MINI PROJECT REPORT

on

Smart Order System using Web Application

Submitted in partial fulfilment of requirements to

CB 352 Mini Project

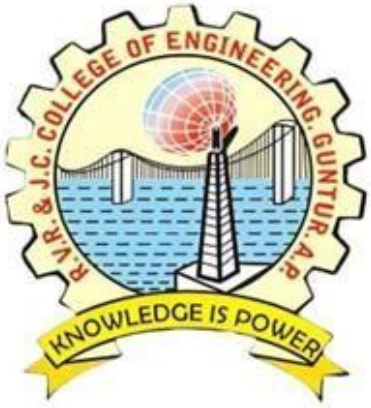
III/IV B. Tech CSBS (V Semester)

By

Y21CB024 – B. KARTHIK

Y21CB031 – M. LAKSHMI PUJITHA

L22CB069 – G. NAVEEN



December 2023

Department of Computer Science and Business System

R.V.R & J.C. COLLEGE OF ENGINEERING

(AUTONOMOUS)

(Approved by A.I.C.T.E) NAAC 'A+' Grade.

(Affiliated to Acharya Nagarjuna University)

Chandramouli Puram, Chowdavaram.

GUNTUR – 522 019

R.V.R & J.C. COLLEGE OF ENGINEERING

DEPARTMENT OF

COMPUTER SCIENCE AND BUSINESS SYSTEM

BONAFIDE CERTIFICATE

This is to certify that this Mini Project work entitled **"Smart Order System using Web Application"** is the Bonafide work of **B.Karthik (Y21CB024), M.Lakshmi Pujitha (Y21CB031), G.Naveen (L22CB069)** of III/IV B. Tech who carried the work under my supervision and submitted in the partial fulfilment of the requirements to **CB352 – MINI PROJECT LAB REPORT** during the year **2023-2024**.

Dr. P. Lakshmi Kanth	**Mr. P. Srinivasa Rao**	**Dr. M. V. P. Chandra Sekhara Rao**
(Project Guide)	(Project In charge)	(Prof. & Head, Dept. of CSBS)

ABSTRACT

Smart Order System (SOS) is a system which will help restaurant to optimize and control over their restaurants. It makes life easier for the waiters because they don't have to go kitchen and give orders to chef easily. The management point of view, the admin will be able to control the restaurant by having all the reports to hand and able to see the records of each employee and orders. The website helps the restaurants to do all functionalities more accurately and enhances the spending of all the tasks. SOS reduces manual work and improves efficiency of restaurants. The SOS sets up menu online and the customers easily places the order from their respective table with one click and with a food menu online you can easily take the orders, maintain customer's database, and improve your food order service. The system allows the user to select the desired food items from the displayed menu. The user orders the food items. The payment can be made through the pay on-delivery system. The user's details are maintained confidential because it maintains a separate account for each user. An ID and password are provided for each user. It provides more secured ordering.

ACKNOWLEDGEMENT

From the idea to the act, from the conception to reality, from the emotion to the response, from the desire to the spasm, we are led by those about whom to write all words seem meek.

We are very much thankful to **Dr. Kolla. Srinivas, Principal of R.V.R. & J.C College of Engineering, Guntur,** for allowing us to work on this project.

We express our sincere thanks to **Dr. M.V.P. Chandra Sekhara Rao, Professor and Head, Department of Computer Science and Business Systems** for encouraging and supporting us to carry out this project successfully.

We are very glad to express our special thanks to **Dr. Lakshmi Kanth Paleti, Associate Professor and Mentor** who has inspired us to select this project according to our choice and for her valuable advice to work on this project.

This Mini Project wouldn't be completed without the help of my friends, family members and other people who are directly or indirectly connected with this work. I also express my sincere thanks to the Technical and Non-Technical staff and all the faculty of the department for their valuable help.

B. Karthik (Y21CB024)
M. Lakshmi Pujitha(Y21CB031)
G. Naveen (L22CB069)

CONTENTS

LIST OF FIGURES

1.INTRODUCTION

SOS is a new generation of restaurant management software. When users/customer will enter on the website, he/she should have an account. User does not have an account, user has to create a new account to order food. Create a new account user should enter unique username, email, and new mobile no. with password. User fill his/her table for food delivery. Once user enters on the website, you can see different types of food available in restaurant. First select category of food from soups, starters, the main course dishes and desserts and after that search food as your interest. User can select food what he/she want to order and after selecting all your meal place your order and confirm your table. The website will saw your various type of payment methods and your total bill amount. User can pay cash on delivery and choose best deal for meal.

3.1. Tools/Technologies Used

Technologies:

1. HTML
2. CSS
3. JavaScript
4. Python
5. Django

Tools:

1. Git
2. Visual Studio Code

2.LITERATURE SURVEY

In today's fast-paced world, convenience and efficiency are highly valued by consumers. This is especially true in the restaurant industry, where customers are increasingly looking for ways to streamline their dining experience. SOS offer a promising solution by allowing customers to place orders from their tables using tablets or smartphones. This not only saves time but also improves customer satisfaction and reduces order errors.

