**ABSTRACT**

Online food delivery system is an order management software for food delivery companies. In recent times, ordering food online is popular and normative for consumers. This system is a third-party platform that integrates information from customers, restaurants and delivery agents. This system accepts the orders placed by the consumers and place these orders to the restaurants and then instructs the delivery agents to collect the food from the specified restaurant and delivery it to the customers. No separate order is required for each restaurant. The system also has an administration interface where members of the delivery company can track and manage the received orders. They can create new restaurants, menus, and modify existing ones. The mobile application helps the delivery personnel to manage the incoming orders. Only users with delivery role can use the application. After a successful login, the orders are displayed for the current date. Both individual and group orders can be accepted partially by selecting individual items from the corresponding list. The heuristics algorithm is used in this system to reduce the time delay in ordering and delivering of food.

**ABSTRACT IN TAMIL**

ஆன்லைன் உணவு விநியோக அமைப்பு என்பது உணவு விநியோக நிறுவனங்களுக்கான ஆர்டர் மேலாண்மை மென்பொருளாகும். சமீபத்திய காலங்களில், ஆன்லைனில் உணவை ஆர்டர் செய்வது பிரபலமானது மற்றும் நுகர்வோருக்கு இயல்பானது. இந்த அமைப்பு வாடிக்கையாளர்கள், உணவகங்கள் மற்றும் விநியோக முகவர்களிடமிருந்து தகவல்களை ஒருங்கிணைக்கும் மூன்றாம் தரப்பு தளமாகும். இந்த அமைப்பு நுகர்வோர் வழங்கும் ஆர்டர்களை ஏற்று, இந்த ஆர்டர்களை உணவகங்களுக்குச் செய்து, பின்னர் குறிப்பிட்ட உணவகத்தில் இருந்து உணவைச் சேகரித்து வாடிக்கையாளர்களுக்கு டெலிவரி செய்யும்படி டெலிவரி முகவர்களை அறிவுறுத்துகிறது. ஒவ்வொரு உணவகத்திற்கும் தனித்தனி ஆர்டர் தேவையில்லை. டெலிவரி நிறுவனத்தின் உறுப்பினர்கள் பெறப்பட்ட ஆர்டர்களைக் கண்காணித்து நிர்வகிக்கக்கூடிய நிர்வாக இடைமுகமும் கணினியில் உள்ளது. அவர்கள் புதிய உணவகங்கள், மெனுக்களை உருவாக்கலாம் மற்றும் ஏற்கனவே உள்ளவற்றை மாற்றலாம். மொபைல் பயன்பாடு டெலிவரி பணியாளர்களுக்கு உள்வரும் ஆர்டர்களை நிர்வகிக்க உதவுகிறது. டெலிவரி ரோல் உள்ள பயனர்கள் மட்டுமே பயன்பாட்டைப் பயன்படுத்த முடியும். வெற்றிகரமான உள்நுழைவுக்குப் பிறகு, தற்போதைய தேதிக்கு ஆர்டர்கள் காட்டப்படும். தொடர்புடைய பட்டியலிலிருந்து தனிப்பட்ட உருப்படிகளைத் தேர்ந்தெடுப்பதன் மூலம் தனிநபர் மற்றும் குழு ஆர்டர்கள் இரண்டையும் ஓரளவு ஏற்றுக்கொள்ளலாம். உணவு ஆர்டர் மற்றும் டெலிவரி செய்வதில் நேர தாமதத்தைக் குறைக்க இந்த அமைப்பில் ஹியூரிஸ்டிக்ஸ் அல்காரிதம் பயன்படுத்தப்படுகிறது.

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Fig.No.** | **Fig. Name** | **Page.No.** |
| **1** | System Architecture | 19 |
| **2** | Data Flow Diagram -Admin module | 22 |
| **3** | Data Flow Diagram -Restaurant module | 22 |
| **4** | Data Flow Diagram -Customer module | 23 |

**CHAPTER 1**

**INTRODUCTION**

**1.1 Overview of the Project**

E-commerce refers to the purchase and sale of goods and or services via the internet. Online Food Ordering System is a part of e-commerce. ONLINE FOOD ORDER SYSTEM is a website designed primarily for use in the food delivery industry. Through these services, restaurants can sell and distribute their resources at minimal resource usage effectively with high profits by gaining customer trust. This Online food order system database will be helpful for the business owners to extend their business just by placing the orders online and not visiting the restaurant. There is no confinement for placing and receiving the orders since the order can be placed online. There will be no waiting time with the vast amount of verities at comfortable prices. To develop this application database is the main part that will communicate through the application to retrieve the details. The database includes Customers who can place their orders from different food categories and restaurant staff will process the orders and deliver the requested order within an expected delivery time, and asking the customers for the reviews, and depending on the order quantity provide the rewards to the customers where they can claim the money.

The "Online Food Ordering System" is developed to override the problems prevailing in the practicing manual system. This software is supported to eliminate and, in some cases, reduce the hardships faced by this existing system. Moreover, this system is designed for the particular need of the company to carry out operations smoothly and effectively.

The application is reduced as much as possible to avoid errors while entering the data. It also provides an error message while entering invalid data. No formal knowledge is needed for the user to use this system. Thus, this, all prove it is a user-friendly Online Food Ordering System, as described above, which can lead to an error-free, secure, reliable, and fast management system. It can assist the user to concentrate on their other activities rather than concentrate on record-keeping. Thus, it will help the organization in better utilization of resources,

Every organization, whether big or small, has challenges to overcome and manage the information of Category. Food Item, Order, Payment, Confirm Order. Every Online Food Ordering System has different Food Item needs, therefore, an exclusive employee management systems that is designed so managerial requirements are adopted.

It is known globally that, in today’s market, it is extremely difficult to start a new small-scale business and live through the competition from the well-established and settled owners. In the fast-paced time of today, when everyone is squeezed for time, the majority of people are finicky when it comes to placing a food order. The customers of today are not only attracted because placing an order online is very convenient but also because they have visibility into the items offered, price, and extremely simplified navigation for the order. The online ordering system that I am proposing here greatly simplifies the ordering process for both the customer and the restaurant. The system presents an interactive and up-to-date menu with all available options in an easy-to-use manner. Customers can choose one or more items to place an order which will land in the Cart. Customers can view all the order details in the cart before checking out. In the end, the customer gets order confirmation details. Once the order is placed it is entered into the database and retrieved in pretty much real-time. This allows Restaurant Employees to quickly go through the orders as they are received and process all orders efficiently and effectively with minimal delays and confusion. This is designed to assist in strategic planning and will help you ensure that your organization is equipped with the right level of information and details for your future goals. Also, for those busy executives who are always on the go, our systems come with remote

access features, which will allow you to manage your workforce anytime, at all times. These systems will ultimately allow you to better manage resources.

**1.2 Existing System**

The labour rates are increasing steadily year on year thus making it difficult to find employees. The food industry is highly labour intensive and the biggest expense in the food industry is the cost of employing the right kind of people to do the work.

In today’s age of fast food and take-out, many restaurants have chosen to focus on quick preparation and speedy delivery of orders rather than offering a rich dining experience. Until very recently, all of these delivery orders were placed over the phone, but there are many disadvantages to this system. first, the customer must have a physical copy of the restaurant’s menu to look at while placing their order and this menu must be up to date. While this expectation is not unreasonable, it is certainly inconvenient.

Second, the orders are placed using strictly oral communication, which makes it far more difficult for the customer to receive immediate feedback on the order they have placed. This often leads to confusion and incorrect orders. The current system is also inconvenient for the restaurant itself, as they must either have a dedicated staff member to answer the phone and take orders, or some employees must perform double-duty, distracting them from their regular tasks.

In addition to young Indians fueling the growing demand for the services of online food delivery, the increase in dual income families in urban areas of India, where both parents work, is dramatically changing the way people live in subtle yet significant ways. The changes in routines, lifestyle and food habits have resulted in an increase in demand for easily accessible and good quality food. It has been estimated that almost 92% of nuclear families who seek out fast food or take out, as against preparing a meal at home, do so in order to save the time and energy. Also, the increase in the number of working women increasing, and the increase in the amount of disposable income has proved to be a key demand driver.

**1.3 Proposed System**

The proposed system is an online food ordering system that enables ease for the customers. It overcomes the disadvantages of the traditional queueing system. The proposed system is a medium to order online food hassle free from restaurants as well as mess service. This system improves the method of taking the order from customer. The online food ordering system sets up a food menu online and customers can easily place the order as per their wish. Also with a food menu, customers can easily track the orders. This system also provides a feedback system in which user can rate the food items. Also, the proposed system can recommend hotels, food, based on the ratings given by the user, the hotel staff will be informed for the improvements along with the quality. The payment can be made online or pay-on- delivery system. For more secured ordering separate accounts are maintained for each user by providing them an ID and a password.

Functionalities already exist in this Online Food Ordering System are as follows: Provides the searching facilities based on various factors. Such as Food items, Customers, Orders, Confirm orders. Online Food Ordering System also manages the Payment details online for Order details, Confirm Order details, and Food items. It tracks all the information of Category, Payment, Order, etc. Manage the information of Category Shows the information and description of the Food Item, Customer To increase the efficiency of managing the Food Item, Category. It deals with monitoring the information and transactions of Orders. Manage the information of Food Item Editing, adding and updating of Records is improved which results in proper resource management of Food Item data. Manage the information of Order Integration of all records of Confirm Order.

The proposed system is a bunch of benefits from various points of view. As this online application enables the end-users to register to the system online, select the food items of their choice from the menu list, and order food online. Also, the payment can be made at the time of home delivery depending upon the customer’s choice and convenience. The selection made by the customers will be available to the hotel reception or the person handling the work assignment Now this same person will assign the orders to the specialist chef to be completed within a fixed duration of time As soon as the chef prepares the food, the theaterperson

forwards the parcels to the delivery persons assigned with the location and customer identity of the customer along with the bill status. With this application, the workload of the waiter in the hotels is reduced or in some situations, their work is abolished. One of the various benefits of this system is that if there is a rush or a huge crowd present in the restaurant then in that case sometimes the unavailability of tables cut downs the restaurant’s customers. Also, there will be chances that the waiters are unavailable as they are busy handling others, so the customer can directly order the food from the chef online by using this application, by checking the seat availability in the restaurant. This system allows the staff to serve customers within less time compared to the manual system.

**1.4 Objectives of the Project**

The main objective of the project is to learn and implement a real-time application on a database for Online-Food Ordering System. The main task is to build a responsive online application for restaurant which helps customers to order foods online. Along with this, the application needs to be responsive as the application can be accessed through devices with different size of screens. Social media websites play a vital role to attract customers and make trust on their mind.

The project concentrates on taking orders, streamlining the orders to a specified restaurant, and billing. This Database will be a great solution for many start-up food businesses, they can just start initially with fewer funds by posting their menu online with this application.

The purpose of the Online Food Ordering System is to automate the existing manual system with the help of computerized equipment and full-fledged computer software, fulfilling their requirements, so that their valuable data/information can be stored for a longer period with easy accessing and manipulation of the same. The required software and hardware are easily available and easy to work with.

Online Food Ordering systems, as described above, can lead to error-free, secure, reliable, and fast management systems. It can assist the user to concentrate on their other activities rather than concentrate on record keeping. Thus, it will help the organization in better utilization of resources. The organization can maintain computerized records without redundant entries. That means that one need not be distracted by information that is not relevant while being able to reach the information.

