Software Requirements Specification

for

Restaurant Management System

Version 1.0

Prepared by Sanchit Singh (2K18/CO/317), Samvit Mukherjee (2K18/CO/315)

Delhi Technological University

Table Of Contents

Revision History :

1. Introduction:

1.1. Purpose :

The purpose of this project is to create a software that helps Restaurant owners log every receipt into a database that will help them keep track of their total income for the day along with helping the kitchen in preparation and keeping a track of the orders that they will be serving. At the end of the day the system can calculate logs and report daily earnings and the materials spent to the manager for review.

1.2. Document Conventions:

- Restaurant Manager: The person who is incharge of overall functioning of the Restaurant..
- Chef: An employee of the Restaurant incharge of Kitchen.
- Waiter: An employee of the Restaurant incharge of Serving Food.
- Cashier: An employee of the Restaurant incharge of handling Receipts and Bills.
- Customer/Client: Any person using any type of services offered by the Restaurant.

1.3. Scope:

The intended product automates the working of a Restaurant, thereby providing efficient services for employees of the Employees as well as the customers. The details of the Restaurant functions being automated are described in Section 3. After installation of the system the Restaurant would be able to make a Digital menu for the restaurant, help with preparation of order in the kitchen and efficiently deliver the orders to the respective customer.

1.4. Overview:

Section 2 of this document describes overview of the system in terms of General characteristics of the system, information about possible users of the system, possible constraints on the system, operating environment of the system etc. The Section 3 describes in detail functional as well as non-functional requirements of the system.

1.5. References:

- IEEE 830¬1998 standard for writing SRS document.
- Software engineering by KK Aggarwal
- Example- SRS document

2. Overall Description:

2.1. Product Perspective:

The system is Designed for showing and deciding the menu by the customer, showing the received order in the kitchen for preparation and logging of the Bills. The system will be connected to multiple terminals in a local network. The Below figure shows the use case diagram and use case description of the Software.

