may affect the functioning of the system.

The software is based on Windows operating system.

3. Specific Requirements

3.1. External Interface Requirements

3.1.1 User Interfaces

The software has a login page for admin. The admin will then be redirected to the homepage of the software where he can perform the following operations:

- ✓ Start new Billing:
  This will take the user to a page where he can perform:
  i. Billing for both registered and unregistered users
  ii. View reports of Sales and Stock.
  iii. Add, delete and modify stock details.
- ✓ The Quick User Adds:
  This will take the admin to the page where he can add, delete and modify a customer.
- ✓ Cancellation of bill
- ✓ Edit profile:
  This will take the admin to the page where he can add, delete and modify users' details.
- ✓ Payment collection: This will take the admin to the page where he can:
  i. View history related to payments.
  ii. Check net profit (daily, monthly, weekly)
- ✓ Change credentials

3.1.2 Hardware Interfaces

For the implementation of our model, the store requires a functional computer with printer to print bills, reports, and fingerprint scanner to identify the registered customer.

3.1.3 Software Interface

This project will be done in mainly two parts. The first one is the desktop application which will run on the system and the user will use it on the same. A second will be a local server running on user's device storing the database.

3.1.4 Communication Interfaces

All the data will be stored on the local server and will be accessed by the application as and when required.

3.2 Functional Requirements

3.2.1 Billing

Whenever the customer wants to buy some items, the admin will scan each item using a barcode scanner. Registered customer will be required to input their thumb impression, which identifies his/her respective account. Unregistered customers need not provide any details.

Items scanned from barcode reader will be added into customer's bill. If the customer is registered then the total bill amount is deducted from his wallet account. If the customer is unregistered, then he will pay manually to the admin according to his convenience. Depending upon the type of customer(member/non-member), the customer will be given a discount which will be decided by the admin. When the payment is done the stock will be updated automatically.

The bill is produced through the printer.

3.2.2 Adding Customer or User

The information of customer/user will be inputted by admin. For a customer, thumb impression and initial wallet amount will be inputted. For a user, the password will be required.

The given details will be stored on the local server in the database.

3.2.3 Account and Stock Management

In this section, admin can view reports of sales and stocks. He can also add, delete, modify item details. The user can select from menu whichever report he wants to see.

Depending on the user's input actions like profit calculation would be done on the basis of the available database.

The system will automatically generate reports in pdf or excel spreadsheet.

3.3 Classes/Objects

3.4 Non-Functional Requirements

3.4.1 Performance

The response time should be low. The system should have 4GB RAM and for the throughput, we need to have 10GB free space.

3.4.2 Reliability

The system should be able to process all the information for analytics and store them permanently as any loss of data will render the whole software useless for the user.

3.4.3 Availability

The software is created by collaborating using a free and open source distributed version control system. Git, and hence contributions of various other programmers and thereby remains updated.
Competent documentation should assist a new programmer to un-added by each code and the goals we aspire to achieve.
Maintenance of the database has been made easy by the user login.

3.4.4 Security

The operating system being used should be updated so that it is free of common vulnerabilities. The application should also have security features to protect it from basic vulnerabilities. The admin should also be able to a timely inspection of the database to ensure safety.

3.4.5 Maintainability

The database should be accessible to the admin so as to carry out maintenance.

3.4.6 Portability

The platforms on which the system runs should be generic enough to allow substantially amount of portability.

3.5 Inverse Requirements

3.6 Design Constraints

3.7 Logical Database Requirements

The database consists of two main tables. One is for customer details like name, address, contact, wallet balance and another one for stock maintenance. A large amount of free storage is recommended for database storage requirements.

3.8 Other Requirements