**Software Requirements**

**Specification**

**for**

**<SureFinventory-Detailed Control Application>**

****

**Version 0.1**

**Prepared by <Esteban, Matias, Abner & Benjamin>**

**<JABAS team>**

**<November 10th 2024>**

***Copyright © 1999 by Karl E. Wiegers. Permission is granted to use, modify, and distribute this document.***

# Table of Contents

[**Table of Contents 2**](#_heading=h.gjdgxs)

[**Revision History 2**](#_heading=h.30j0zll)

[**1. Introduction 1**](#_heading=h.1fob9te)

[1.1 Purpose 1](#_heading=h.3znysh7)

[1.2 Document Conventions 1](#_heading=h.2et92p0)

[1.3 Intended Audience and Reading Suggestions 1](#_heading=h.tyjcwt)

[1.4 Product Scope 1](#_heading=h.3dy6vkm)

[**2. Overall Description 2**](#_heading=h.4d34og8)

[2.1 Product Perspective 2](#_heading=h.2s8eyo1)

[2.2 Product Functions 2](#_heading=h.17dp8vu)

[2.3 Operating Environment 3](#_heading=h.26in1rg)

[2.4 Design and Implementation Constraints 3](#_heading=h.lnxbz9)

[2.5 User Documentation 3](#_heading=h.35nkun2)

[2.6 Assumptions and Dependencies 4](#_heading=h.1ksv4uv)

[**3. External Interface Requirements 4**](#_heading=h.44sinio)

[3.1 User Interfaces 4](#_heading=h.2jxsxqh)

[3.2 Hardware Interfaces 4](#_heading=h.z337ya)

[**4. Other Nonfunctional Requirements 5**](#_heading=h.3whwml4)

[4.1 Performance Requirements 5](#_heading=h.2bn6wsx)

[4.2 Security Requirements 6](#_heading=h.3as4poj)

[4.3 Software Quality Attributes 6](#_heading=h.1pxezwc)

[4.4 Business Rules 6](#_heading=h.49x2ik5)

[**5. Other Requirements 6**](#_heading=h.2p2csry)

[**Appendix A: Glossary 6**](#_heading=h.147n2zr)

[**Appendix B: Analysis Models 6**](#_heading=h.3o7alnk)

[**Appendix C: To Be Determined List 7**](#_heading=h.23ckvvd)

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose

*The purpose of the following document is to define and structure the software requirements for the SureFinventory application.*

1. *Establish the functional requirements for SureFinventory to provide user authentication, inventory management, cash closing, basic accounting management and generate monthly reports.*
2. *Describe the non-functional requirements that ensure a secure, accessible, intuitive and customizable user experience.*
3. *Delimit the scope of the system to specify operations that the system will cover.*

## Document Conventions

*As the main source of requirements we use the book Clean Code, the author of the book, Robert C. Martin, invites each of the group members to review and analyze our code, following and standard and getting our best work.*

## Intended Audience and Reading Suggestions

*The document is addressed to the profiles involved within the development of the SureFinventory application, we have 3 main recipients within the project, which are:*

1. *Developers: They are the ones who provide a detailed description of the functional and non-functional requirements necessary for the creation of SureFinventory must read all the specific requirements, focusing on the essential functionalities and quality standards of the system.*
2. *Project Managers: For managers, it is recommended to start with the introductory and product purpose sections to get an overview of the project scope and objectives.*
3. *End Users: They can review functional requirements and configurations to understand the application's operation and options.*

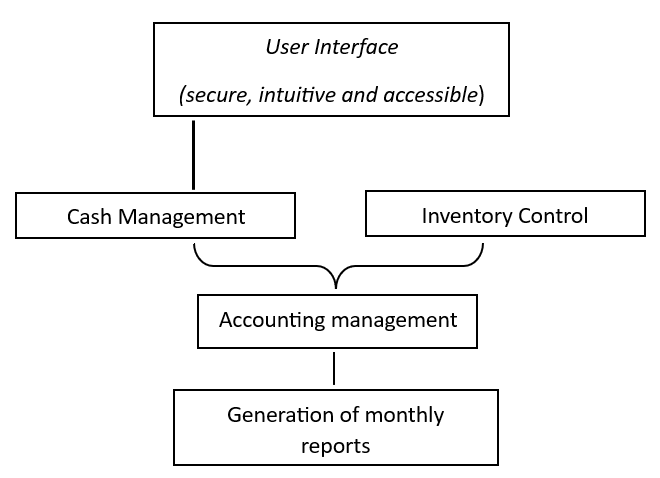
## Product Scope

*SureFinventory is an application designed to comprehensively manage the operations of a cafeteria, improving efficiency in inventory management, cash management and basic accounting control. Its main purpose is to optimize operational and administrative processes, allowing staff to focus on providing better customer service.*

