CSE3005 Software Engineering

Digital Restaurant

(Group 11)

Aashirwad Mishra 19BCE10008
Himanshi Tiwari 19BCE10013
Pushkal Singhal 19BCE10330
Pallavi Dhere 19BCE10345

Software Requirements Specification

Document

Table of Contents

1. Introduction

The following section provides an overview of the derived Software Requirements Specification (SRS) for the software Digital Restaurant. To begin with, the purpose of the document is presented and its intended audience outlined. Subsequently, the scope of the project specified by the document is given with a particular focus on what the resultant software will do and the relevant benefits associated with it. To conclude, a complete document overview is provided to facilitate increased reader comprehension and navigation.

1.1 Purpose

The purpose of this SRS is to outline both the functional and non-functional requirements of the software Digital Restaurant. In addition to said requirements, the document also provides a detailed profile of the external interfaces, performance considerations and design constraints imposed on the subsequent implementation. It is the intention that the presented set of requirements possesses the following qualities; correctness, unambiguousness, completeness, consistency, verifiability, modifiability and traceability. Consequently, the document should act as a foundation for efficient and well-managed project completion and further serve as an accurate reference in the future. The primary audience of this SRS document will be the development team employed to implement the specified RMOS. It will not only provide an extensive capacity for project planning and progress assessment but it will further assist with developer/stakeholder interactions. The secondary document audience comprises the stakeholders of the project, that is, restaurateurs and associated staff. To this audience group, this SRS should convey and confirm the required functionality and represent a contractual agreement between the involved parties.

1.2 Scope

In current formal dining environments, some form of physical static menu is utilised to convey the available food and beverage choices to customers. Said menus are generally paper based and hence impose restrictions on the textual real estate available and the ability a restaurateur has to update them. This document specifies the requirements for a restaurant paper menu and ordering replacement strategy to alleviate the problems associated with the current archaic method.. The first pertains to the replacement of paper-based menus using an electronic format, the second relates to a complementary electronic strategy for the front of house handling of a customer’s order and the third surrounds the process of transferring said electronic orders to the kitchen for preparation. It should be noted that while the suggested strategy incorporates the use of various hardware components, the primary focus of the presented SRS relates to the constituent software elements.

1.3 Definitions, Acronyms, and Abbreviations.

1.3.1 Definitions

- **Register -** If a customer wants to order the food, then he/she must be registered, unregistered users can't order.
- **Login** - The customer begins to use the system by entering a valid user id and password for ordering. Hotels can also login in the app to provide its services.
- **Home Delivery -** If a user wants the food to be delivered at the doorstep, then he/she can select this Home Delivery option in the beginning after logging in to the app.
- **Table Booking -** If a user is at a restaurant or cafe and wants to order food from the app, he/she can select the Table Booking option in the beginning after logging in to the app. The user just needs to scan the QR code on the table of the restaurant or cafe and it will show the menu and then he/she can select items from the digital menu to order.
- **Add to cart -** The items displayed in the menu will have an option to add to cart. Users can click on this option and add as many items as they want to the cart.
- **Place Order -** Once a user adds the items to the cart, he/she can then click on the Place Order button to confirm the order and the order will be placed by the user.
- **Payment -** Once a user places an order, he/she will get an option to pay the required amount. The payment will be authenticated by the payment gateway.
- **Hotel Registration** - If a hotel wants to provide its services on this app, they first need to register on the app by filling out the necessary details.
- **Hotel Login -** If a hotel has already registered on the app, then he/she can directly login to the app to make changes on its account.
- **Managing order/booking -** Once a user has placed an order, then the hotel will receive a notification of the summary of the order. After receiving the order summary, the hotel can start processing the order.
- **Assigning Delivery Boy -** Our app will assign the nearest delivery boy after the order has been processed by the hotel. If the order is of Table Booking type, then there is no need to assign the delivery boy. The hotel will manually assign a waiter to deliver the food or it can be self-service to pick up food.

