# Food Delivery App

*A*

***Report***

*Submitted*

*In partial fulfillment for 6th Semester*

***In Department of \_\_\_\_\_IT\_\_\_\_\_\_\_\_\_\_\_***

Guide: Submitted by:

Dr. Parita Shah Zinzuwadia Yash - 20BEIT54068

Soni Devarsh - 20BEIT54059

Padhiyar Nimesh - 20BEIT54014

Patel Venish – 20BEIT54044

Bhavsar Vraj – 20BEIT54003



**Department of Information Technology**

**ACKNOWLEDGEMENT**

I wish to express my heartfelt appreciation to all those who have contributed to this project, both explicitly and implicitly, without the co-operation of whom, it would not have been possible to complete this project.

I would like to thank our Dr. Parita Shah for guiding and showing us the correct path to reach towards our desired goal. Also, we thank them for sharing their experience, knowledge and valuable time with me and showing their concern in my project to make it better.

I am grateful to my college V.S.I.T.R for providing me all the required resources and a good working environment.

At last, I would like to thank my batchmates who have directly or indirectly helped me in making the project work successful.

I

# ABSTRACT

The Food Delivery System delineated during this paper has been designed to fill a particular niche within the market by providing little restaurants with the power to supply their customers a web ordering possibility while not having to speculate giant amounts of your time and cash in having custom software system designed particularly for them. The extremely customizable system permits the building workers to simply manage the positioning content, most significantly the menu, themselves through an intuitive graphical interface. The website, which is the sole element seen by the building customers, is then designed dynamically to support the present state of the system, thus any changes created square measure mirrored in real-time. Guests to the current website, once registered, square measure then able to simply navigate this menu, add food things to their order, and specify delivery choices with solely some clicks, greatly simplifying the ordering method. Back within the building, placed orders square measure promptly retrieved and displayed in a simply decipherable format for the economical process. This abstract aims to supply in-depth descriptions of the style and implementation details of the system, additionally with descriptions of all offered practicality and plans for evolution. Additionally, user manuals and troubleshooting tips are enclosed for all three parts to allow the reader a transparent plan of supposed typical use cases for the system.

II

|  |  |
| --- | --- |
| **INDEX**    **Sr. Content**  **No**  **ACKNOWLEDGEMENT…………………………………………………………. ABSTRACT…………………………………………………………………………**  **INDEX………………………………………………………………………………. LIST OF FIGURES…………………………………………………………………**   1. **INTRODUCTION…………………………………………………………………..**     1. Purpose    2. Scope    3. Definitions, Acronyms and Abbreviations    4. References 2. **OVERALL DESCRIPTION………………………………………………………..**     1. Product Perspective    2. Product Features    3. Operating Environment    4. Design and Implementation Constraints    5. Assumptions and Dependencies 3. **FUNCTIONAL REQUIREMENT…………………………………………….......** 4. **NON FUNCTIONAL REQUIREMENT…………………………………………..** 5. **DESIGN AND ANALYSIS…………………………………………………….......**     1. Use Case Diagram    2. Data Flow Diagram    3. Class Diagram    4. Flow Chart Diagram 6. **TESTING…………………………………………………………………………..**     1. White Box Testing    2. Black Box Testing 7. **FUTURE ENHANCEMENTS AND CONCLUSION…………………………..** 8. **BIBLIOGRAPHY ………………………………………………………………...**   III | **Page No.**  **I**  **II**  **III**  **IV**  **1**            **4**            **8**  **11**  **13**          **17**        **19**  **20** |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Sr No.** | **Figure** | **Page No.** |
| **1.** | **5(a)** | **13** |
| **2.** | **5(b)** | **14** |
| **3.** | **5(c)** | **15** |
| **4.** | **5(d)** | **16** |
| **5.** | **6(a)** | **17** |
| **6.** | **6(b)** | **18** |

IV

# 1 : Introduction

## 1.1 Purpose

Food-delivery apps allow customers to order from a nearby restaurant at their convenience. The customers can get their order delivered, they can pick it up themselves or they can dine in. The restaurants receive the order on the restaurant app and prepare the meal. The delivery guy picks the order from the restaurant and drops it at the dropping location.

## 1.2 Scope

With many food apps coming up in the market, it’s been easy to utilize and incorporate into your daily life. It helps with food ordering, table booking, online delivery or reservations. This ease can be easily adaptable by people around, which can never fail. The **food delivery app market** faces development challenges because of low Internet network on cell phones. Moreover, extra expense acquired by eatery or service, franchisee to dispatch a mobile application, and further for support, establishment, and promoting is constraining the development of Food delivery app market somewhat.