# Overall Description

## Product Perspective

*SureFinventory is a new, stand-alone product developed specifically to meet the operational and administrative needs of a small to medium-sized coffee shop. Its design responds to the growing demand for digital solutions in the restaurant industry that facilitate daily management, inventory control, cash management, and tracking of accounts and reports. SureFinventory arose from the need of our customers who rely on manual methods or costly and complex solutions to manage their operations. The lack of tools that combine simplicity with functionality drove the development of this application.*

**

## Product Functions

1. *User Management*

* *Allow secure login with username and password.*
* *Configuration of user permissions according to specific roles (administrator, employee).*

1. *Application Configuration*

* *Allow a settings option to customize the menu’s products and its prices.*

1. *Inventory Management*

* *Easily record and update product inventory.*
* *Allow the manager to modify the products in the inventory.*

1. *Accounting Management*

* *Generate transaction and cash flow reports (incomes and expenses) daily or monthly to provide an overview of the cafeteria's finances.*

1. *Menu Management*

*Allow the manager to change the menu product’s.*

*Allow the manager to modify the products on the menu.*

1. *Order Management*

*Allow the manager and the cashier to take orders.*

*Generate invoices.*

*Charge the customer for the products bought.*

## User Classes and Characteristics

**Manager:** used to administer the system, modify inventory, create orders, modify restaurant menu.

**Menu:** gets all the products in a list, it’s used by the manager and cashier.

**Cashier:** use the menu to take orders and collect money.

## Operating Environment

*SureFinventory is designed to run in a variety of operating environments, ensuring compatibility with various hardware and software platforms. It requires a device with a minimum Intel Core i3 processor, 4 GB of RAM and 500 MB of free disk space, and the application is compatible with Windows 10 or higher, macOS 10.15 or higher, and Ubuntu 18.04 or later versions of Linux operating systems and is accessible from mobile devices running iOS 12.0 or higher and Android 9.0 or higher. SureFinventory is based on a web platform and is compatible with the most common browsers, such as Google Chrome, Mozilla Firefox, Microsoft Edge and Safari on Apple devices.*

## Design and Implementation Constraints

*There are some factors that will shape and constrain the design and implementation of SureFinventory. These constraints include knowledges limitations, hardware limitations, and specific technical requirements, such as:*

* *The scalability of the software is limited by the knowledges that the development team has at the time of creating this system.*
* *The application must be designed to run on devices with a minimum storage specification and if it does not meet these specifications, the application will not be able to run and cannot be used by the customer.*

## User Documentation

* *User Manual: will be delivered with functionalities about the program and how to use them. This manual has the purpose of teaching users on their way to use our software.*
* *Programmer Manual: aimed at the team and other developers that may join the project on future versions.*

## Assumptions and Dependencies

* *The aim of the project may change, these changes rely on the knowledge learned by the developers in the OOP (Object Oriented Programming) class. We assume that the requirements in this document are aligned with our abilities, yet it is possible that we find advanced programming related problems we might not be able to solve.*

# External Interface Requirements

## User Interfaces

*The interface between the user and the application is going to be in graphic mode, these means that all the features that are showed in these document will be coded in a VMC framework.*

## Hardware Interfaces

*The only interface is between the software and the device where it runs.*

## Software Interfaces

Data will be saved in json files that are managed with Gson library, these files are located in local databases and progressively uploaded to a remote database.

* + **Communication Interfaces**

*A communicative tool with the user's email will be needed and implemented.*

# 

# System Features

# Login

### 4.1.1 Description and Priority

An interface to validate users and passwords, it has a high-level priority in order that is the main security characteristic of the program.

### 4.1.2 Stimulus/Response Sequences

### Manager or cashier insert their respective username.

### System searches the username in a json file, it extracts the passwords and saves it.

### Manager or cashier insert their password.

### System compare passwords and decide if the user can continue in the program.

### Functional Requirements

REQ-1: The system must display a login dashboard with three options: log in as manager, log in as cashier, and close the application.

REQ-2: The user must be able to select one of the options from the login dashboard.

REQ-3: If the user selects "log in as manager" or "log in as cashier," the system must prompt for a username and password.