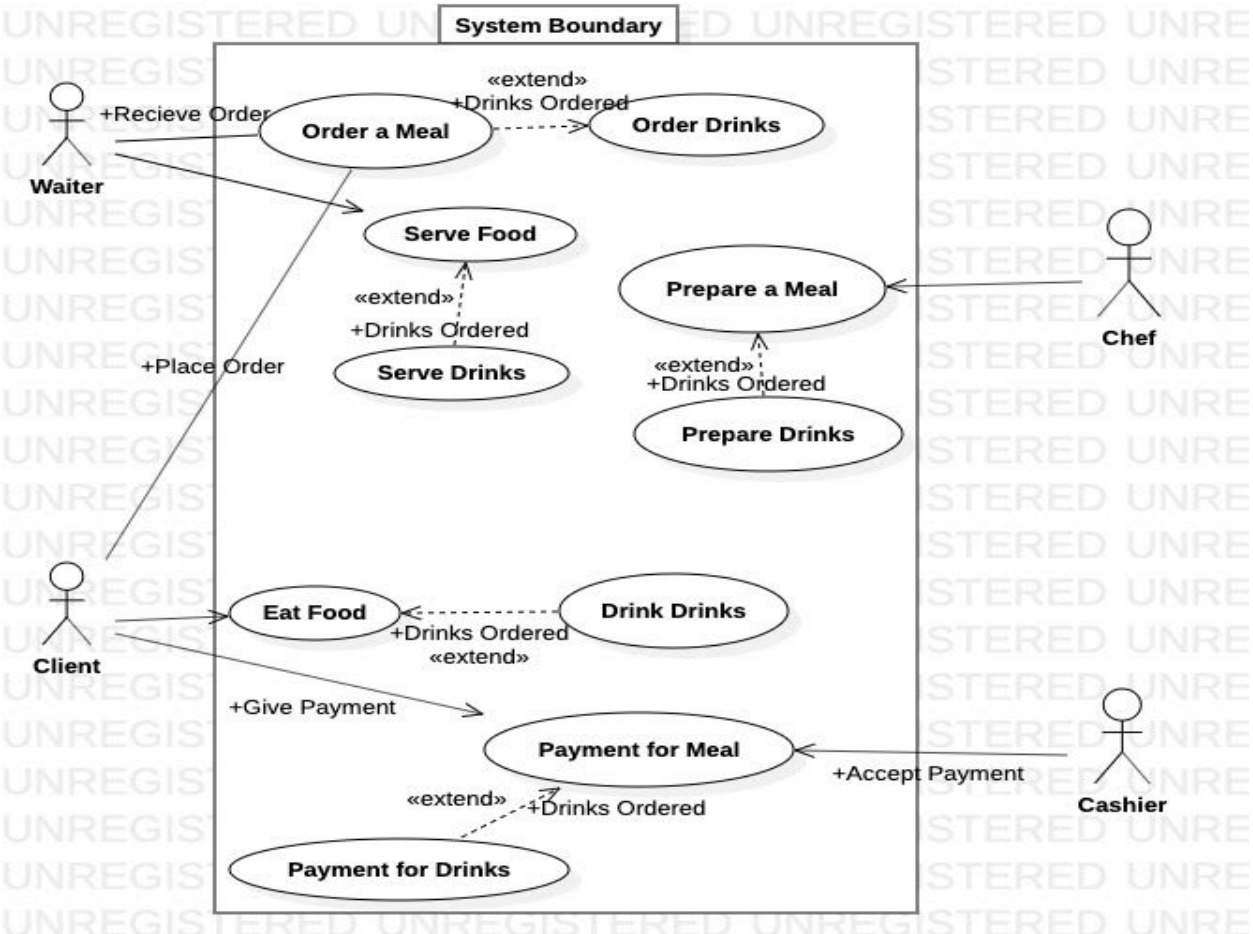


Fig 1. - Use Case Diagram of the Project

Use Case Description:

Use Case	
Use Case Name :	Order a meal.
Actors:	Customer, Waiter and Cashier
Description:	Customer looks at the menu, selects a food item/s and places an order for a meal. After the meal, pay according to the receipt.
Pre-Conditions:	● Customer intended to eat at this restaurant. ● Waiter is ready to take the customer's order.
Post-Conditions:	The Order is stored in the system with a status of delivered.
Normal Flow:	**1.0 Order a meal** 1. Customer is given the menu. 2. Customer selects one or more food items from the menu. 3. Customer indicates the meal order is complete. 4. System stores the order in the database and calculates the bill. 5. Customer is given the meal. **2.0 Payment for the meal** 1. After the customer eats, the entire bill is generated 2. The generated bill is handed over to the customer at Cashier or by the Waiter at the Table itself. 3. After Payment of the bill, a receipt is generated and handed over to the customer.
Alternative Flow:	1.1 Order Another meal. 1. Customer asks for another meal. 2. Return to Step 1 of **1.0**

Use Case	
Use Case Name :	Prepare a meal.
Actors:	Waiter and Chef
Description:	Waiter creates the order in his tablet, the system shows it to the Chef who prepares the meal. The order once given

	to the customer is added into the database to generate a bill for the Customer and store the receipt.
Pre-Conditions:	● Chef is ready to cook. ● Waiter has already taken the customer's order.
Post-Conditions:	A Bill is generated by the system
Normal Flow:	**1.0 Prepare a meal** 1. The Waiter takes the order from the customer and updates it in the system. 2. The Chef prepares the meal by looking at the order through the system. 3. The Waiter takes the meal and delivers it to the Customer. 4. The Waiter takes the Bill from the Cashier and gives it to the Customer.

2.2. Product Functions:

The RMS Supports the Following Functions:

- Functions by which the customer can view the menu and decide on an order.
- Functions by which the Waiter can add the order.
- Function by which the Kitchen Staff can view Orders to prepare.
- Function by which the Cashier can generate Receipts.
- Function by which the Restaurant Manager can see Daily Reports.
- Function to manage the Inventory of the restaurant.

The access to these different functions is restricted. E.g only the restaurant manager can view the Daily Reports.

2.3. User Characteristics:

Users of this System are Restaurant Staff and the customer. The customer doesn't need much knowledge since they will just be interacting with the menu but the Restaurant staff needs some in house training on how to operate the software.

2.4. User Constraints:

Once the order has been given, no editing can be done and any more orders taken after Receipt has been generated will be treated as new orders.

3. External Interface and Requirements:

3.1. Operating System Requirements:

All hardware must support Windows XP operating systems. It must also incorporate existing client server software in the product.

3.2. Software Requirements:

We will need an Software that can help in printing Receipts.

3.3. Hardware Interfaces:

We will need multiple tablets that run Windows Operating System as portable terminals as well as at least one terminal which will be responsible for printing the Bills.