## 1.3 Definition

Food delivery is a home delivery service in which a store , restaurant , or third party app delivers food to consumers , whenever they ask for it . These days offers are generally placed through a mobile app , website or phone. Delivery includes cooked meals and groceries from supermarkets. Other ways to deliver food include catering or wholesale.

**1.4 References**

## ❖ How to write software requirements specification at <http://www.microtoolsinc.com/Howsrs.php>

❖  **SRS Template at** [**http://www.processimpact.com/process\_assets/srs\_template.doc.**](http://www.processimpact.com/process_assets/srs_template.doc)

# 2 : Overall Description

**2.1 Product Perspective**

The software described in this SRS is the software for a complete restaurant food ordering system. The system merges various hardware and software elements and further interfaces with external systems. It relies on a number of external interfaces for persistence and unhandled tasks , as well as physically interfacing with humans.

**2.2 Product Features**

The restaurant food ordering system interfacing with an existing payment system, including a cash register and software accessible credit system , in order to quickly and easily handle customer billing. The payment system should be operable such that it can return information to the RFOS system as to whether payment was successful or failed.

**2.3 User Classes and Characteristics**

There are three separate user interfaces used by the RFOS software , each related to an interfaced physical hardware device. These three user interfaces are the Surface Computer UI , Tablet UI , Display UI.

**2.4 Operating Environment**

The Surface Computer UI is the interface used by restaurant customer. This interface uses the surface computer paradigm-user interact with the system by dragging ‘objects’ around on the flatscreen touch-sensitive display.

The Tablet UI is designed to run on a small , wireless-enabled touch-screen tablet PC, to be used by waiters to accommodate customer needs.

The Display UI provides kitchen staff with simple functionality related to ordered items.

* 1. **Design and Implementation Constraints**

The RFOS should be written in an object-oriented language with strong GUI links and a simple , accessible networks 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 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.

* 1. **User Documentation**

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

**2.7** **Assumptions and Dependencies**

The SRS assumes that none of the constituent system components will be implemented as embedded applications. It is further assumed that tablet PCs of sufficient processing capability and battery life will be utilised.

# 3 : Functional Requirement

**Customer Requirements :**

* Customer shall be able to login to his account through username and password.
* Customer shall be able to choose restaurant from a list of restaurant.
* Customer shall be able to see a the menu of selected restaurant.
* Customer shall be able to order from the menu of a selected restaurant.
* Customer shall be able to add an item to a favourite list.
* Customer shall be able to delete an item from a favourite list.
* Customer shall be able to communicate through messages or call with restaurant directly.
* Customer shall be able to reserve a table.
* Customer shall be able to see the offers of a restaurant.
* Customer shall be able to track the time spent on preparing his order and be able to track the delivery rider after the order has been prepared.
* Customer shall be able to see the history of the order.

**Manager Requirements:**

* Manager can login through their unique ID and Password.
* Manager shall be able manage a reserve tables.
* Manager shall be able to add the item to the menu list.
* Manager shall be able to delete items from the menu list.
* Manager shall be able to update the list.
* Manager shall be able to manage users.
* Manager shall be able to manage orders.

**Delivery Order:**

* Delivery rider shall be able to locate the customer which he has been assigned to deliver the order.

Non functional Requirements Chapter : 4

# 4 : Non Functional Requirement

**Security :**

* The system should prevent unauthorised access or misuse of sensitive information, such as consumer payment and personal information.  
  This could include regulations for the use of encryption, secure servers, and other data integrity safeguards.

**Scalability:**

* Scalability refers to the system’s ability to accommodate increases in the number of users or orders without deteriorating performance.  
  This could include the capacity to add more servers or other hardware as needed to accommodate rising demand.

**Reliability :**

* The system should be available and working when required, with as little downtime as possible.  
  This could include requirements for the system’s ability to handle failures or unforeseen events, as well as the utilisation of backup systems and processes to assure service continuity.

Non functional Requirements Chapter : 4

**Maintainability:**

* With a clear and well-documented codebase and a solid testing and deployment procedure, the system should be simple to upgrade and maintain over time.  
  This could include requirements for using version control, automated testing, and other tools and processes to keep the system reliable and up to date.