REQ-4: The user must be able to enter the requested username and password.

REQ-5: The system must validate the provided credentials against a JSON file containing the information of authorized users.REQ-6: If it is correct the system allows the user to continue using the app and open the next dashboard depending if the user is a manager or a cashier.

REQ-6: If the credentials are correct, the system must allow the user to proceed to the next interface, depending on whether they are a manager or a cashier.

REQ-7: If the credentials are incorrect, the system must display an error message and return to the login dashboard.

# Create Cashier User

### Description and Priority

Allows the manager to create a cashier with a username and a password, this feature has a low-level priority because probably it will not be used continuously, and has a low dependency with other features.

### Stimulus/Response Sequences

### -Manager inserts a new username and password.

### -System validates if the username already exists, if it does shows a message advising that the username already exists and ask for a new username and password or suggest to leave.

### -If it doesn´t exist, the system instantiates a new cashier object and pass these parameters.

### -The system transforms this information and saves it in the credentials Json file.

### -System validates the saving process and shows and successful or failure message.

### Functional Requirements

REQ-1: Display a message asking for username.

REQ-2: Scan the input in a String variable.

REQ-3: Validate if the username already exists in the passwords json file using the Manage Json Files feature

REQ-4: If the username already exists throw a message advising that the cashier already exist, print a message asking if manager wants to repeat or leave. If manager wants to repeat, start again from REQ-1, otherwise return to the manager menu.

REQ-5: If the username does not exist, ask for a password.

REQ-6: Scan the password and save it in a String variable.

REQ-7: Create a new cashier object.

REQ-8: Instantiate the object with the new username and password.

REQ-9: Transform the object into a json format object with the Manage Json Files feature.

REQ-10: Save the json object into the passwords json file.

# Create Accounting Report

### Description and Priority

Used to create a report about the accounting of the restaurant daily or monthly. It has a high-level priority, because it depends on some other classes and is used just by the manager.

### Stimulus/Response Sequences

### -Manager inserts the day or month of the report expected.

### -System looks for incomes and expenses in the period established and recollects all the needed data.

### -System shows a report that contains the balance that equals the incomes minus the expenses and then shows a total.

### Functional Requirements

### REQ-1: Print a message asking if the user wants a balance report.

REQ-2: Scan one of the previous options.

REQ-1: Print a message asking if the user wants a daily or monthly report.

REQ-2: Scan one of the previous options.

REQ-3: Open the json that manages incomes and expenses and filter the data by date.

REQ-4: Transform the json into objects.

REQ-5: Extract the objects that validate the date proportioned by the user.

REQ-6: Print the balance required.

# 4.5 Manage Json Files

### Description and Priority

Management and store of Information inside Json files is a high level priority in order to take account of the data.

### Stimulus/Response Sequences

### Manager or cashier insert data about inventory, sales, invoices.

### The new data is updated, created, deleted or searched in a json file.

### Functional Requirements

REQ-1: Enter a menu that guides trough the operation

REQ-2: Choose the option needeed

REQ-3: Perform CRUD operations working with the data provided

REQ-4: Implement an error handling and recovery strategy

REQ-5: Save all changes in the Json File

**4.6 Manage Menu**

### Description and Priority

It allows you to customize and manage the products offered on the cafeteria menu, including prices, descriptions and availability according to the season.

### Stimulus/Response Sequences

### Manager or cashier insert data about inventory, sales, invoices.

### The new data is updated, created, deleted or searched in a json file.

### Functional Requirements

REQ-1: Enter a menu that guides trough the operation.

**4.8. Take Order**

**4.8.1. Description and Priority**

- Allows the cashier or manager to register customer orders by selecting products from the menú.

**4.8.2.** **Stimulus/Response Sequences**

- The customer is going to buy at the coffeshop.

- The manager or cashier takes the customer's order

- System searches the customer's order in a json file

- System shows if the order is available

-System displays the payment methods

**4.8.3.** **Functional Requirements**

REQ-1: The customer will choose one of the orders available in the system.

REQ-2: The manager or cashier takes the order from the customer, of what he wants to buy

REQ-3: System searches the Json files to see if the requested order is available.

REQ-4: System will send a message if the order is available or not.

REQ-5: System will send to bring the order if the order is available.

REQ-6: In case it is not, the system will send for another order to the customer.

REQ-7: System will display the payment methods once the customer has been served his order.

**4.9. Cancel Order**

**4.9.1. Description and Priority**

- Allows the cashier or manager to cancel an order previously registered in the system.