4. Specific Requirements:

4.1. Functional Requirements:

4.1.1. Customer Functions:

4.1.1.1. View Menu

4.1.1.2. View Combo and Deals

4.1.2. Waiter Functions:

4.1.2.1. Edit the Order.

4.1.2.2. Check order Status.

4.1.3. Chef Functions:

4.1.3.1. Check Order.

4.1.3.2. Confirm Order is ready for Serving.

4.1.4. Cashier Functions:

4.1.4.1. Check Order

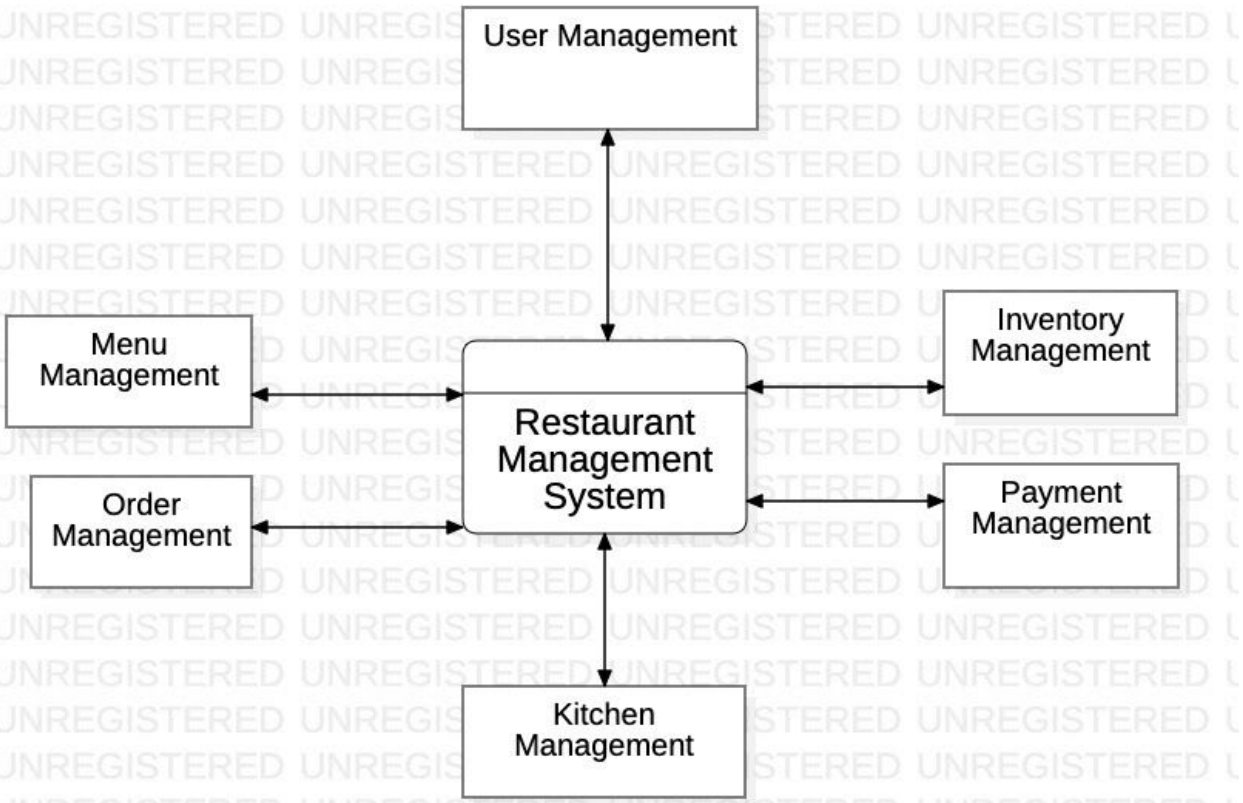
4.1.4.2. Generate Receipt

4.1.5. Restaurant Manager Functions:

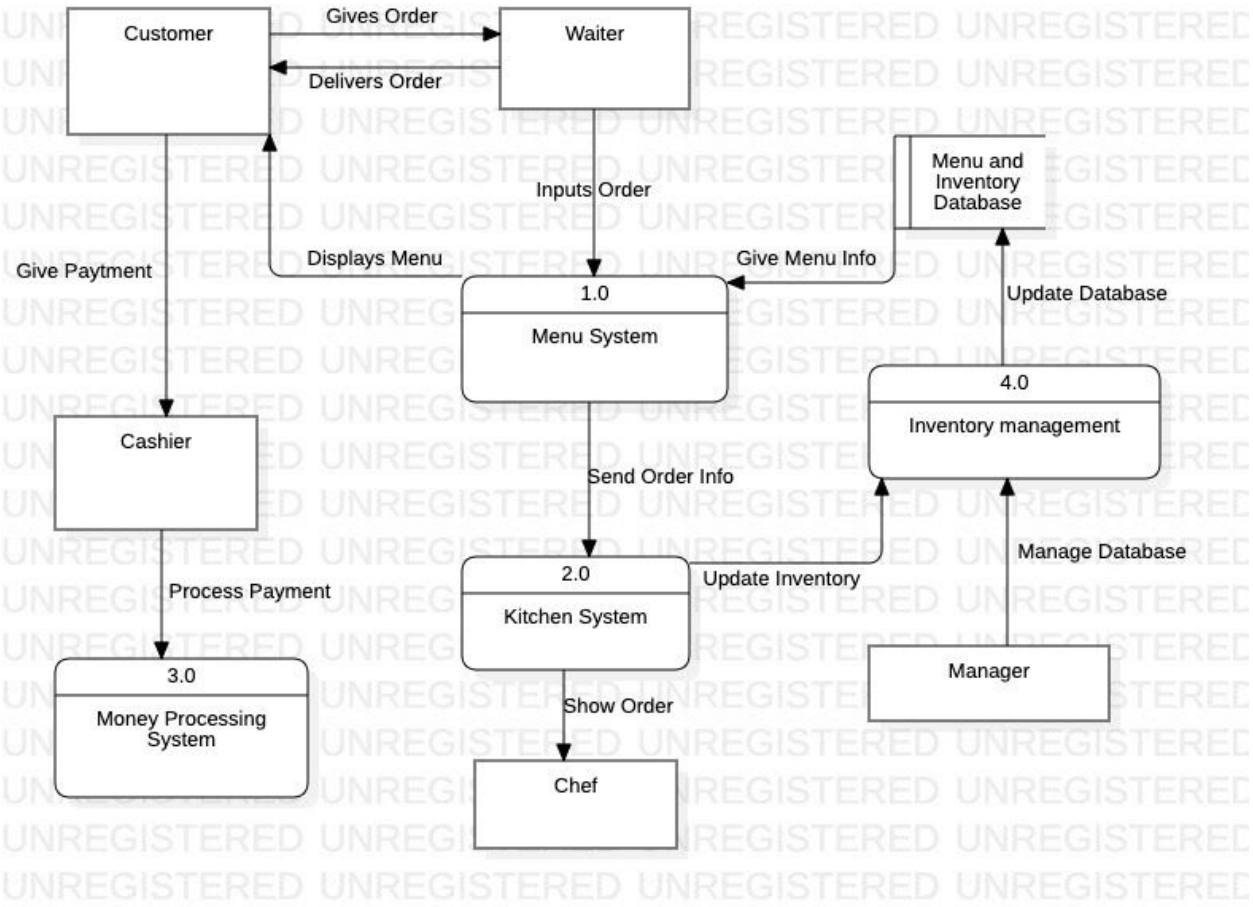
4.1.5.1. Check Daily Reports

4.1.5.2. Check Inventory

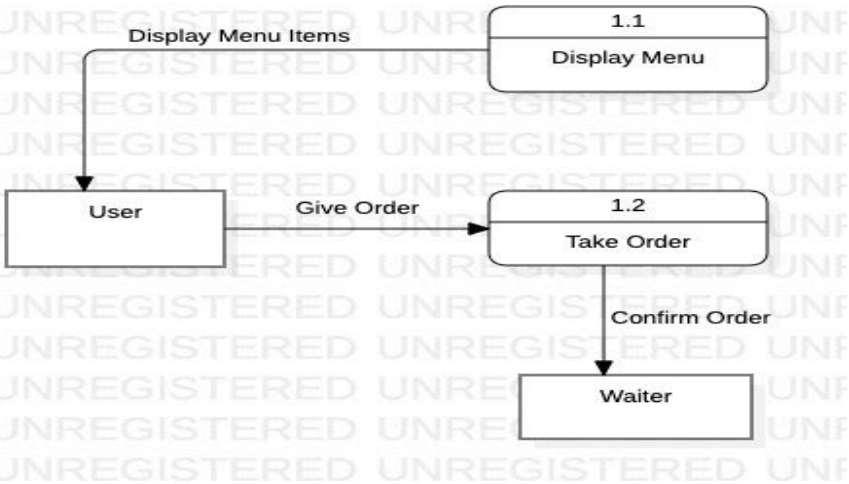
4.1.5.3. Update Menu, Combo or Deals.