**Usability:**

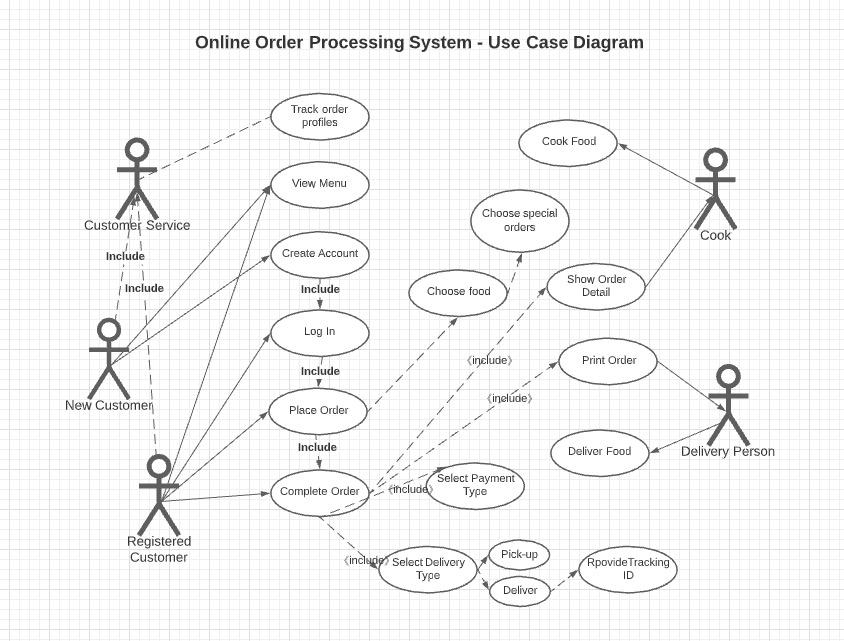
* The system should be simple to use for both customers and restaurant employees, with a clear and intuitive interface and simple navigation.  
  This could include criteria for the system’s layout and design, the use of clear and simple language, and the provision of assistance and support.

**Performance:**

* The system should be able to process a high volume of orders efficiently.  
  This could include system speed, the quantity of orders it can process at once, and the ability to handle peak periods of activity.