The aim is to automate its existing manual system with the help of computerized equipment and full-fledged computer software, fulfilling their requirements, so that their valuable data/information can be stored for a longer period with easy accessing and manipulation of the same the project describes how to manage for good performance and better services for the clients

1.5 Organization of the Report

Chapter 2: Discuss the Literature Survey

Chapter 3: Discuss the System Analysis

Chapter 4: Discuss the System Specification

Chapter 5: Discuss the System Design

Chapter 6: Discuss the System Testing

Chapter 7: Discuss the Implementation

Chapter 8: Conclusion

Chapter 9: Future work

**CHAPTER 2**

**LITERATURE SURVEY**

In the paper, “ Netfood: A Software Systems for Food ordering and Delivery”, by Cristina-Edina Domokos, Barna Sera, Karoly Simon, Lajos kovacs, Tas -Bela Szakacs in International Symposium on intelligent system and Informatics, IEEE,2018, the authors say, netfood is a delivery –oriented system that allows clients to order from multiple restaurant at the same time, and provides the possibility to order individually or in a group. The communication between the peers is implemented by a Restful API. The data is stored in a relational database. The users and the administrators can log in to the system through the web Interface, with a Facebook account or using a registered username. After a successful registration the system sends a confirmation e-mail to the user. This notification system is implemented using the Java Mail Sender interface provided by the spring framework. Liqui base is a database version tracker for managing the state change of the database during development. The database update operations are partially automated using Liquibase. An interface for restaurant owners for directly managing their offers; the possibility for assembling daily menus from already uploaded foods, after placing an order the customer should receive a message with the estimated time of delivery, IOS version for the mobile application, Google maps integration into the mobile application, to navigate the delivery personnel from the current location to the delivery address, a stand- alone mobile application used by customers for placing orders.

The paper, “A Heuristic Hybrid Recommended Order Modle”, by Yi Yang, Baolin Li, Jian Hu, in International Conference on Computer Science & education(ICCSE 2019)IEEE, Proposes a heuristics hybrid recommendation ordering model, which combines the traditional association rule recommendation algorithm with the recommendation algorithm based on the dishes to from the dish recommendation rules, and then ,make the recommendation list based on the dish each time after ordering. The

collaborative filtering-based recommendation algorithm aims at calculating the similarity among different users and items, and recommendation by using the result to compute the prediction of users scores on items. These recommendation algorithms have been widely applied in the field such as e-commerce, entertainment, games, etc. since it is difficult to collect accurate customer characteristics data, the Heuristic Hybrid Recommendation Order Model (HHROM) mainly recommends dishes to customers by analyzing existing historical order data and combining the four dishes attributes of dish prices, types, tastes and ingredients. The algorithmic thinking of HHROM model is mainly to recommend by combining association rules and recommendation algorithm based on dishes attributes. The FP-growth algorithm and HHROM recommendation model are compared and analyzed with two indicators of recommendation accuracy and coverage. The results show that the two recommendation models have similar effect when the number of order is relatively small. When the number of recommended dishes reaches a certain level, however, HHROM recommendation model is better than FP-growth algorithm.

The paper “A Two-stage Algorithm for Fuzzy Online Order Dispatching Problem” by Jie Zheng,Shengyao Wang, Ling Wang ,Jing-fang chen,Li wang, Jinghua Hao, Renqing He, Zhizhao Sun, IEEE Xplore, addresses an online order dispatching problem with fuzzy preparation times(FOODP).According to the characteristics of the problem ,the FOODP is decomposed into two sub-problems, an order assignment problem and a fuzzy traveling salesman problem with pickup and delivery. To deal with the two sub-problems efficiently, a two-stage algorithm is proposed by reasonably fusing a modified greedy search (MGS) and the fruit fly optimized algorithm (FOA). In the MGS phase, agreement index is employed in the multi-stage decision to search for robust optimal solutions. In the FOA- based search phase, a modified heuristic is used to generate the initial route. To enhance the exploration of the algorithm, an olfactory search is designed by the cooperation of several problem-specific search operators. More over a specific local intensification is employed to further improve the performance of solutions. Numerical tests and statistical analysis demonstrate the effectiveness and efficiency of the proposed algorithm.

The paper “Android Application Food Delivery Services”,by Lidya Chitra Laoh,Timothy Adithia pongantung, Carolin Mulalinda,IEEE Xplore, aims to offer an android based food delivery services application to ease customer to meet their order. Developed application works only in Android based smartphone. Enable the buyer to set the address coordinate directly from Google Maps. Add the ID verification menu for the driver who wants to register. Provide back-end of the application that might be managed by an admin.

In the paper “Stimulus Factors of Order online Food Delivery”, by Yakob Utama Chandra, Cadelina Cassandra, in International Conference on information Management and Technology (ICIMTech) IEEE, 2019, the authors have used quantitative approach with 187 respondents to see what stimulus factors that make people interest in online order for food delivery. Questions and which stimulus factors that make people interest to do online order for food delivery There are 6 (six) hypotheses for this research and the answers to these questions are privacy factor and informativeness factors as the stimulus to make the customer order online food delivery. The SOR model instead of stimulus Organism response Model is a framework that used to find out how human process receives a stimulus factor. The stimulus factors of user online food delivery that can influence the behavior of their customers using the SOR Mode. Perceived Ease of use (PEOU) significant to the value of customer (VC). Perceived Usefulness (PU) significant to the value of customers (VC). Value of customers (VC) significant to the Behavior to use (BU).

In the paper “Two Fast Heuristics for Online Order Dispatching”, Qingte Zhou, Huanya Zheng, Shengyaowang,Jinghua hao, Renqing He, Zhizhao, in IEEE Xplore, 2020, In the authors have studied online dispatching problem considering some realistic constraints they have presented two fast heuristics, namely modified greedy insertion (MGI)and modified regret insertion (MRI),based on greedy insertion (GI) heuristics and regret insertion (RI)heuristics are ,respectively. The heuristics are tested on a series of instances that are randomly sampled from real order dispatching problems. The result of these experiments shows that the solution quality of MGI and MRI are consistently improved with respect to GI and are very close to RI. As for computation time, MGI is faster than GI and MRI is also faster than RI, which demonstrates the efficiency of proposed heuristics.

In this paper “Online Food Delivery Platforms and Restaurants Interactions in the Context of the Ban on Using Single-use Plastics”, KEQAN LI, YAN CHEN, JIACHEN LIU, LIN ZHANG, XIANGWEI MU in, IEEE ACCESS, 2021, the authors have applied evolutionary game theory to examine the interaction mechanism of the problematic behaviors between OFD platforms and restaurants. They have developed an evolutionary game model to analyze an OFD platforms and restaurants behavioral strategies in the context of the ban on using single-use plastics. The game players in the process are an OFD platform and a restaurant.Both players are stalk holders in the process, and they are finitely rational.

The paper “Food Delivery Automation in Restaurants Using Collaborative Robotics”, Albin Antony, Sivraj.P, International Conference on Inventive in Computing Applications IEEE,2018, highlights the issues and challenges in collaborative robotics by considering restaurant automation using multiple robots and a central station. A selection algorithm for the optimal selection of robots for a particular task and modified A\* algorithm for line following robot motion is simulated and validated in MATLAB. For hardware validation of the same a gridbased restaurant environment is created with cells representing tables and black lines as paths for robot movements. Firebird IV robotic experimenter kits interfaced to android phones with android applications for taking order from customers acts as waiters enabling automation. These robotic waiters are programmed with modified A\* algorithms for moving from current positions to destinations via shortest path in a line following fashion.

**CHAPTER 3**

**SYSTEM ANALYSIS**

The world is complex and full of problems to solve. It's probably not surprising, therefore, that problem solving is one of the most sought-after skills. If you can break a problem apart, and come up with a solution, your skills will always be needed. One type of problem solving is called systems analysis.

**3.1 System Requirements**

System requirements are the required specifications a device must have in order to use certain hardware or software. For example, a computer may require a specific I/O port to work with a peripheral device. A smartphone may need a specific operating system to run a particular app.

**3.1.1 Hardware Requirements**

• Processor - Intel(R) Core(TM)

• Speed - 1.1 GHz

• RAM - 4.00 GB

• Hard Disk - 20 GB

• Key Board - Standard Windows Keyboard

• Mouse - Wireless

• Monitor - SVGA

**3.1.2 Software Requirements**

• Web Server - Xampp [8.0.18]

• Web Browser - Internet Explorer, Mozilla

• Tools - Dream wear 6.0, PyCharm 3.21

• Technology - Python3.9,PHP

• Database Server - My sql [6.0.21]

**CHAPTER 4**

**SYSTEM SPECIFICATION**

**4.1 Operating System**

In computer software, an operating environment or integrated applications environment is the environment in which users run application software. The environment consists of a user interface provided by an applications manager and usually an application programming interface (API) to the applications manager.

An operating environment is usually not a full operating system, but is a form of middleware that rests between the OS and the application. For example,the first version of Microsoft Windows, Windows 1.0, was not a full operating system, but a GUI laid over DOS albeit with an API of its own. Similarly,the IBM U2 system operates on both Unix/Linux and Windows NT. Usually, theuser interface is text-based or graphical, rather than a commandline interface (e.g., DOS or the Unix shell), which is often the interface of the underlying operating system. In the mid-1980s, text-based and graphical user interface operating environments surrounded DOS operating systems with a shell that turned the user's display into a menu-oriented "desktop" for selectin and running PC applications. These operating environment systems gave users muchof the convenience of software without locking them into a single package.

**Windows 7**

Windows 7 has been redesigned from scratch to run faster, more smoothly, and more responsive. It runs faster with other devices, including media equipment. User changes are simpler, such as changing the desktop or installing printers, and the desktop has been streamlined so there’s less clutter. The number of alerts has been reduced and the user has

more control over which are displayed. Domestic benefits include better home networking, improved access to media, and better photo editing options.

The main elements introduced by Windows 7 are opening programs fasterusing Jump Lists and Pin, navigating open windows more easily using Snap, Peek and Shake, and organizing files and documents in different locations usingLibraries. Enhancements include faster searching, quicker wake up, improved power management, better protection against spyware and malicious software, and improved desktop personalization.

**4.2 Software Description**

The windows XP environment is a boon for the users of DOS. A system is a set of software tools designed to make it easy for people and programs to make optimum use of the computer.

**Python**

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

**PHP**

PHP is the large no of internet service providers (ISPs) and web hosting companies that support it. Today there are hundreds of thousands of developers using PHP, and it’s not surprising that there are so many, considering that several that several million sites are reported to have PHP installed. PHP is a cross-platform technology and once you’ve written your web page, it’s easy to get it up and running on our web server but how does PHP compare with other technologies out there? Well, comparing PHP with Perl is a bit tricky because they were designed for different things in. PHP was specifically designed to rapidly create dynamic web content; Perl was not. As a result, Perl can sometimes be a complicated language that can become prohibitive for users who want to create web pages. Comparing PHP with asp is a more balanced comparison, but then you have to pay for asp, and asp doesn’t work well on a variety of platforms it needs to be used on another proprietary for which you also must pay.

**Java Script**

Today’s website needs to go much beyond HTML. There is a definite need to allow users, browsing through the website to interact with the website. The web must be intelligent enough to accept user input and dynamically structure web page content, tailor-made, to a user’s requirements. Users, who browse through a website today, prefer to choose to view what interests them. Hence even the content of a web page needs to be dynamic based on what a user wishes to see. This requires a website development that will allow the creation of interactive web pages. Capturing user requests is traditionally done via a form. The website development environment should also provide the facility for validating. JavaScript is an object-oriented language that allows the creation of interactive web pages.