1.3.2 Acronyms

- RAM - Random Access Memory
- RFC - Request For Comments
- DBMS - Database Management Systems
- IEEE - Institute of Electrical and Electronics Engineers
- UML - Unified Modeling Language

1.3.3 Abbreviations

- GUI - Graphical User Interface
- UI - User Interface
- DFD - Data Flow Diagram

1.4 References

- C. Larman, APPLYING UML AND PATTERNS An Introduction to Object-Oriented Analysis and Design and Iterative Development, 3rd edMassachusetts: Pearson Education, 2005.
- D. Carrington, CSSE3002 Course Notes, School of ITEE University of Queensland, 2008
- IEEE Recommended Practice for Software Requirements Specifications, IEEE Standard 830, 1998

1.5 Overview

The structure of this Software Requirements Specification is as follows. Section 2 presents an overall description of the software Digital Restaurant. Attention is given to putting the resultant software product into perspective and further outlining end-user characteristics, system constraints and assumptions. Section 3 is devoted to the explicit specification of software requirements both functional and non-functional in nature. The functional requirements listed have been demarcated according to the categories of system users. For completeness, Section 4 extends upon Section 3 through the inclusion of UML analysis models and diagrams. To begin with, the identified Digital Restaurant use cases are given. In addition, Data Flow Diagram, Sequence Diagram and class diagram are given.

2. The Overall Description

The following section presents an overall description of the software Digital Restaurant. In particular, the product has been put into perspective through a detailed assessment of the system, user, hardware, software and communication interfaces, memory considerations, operational modes and site adaptation requirements. Further, characteristics of the system's end-users are discussed along with the identified system constraints and assumptions. To conclude the section, an apportioning of requirements has been outlined.

2.1 Product Perspective

The software described in this SRS is the software for a complete Digital Restaurant system. The system merges various hardware and software elements and further interfaces with external systems. Thus, while the software covers the majority of the system's functionality, it relies on a number of external interfaces for persistence and unhandled tasks, as well as physically interfacing with humans.

The following subsections describe how the software operates inside various constraints.

2.1.1 System Interfaces

The Digital Restaurant interfaces with an existing payment system, in order to quickly and easily handle customer billing. The payment system should be operable such that it can return information to the Digital Restaurant system as to whether payment was successful or failed.

2.1.2 Interfaces

There are three separate user interfaces used by App, each related to an interfaced physical hardware device These three user interfaces are the

Customer Mobile UI
The Mobile UI is the interface used by restaurant customers. This interface uses the surface computer paradigm - users interact with the system by dragging 'objects' around on the flat-screen touch-sensitive display. For the App, users can manipulate objects such as items of food, dietary requirements, tips and menus on the surface of their table. Such objects can be moved into static objects such as meals and payments to perform various functions. In addition to this object manipulation paradigm, a limited system menu is necessary. Users will summon their restaurant menu, which is combined with a system/command menu, using an easy touch gesture, a double-tap on the touch surface, and dismiss it with a similar gesture or by tapping a close button GUI element.

Restaurant UI
The Restaurant UI is designed to be used by Restaurant to accommodate customer needs. This UI will be designed for use with a stylus input into the touch-screen. Because the number of operations the UI needs to support is relatively limited, there will be no nested menu structure. The UI shall provide simple graphical interfaces.

Delivery Boy UI
This UI provides Delivery boy with simple functionality related to ordered items. The UI will display the address of the customer or other contact details.

2.1.3 Hardware Interfaces

Specify the logical characteristics of each interface between the software product and the hardware components of the system. This includes configuration characteristics. It also covers such matters as what devices are to be supported, how they are to be supported and protocols. This is not a description of hardware requirements in the sense that “This program must run on a Mac with 64M of RAM”. This section is for detailing the actual hardware devices your application will interact with and control. For instance, if you are controlling X10 type home devices, what is the interface to those devices? Designers

should be able to look at this and know what hardware they need to worry about in the design. Many business type applications will have no hardware interfaces. If none, just state "The system has no hardware interface requirements" If you just delete sections that are not applicable, then readers do not know if: a. this does not apply or b. you forgot to include the section in the first place.

2.1.4 Software Interfaces

The Digital Restaurant will interface with a Database Management System (DBMS) that stores the information necessary for the App to operate. The DBMS must be able to provide, on request and with low latency, data concerning the restaurant's menu, employees (and their passwords) and available dietary requirements. Additionally, it should take and archive data provided to it by the App. This data will include records of all orders and transactions (system states and state changes) executed by the App. The DBMS must store all data such that it can be used for accounting, as well as accountability.

2.1.5 Communications Interfaces

Specify the various interfaces to communications such as local network protocols, etc. These are protocols you will need to directly interact with. If you happen to use web services transparently to your application then do not list it here. If you are using a custom protocol to communicate between systems, then document that protocol here so designers know what to design. If it is a standard protocol, you can reference an existing document or RFC.

