Analysis on

RESTAURANT MANAGEMENT SYSTEM

(ONLINE FOOD ORDERING SYSTEM)

AMAN GARU

00162791

COMPUTING PROJECT

Level 5 Diploma in Computing

Softwarica College of IT & E-Commerce

Kathmandu, Nepal

23rd, April 2017

Contents

[2. Analysis Specification 4](#_Toc480732269)

[2.1 Requirements 4](#_Toc480732270)

[2.1.1 Functional Requirements 5](#_Toc480732271)

[2.1.2 Non-functional Requirements 7](#_Toc480732272)

[2.1.3 Prioritization 10](#_Toc480732273)

[2.2 Use Case 11](#_Toc480732274)

[2.3 Architecture 15](#_Toc480732275)

[2.3.1 System Architecture 15](#_Toc480732276)

[2.3.2 Initial Class Diagram 15](#_Toc480732277)

[2.4 Conclusion 17](#_Toc480732278)

Table of Figures

[Figure 1: Rich Picture 4](#_Toc480732299)

[Figure 2: Use Case Diagram for Customer 12](#_Toc480732300)

[Figure 3: Use Case Diagram for Manager 14](#_Toc480732301)

[Figure 4: Inital Class Diagram 16](#_Toc480732302)

# 2. Analysis Specification

Before making a system, it is important to analyze the system, its boundary and its requirement. Requirement analysis is the most important step during the system development. It is the stage, which require least technical aspects. However, it requires more social, communication and managerial skill. Here the user is focused more. Analysis help to track out the problems as well. Analysis can be performed by breaking the system into different pieces.  
  
First brainstorming were performed for analysis purpose. Then after market feasibility study were conducted. To collect the requirements, interview were conducted among the members of restaurant. Planning were performed for further analysis. Rich Picture is also made during this process, which is shown below.

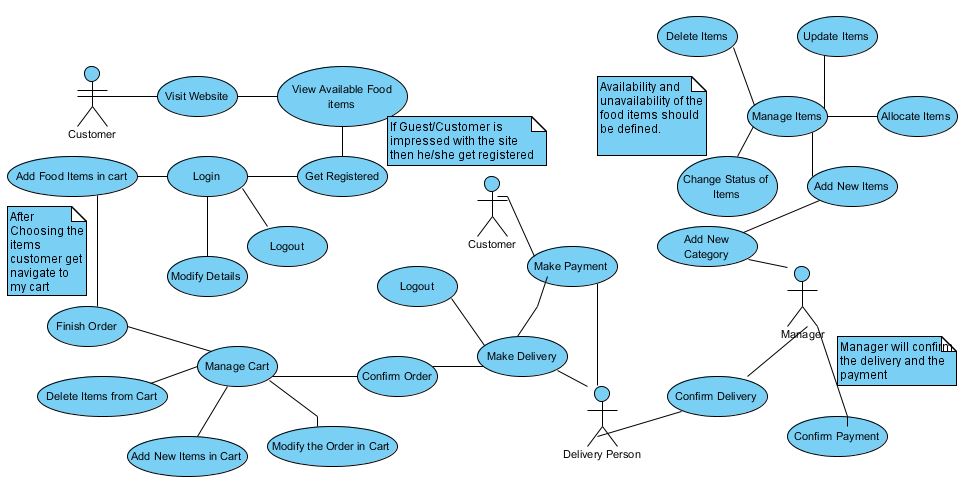


Figure 1: Rich Picture

## 2.1 Requirements

We can use some traditional methods like interview, observation, study of system and documents to gain the requirements. These methods are simple and cost effective but they are only effective if the objective is clear. Nowadays modern methods are used such as Prototyping, where the preliminary model is developed. In addition, Joint Application Design is popular where the requirement is discussed in the group of project team, user and managers. These methods give the clear vision but they are relatively expensive.

Similarly requirement for this system are gathered via interview session and via reading the existing documents.

### 2.1.1 Functional Requirements

Functional requirements are those requirements that describe what the system do. It defines a system and its component. It may include technical details, calculation, authentication, business rules, data control, processing and many other functions. It should describe clearly about the system service in brief. This are the main things that the user are waiting in the system while the nonfunctional requirement support this functional requirement. This requirement define how software behaves to meet user needs. Generally, functional requirements create a framework that allows the clients to influence and control cost. Moreover, it create a concrete link between what is wanted and what will be created.