The advantages of JavaScript:-

• An interpreted language.

• Embedded within HTML.

• Minimal syntax is easy to learn.

• Performance sign for simple, small programs.

**4.3 Database Description**

**MY SQL**

MYSQL is a freely available RDBMS. Which fully joined the open-source community only recently, when it was released under the GNU public license (GPL). Even before it want free, you didn’t need a license unless you wanted to make money out of it or run the server on the Windows platform. (the windows version of MySQL was shareware). Because you now don’t have to pay a dime to use it, this alone makes MySQL a solid candidate for developing applications.

MySQL, pronounced either "My S-Q-L" or "My Sequel," is an open source relational database management system. It is based on the Structure query Language ( SQL ), which is used for adding, removing, and modifying information in the database.

**CHAPTER 5**

**SYSTEM DESIGN**

**5.1 System Architecture**

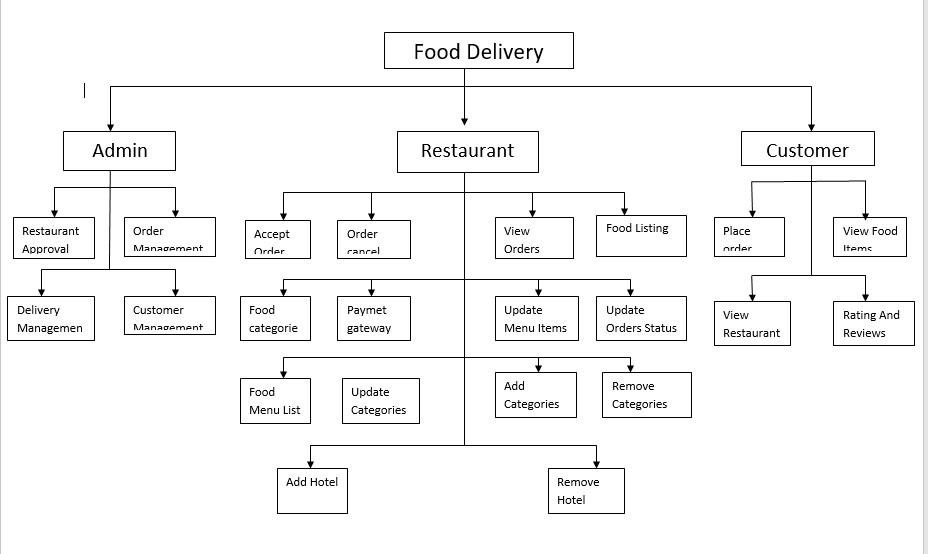


Fig.1

**5.2 Data Flow Diagram**

The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system. The DFD is also called a bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data generated by this system. DFD shows how

The information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output. DFD is also known as a bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail moves from input to output. DFD is also known as a bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

|  |  |  |
| --- | --- | --- |
| **NOTATION** | **COMPONENT** | **DESCRIPTION** |
|  | Process | An oval represents a process or transform that is applied to data or control and changes it in some way. |
|  | External Entity | A rectangle is used to represent an external entity, that is, a system element that produces information for transformation by the software or receives information produced by he software. |
|  | Data Flows | An arrow represents one or more data items or data objects. |
|  | Data Store | The open box represents t h e datastore- stored information that is used by the software. |

5.2.1 Data Flow Diagram –Admin module:

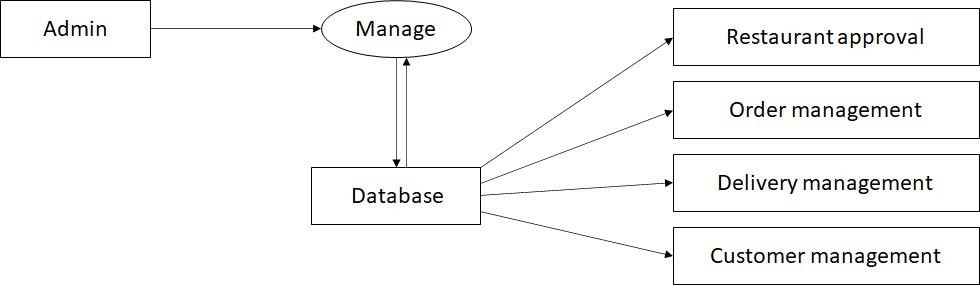


Fig.2

5.2.2 Data Flow Diagram –Restaurant module:

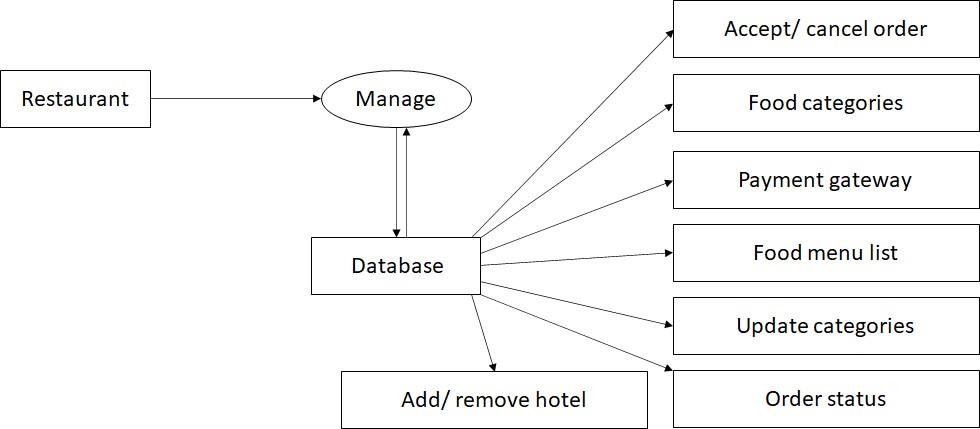


Fig.3

5.2.3 Data Flow Diagram –customer module:

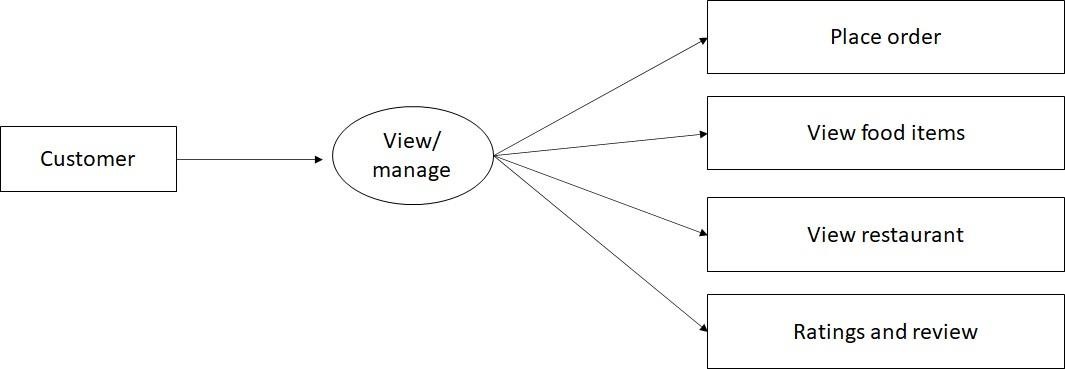


Fig.4

* 1. **Database Design**

**5.31 Order Items**

|  |  |
| --- | --- |
| **Column Name** | **Data Type** |
| order\_item\_id | bigint(20) |
| order\_item\_name | Longtext |
| order\_item\_type | varchar(200) |
| order\_id | bigint(20)g |

**5.3.2 Order Products**

|  |  |
| --- | --- |
| **Column Name** | **Data Type** |
| order\_item\_id | bigint(20) |
| order\_id | bigint(20) |
| product\_id | bigint(20) |
| variation\_id | bigint(20) |
| customer\_id | bigint(20) |
| date\_created | Datetime |
| product\_qty | int(11) |
| product\_net\_revenue | Double |
| product\_gross\_revenue | Double |
| coupon\_amount | Double |
| tax\_amount | Double |
| shipping\_amount | Double |

**5.3.3 Post Code**

|  |  |
| --- | --- |
| **Column Name** | **Data Type** |
| order\_item\_id | bigint(20) |
| order\_item\_name | Longtext |
| order\_item\_type | varchar(200) |
| order\_id | bigint(20) |

**5.3.4 Post Meta**

|  |  |
| --- | --- |
| **Column Name** | **Data Type** |
| meta id | bigint(20) |
| Post\_id | bigint(20) |
| meta\_key | vatchar(256) |
| meta\_value \_ | Longtext |

**5.3.5 Posts**

|  |  |
| --- | --- |
| **Column Name** | **Data Type** |
| Id | bigint(20) |
| post\_author | bigint(20) |
| post\_date | Datetime |
| post\_date\_gmt | Datetime |
| post\_content | Longtext |
| post\_title | Text |
| post\_excerpt | Text |
| post\_status | varchar(20) |
| comment\_status | varchar(20) |
| ping\_status | varchar(20) |
| post\_password | varchar(255) |
| post\_name | varchar(200) |
| to\_ping | Text |

|  |  |
| --- | --- |
| Pinged | Text |
| Post\_modified | Datatime |
| Post\_modified\_gmt | Datatime |
| Post\_content\_filtered | Longtext |
| Post\_parent | bigint(20) |
| Guid | varchar(255) |
| menu\_order | int(11) |
| post\_type | varchar(20) |
| Post\_mime\_type | varchar(100) |
| Comment\_count | bigint(20) |

**5.3.6 User Meta**

|  |  |
| --- | --- |
| **Column Name** | **Data Type** |
| umeta\_id | bigint(20) |
| user\_id | bigint(20) |
| meta\_key | varchar(255) |
| meta\_value | Longtex |

**5.3.7 User**

|  |  |
| --- | --- |
| **Column Name** | **Data Type** |
| ID | bigint(20) |
| user\_login | varchar(60) |
| user\_pass | varchar(255) |
| user\_nicename | varchar(50) |

**5.3.8 Wc\_order\_ststs**

|  |  |
| --- | --- |
| **Column Name** | **Data Type** |
| Order\_id | bigint(20) |
| Parent\_id | bigint(20) |
| date\_created | Datetime |
| date\_created\_gmt | Datetime |
| num\_items\_sold | int(11) |
| total\_sales | Double |
| tax\_total | Double |
| shipping\_tota | Doubl |
| net\_total | Double |
| returniong\_customer | tinyint(1) |
| Status | varchar(200) |
| customer\_id | bigint(20) |

**5.3 System Description**

Users of the web ordering system will interact with the application through a series of simple forms. Each category of food has its own form associated with it which presents a drop down menu for choosing which specific item from the category should be added to the order, and a series of check boxes and radio button click. Users select which category of food they would like to order , and therefore which from should be displayed , by navigating a menu bar, an approach which should be familiar to most users.

Entering delivery and payment deals is done in a similar manner. The user is presented with a form and, must complete the required fields, which include both drop down and text boxes, before checking out and receiving a confirmation number.

Systems design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements.

The online food delivery system has three modules like admin, restaurant and customer. In the admin module, the admin can manage restaurant approval, order management, delivery management and customer management. In the Restaurant module, The organistion can manage accepting and cancelling the order, view the orders, food listing, food categories, update gateway, update menu items, update order status, add update and remove categories and add or remove hotel. In the customer module, they can view food items view restaurant, placing the order and add ratings and reviews.