2.1.6 Memory Constraints

Specify any applicable characteristics and limits on primary and secondary memory. Don't just make up something here. If all the customer's machines have only 128K of RAM, then your target design has to come in under 128K so there is an actual requirement. You could also cite market research here for shrink-wrap type applications "Focus groups have determined that our target market has between 256-512M of RAM, therefore the design footprint should not exceed 256M." If there are no memory constraints, so state.

2.1.7 Operations

The Application has two modes of operation. However, because of the restaurant environment it is used in, it must be able to operate for long periods, without error. The server must be able to operate unattended indefinitely. It should not need physical interaction except for upgrades and failure of hardware elements. Backup and recovery should be handled by the DBMS and operating system, or external software running on a timed backup system. Interaction from the App should not be required. Since stateful data should not be stored on any of the devices other than the server, keeping a system image

on the server for each device may be a sufficient operational method to facilitate restoration should a device become corrupted.

2.1.8 Site Adaptation Requirements

Site configuration for the Digital software is expected to encompass the following steps:

- Install the server, surface computers and displays
- Acquire sufficient tablets for all staff that need to use them
- Network all devices, install operating systems, server software and DBMS
- Secure network, distribute initial passkeys
- Install Digital Restaurant software
- Configure server software

2.2 Product Functions

Digital Restaurant is a Mobile Application built to manage the Hotel/Restaurant ordering system and home delivery services. The app will give an option for table booking and online delivery. When Customer Chooses the Option of Table Booking, after visiting, they will find a QR scanner in the Hotel and scan to see the menu from where they can directly order and pay. On choosing Home Delivery, they will Order the food which gets delivered to their home. Also, various hotels can Register themselves, add their menu, details and manage orders to give service.

2.3 User Characteristics

The end-users of the App fall into three primary categories, unskilled, partly skilled and highly skilled.

- **Unskilled user :-**
  The users of the mobile phone are walk-in customers and should therefore be assumed to have no relevant prior skills or education other than basic abilities to operate an App; no more complex than a parking meter or vending machine.
- **Partly skilled users :-**
  The users of the mobiles and displays are waiters and chefs respectively and they should be able to use the system and further be able to train others with minimal training themselves. They must be able to explain all elements of the user interfaces except the server. Supervisors also fall into the same category, though they will have to learn other sections of the system (refunds etc); these should not be of notably greater complexity than the standard functions. This class of user would be expected to have a junior high-school certificate education or equivalent.
- **Highly skilled user :-**
  The initial installation and configuration of hardware and the constituent App components (especially the server) is guaranteed to require someone with notable computer experience, including extensive experience with network and operating systems to complete it. The software should not be needlessly complex, but it is still expected not to be entirely 'plug and play'. This class of user is expected to

have a high-school certificate or equivalent, as well as extensive computer experience.

2.4 Constraints

The Application should be written in an object-oriented language with strong GUI links and a simple, accessible network API. The primary candidate tool chains are Java/Swing, C++/Qt and Python/Qt. The system must provide a capacity for parallel operation and system design should not introduce scalability issues with regard to the number of surface computers, tablets or displays connected at any one time. The end system should also allow for seamless recovery, without data loss, from individual device failure. There must be a strong audit chain with all system actions logged. While interfaces are covered in Section 2.1, it is worth noting that this system is likely to conform to what is available. With that in mind, the most adaptable and portable technologies should be used for the implementation. The system has criticality insofar as it is a live system. If the system is down, then customers must not notice, or notice that the system recovers quickly (seconds). The system must be reliable enough to run crash and glitch free more or less indefinitely, or facilitate error recovery strong enough such that glitches are never revealed to its end-users.

2.5 Assumptions and Dependencies

The SRS assumes that none of the constituent system components will be implemented as embedded applications. The implication is that the target hardware will provide a capacity for standalone program/application deployment and not require customised embedded firmware to be written. It is further assumed that a mobile of sufficient processing capability and battery life will be utilised. The mobile employed by the system should facilitate being utilised/left on for extended periods (sufficient for daily use) and that they are programmable in the same fashion as x86 architecture computers. Finally, it is further assumed that the deployment environment is capable of supporting an IEEE 802.11 wireless network for system communication.