Some of the functional requirement of the system are shown below:

**ID**: R1

**Title**: User Registration

**Description**: Customer/user of the system should register through the web portal. User must provide his/her name along with the contact address and contact number.

**Rational**: To capture user information for login process.

**ID**: R2

**Title**: Login

**Description**: Customer/user should be able to login using the username and password provided. Appropriate message should be displayed if something went wrong.

**Rational**: To enforce security in system.

**ID**: R3

**Title**: Display Menu

**Description**: The system shall display all the items available with the proper price sheet. Here the customer must be able to choose the item that need to be order.

**Rational**: To see the available items in the Restaurant and added them to.

**ID**: R4

**Title**: Add to Cart Facility

**Description**: System should provide the add to cart facility. Here all the items selected by customer is displayed.

**Rational**: Customer can check if they have mistakenly order any items and if so then the customer can deselect the order.

**ID**: R5

**Title**: Manage Cart

**Description**: Facility for deleting items from cart, updating the items from cart, adding new items in cart should be provided.

**Rational**: Customer can modify the order if they have mistakenly order items and update the previous order as well.

**ID**: R6

**Title**: Confirm Order

**Description**: Confirm Order Button should be placed after checking food cart and if customers are ready to order their food items.

**Rational**: No body is perfect. Some fault may occur at any time. Likewise, at the time of checking cart, order may give within a single click. Hence, confirm order facility should be placed to remove the unwanted order and to provide more security.

**ID**: R7

**Title**: Automatic Bill Generation

**Description**: The system should generate automatic bill after ordering food items along with vat and delivery charge.

**Rational**: To provide the automatic service in the system. It makes customer easy to generate invoice as well. In addition, customer can order the item according to their payment capacity.

**ID**: R8

**Title**: Confirm Delivery of food items

**Description**: Admin panel should confirmation the delivery of goods to the particular person.

**Rational**: It proves that the food items is delivered to real customer, which can be confirmed through the cash payment.

**ID**: R9

**Title**: Confirm Payment

**Description**: Along with ordering of food items, customer should be informed to have cash payment to delivery person and the confirmation should be done by admin panel after the delivery person.

**Rational**: It makes sure that there will not be any credit in transaction and it make sure that the goods are not delivered to unknown one.

**ID**: R10

**Title**: View Report

**Description**: Amount of food items delivered and the per day income should be generated in a single click.

**Rational**: It helps to analyze the ordering of food items at particular date.

### 2.1.2 Non-functional Requirements

Beyond the features and functions of the system, there are some extra requirement that add the charm in the system called Non-Functional Requirement. Non-functional requirement are those requirement that are used to judge the effectiveness of the system and it is called as system qualities. Functional requirement describe what the system do while Non-functional requirement describe how the system works. It determine technical specification of the product that will be delivered. The non-functional requirement are:

**Usability**: It deals in how difficult it would be to learn and operate the system. For e.g. if the website is difficult to use user immediately leaves the site. Hence, the system should be kept satisfied for users, it should be designed attracting the interest of user and it should focused on how to enjoy the user. It may leads in quickness in learning and easiness in remembering as well.

**Reliability**: User have to trust on the system. However, the software may often fails. The measurement is measured in term of MTBF (Mean Time between Failures). Less the failure, more will be reliability. Hence, to allow a user to reliable on a system, the system should be made without errors. Downtime should not exceed more than 1% of total operating time. Review must be taken at least once a month.

**Performance**: It includes the resource required, response time, benchmarking etc. Retrieving of data should be fast. If the system is passive then the user loses their interest in the system. Hence, the overall performance of the system should be enhanced.

**Portability**: It defines the effort required for the software to change its platform. If possible, the software should be made in such a way that it would run in every platform.

**Security**: The most important things in our system is data. Both the system and data should be protect securely. Unauthorized person should be strictly banned. Cryptographic techniques should be used. Moreover, firewall and antimalware software’s can be used to enhance security.