One of the ways to reduce this expense is to use modern technology to replace some of the jobs done by human beings and make machines do the work. Here we propose an “Online Food Ordering System” that has been designed for Fast Food restaurant, Take-Out college or College Cafeterias. The system can also be used in any food delivery industry. This simplifies the process of food ordering for both the customer and the restaurant, as the entire process of taking orders is automated.

With this method, food is ordered online and delivered to the customer. This is made possible through the use of electronic payment system. Customers pay with their credit cards, although credit card customers can be served even before they make payment either through cash or cheque. So, the system designed in this project will enable customers go online and place order for their food. Due to the great increase in the awareness of internet and the technologies associated with it, several opportunities are coming up on the web. So many businesses and companies now venture into their business with ease because of the internet. One of such business that the internet introduced is an online food ordering system. In today’s age of fast food and take out, many restaurants have chosen to focus on quick preparation and speedy delivery of orders rather than offering a rich dining experience. Until recently, most of this delivery orders were placed over the phone, but there are many disadvantages to this system. It is possible for anybody to order any goods via the internet and have the goods delivered at his/her doorsteps. But while trying to discuss the transfer method of the goods and services, attention is focused on the payment mode Since the world is fast becoming a global village, the necessary tool for this process is communication of which telecommunication is a key player. A major breakthrough is the wireless 2 telephone system which comes in either fixed wireless telephone lines or the Global System of Mobile communication (GSM). What I propose is an online ordering system originally designed for use in college cafeterias, but just as applicable in any food delivery industry. The main advantage of this system is that it greatly simplifies the ordering process for both the customer and the restaurant. The system also greatly lightens the load on the restaurants end, as the entire process of taking orders is automated. Once an order is placed on the webpage that will be designed, it is placed into the database and then retrieved, in pretty much real-time, by a desktop application on the restaurants end. Within this application, all items in the order are displayed, along with their corresponding options and delivery details, in a concise and easy to read manner. This allows the restaurant employees to quickly go through the orders as they are placed and produce the necessary items with minimal delay and confusion. The greatest advantage of this system is its FLEXIBILITY

The simulation first starts with the customer entering his/her credentials (name, ID and password). Once that has been verified, the customer can place an order specifying the quantity of the food required. Now we get a window that displays the order number, customer ID, food name, price and quantity. Once the customer finalizes his/her order, they are redirected to the payment window where the total price is displayed and the customer can select the payment method of their choice and then the customer gets a message of confirmation of order. The block diagram and the ER Diagram of the proposed Online Food Ordering System is given in Figure 1 (a) and (b). The above mentioned simulation flow is with respect to the customer's point of view. Now if you are an admin, you can select the normal login option and enter the admin credentials (email ID and password). Once you enter the admin portal, you get the option of adding food, deleting food or updating food. Any option of choice leads you to the food menu. Once the selected operation is carried out, the end result, i.e, the added food or the updated food list is displayed and if you have deleted a food, that particular food disappears from the main

**CHAPTER 6**

**SYSTEM TESTING**

System Testing is a level of testing that validates the complete and fully integrated software product. The purpose of a system test is to evaluate the end-to-end system specifications. Usually, the software is only one element of a larger computer-based system. Ultimately, the software is interfaced with other software/hardware systems. System Testing is actually a series of different tests whose sole purpose is to exercise the full computer-based systems.

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies, and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail unacceptably. There are various types of tests. Each test type addresses a specific testing requirement.

**Unit testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at the component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

**Integration testing**

Integration tests are designed to test integrated software components to determine if they run as one program. Testing is event-driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfied, as shown by successful unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

**Functional test**

Functional tests provide systematic demonstrations that functions tested areavailable as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centered on the following items:

Valid : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected

.Function : identified functions must be exercised.

Output :identified classes of application outputs must beexercised.

Systems/Procedures : interfacing systems or procedures must be invoked.

The organization and preparation of functional tests are focused on requirements, key functions, or special test cases. In addition, systematic coverage about identifying Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**System Test**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is

the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**White Box Testing**

White Box Testing is a testing in which the software tester knows the inner workings, structure, and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black-box level.

**Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner workings, structure, or language of the module being tested. Black box tests, like most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

**CHAPTER 7**

**IMPLEMENTATION AND RESULT**

Implementation is one of the most important tasks in a project. Implementation is the phase, in which one has to be cautious because all the efforts undertake during this project will be fruitful only if the software is properly implemented according to the plans made. Implementation is the stage in the project where the theoretical design is turned into a working system. The crucial stage is achieving a successful new system and giving the users confidence in that the system will work effectively and efficiently.

**7.1 Implementation**

The Online Food Ordering System has been designed to fill a specific niche in the market by providing small restaurants with the ability to offer their customers an online ordering option without having to invest large amounts of time and money in having custom software designed specifically for them. The system, which is highly customizable, allows the restaurant employees to easily manage the site content, most importantly the menu, themselves through a very intuitive graphical interface.

After developing the system, different testing methods have been used to find bugs in the system and ensure the quality.

Implementation of Online Food Ordering system Using Machine Learning Algorithm is done perfectly in Tunfoods app. It has worked out well as to the expectations.

Python is used as the front-end and MySQL is used as the back-end for storing data.

**Algorithm**

TSP solution Heuristic Algorithm for Routing Optimization Heuristics algorithm is based on the model of undirected weighted graph. The constructive heuristics algorithm is applied in the developed system and uses nearest –neighbor (NM) approach. The algorithm that is used to solve TSP has several iterations, which are [9,10]:

1. Salesman is in initial point as the current vertex.

2. Search the edge that has the least weight between the current vertex and the unvisited vertex V.

3. V is set as the current vertex.

4. Mark V as visited vertex.

5. Repeat step 2.

6. If all vertices have been visited, stop the iteration and go back to initial point. The output of this algorithm is the sequence of steps as the

7.2 Result

7.2.1 Source Code

models.py:

from django.db import models

from django.contrib.auth.models import AbstractUser from django.conf import settings

from django.db.models.signals import post\_save

class User(AbstractUser):

is\_customer = models.BooleanField(default=False) is\_restaurant

= models.BooleanField(default=False)

class Customer(models.Model): user=models.OneToOneField(settings.AUTH\_USER\_MODEL,on\_delete=models.

CASCADE)

f\_name = models.CharField(max\_length=20,blank=False)

l\_name = models.CharField(max\_length=20,blank=False)

city = models.CharField(max\_length=40,blank=False)

phone = models.CharField(max\_length=10,blank=False)

address = models.TextField()

def \_str\_(self):

return self.user.username

class Restaurant(models.Model):

user = models.OneToOneField(settings.AUTH\_USER\_MODEL,on\_delete=models.CASC

ADE) rname = models.CharField(max\_length=100,blank=False) info =

models.CharField(max\_length=40,blank=False) min\_ord =

models.CharField(max\_length=5,blank=False) location =

models.CharField(max\_length=40,blank=False)

r\_logo = models.FileField(blank=False)

REST\_STATE\_OPEN = "Open" REST\_STATE\_CLOSE = "Closed" REST\_STATE\_CHOICES =(

(REST\_STATE\_OPEN,REST\_STATE\_OPEN), (REST\_STATE\_CLOSE,REST\_STATE\_CLOSE)

)

status =

models.CharField(max\_length=50,choices=REST\_STATE\_CHOICES,default=REST\_

ST

ATE\_OPEN,blank=False)

approved = models.BooleanField(blank=False,default=True)

def \_str\_(self): return self.rname

class Item(models.Model):

id = models.AutoField(primary\_key=True)

fname = models.CharField(max\_length=30,blank=False) category = models.CharField(max\_length=50,blank=False)

def \_str\_(self):

return self.fname

class Menu(models.Model):

id = models.AutoField(primary\_key=True)

item\_id = models.ForeignKey(Item,on\_delete=models.CASCADE) r\_id

= models.ForeignKey(Restaurant,on\_delete=models.CASCADE) price

= models.IntegerField(blank=False)

quantity = models.IntegerField(blank=False,default=0)

def \_str\_(self):

return self.item\_id.fname+' - '+str(self.price)

class Order(models.Model):

id = models.AutoField(primary\_key=True) total\_amount = models.IntegerField(default=0) timestamp = models.DateTimeField(auto\_now\_add=True) delivery\_addr = models.CharField(max\_length=50,blank=True) orderedBy =

models.ForeignKey(User ,on\_delete=models.CASCADE) r\_id = models.ForeignKey(Restaurant

,on\_delete=models.CASCADE)

ORDER\_STATE\_WAITING = "Waiting" ORDER\_STATE\_PLACED = "Placed"

ORDER\_STATE\_ACKNOWLEDGED = "Acknowledged" ORDER\_STATE\_COMPLETED = "Completed" ORDER\_STATE\_CANCELLED = "Cancelled"

ORDER\_STATE\_DISPATCHED = "Dispatched" ORDER\_STATE\_CHOICES = (

(ORDER\_STATE\_WAITING,ORDER\_STATE\_WAITING), (ORDER\_STATE\_PLACED, ORDER\_STATE\_PLACED),

(ORDER\_STATE\_ACKNOWLEDGED, ORDER\_STATE\_ACKNOWLEDGED),

(ORDER\_STATE\_COMPLETED, ORDER\_STATE\_COMPLETED), (ORDER\_STATE\_CANCELLED, ORDER\_STATE\_CANCELLED), (ORDER\_STATE\_DISPATCHED, ORDER\_STATE\_DISPATCHED)

status=models.CharField(max\_length=50,choices=ORDER\_STATE\_CHOICES,de fault=ORDER\_STATE\_WAITING)

def \_str\_(self):

return str(self.id) +' '+self.status class orderItem(models.Model):

id = models.AutoField(primary\_key=True)

item\_id = models.ForeignKey(Menu ,on\_delete=models.CASCADE) ord\_id = models.ForeignKey(Order,on\_delete=models.CASCADE)

quantity = models.IntegerField(default=0)

def \_str\_(self):

return str(self.id) views.py:

from django.shortcuts import render, redirect,get\_object\_or\_404 from django.contrib.auth import authenticate, login,logout from

.forms import CustomerSignUpForm,RestuarantSignUpForm,CustomerForm,Restuarant Form from django.contrib.auth.decorators import login\_required from collections import Counter from django.urls import reverse from django.db.models import Q

from .models import Customer,Restaurant,Item,Menu,Order,orderItem,User

#### ---------- General Side #####

# Showing index page def index(request):

return render(request,'webapp/index.html',{})

def orderplaced(request):

return render(request,'webapp/orderplaced.html',{}) # Showing Restaurants list to Customer def

restuarent(request):

r\_object = Restaurant.objects.all() query = request.GET.get('q') if query:

r\_object=Restaurant.objects.filter(Q(rname\_\_icontains=query)).distinct() return render(request,'webapp/restaurents.html',{'r\_object':r\_object})

return render(request,'webapp/restaurents.html',{'r\_object':r\_object})

# logout

def Logout(request):

if request.user.is\_restaurant: logout(request)

return redirect("rlogin")

else: logout(request)

return redirect("login")

#### Customer Side ######

# Creating Customer Account def customerRegister(request): form =CustomerSignUpForm(request.POST or None)

if form.is\_valid():

user = form.save(commit=False) username = form.cleaned\_data['username'] password = form.cleaned\_data['password'] user.is\_customer=True user.set\_password(password)

user.save()

user = authenticate(username=username,password=password) if user is not None:

if user.is\_active:

login(request,user) return redirect("ccreate")

context ={

'form':form

}

return render(request,'webapp/signup.html',context)

