Software Design Specification

for

Inventory System

Version 1.0

Prepared by

|  |  |  |
| --- | --- | --- |
| Patricia Surf | 620100929 | Patsurf38@gmail.com |
| Javier Stewart | 620100960 | Javierlamarstewar68@gmail.com |
| Fay Mcintosh | 620113856 | Fay.mcintosh17@gmai. |
| Matthew Johnson | 620098591 | Mat.a.john.mj@gmail.com |
| Ronaldo Willie | 620099445 | Ronaldo\_kc@outlook.com |
|  |  |  |
|  |  |  |

|  |  |
| --- | --- |
| Course Instructor: | Dr Ricardo Anderson |
| Course: | COMP2140 – Introduction to Software Engineering |
| Studio Facilitator: | Dr Ricardo Anderson |
| Date: | 11/21/2020 |

## TABLE of CONTENTS

**1.0 Project Overview …………………………………………………………………. 1**

**2.0 Architectural Design …………………………………………………………… 1-4**

* **2.1 General Constraints**
* **2.2 Alternatives Considered**
* **2.3 System Architecture Diagram**
* 2.3.1 Architecture Description
* 2.3.2 Architecture Justification

**3.0 Architectural Decomposition …………………………………………………… 4-6**

* **3.1 Component Decomposition**
* **3.2 Structural Design**
* 3.2.1 Design Notes

1.0 Project Overview  
 Bobbie Bread and Dawn Egan are both located in Misourri, St.louis. The business has had difficulty in keeping stock of their inventory, resulting in a misrepresentation of their business’s performance. The software is a web-based application that will keep track of the client’s inventory, track sales and report trends in sales and profits. This will assist in giving the owner a more comprehensive and accurate representation of how well the business is performing. The intended users of the System will be the managers (Bobbie bread and Dawn Egan) and the Sales person (cashier).

# 2.0 Architectural Design

2.1 General Constraints

The business comprises of two clients where only one user needs to use the application at a time. Therefore, security login only needs one username and one password to allow access to the web site for the client's use and maintenance. The applications require a standard mobile web browser (e.g., Safari and chrome). Since the website is accessed on the browser there is no special memory requirement needed for the application. The web application will be designed only for mobile devices such as tablets and smart phones, so therefore on a desktop, it will have a windowed display. (Hence, the time constraints will be at most five seconds per image on each web page.) The database for the application will be updated by the client. Items and issues that will limit the options available to the developer is that the software may not be completely be compatible with the hardware it will be implemented for. The time of execution may vary from hardware to hardware. The software requires the developer to have an understanding of java script as such if the writer does not have a full understanding of the language it may take a longer time to design and implement the software itself.

The hardware and software environment will be a web-based application. This will allow the user to access the system on any platform, Windows, Linux etc. The system also will be able to be used on desktop or mobile devices, such as cell phones, laptops, tablets, etc. The system will be able to operate on a device running Pentium 4 processor higher and 256MB of RAM.

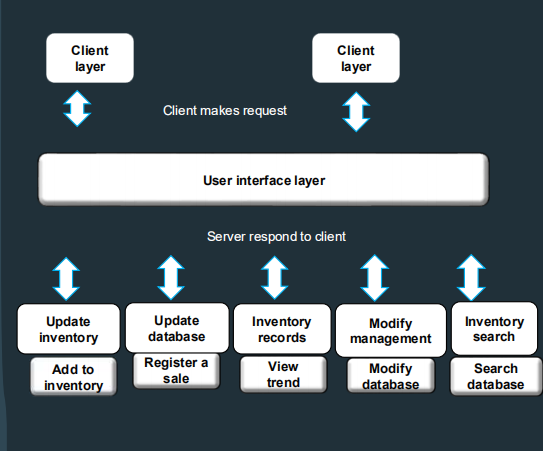
The software interfaces will rely on a database that will be implemented to record the Arch Cafe inventory. The database will be created using MySQL, the database will also need to do other functions outside of storing inventory, such as calculating the overall sales in a given time period, calculate profit and loss. The database will be compatible with an Android and Windows operating system. The capacitive screen will allow the user to interact with the varying parts of the software, which will include a Graphic User Interface (GUI), which will prompt the user to login to the platform.

2.2 Alternatives Considered

Two alternatives considered are:

1. The Model View Controller Architecture was one of the alternatives considered this is because the MVC consists of three modelled components: view, model and controller. The view component is responsible for the presentation of the data. The controller handles events caused by user interaction and facilitates the community of these events to the model and view components. The Rationale behind considering this Architecture is that the data can be presented in different representation known as “views” and changes in the data affect all representations. As it relates to the System this would allow for increase in the control of information, however there would have been a negative impact to as it relates to code and complexity.
2. The Layered Architecture was considered because it allows for independent modification of layers which affords a more flexible means of developing the software. The architecture is suitable for interactive systems; however, Security and dependability would have proposed a problem throughout the layers in the software.