**Maintainability**: This should specify attributes of software that relate to the ease of maintenance of the software itself. There may be some requirement for certain modularity, interfaces, complexity, etc. Requirements should not be placed here just because they are thought to be good design practices. Maintenance of the system should be conducted at least once a month.

**Adaptability**: System should be flexible enough to withstand any future change that might be taken in the later phase.

**Compatibility**: The site should be compatible with the most used browsers i.e. Chrome, Mozilla Firefox, Internet Explorer.

**Availability**: In today’s world, a system should work 24/7. Data can be required during any time at any places. Even though the system fails, data should be available in the fraction of second. For this appropriate method, called daily backups or cloud backups should be used. If the user cannot access the data, at the time of need user may show their anger and they might not visit our system repeatedly. Hence, the data should be available any time, mostly at the site working time.

**Backup**: Backup server should be placed to prevent the loss of data when the main server or database are down. Downtime should not last more then 30sec when switching from main server to the backup server in case of a breakdown.

Examples of non-functional requirements are given below:

**ID**: Q1

**Title**: Usability

**Description**: Customers/Visitors should fell easy to use the site.

**Rational**: To make user use the site without confusion and to attract the user.

**ID**: Q2

**Title**: Reliability

**Description**: Admins, Managers, Customers should rely/trust on system.

**Rational**: To make users use the system. If the user do not trust on the system then there is ta high chance of failure of system.

**ID**: Q3

**Title**: Performance

**Description**: Customers/Users should be able to book the items or retrieve the data within a short time.

**Rational**: To access the data in a quick time.

**ID**: Q4

**Title**: Portability

**Description**: System should run in every platform.

**Rational**: To provide extra facility to client and to increase the customer.

**ID**: Q5

**Title**: Security

**Description**: Website should be secured.

**Rational**: To make the private information secure and to make hack free.

**ID**: Q6

**Title**: Maintainability

**Description**: Developer should performed maintenance check at least once a month.

**Rational**: To eradicate any problems or error arose in the system.

**ID**: Q7

**Title**: Adaptability

**Description**: System should be adjustable to maintain future change.

**Rational**: To withstand future change in the system.

**ID**: Q8

**Title**: Compatibility

**Description**: System should be compatible with all the browsers.

**Rational**: To provide the facility to the customer and attract more customers.

**ID**: Q9

**Title**: Availability

**Description**: Website should be available 24/7 during the running time. i.e. between (6 am to 9 pm).

**Rational**: To access the website at any time during active hours.

**ID**: Q10

**Title**: Backup

**Description**: Regular backup should be done.

**Rational**: To recover the data easily if the system crash.

### 2.1.3 Prioritization

Numbers of requirement are listed during the development of the system. Some of the requirement may not be covered during the development stages. For this, the system called MoSCoW rule is followed which help in prioritization of the requirement. Requirements may be of high, medium and low. Using Moscow means to have a specific prioritization. MoSCoW is considered a simple way to sort features into priority order. The word MoSCoW defines the following term.

**Mo (Must Have)**: It defines the most important requirement, which is compulsory in the system and if not fulfilled then the software will failed.

**S (Should Have)**: It defines the requirement that are important and that may not be fulfilled during the time of release.

**Co (Could Have)**: It defines the desirable requirement but not necessary as like others. It adds the extra functionality in the system and help in increasing user experience and customer satisfaction.

**W (Won’t have)**: It defines the requirement that are not appropriate at his time and are least critical.

The MoSCoW prioritization table is shown below:

|  |  |
| --- | --- |
| **Requirement** | **MoSCoW** **Rules** |
| R1: User Registration | Must Have |
| R2: Login | Must Have |
| R3: Display Menu | Must Have |
| R4: Add To Cart Facility | Should Have |
| R5: Manage Cart | Could Have |
| R6: Confirm Order | Could Have |
| R7: Automatic Bill Generation | Must Have |
| R8: Confirm Delivery of Food Items | Must Have |
| R9: Confirm Payment | Must Have |
| R10: View Report | Should Have |
| Q1: Usability | Should Have |
| Q2: Reliability | Should Have |
| Q3: Performance | Must Have |
| Q4: Portability | Could Have |
| Q5: Security | Must Have |
| Q6: Maintainability | Should Have |
| Q7: Adaptability | Could Have |
| Q8: Compatibility | Should Have |
| Q9: Availability | Must Have |
| Q10: Backup | Must Have |

## 2.2 Use Case

Use Case Diagram is a behavioral diagram used in analyzing various system. It helps in visualizing the role of actor with the system. It not only show the interaction between actors only but also shows the interaction between system and the actor.