2.6 Apportioning of Requirements.

This subsection pertains to both the functional and non-functional requirements omitted unintentionally from this SRS document.

3. Specific Requirements

The following section presents the complete set of functional and non-functional requirements identified for the software Digital Restaurant. Functional requirements are listed first, according to their relationship to the overall system, customers, waiters, chefs

and supervisors. The non-functional requirements that pertain to safety, security, the interface, human engineering, qualification, operation, maintenance and performance are subsequently presented. The functional requirements have been specified using a natural language description and as such, the reader is directed to Section 4 (UML Analysis Models) for further detail.

3.1 Functions

1. Registration:- If customer wants to order the food then he/she must be registered, unregistered user can't go for ordering
2. Login:- The customer begins to use the system by entering valid user id and password for ordering. Also hotels can login in the app and manage orders , add/Update menu.
3. Options :- customer can choose which mode he needs to use (Table booking or Home delivery).
4. Display the menu:- In the system all the items are displayed with their rates.
5. Modify menu :- System can make changes in the menu like adding or removing food items which are not available.
6. Select food items:- Items are selected, customers feel free to order.
7. Changes to order:- Changes to order means the customer can make changings in order like he/she can delete or add food items in order.
8. Review the order before submitting Before submitting the complete order is reviewed to the customer. Customer name, phone number, location (address) and placed order, then finally order is submitted.
9. Payment :-For customers there are many types of secure billing that will be prepaid as debit or credit card, postpaid as after delivering, check or bank draft.
10. Provide delivery and payment details:- Here bill is generated, order no. and payment is given and confirmation of delivery is done.

3.2 Performance Requirements

- The server shall be capable of supporting no less than 200 concurrent connections from any combination of mobiles, tablets and displays.

- The server shall be capable of supporting an arbitrary number of mobiles, tablets and displays, that is, it shall provide no limit on how many devices are in the system.

- The server shall be capable of supporting an arbitrary number of active meals/orders, that is, no meals/orders shall be lost under any circumstances.

- The server shall be capable of supporting an arbitrary number of active customer payments, that is, no payments shall be lost under any circumstances.

3.3 Software System Attributes

There are a number of attributes of software that can serve as requirements. It is important that required attributes are specified so that their achievement can be objectively verified. The following items provide a partial list of examples. These are also known as non-functional requirements or quality attributes.

These are characteristics the system must possess, but that pervade (or cross-cut) the design. These requirements have to be testable just like the functional requirements. It's easy to start philosophizing here, but keep it specific.

3.3.1 Reliability

The ability of the system to behave consistently in a user-acceptable manner when operating within the environment for which the system was intended.

3.3.2 Availability

The system should be available at all times, meaning the user can access it using a mobile app, only restricted by the down time of the server on which the system runs.

3.3.3 Security

Secure access of confidential data (customer information).

3.3.4 Maintainability

A commercial database is used for maintaining the database and the application server takes care of the site.

3.3.5 Portability

Systems running on one platform can easily be converted to run on another platform.

3.4 Organizing the Specific Requirements

This subsection extends upon the functional requirements given in Section 3.1 through the presentation of detailed use cases. To facilitate an unambiguous and clear view of how the end-users interact with the Application, the actors (end-users) involved in the use cases, a use case diagram and detailed use case descriptions are provided. The use cases that find

representation are Log In, Log Out, Place Order, Pay Bill, Accept/Reject Item and Indicate Item Ready etc.

3.4.1 Use case Diagram

Actors:

- **Customer :-** They can Register/Login for home Delivery or Table Booking , Access the menu, Add item to cart, Place order, Pay through payment gateways, Track Status of Order.
- **Hotel :-** They can Register, add/update menu, manage orders and delivery.
- **Payment Gateway :-** They Authenticate things for Payment.
- **App Administration :-** Keep user Details for further use, Track of Login/ customer and Hotel , Assigns Delivery Boy.
- **Delivery Boy :-** Updates food status and delivers it.