# Customer Login

def customerLogin(request):

if request.method=="POST":

username = request.POST['username'] password = request.POST['password'] user = authenticate(username=username,password=password)

if user is not None:

if user.is\_active:

login(request,user) return redirect("profile")

else:

return

render(request,'webapp/login.html',{'error\_message':'Your account disable'}) else:

return render(request,'webapp/login.html',{'error\_message': 'Invalid

Login'})

return render(request,'webapp/login.html') # customer profile view def

customerProfile(request,pk=None): if pk:

user = User.objects.get(pk=pk)

else

:

user=request.user

return render(request,'webapp/profile.html',{'user':user})

#Create customer profile

def createCustomer(request):

form = CustomerForm(request.POST or None) if form.is\_valid():

instance = form.save(commit=False) instance.user = request.user instance.save() return redirect("profile") context={ 'form':form,

'title':"Complete Your profile"

}

return render(request,'webapp/profile\_form.html',context)

# Update customer detail

def updateCustomer(request,id):

form = CustomerForm(request.POST or None,instance=request.user.customer) if form.is\_valid():

form.save()

return redirect('profile') context={ 'form':form,

'title':"Update Your profile"

}

return render(request,'webapp/profile\_form.html',context)

def restuarantMenu(request,pk=None):

menu = Menu.objects.filter(r\_id=pk) rest = Restaurant.objects.filter(id=pk) items =[]

for i in menu:

item = Item.objects.filter(fname=i.item\_id) for content in item:

temp=[] temp.append(content.fname) temp.append(content.category) temp.append(i.price) temp.append(i.id) temp.append(rest[0].status)

temp.append(i.quantity) items.append(temp)

context = {

'items' : items, 'rid' : pk,

'rname' : rest[0].rname,

'rmin' : rest[0].min\_ord,

'rinfo' : rest[0].info, 'rlocation':rest[0].location,

} return render(request,'webapp/menu.html',context) @login\_required(login\_url='/login/user/') def checkout(request):

if request.POST:

addr = request.POST['address'] ordid

= request.POST['oid'] Order.objects.filter(id=int(ordid)).update(delivery\_addr = addr,

status=Order.ORDER\_STATE\_PLACED)

return redirect('/orderplaced/')

else:

cart = request.COOKIES['cart'].split(",") cart =

dict(Counter(cart)) items = [] totalprice = 0 uid

= User.objects.filter(username=request.user) oid = Order() oid.orderedBy = uid[0] for x,y in cart.items():

item = [] it = Menu.objects.filter(id=int(x)) if len(it):

oiid=orderItem() oiid.item\_id=it[0] oiid.quantity=int(y) oid.r\_id=it[0].r\_id oid.save() oiid.ord\_id =oid oiid.save() totalprice += int(y)\*it[0].price item.append(it[0].item\_id.fnam e) it[0].quantity = it[0].quantity

- y it[0].save() item.append(y) item.append(it[0].price\*int(y))

items.append(item) oid.total\_amount=totalprice oid.save()

context={

"items":items,

"totalprice":totalprice, "oid":oid.id

}

return render(request,'webapp/order.html',context)

####### Restaurant Side #####

# creating restuarant account def restRegister(request): form

=RestuarantSignUpForm(request.POST or None) if form.is\_valid():

user = form.save(commit=False) username

= form.cleaned\_data['username'] password

= form.cleaned\_data['password'] user.is\_restaurant=True user.set\_password(password)

user.save()ser = authenticate(username=username,password=password) if user is not None:

if user.is\_active:

login(request,user) return redirect("rcreate")

context ={

'form':form

}

return render(request,'webapp/restsignup.html',context)

# restuarant login def restLogin(request):

if request.method=="POST":

username = request.POST['username'] password = request.POST['password'] user = authenticate(username=username,password=password)

if user is not None:

if user.is\_active:

login(request,user)

return redirect("rprofile")

else

:

return

render(request,'webapp/restlogin.html',{'error\_message':'Your account disable'}) else:

return render(request,'webapp/restlogin.html',{'error\_message':

'Invalid Login'})

return render(request,'webapp/restlogin.html')

# restaurant profile view

def restaurantProfile(request,pk=None): if pk:

user = User.objects.get(pk=pk)

else:

user=request.user

return render(request,'webapp/rest\_profile.html',{'user':user})

# create restaurant detail @login\_required(login\_url='/login/restaurant/') def createRestaurant(request):

form=RestuarantForm(request.POST or None,request.FILES or None) if form.is\_valid():

instance = form.save(commit=False) instance.user = request.user instance.save() return redirect("rprofile") context={ 'form':form,

) from exc execute\_from\_command\_line(sys.argv)

index page:

<!DOCTYPE html>

<html>

<head>

{% load staticfiles %}

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">

<title>{% block title %}FoodApp{% endblock title %}</title>

<link rel="stylesheet" type="text/css" href="{% static 'webapp/css/bootstrap.min.css'%}">

<link href="/static/webapp/css/style.css" rel="stylesheet" type="text/css" media="all" />

<link rel='stylesheet' href='https://maxcdn.bootstrapcdn.com/font- awesome/4.5.0/css/font-awesome.min.css' >

<link href='http://fonts.googleapis.com/css?family=Open+Sans:400,300,700' rel='stylesheet' type='text/css'>

<link href='http://fonts.googleapis.com/css?family=Lobster+Two:400,400italic,700,700italic' rel='stylesheet' type='text/css'>

<!--Animation-->

<button class="navbar-toggler" type="button" data-toggle="collapse" data- target=".dual-collapse2">

<span class="navbar-toggler-icon"></span>

</button>

</div>

<div class="navbar-collapse collapse w-100 dual-collapse2 order-2 order-md-2">

<ul class="navbar-nav mr-auto text-center">

<li class="nav-item">

<a class="nav-link text-light" href="{% url 'rlogin'%}"> &nbsp;<strong>For Restaurants</strong> </a>

</li>

</ul>

</div>

</div>

</nav>

<div class="jumbotron big-banner" style="height: 500px; width: 100%; padding- top:100px;" >

<div class="container">

<div class="row">

<div class="col">

<div class="col4 text-light ">

<h1 style="font-style:italic; font-family:sans-serif; "><strong>Feeling Hungry</strong></h1>

<h2><strong>Register Now<br/>and<br/>Order

Online</strong></h2>

</div>

<br>

<button type="button" class="btn btn-success btn-lg"><strong><a href="{% url 'register'%}" class="text-light">Register</a></strong></button>

</div>

<div class="col">

<div class="col4 text-light">

<h1 style="font-style:italic; font-family:sans-serif; "><strong>Be a Partner</strong></h1>

<h2><strong>Join Us Today<br/>and<br>Increase Your Sale</strong></h2>

</div>

<br>

<button type="button" class="btn btn-success btn-lg"><strong><a href="{% url 'rregister'%}" class="text-light">Join Us</a></strong></button>

</div>

</div>

</div>

</div>

</div>

<div class="content">

<div class="ordering-section" id="Order">

<div class="container">

<div class="ordering-section-head text-center wow bounceInRight" data-wow-delay="0.4s">

<h3>Ordering food was never so easy</h3>

<div class="dotted-line">

<h4>Just 4 steps to follow </h4>

</div>

</div>

<div class="i-am-center">

<div class="row ">

<div class="col-sm-2 col-md-2">

<div class="thumbnail borders text-light">

<p>Choose <span>Your

Restaurant</span></p>

</div>

</div>

<div class="col-sm-2 col-md-2 ">

<div class="thumbnail borders text-light">

<p>Order <span>Your

Cuisine</span></p>

</div>

</div>

<div class="col-sm-2 col-md-2">

<div class="thumbnail borders text-light" >

<p>Pay <span> online </span></p>

<label></label>

</div>

</div>

<div class="col-sm-2 col-md-2">

<div class="thumbnail borders text-light">

<p>Enjoy <span>your food </span></p>

<label></label>

</div>

</div>

<div class="clearfix"></div>

</div>

</div>

</div>

</div>

</div>

<div class="container col-md-9">

<div class="top-restaurent-head">

<h3>Top Restaurants</h3>

</div>

<div class="row">

<div class="col-md-4 ">

<div class="thumbnail">

<a href="">

<img src="/static/webapp/images/restaurent-2.jpg" class="img-responsive"

alt="" />

</a>

</div>

</div>

<br>

<div class="col-md-4 ">

<div class="thumbnail">

<a href="">

<img src="/static/webapp/images/restaurent-3.jpg" class="img-responsive"

alt="" />

</a>

</div>

</div>

<br>

<div class="col-md-4 ">

<div class="thumbnail">

<a href="">

<img src="/static/webapp/images/restaurent-4.jpg" class="img-responsive"

alt="" />

</a>

</div>

</div>

<br>

<div class="col-md-4 ">

<div class="thumbnail">

<a href="">

<img src="/static/webapp/images/restaurent-5.jpg" class="img-responsive"

alt="" />

</a>

</div>

</div>

<br>

<div class="col-md-4 ">

<div class="thumbnail">

<a href="">

<img src="/static/webapp/images/restaurent-6.jpg" class="img-responsive"

alt="" />

</a>

</div>

</div>

</div>

</div>

<br>

<br>

<div class="container col-md-9">

<div class="top-cuisine-head">

<h3>Top Cuisines</h3>

</div>

<div class="row">

<div class="col-md-4 ">

<div class="thumbnail">

<a href="">

<img src="/static/webapp/images/cuisine1.jpg" class="img-responsive"

alt="" />

</a>

</div>

<div class="caption border col-sm-4">

<h5 class="text-secondary"><strong>Pizza</strong></h5>

</div>

</div>

<div class="col-md-4 mb-3">

<div class="thumbnail">

<a href="">

<img src="/static/webapp/images/cuisine2.jpg" class="img-responsive"

alt="" />

</a>

</div>

<div class="caption border col-sm-4">

<h5 class="text-secondary"><strong>Burger</strong></h5>

</div>

</div>

<div class="col-md-4 ">

<div class="thumbnail">

<a href="">

<img src="/static/webapp/images/cuisine3.jpg" class="img-responsive"

alt="" />

</a>

</div>

<div class="caption border col-sm-6">

<h5 class="text-secondary"><strong>Dough Nut</strong></h5>

</div>

</div>

<div class="col-md-4 mb-3">

<div class="thumbnail">

<a href="">

<img src="/static/webapp/images/cuisine4.jpg" class="img-responsive"

</a>

</div>

<div class="caption border col-sm-4">

<h5 class="text-secondary"><strong>Curry</strong></h5>

</div>

</div>

<div class="col-md-4 ">

<div class="thumbnail">

<a href="">

<img src="/static/webapp/images/cuisine5.jpg" class="img-responsive"

alt="" />

</a>

</div>

<div class="caption border col-sm-4">

<h5 class="text-secondary"><strong>trips</strong></h5>

</div>

</div>

<div class="col-md-4 ">

<div class="thumbnail">

alt="" /><a href="">

<img src="/static/webapp/images/cuisine6.jpg" class="img-responsive"