It help us to identify the functions in the systems and the roles of the user with that particular function. Moreover, it helps in identifying, clarifying and organizing system requirement. It also helps in gaining external and internal factors influencing the system.

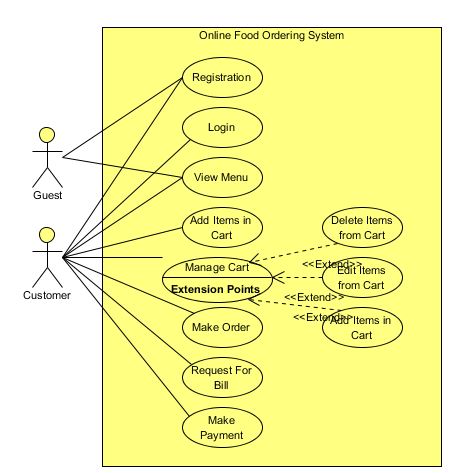


Figure 2: Use Case Diagram for Customer

Scenario Description

Scenario Description for Registration

* Customer can register to the system.
* Guest can also register to the system.
* Appropriate username and password along with the name of customer, address and mobile number should be given.
* Fake Contact should not be registered as the system is related with the delivery system.

Scenario Description for Login

* After the customer is registered, they can login to the system.
* Valid username and password should be provided.

Scenario Description for View Menu

* All the items available in the system (as filled by restaurant) should be available.
* Items should be grouped by the category available.
* Price of each items should be displayed.
* Name of items should be given clearly, which may leads in confusion.

Scenario Description for Add Items in Cart

* Customer select the items from the menu and it will be added to cart.
* After clicking order, the items will be automatically added to cart.
* It helps in future review.

Scenario Description for Manage Cart

* After clicking the done button in menu list, it will navigate to the cart page.
* Here the customer can review the items that he/she ordered.
* If the customer want more items then, he/she can add from here.
* It allows the customer to increase/decrease the number of food items quantity.
* In addition, it allows the customer to delete the items if they do not want.

Scenario Description for Make Order

* User can make order after reviewing the cart.
* User must click on make order button for ordering the listed item.
* In addition, user can approve the order with confirm order option.

Scenario Description for Request for Bill

* User can request a bill after making order.
* User need to click prepare bill button to generate bill.
* User can print the bill if they want.

Scenario Description for Make Payment

* User can make online/offline payment.
* For offline payment user must choose offline payment and they will pay the total amount to delivery person to make their payment.
* For online payment user must choose online payment option.

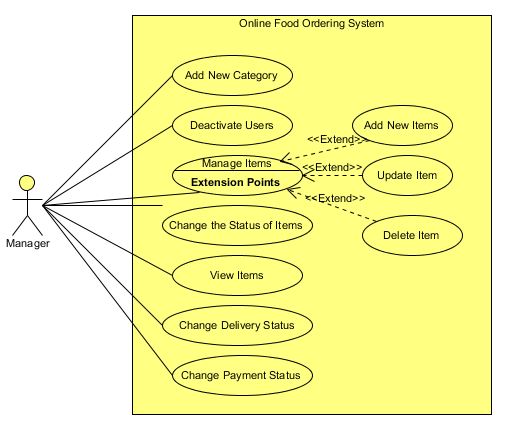


Figure 3: Use Case Diagram for Manager

Scenario Description

Scenario Description for Add New Category

* Manager can add the new category in the menu list.
* Manager must login through admin password.

Scenario Description for Deactivate User

* Manager can deactivate the fake account available in the system.
* Manager might deactivate the account if the customer misbehave with the system.

Scenario Description for Manage Items

* Manager can manage the items available in the menu list.
* Manager can add new items in the menu.
* If needed, the category for the items can be modified.
* Items can be deleted.
* Price of item can be modified.
* Items can be renamed in the needed situation.