Use Case Description:

Use Case	Login/Register
Actor	Customer , App Administrator
Purpose	If user need to order or book table he needs to Register and if already registered then login
Extensions	Home Delivery Table Booking Add Details : it include addition of Details of customers

Use Case	Home Delivery
Actor	Customer , Hotel
Purpose	Users can choose items from the menu to order food at home. Hotel will take the order and deliver it through delivery boy
Extensions	Menu - Shows items

Use Case	Table Booking
Actor	Customer , Hotel
Purpose	Users can book the table and after visiting they can scan the QR code from where they can access the menu and order it. Hotel need to take the order and serve it
Extensions	QR scanner - Shows the menu Menu Visible - shows item that can be ordered

Use Case	Add to cart
Actor	Customer
Purpose	Select the items from menu which they want to order
Extensions	Edit Item Quantity - Can add or sub items added to cart Give instructions - Can give instruction about food Place Order - Customer can place order

Use Case	Place Order
Actor	Customer
Purpose	Places the order
Extensions	Track Status - Updates by delivery boy and hotel , Customer can track their order Payment - adding items to card shows bill payment Add New Address - If want to change the location of delivery from location added while registration.

Use Case	Payment

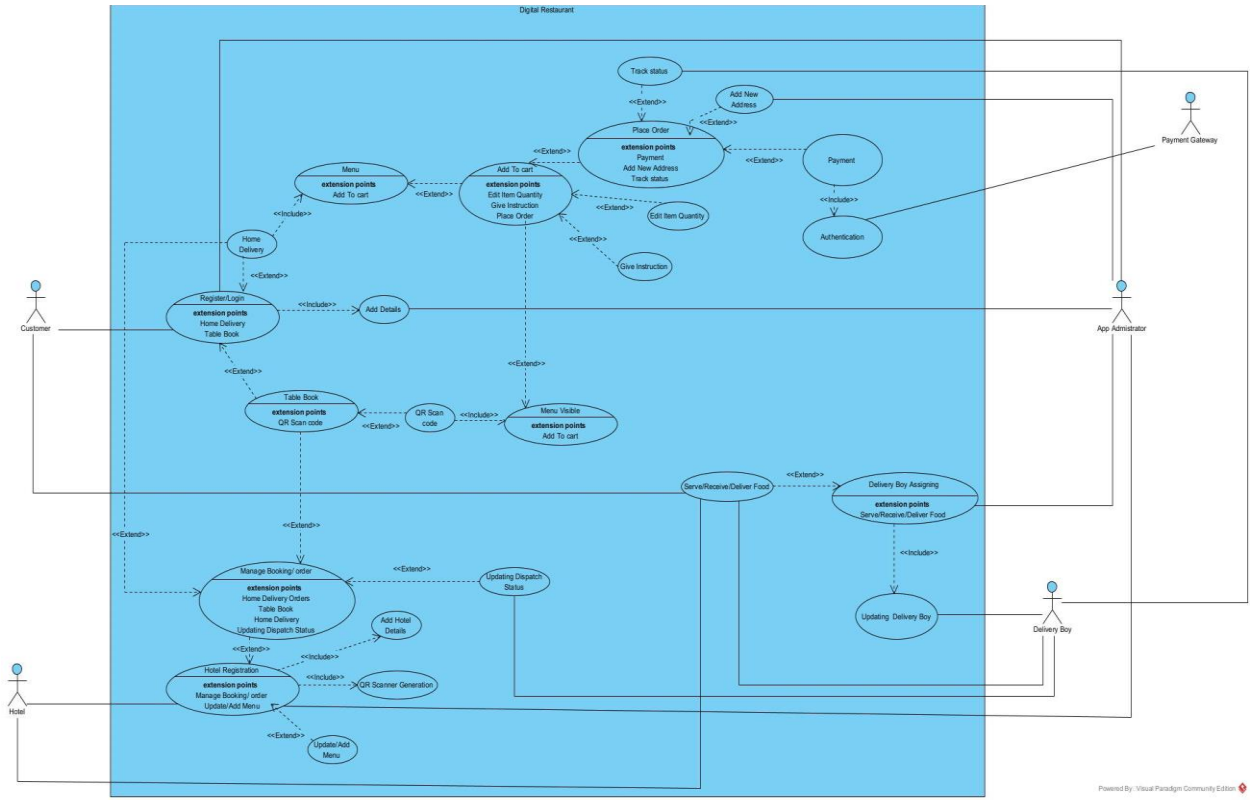
Actor	Customer, Payment Gateway
Purpose	Payment of bill from customer
Extensions	Authentication - Payment Gateway will authenticate payment.