</a>

</div>

<div class="caption border col-md-6">

<h5 class="text-secondary"><strong>Frawn Rice</strong></h5>

</div>

</div>

<div class="col-md-4 ">

<div class="thumbnail">

<a href="">

<img src="/static/webapp/images/cuisine7.jpg" class="img-responsive"

alt="" />

</a>

</div>

<div class="caption border col-md-4">

<h5 class="text-secondary"><strong>Fingers</strong></h5>

</div>

</div>

<div class="col-md-6 ">

alt="" />

<div class="thumbnail">

<img src="/static/webapp/images/cuisine8.jpg" class="img-responsive"

<div class="caption border col-md-3">

<h5 class="text-secondary"><strong>Cuisine</strong></h5>

</div>

</div>

</div>

<div class="clearfix"></div>

</div>

</div>

<br>

<br>

<br>

<div class="footer">

<div class="container">

<div class="footer-copyright text-center py-3">© 2018 Copyright:

<a href="#"> MohdAsif</a>

</div>

</div>

<script type="text/javascript" src="{%static 'webapp/js/jquery.min.js'%}">

</script>

<script type="text/javascript" src="{%static 'webapp/js/bootstrap.min.js'%}"></script>

<script> new WOW().init();

</script>

<script src="/static/webapp/js/simpleCart.min.js"> </script>

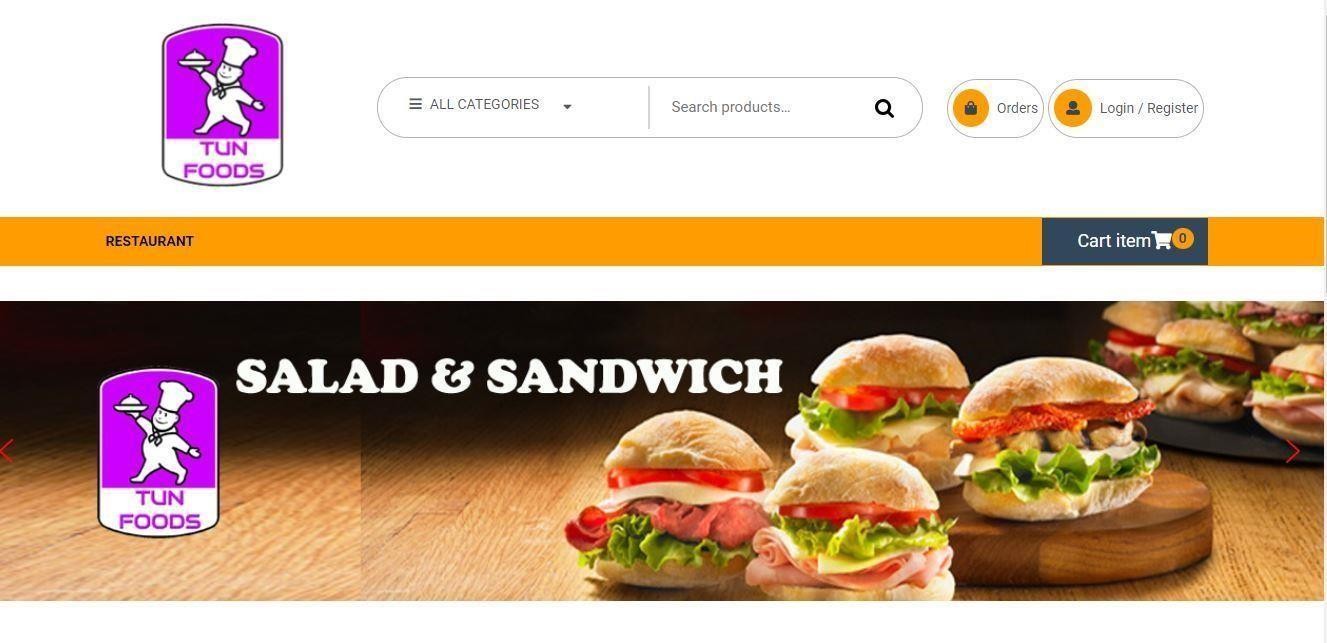
<script type="text/javascript" src="/static/webapp/js/move-top.js"></script>

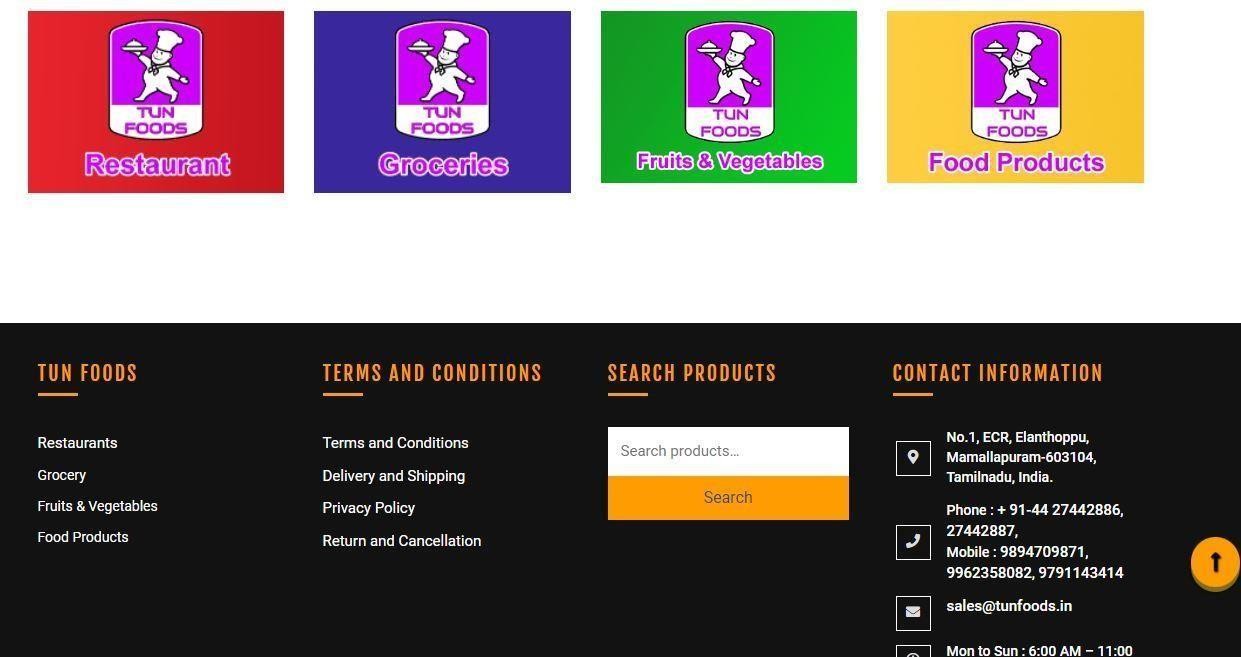
<script type="text/javascript" src="/static/webapp/js/easing.js"></script>

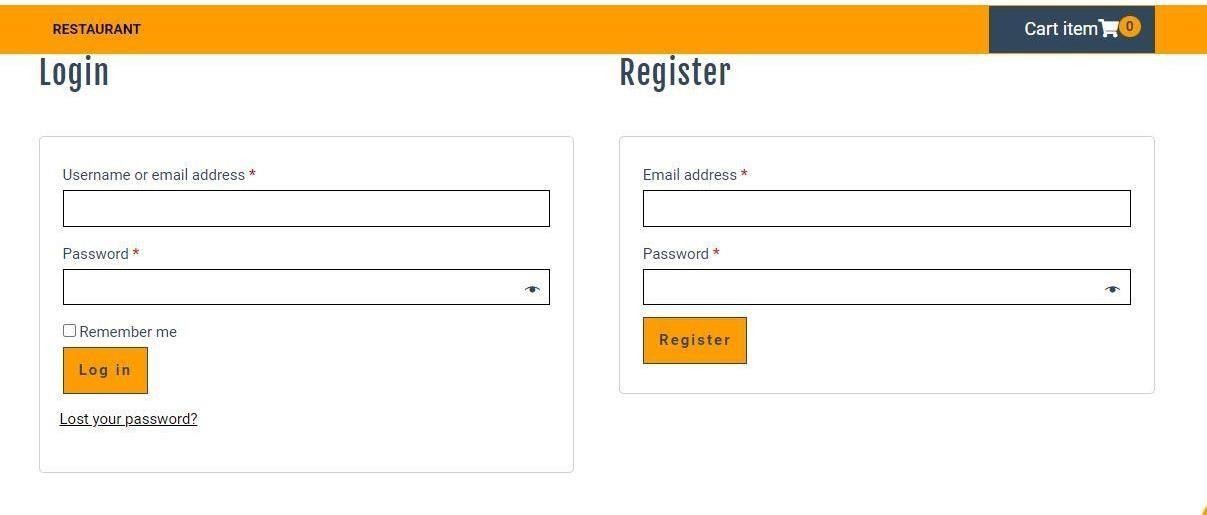
</body>

</html>

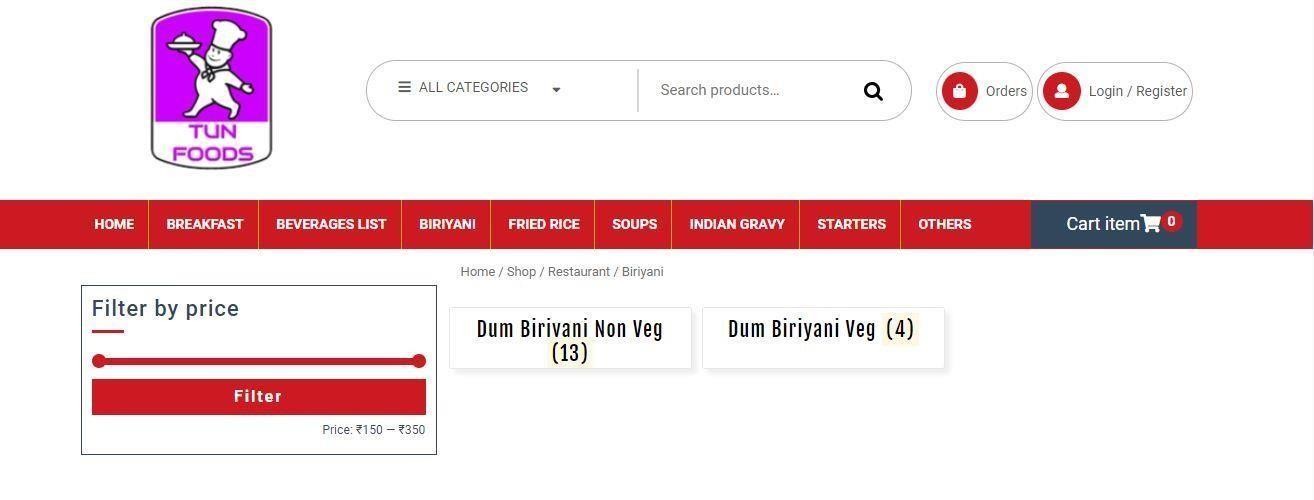
7.2.2 Screenshots:

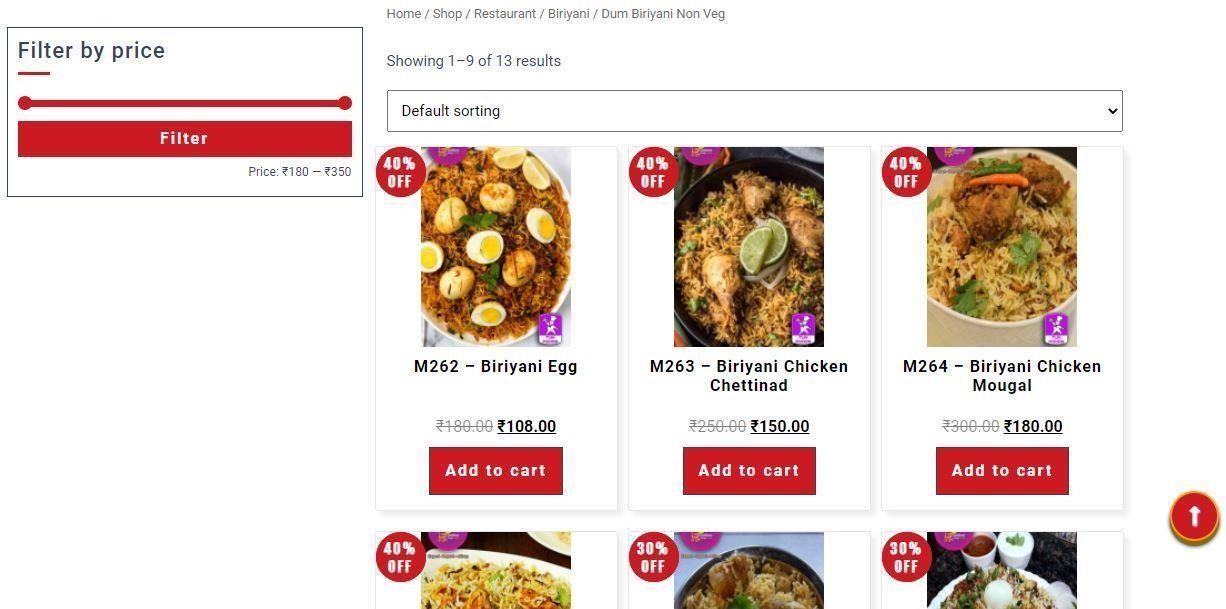


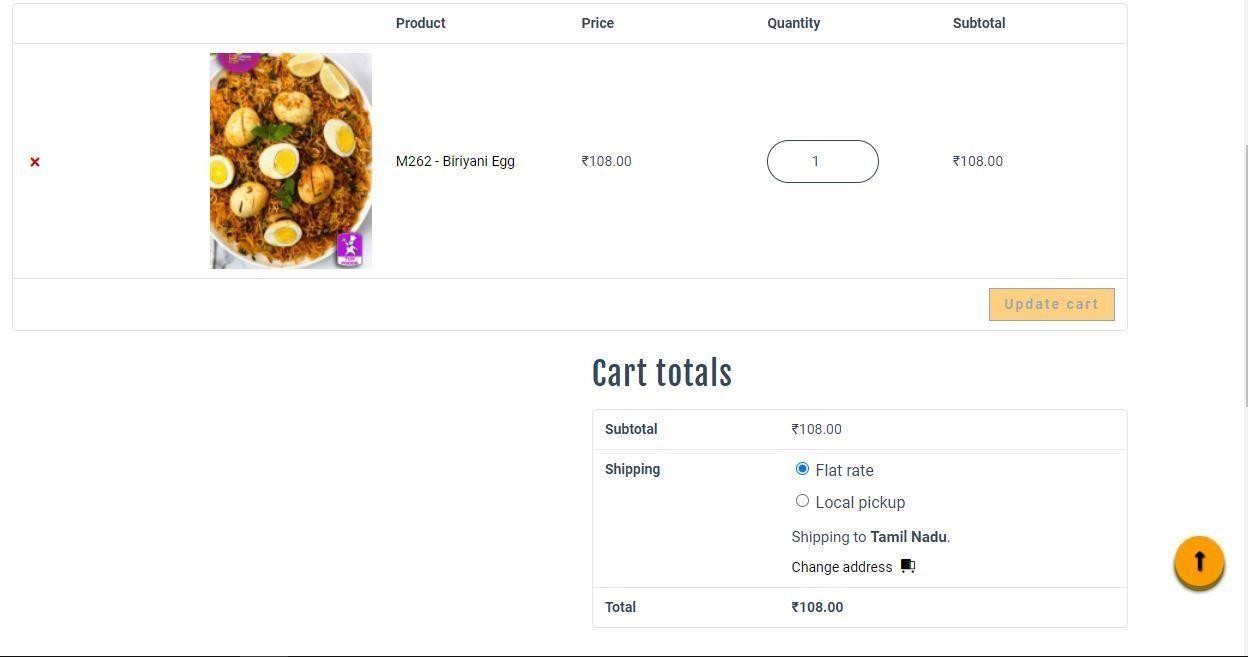


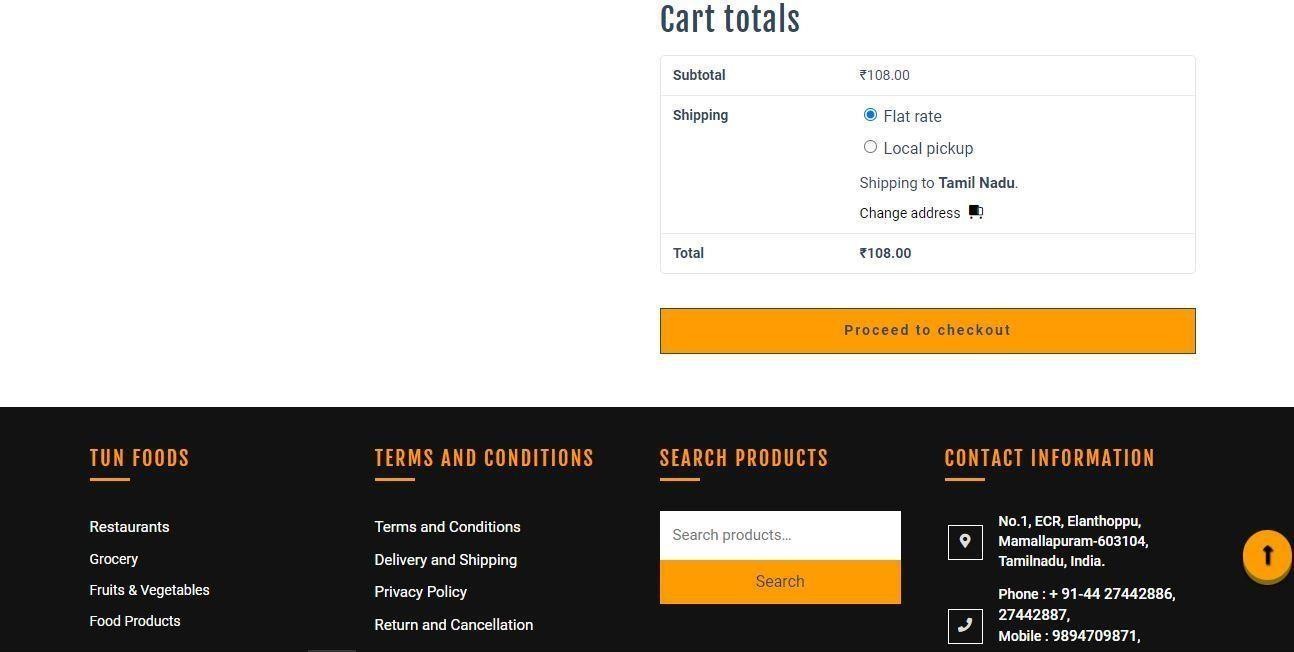
 

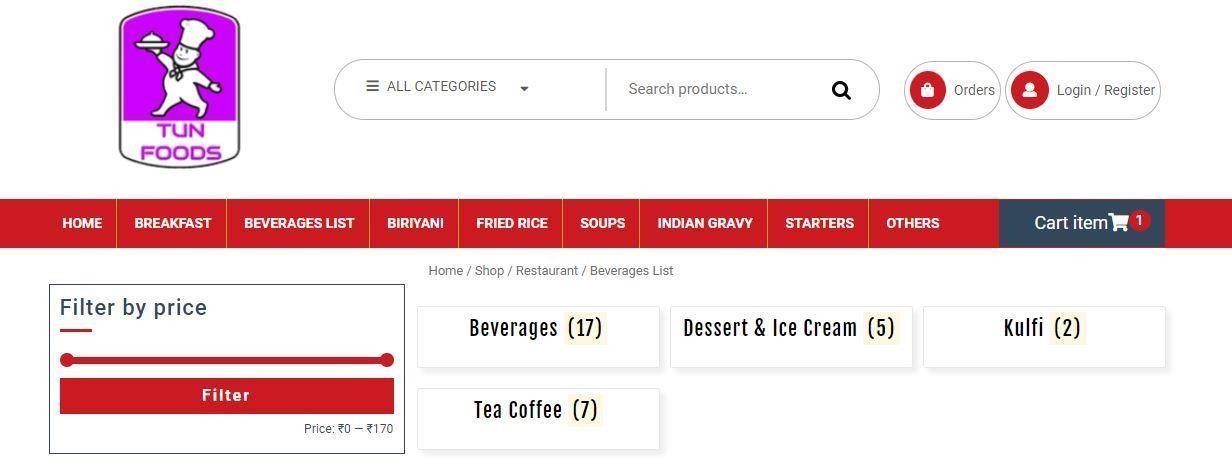


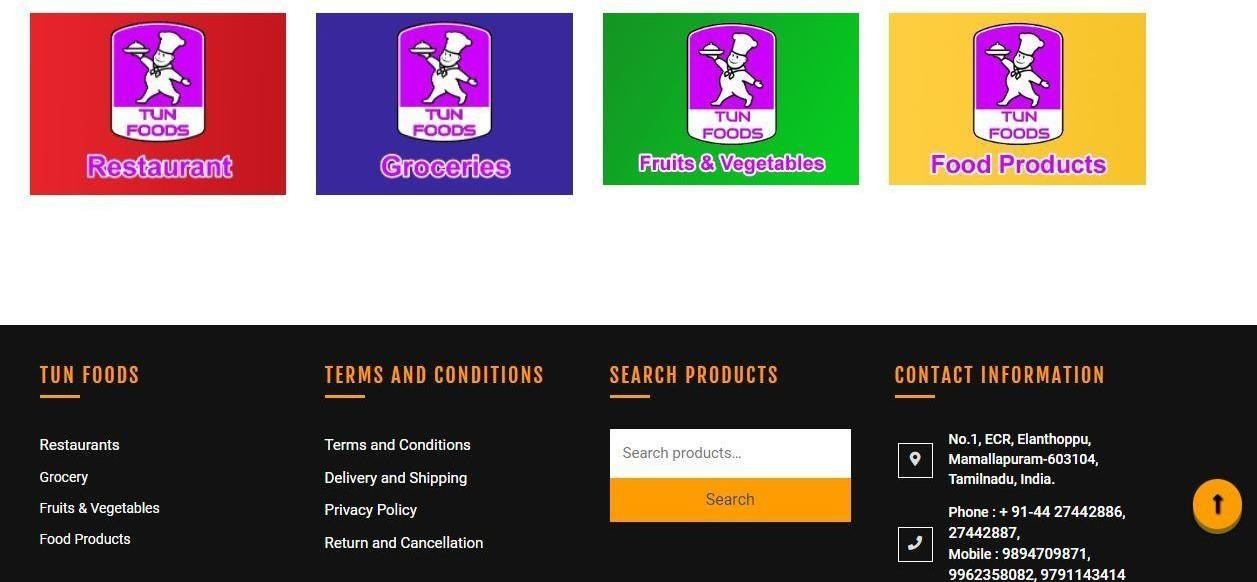


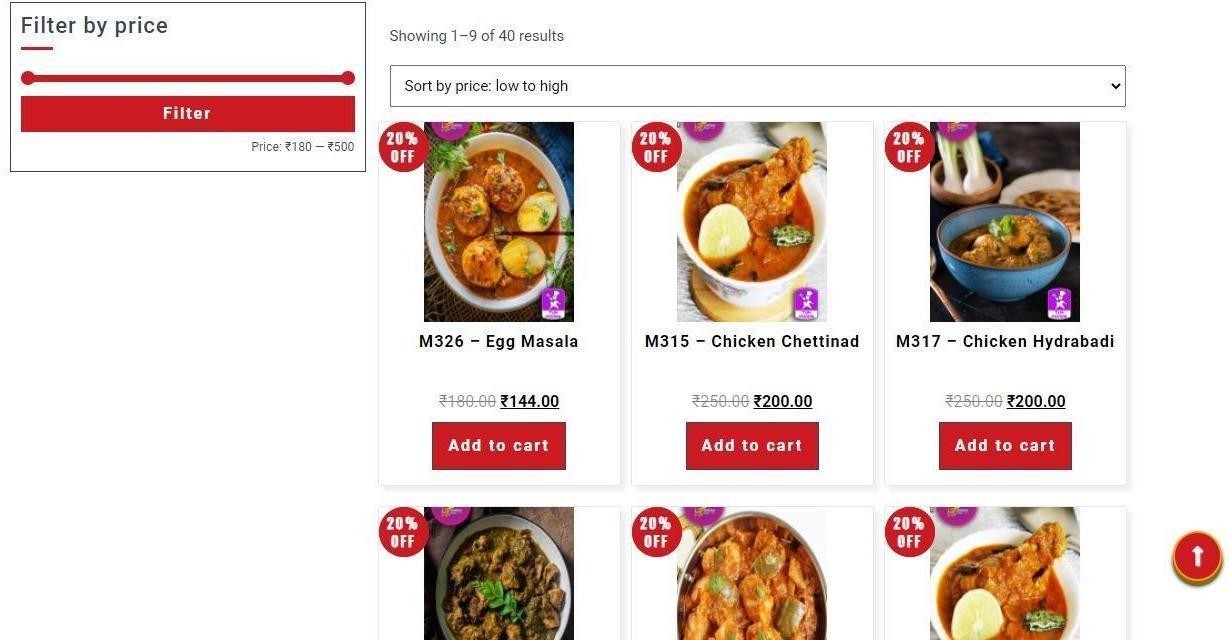
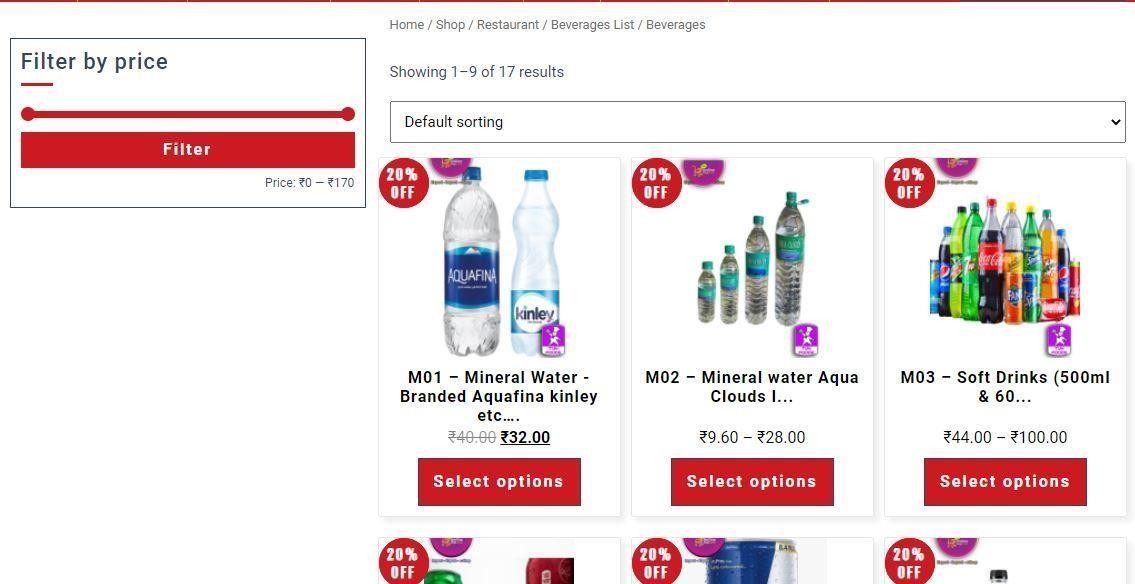












**CHAPTER 8**

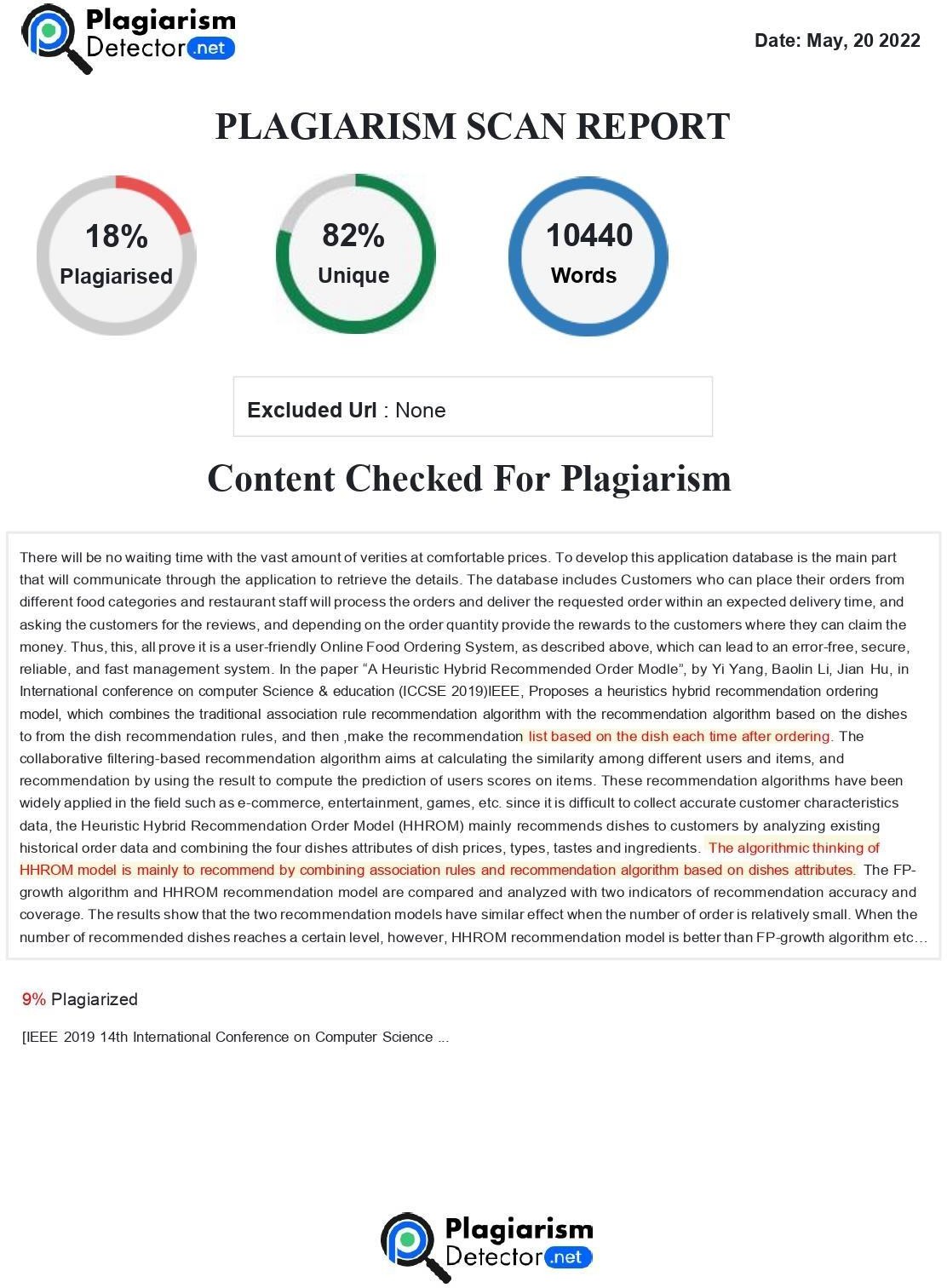
**CONCLUSION**

An online food ordering system with live messenger is a web-based system for restaurant, which automates food ordering system. If is a secure, user-friendly food ordering Management System. It helps the management to manage the online orders and view the status. The management can add menus and take orders with the system. The system also has a simple mobile-friendly user interface which can be used through different types of devices and screens. This System is completely secure since every user is provided with user ID and Password so there is no chance of any unauthorized access. Online Payment, Registration and cancellation make it easier to use. So, using this system will help in reducing the labor and provide more facility for Customer to like the services.

**CHAPTER 9**

**FUTURE ENHANCEMENT**

Future Enhancement With the increasing competition, evolving technology trends, and consumer needs, online food Ordering system can update their system and adopt unique trends in order to meet requirements. Ordering Through New Channels. It means owners have to update their existing platform and allow users to place orders via social media, virtual assistants, smart devices, and so on. Food Ordering Via Tweet. Food Ordering Via Smartwatch. With the help of Big Data technology, food delivery companies can accurately estimate the above data and immediately respond to customers’ query, if it arises.