Scenario Description for Change the Status of Items

* For unavailable products, manger can choose unavailable option.
* Customers cannot book unavailable products.

Scenario Description for View Items

* Manager are allow to view the items available in the system (restaurant).
* Items are grouped in category wise.

Scenario for Change Delivery Status

* Delivery status should be changed after the items is sent for delivery.
* Manager must change the option to Delivered after the delivery process.

Scenario for Change Payment Status

* Option for payment after being paid for the items should be changed.
* Manager should change the status to paid from unpaid after he/she get the money from delivery boy paid for him/her by customer.

## 2.3 Architecture

Architecture is defined as the structured representation of behaviors and structures of the system. It can be called as skeleton of the program, which shows the abstraction of a system. It may also impact on security, performance as well.

### 2.3.1 System Architecture

A system architecture is a conceptual model that characterizes the structure, behavior, and more views of a system. The motivation behind system architecture is to characterize a thorough arrangement in view of principles, ideas, and properties sensibly related and predictable with each other. The solution architecture has features, properties, and characteristics satisfying, as far as possible, the problem or opportunity expressed by a set of system requirements

Among the various type of architecture found, MVC architecture is the mostly used architecture. This system is made based on MVC i.e. Model View Controller Architecture. Model represent data structure usually by interfacing with database. View consists all the design pages and controller handles page requests and bind everything together.

### 2.3.2 Initial Class Diagram

Class Diagram is one of the static diagram that describe the structure of the system showing its class, attributes, its methods(Operations) and the relationship exists in them. Along visualizing, describing and documenting the system it also help in constructing code of application as well.

For making Class Diagram, the process called NLA (Natural Language Analysis) was performed. Here nouns and verbs were extracted from the scenario given and then listed in the tabular format as shown below.

|  |  |
| --- | --- |
| Classes(noun) | Methods(verb) |
| Customer, Admin, Restaurant,  Owner, Category, Staff, Cook,  Item, Guest, Manager, Order, Cart | Update, edit, order, view, manage,  delete, deactivate, generate, add,  change, confirm, register , monitor, |

After listing all the possible classes and methods filtration were done. Filtration include the process to remove the classes that are repeated, the classes that are synonym, and the classes that are beyond the scope. In addition, the repeating methods were removed and only those methods, which make sense within the class, are extracted. While some of the missing, classes and methods are added during the development of class diagram. After the refining process, again the noun and verb are allocated in the table that in shown below.

|  |  |
| --- | --- |
| Classes(noun) | Methods(verb) |
| Customer, Category, Item, Manager, Order, Cart | Update, add, delete,  Confirm, manage, view |

The initial class diagram is shown below.

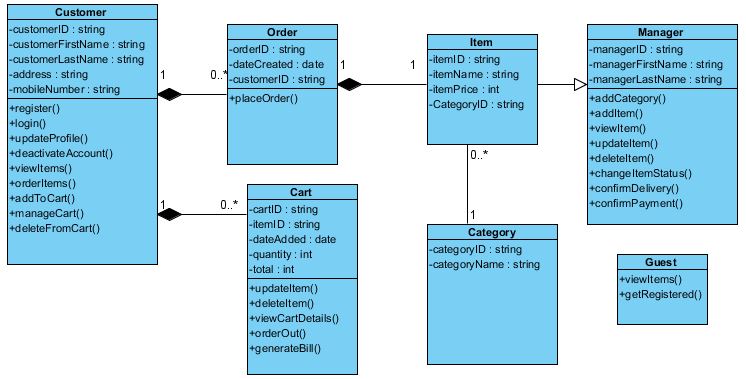


Figure 4: Initial Class Diagram

## 2.4 Conclusion

Finally, analysis was conducted successfully and the documentation were made. We have come cross rich picture, use case diagram with scenario, and initial class diagram. SRS (System Requirement Specification) which include the functional and non-functional requirement were made. First, we performed brainstorming, and then feasibility status of market were studied. After that interview were conducted with one of the member of the Restaurant. Similarly, appropriate architecture were choosen for the better development of system.