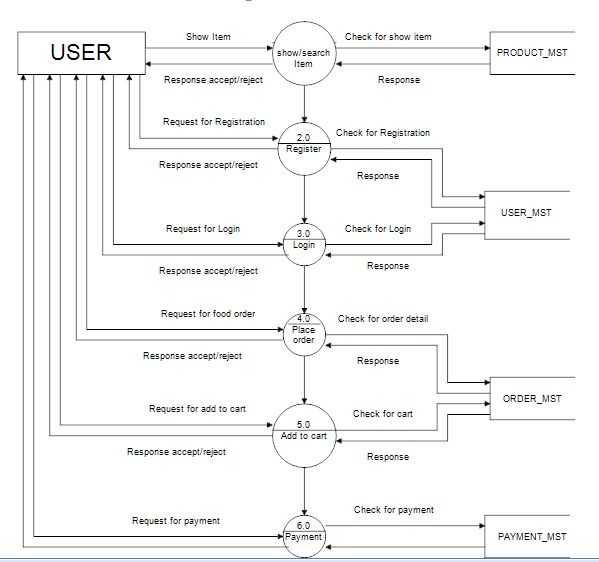
# 5 : Design And Analysis

## 5.1



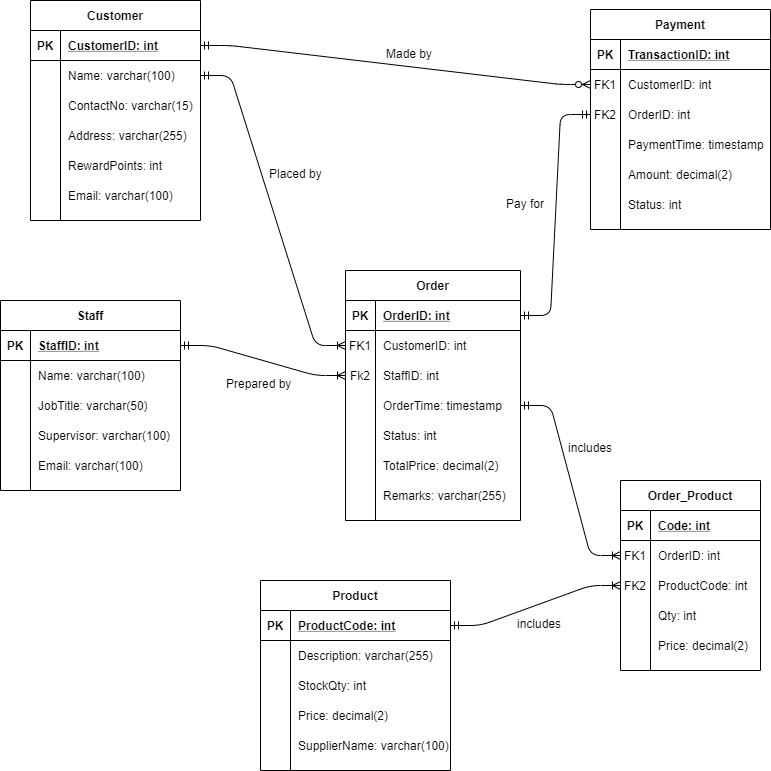
**Fig. 5(a) of Use Case Diagram**

## 5.2



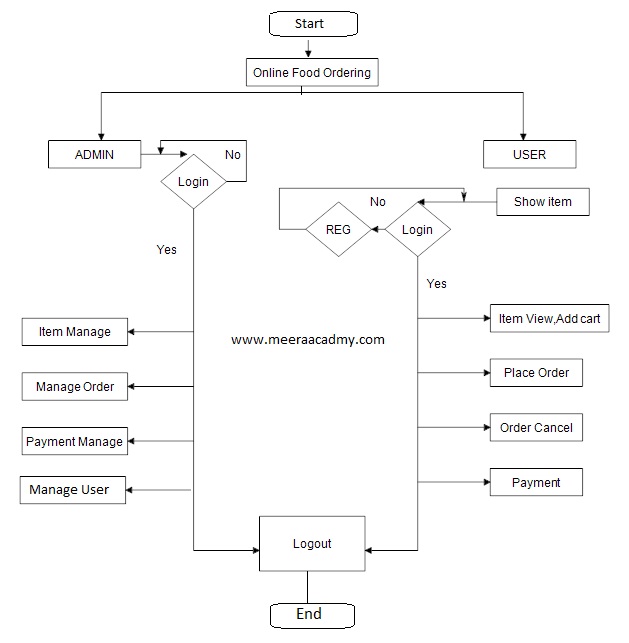
**Fig 5(b) Of Data Flow Diagram**

## 5.3



**Fig 5(c) of Class Diagram**

## 5.4



**Fig 5(d) of Flow Chart**

Testing Chapter : 6

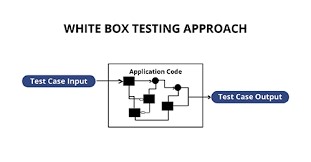
# 6 : Testing

**6.1 White Box Testing**

White Box Testing is a technique that is applied for checking the system's internal functioning. This testing process is entirely based on the coverage of the code statements, paths, branches, or conditions. The White Box testing is generally considered to be a lower-level software testing.

Steps :

1. First of all, we will understand the inner structure or source code of that particular part which is to be tested.
2. Then, we will create some test classes by which the whole process of testing can come to an efficient result.



**Fig 6(a)**

Testing Chapter : 6

**6.2 Black Box Testing**

Black Box Testing is a software testing method in which the functionalities of software applications are tested without having knowledge of internal code structure, implementation details and internal paths. Black Box Testing mainly focuses on input and output of software applications and it is entirely based on software requirements and specifications. It is also known as Behavioral Testing.

Steps :

1. First of all, we will take some valid and also invalid inputs for our system and will check that it gives the correct output or not.
2. Then, we will create some test cases from selected inputs and will execute those classes.
3. After execution of test classes, we will compare the outputs with the correct outputs and fix it.

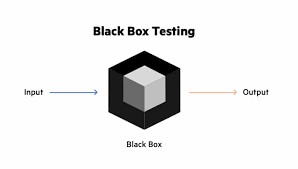


Fig6(b)

Future Enhancement And Conclusion Chapter:7

**7 : Future Enhancement And Conclusion**

**Future Enhancement :**

1. Improving Easibility for Internal use.
2. Improving Tracking System of food delivery rider throughout the whole process.
3. Improve to keep Syncing the Data of the System.
4. Improvement in Data Updation.

**Conclusion :**

Our project is only a humble venture to satisfy the needs to manage their project work. Several user friendly coding have also adopted. This package shall prove to be a powerful package in satisfying all the requirements of the school. The objective of software planning is to provide a frame work that enables the manger to make reasonable estimates made within a limited time frame at the beginning of the software project and should be updated regularly as the project progresses. At the end it is concluded that we have made effort on following points...

• A description of the background and context of the project and its relation to work already done in the area.

• Made statement of the aims and objectives of the project.

• The description of Purpose. Scope, and applicability

• We define the problem on which we are working in the project.

• We describe the requirement Specifications of the system and the actions that can be done on these things.

• We understand the problem domain and produce a model of the system, which describes operations that can be performed on the system.

• We included features and operations in detail, including screen layouts

• We designed user interface and security issues related to system . Finally the system is implemented and tested according to test cases.