Use Case	Hotel Registration/Login
Actor	Hotel
Purpose	They register so they can manage orders, add/update menu, QR code is generated for them
Extensions	QR Generation - QR scanner will generate for hotel menu Menu addition/update - Hotel will add menu / update menu Add Details - Hotel Details will be added Manage Booking/order - manage the table booking and home delivery

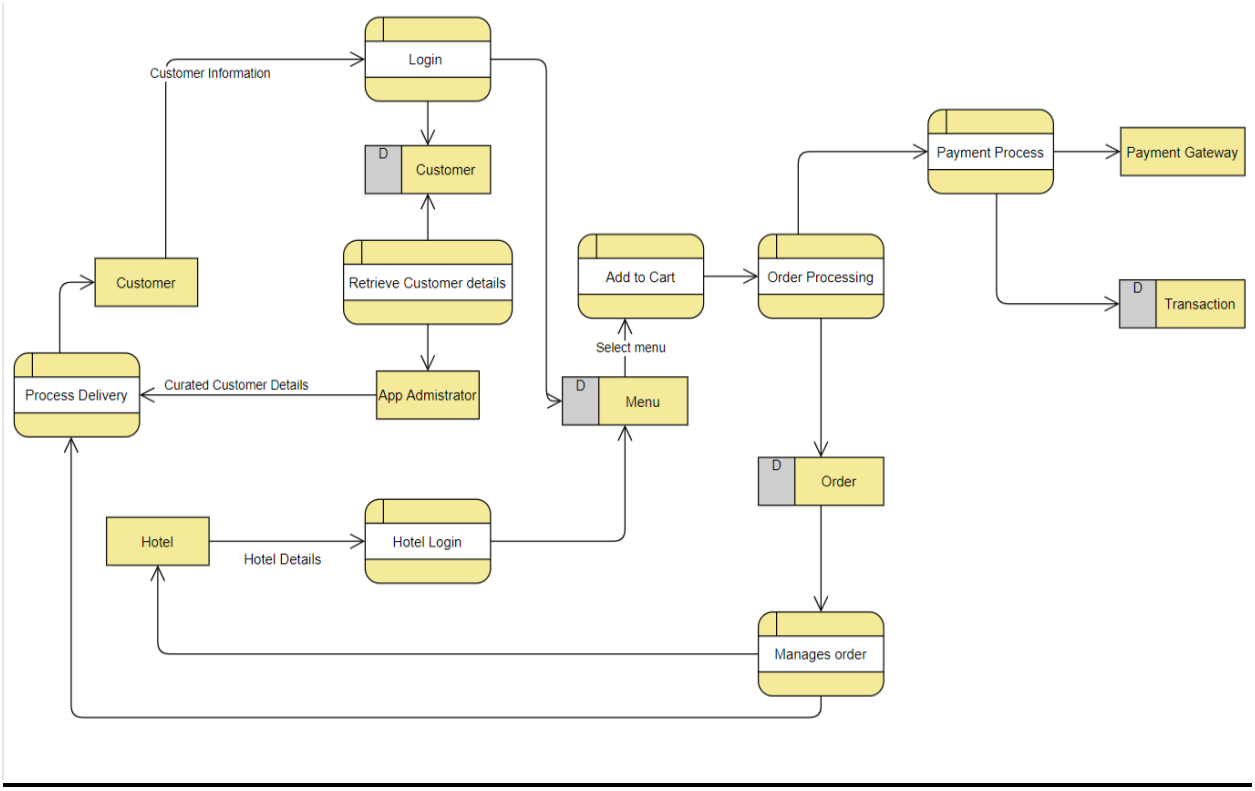
Use Case	Manages Booking / Order
Actor	Hotel
Purpose	Manage the table booking and home delivery
Extensions	Table Booking - Managed by this Home Delivery - Managed by this Updating dispatching details - Update the status of order when get dispatched.

Use Case	Assigning Delivery Boy
Actor	App Administrator
Purpose	They assign the Delivery Boy for particular order
Extensions	Updating delivery boy - he will get the