**4.9.2.** **Stimulus/Response Sequences**

- The customer is going to buy at the coffeshop.

- The manager or cashier takes the customer's order

- A customer requests to cancel an order or the manager identifies the need to cancel it.

- The manager or cashier changes the specific order in the system.

- The system validates if the order can be canceled based on its current status

- If cancellation is possible, the system removes the order from the active list and updates the inventory.

- The system confirms the cancellation to the user and registers a new order.

**4.9.3.** **Functional Requirements**

REQ-1: the customer will choose one of the orders available in the system.

REQ-2: The manager or cashier takes the order from the customer, of what he wants to buy

REQ-3: The system searches the Json files to see if the requested order is available.

REQ-4: The system will send a message if the order is available or not.

REQ-5: The system will send to bring the order if the order is available.

REQ-6: In case it is not, the system will send for another order to the customer.

REQ-7: The system will display the payment methods once the customer has been served his order.

# 4.10 Invoice Generator

### 4.10.1 Description and Priority

- System that allows creating invoices with all the data required and a product list inside of it. High level priority.

### 4.10.2 Stimulus/Response Sequences - After an order is taken and charged it is possible to generate an invoice with all the specifications.

### - It saves the invoice in a Json file where we can accede to it searching inside the file.

### 4.10.3 Functional Requirements

REQ-1: Enter a order menu

REQ-2: Choose the option needeed

REQ-3: Generate the invoice according to the order

REQ-4: Get the object’s price with a bucle and sum them in a variable.

REQ-5: Print the name of each product and the price that is added in the sum variable.

REQ-6: Print the sum, with a message that explains that is the total incomes or expenses in the date.

REQ-7: Save the Invoice generated in a Json File

# 

# Other Nonfunctional Requirements

## Performance Requirements

* *The software must be useful for daily work;*
* *About the data performance, an application that can save several amounts of data is needed, so it's important to find an efficient way to save and pull data in a database located at the computer where the app will be running.*
* *Talking about the users traffic, we just need the app to perform with one user at a time.*
* *The focus in the performance requirement is essentially, the speed at managing data.*
* *Customize the interface according to user preferences.*
* *Change the application language between English and Spanish to suit users.*
* *Use turquoise and red as their brand colors at the app’s design.*

## Safety Requirements

## *If there is not an appropriate usage of this program the user could loose several important data, to avoid these the system must have backups for the data and versions to retrieve in case of any human failure when using the platform.*

## Security Requirements

*The application requires to have different types of users, including an admin that can control everything in the system. Therefore, the system must have authentications methods to assure the security of the restaurant.*

## Software Quality Attributes

*The software needs to be intuitive for the user. Usefulness is the main objective in this project so the quality is going to adapt to these terms.*

## Business Rules

*The business follows the accounting rules established by SRI in the regimen of RIMPE.*

# Other Requirements

*There may be an attempt to use a simple database for the user*

# Appendix A: Glossary

# Appendix B: Analysis Models

# Appendix C: To Be Determined List

*Second Interview*

***Good evening, Mrs. Mayra.****I’m here to ask you some questions regarding the previous interview we had. I need to fill out some more information. Now, I need a bit more specific information, as I mentioned in the message.  
I would like to know what processes you follow here and also, based on those processes, what requirements you would need the app to handle for you.  
To do that, I would need to understand the processes you have in place. For example, how do you manage the accounting for the day?  
How do you enter the daily orders? Let’s start there.*

***Good evening. As I explained last time, we do everything manually. We have sales notes and order notes.****So, in the order notes, we take all the customer orders, and that’s how we keep a sales record. Then, we use a notebook in which we manually record all the sales of the day, and we do a cash closing where we detail sales by card, cash, or transfers.  
And, well, we also record any expenses depending on the day, for example, purchases of products required by the cafeteria, but this can vary because, in general, we make purchases once a week.  
And this is also recorded manually.*

***Where do you make the purchases, Mrs. Mayra?****We use several methods. Some products are delivered to us, and payments are made via transfer. For other products, I go to the supermarket to buy them, and, well, I also record those purchases.  
And I also order online and have the products delivered here.*

***In this case, would you have the documentation, such as receipts, transfers, or card statements, for the products you purchase?****Yes, I have all the invoices, exactly.*

***Okay, perfect. I would need your help to provide this information so I can enter the expenses into the app.****It would really be independent of the payment method, but rather just the amount of the expense for the purchases made.  
So, apart from pantry items, what other expenses do you have?*