## 2.3 System Architecture Diagram



2.3.1 Architectural Description

Client layer - refers to local interfaces used to author, model, analyze, present, report, and distribute diverse content.

User Interface layer- represents the front end of the Web Client, and contains the actual GUI elements that users view and click.

Update Inventory Server- allows the user to add to the inventory.

Update Database Server- allows the user to register sales whenever a sale is made.

Inventory Records Server- allows the user to access and view trends.

Modify Management Server- allows the user to modify the database and edit the data.

Inventory Search Server- allows the user to traverse the database and search for items in the database.

2.3.2 Architecture Justification

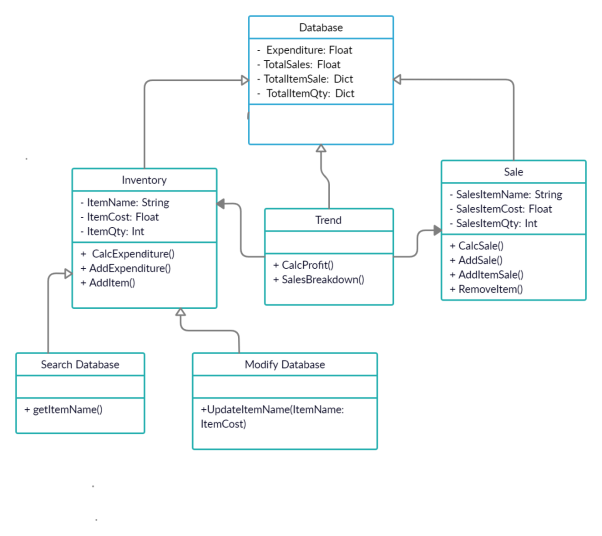
*The Layered Architecture was considered because it allows for independent modification of layers which affords a more flexible means of developing the software. Also the layers are not dependable on each other. This also suited our approach because this architecture is good for low count user usage (i.e. <100 users) and can be accessed to all users without being implemented by all services. The other architecture considered, the Model View Controller (MVC) architecture, was effectively ruled out because of its complexity. Each component were very dependent on each other; so if complications were in the controller, the view would be affected, and vice-versa. It would also require a great amount of coding to be done which was no necessary.*

# 3.0 Architectural Decomposition

### 3.1 Component Decomposition – Classes

|  |  |  |  |
| --- | --- | --- | --- |
| **Requirement ID** | **Architecture Component** | **Class Name** | **Description** |
| Req. 1  Req. 2  Req. 3  Req. 5  Req. 6 | Update inventory  Update database  Inventory records  Modify management  Inventory search | Inventory  Register  Trend  Modify database  Search database | This will allow the user to update the database on items added.  This will allow the user to update the database on sales.  Displays a percentage decomposition of the inventory.  This will enable the user make adjustments to specific items.  This will provide a minimal search field for enabling the user to search and view a specific item. |

## 3.2 Structural Design – Class Diagram



### 3.2.1 Design Notes

In the diagram, Aggregation and Composition are the main relationships shown between the classes. An aggregation is a special type of association in which objects are assembled or configured together to create a more complex object and describes a group and how you interact with them. In the class diagram represented in 3.2, multiple aggregation can be seen.

- ∆ - Represents Aggregation

- Search Database ∆ Inventory – The aggregation relationship between Search database and Inventory was chosen since Search database cannot exist on its own. The Search database would search the database and represent the information needed in the Inventory.

- Modify Database ∆ Inventory - The aggregation relationship between Modify database and Inventory was chosen since Modify database cannot exist on its own. With Modify Database, this would allow for item costs and items names to be added which would be done by the Inventory.

- Inventory ∆ Database – The aggregation relationship between Inventory and Database is caused by the fact that Database would tally all the information gained from the Inventory to produce a general total amount of each information received from the inventory.

- Trend ∆ Database – The aggregation relationship between Trend and Database is caused by the processes in trend is needed to calculate the information to be generated in the Database.

- Sale ∆ Database – The aggregation relationship between Sale and Database is due to the fact that sale would calculate the amount of products and the cost of the products which would link to the name of the product which would be represented in the database.

It is also seen that composition, which can be seen by the solid diamond is represented, which is a form of aggregation where components cannot exist in the aggregate.

- ♦ - Represents Composition

- Trend ♦ Inventory – The composition relationship between Trend and Inventory was selected as Composition because both classes are dependent of each other. Trend is dependent upon the Inventory to gain information to carry out the processes occurring in Trend which calculates profit and gives sale breakdown.

- Trend ♦ Sale - The composition relationship between Trend and Sale was selected as Composition because both classes are dependent of each other. Trend is dependent upon the Sale to gain information to carry out the processes occurring in Trend which calculates profit and gives sale breakdown.

Assumptions taking into consideration are the relationships between each class having either aggregation or composition when creating the diagram.