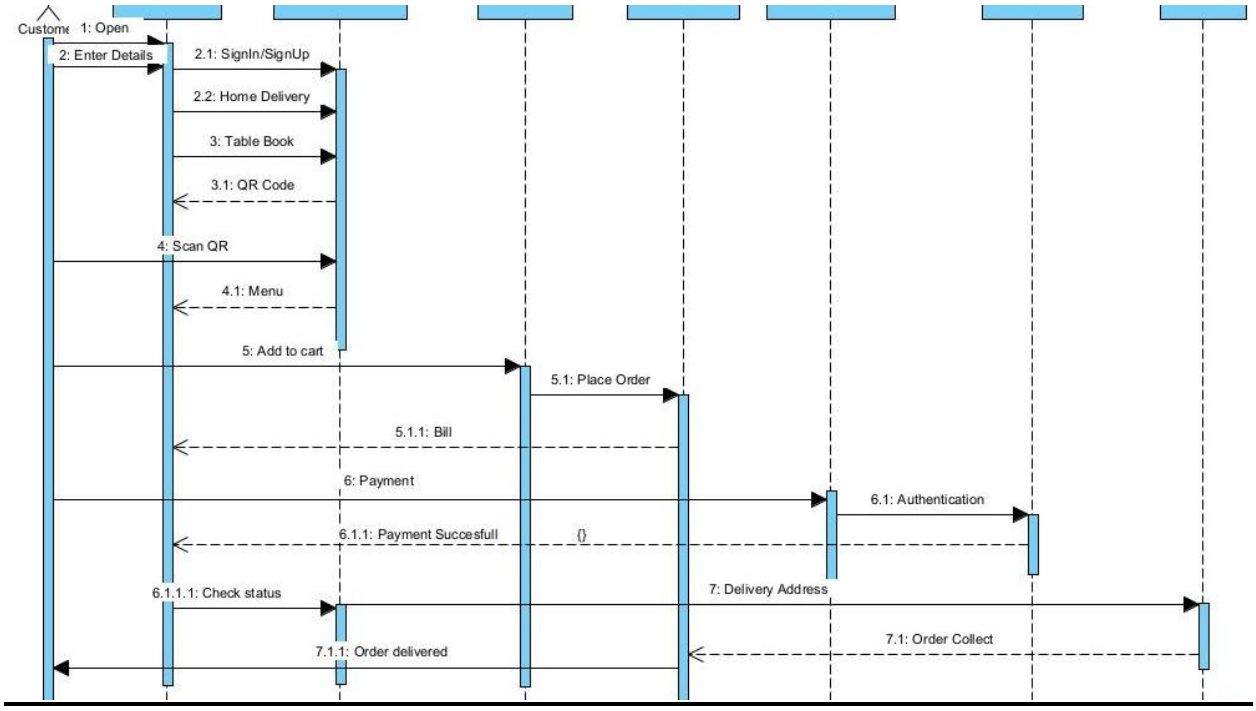
	notification Serve/Receive/deliver food - information Is updating from customers end that food is Received or served and from delivery boys side that food is delivered.