**REFERENCES**

1. “ Netfood: A Software Systems for Food ordering and Delivery”, by CristinaEdina Domokos, Barna Sera, Karoly Simon, Lajos kovacs, Tas -Bela

Szakacs in International Symposium on intelligent system and Informatics, IEEE,2018,

2. “A Heuristic Hybrid Recommended Order Modle”, by Yi Yang, Baolin Li, Jian Hu, in International conference on computer Science & education(ICCSE 2019)IEEE,

3. “A Two-stage Algorithm for Fuzzy Online Order Dispatching Problem” ,by Jie Zheng,Shengyao Wang, Ling Wang ,Jing-fang chen,Li wang, Jinghua Hao, Renqing He, Zhizhao Sun, IEEE

4. “Android Application Food Delivery Services”,by Lidya Chitra Laoh,Timothy Adithia pongantung, Carolin Mulalinda,IEEE

5. “Stimulus Factors of Order online Food Delivery”, by Yakob Utama Chandra, Cadelina Cassandra, in International Conference on information Management and Technology (ICIMTech) IEEE, 2019,

6. paper “Two Fast Heuristics for Online Order Dispatching”, Qingte Zhou, Huanya Zheng, Shengyaowang,Jinghua hao, Renqing He, Zhizhao, in IEEE Xplore, 2020

7. “Online Food Delivery Platforms and Restaurants Interactions in the Context of the Ban on Using Single-use Plastics”, KEQAN LI, YAN CHEN, JIACHEN LIU, LIN ZHANG, XIANGWEI MU in, IEEE ACCESS, 2021,

8. “Food Delivery Automation in Restaurants Using Collaborative Robotics”, Albin Antony, Sivraj.P, International Conference on Inventive in Computing Applications IEEE,2018,

• https://www.tutorialspoint.com/index.htm

• https://www.javatpoint.com

• https://www.w3schools.com

• https://html.com