***Well, we have direct and indirect expenses. Direct expenses include products required by the cafeteria, employee salaries, and also the payments for basic services, like water and electricity.  
Is the location owned or rented?****No, it’s rented.****Okay, so that would be an additional cost too.****Yes, of course.****And do you also have costs related to renovations or repairs within the premises?****Yes, for example, we also have maintenance expenses for the cafeteria and the premises, and we also process documents, such as with the SRI (Tax Authority), which is annual.  
We also do renovations, and sometimes we sell products or items we have here, so we also need to replace that space.*

***Okay, but at the moment, do you not specify the expenses separately? You group them all together and then subtract them from the income? How do you report your final monthly income and expenses?****No, we detail our direct and indirect expenses. For example, maintenance and renovation costs are indirect expenses that the cafeteria incurs, and those are necessary.  
At the moment, are you keeping track of this in Excel or...?  
Everything is manual. Everything is done manually.  
Okay, I recall you mentioned an accounting book—could you send me the format you use?  
Yes, we keep a record of only the most relevant things, like the direct expenses the cafeteria needs.*

***Alright, I now have a clearer idea of what we would need for the app.****I’d also like to know, in terms of inventory, which products should be tracked?  
I understand you have a lot of products coming in and out, so what are the finished products that should be tracked?  
For example, how many cakes or coffee do you have? Could you tell me which products need to be inventoried to have more precision?*

***Yes, the dishes are specific, so we do an inventory of all the products and also track them with the purchases we make online.****For example, I buy packs of a dish we have on the menu, which is the "portavelos."  
I know that each pack has three portavelos, and I buy four packs a week, once a week, so I know how many will be sold.  
So, in that way, I keep track. On weekends, I also do a detailed record of the weekly sales to know which products I’ve sold.*

***Okay, you mentioned an Excel sheet before. Could you share that with me?****Yes, we have an Excel sheet where we track the products we have in the cafeteria. No problem.*

***Perfect, I think that’s basically the information I need.****We now know where you buy products, how payments are made, and how you’ve been managing things so far.  
So, what I understand is that you need a basic report, correct?  
I would also like to know, by the end of the month, do you know your net income or your profits, or have you had trouble identifying that benefit?*

***The truth is, it’s complicated for me. I’ve only managed it weekly, but I haven’t done a monthly report. The people who work here help me, but it’s very basic.***

***Now, let’s get into specifics. What would you need the app to do?****It’s mostly for inventory management and also for the orders. For example, when a customer comes in and orders a cappuccino, the staff sometimes says, "Let’s note it down," but for one order, they use a comanda. It’s quite informal, and it can’t be managed this way. The app would help visualize and track everything properly.*

***Would you need the app to have a username and password?****Yes.****And should there be profiles, one for the administrator and one for employees, or would it be just for you?****I think both would be necessary.  
Yes, that would be fine.*

***Perfect. We could also add features like inventory management, sales tracking, and updates.****As I understand, you make purchases weekly, so we could have a report before each purchase to show which products are running low. Would that be useful?****Yes, that way we can check which products are running out and which ones have been sold because we don’t keep track of that at the moment.***

***Okay, based on our previous conversation, I’ve compiled a list of what would be necessary to solve this.****The app should have an interface for entering daily orders, allowing you to close the cash register, showing the amount that should be in the cash drawer, the sales by card, and everything. This would make it easier to close the cash.  
It should also be connected to a database that can break down basic accounting, showing income and expenses, and generating a monthly report, which I believe you requested.****Yes.***

***Also, since I understand you are about to change the menu, it would be important to modify the product costs and the menu itself.****It wouldn’t be practical for me to make all the changes. It should be intuitive and easy for you to enter the settings and change the product prices if that works for you.****Yes. I would totally need to be able to change the product’s price and the menu’s products. We often change that.***

***Apart from all that, those would be the main functions of our program. I’d like to ask if there’s a specific color you’d like the app to have.****We use turquoise and red as our brand colors.****Perfect, turquoise and red it is.  
Would you prefer the app to be in any specific language?****Yes, it’s important for it to be available in both English and Spanish.****Great, we can set up a language toggle between English and Spanish.****If possible, there will also be design customization options, but language switching seems very feasible.*

***Putting all that together, I think I now have a solid list of requirements, and I have the necessary information.****I just need the rest of the details that we agreed to follow up on, such as how you manage everything manually, so we can prepare a format for automating these processes.*

**Procedures**