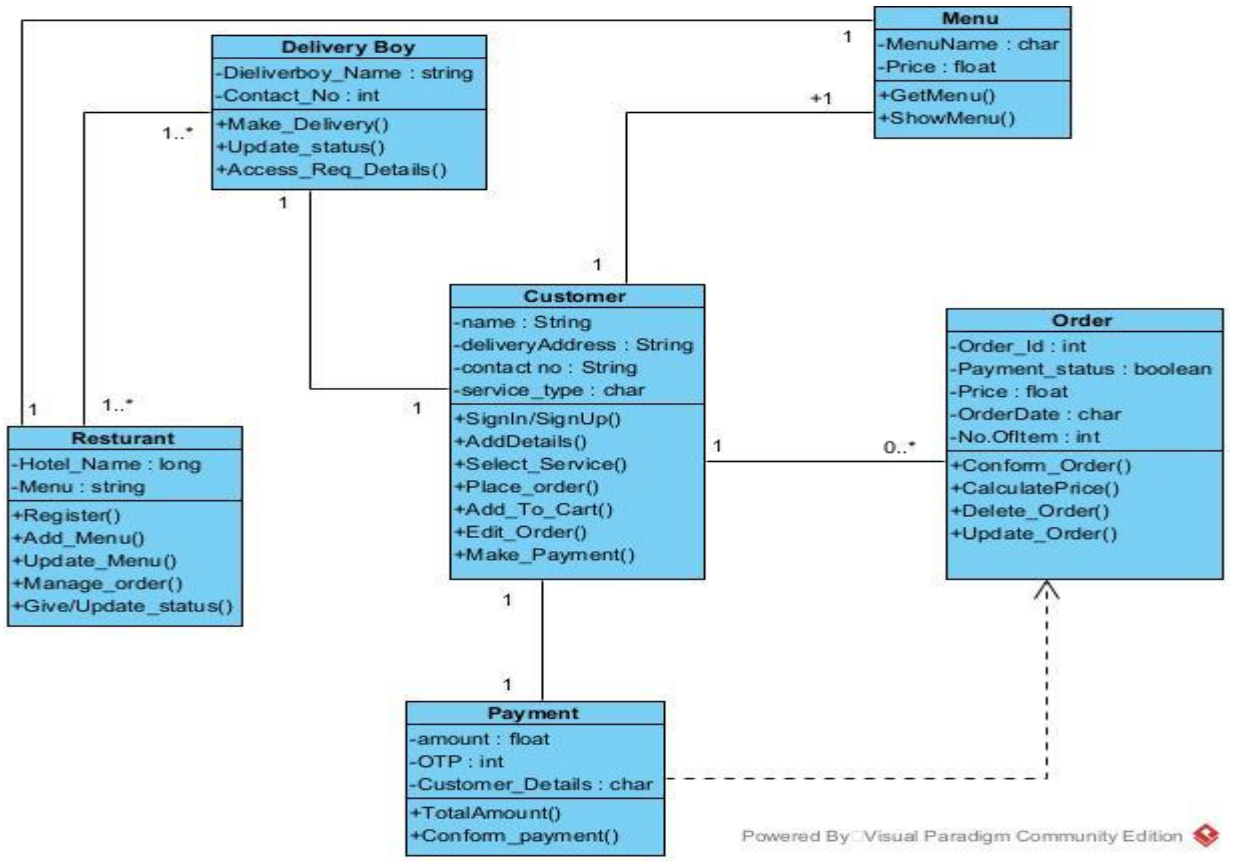
3.4.2 Data Flow Diagram



3.4.3 Sequence Diagram



3.4.4 Class Diagram



Class Diagram Description:-

Here Class Diagram consist of 6 classes as follows:-

- **Customer : -** This class will have all the functions which a customer can perform and then take the attributes name,Delivery Address,contact no. service type. The functions are as follows:-
    - **SignIn/SignUp() -** This function will allow users to register in app/login. This function will invoke the AddDetails() function.
    - **AddDetails() :-** This function will allow users to add their details for signing up.
    - **Select_Service() :-** This Function will get options to customers that they want to book the table or order food for online delivery.
    - **Place_order() :-** This function will allow users to get access to the menu.Invoke the function show_menu() from menu class.
    - **Add_to_cart() :-** Here customers can add items to cart from the menu.
    - **Edit_Order() :-** Customers can edit the order he/she will be going to place.
    - **Make_Payment :-** Customers can make payment from this function which invokes the class Payment.

- **Order :-** This class has attributes of order details and function to place order successfully.

- **Conform_order() :-** After Adding items to cart customer needs to confirm before placing the final order.
- **CalculatePrice() :-** It calculates the amount of order placed.
- **Delete_order() :-** It gives users an option that they can cancel the order.
- **Update_order() :-** Customers can add/delete items in the order.

- **Menu :-** It has the attribute of the item stated in the menu.
  - **ShowMenu():-** Customers class Function Place_order() invokes this function so customers can see the menu.
  - **GetMenu() :-** When Customer has chosen a book table service type it will give them QR code through GetMenu() function after scanning it showmenu() get invoked.

- **Payment :-** It has attributes of customer [payment details.
  - **TotalAmount() :-** Get the calculated price from the order class function called CalculatePrice().
  - **Conform_payment():-** This function will confirm users and Restaurant about whether the payment has succeeded or failed.

- **Delivery Boy :-** It has the attribute of Delivery boy information and customer contact details.
  - **Make_delivery() :-** This Function tells the Delivery boy when, where he needs to make delivery.
  - **Update_status() :-** This Function enables Delivery Boy the status of delivery.
  - **Access_Req_Details() :-** These functions give the details of the customer to make delivery.

- **Restaurant :-** It has attributes related to hotel details and functions performed by Restaurant.
  - **Register() :-** Hotels/Restaurant can register themselves to provide service.
  - **Add_Menu():-** After Registering they can add the menu.
  - **Update_Menu() :-** Restaurants can bring changes in menu.
  - **Manage_ Order():-** Restaurants need to manage table bookings and home deliveries.
  - **Give/update_status():-** They will update the status of order.