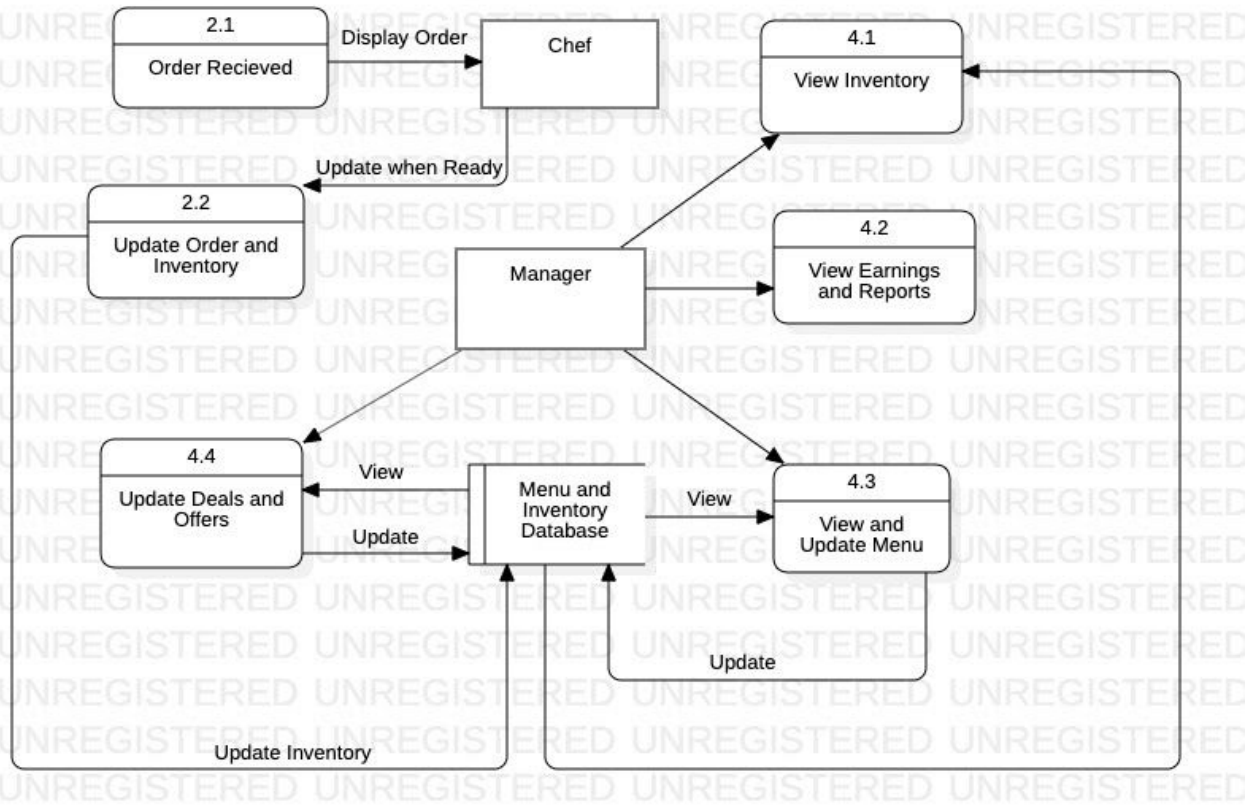
Level 0 DFD of Restaurant Management System



Level 1 DFD Diagram of Restaurant Management System



Level 2 DFD of Menu Management of Restaurant Management System



Level 2 DFD of Kitchen Management and Inventory Management in Restaurant Management System

4.2. Performance Requirements:

It will support all devices it is installed upon, the updates from database might get slower and laggy with every increase in number of devices initially about 8 devices are easily supported.

4.3. Security Requirements:

A login system will be present for Restaurant Manager and the Employees for added security.

5. Process Model:

5.1. Spiral Model:

This model is a combination of Waterfall and Iterative model where each phase begins with a Design goal and Ends with client reviewing. So our software is

developed in a series of incremental releases. It will be worked upon in multiple stages:

- The Communication stage where the requirements are discussed. The next phase will be planning where estimation of the project, scheduling the project and risk analysis while developing the project is done.
- The Modeling phase where analysis and design of the software is done.
- The Construction phase coding followed by testing is done.
- The Deployment phase of the project is delivered along with support. Then the feedback is taken from the customer.

This is how our first task region is completed. So after the first task region concept development takes place followed by software development in the next task region. Then system in enhancement and system maintenance takes place in the next tasks region accordingly.

Spiral model is useful because it's a large project, the releases are required to be frequent , risk and cost evaluation is there, the requirements are unclear and complex and the images might be required any time. So additional functionality or changes can be done at a later stage. Cost estimation becomes easy and development is fast. Also the features are added in a systematic way. Most importantly there is always space for customer feedback.