Several studies have investigated the benefits of SOS for restaurants. A study by found that restaurants that implemented SOS experienced a 15% increase in average order value and a 20% increase in table turnover rate. Another study by found that SOSs led to a 50% reduction in order errors and a 30% decrease in customer wait times.

In addition to these quantitative benefits, SOS also offer several qualitative benefits for restaurants. These include:

- Improved customer experience: SOS allow customers to place orders at their own pace and without having to wait for a server. This can make the dining experience more enjoyable and relaxing.
- Increased staff efficiency: SOS can free up servers to focus on providing better customer service. This can lead to increased customer satisfaction and loyalty.
- Reduced operating costs: SOS can help to reduce labor costs by eliminating the need for servers to take orders and input them into a POS system.

When implementing a SOS, it is important to consider several factors, including:

- User interface: The user interface of the SOS should be easy to use and navigate. It should also be visually appealing and consistent with the restaurant's branding.
- Technology: The SOS should be compatible with the restaurant's existing POS system. It should also be reliable and secure.
- Menu: The menu on the SOS should be up-to-date and accurate. It should also include images of the dishes to help customers make informed decisions.

SOS offer a number of benefits for restaurants, including increased customer satisfaction, improved staff efficiency, and reduced operating costs. When implementing a SOS, it is important to carefully consider the user interface, technology, and menu.

3.METHODOLOGY

5.1 Approaches

Here we followed Full Stack methodologies. Here we connected the Full Stack with the PYTHON.

FULL STACK METHODOLOGY

5.1.1 Front End Languages

The Front-End languages used for the SOS web application are HTML, CSS and JavaScript.HTML is used for the structure of web pages, CSS is used for the styling for the web pages and JavaScript is used to make web pages interactive. Firstly, we create the HTML files for the web page structure. They are index, index1, checkout, about,contact,login,search html files and then we create the CSS file to give the structure for the html files. They are style.css and admin.css and we connect all the html files to the css files and we include the JavaScript code in the html files.

Fig.5.1 HTML and CSS files

5.1.2 Back End Languages

The Back End languages used for the SOS web application are python with the connect the Django framework. Now we have to connect the Django libraries with the python version 3.11. The python files manage the order details, timestamp, order ID, total amount of order. Django manages the admin side modifications, views and records the data of the Users and it allows the major changes for the admin. They are delete, update and view the orders and Users.

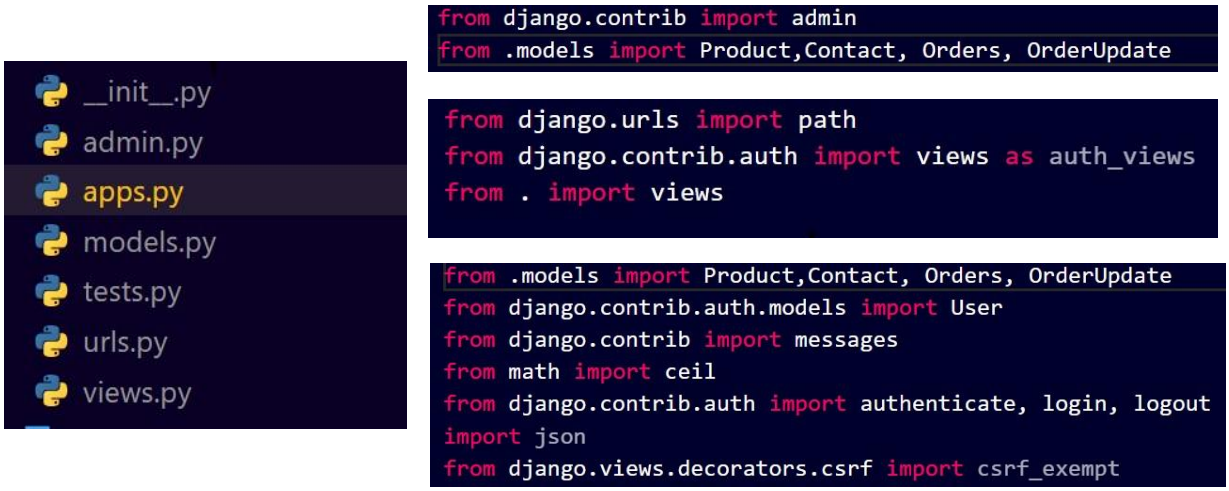


Fig. 5.2 Python Files & Django Framework

5.1.3 Database

The Database is used for storing the data for the user, order details, items. We create the tables and modify all the details of the users and orders. We use the database version of DB browser (SQLite).

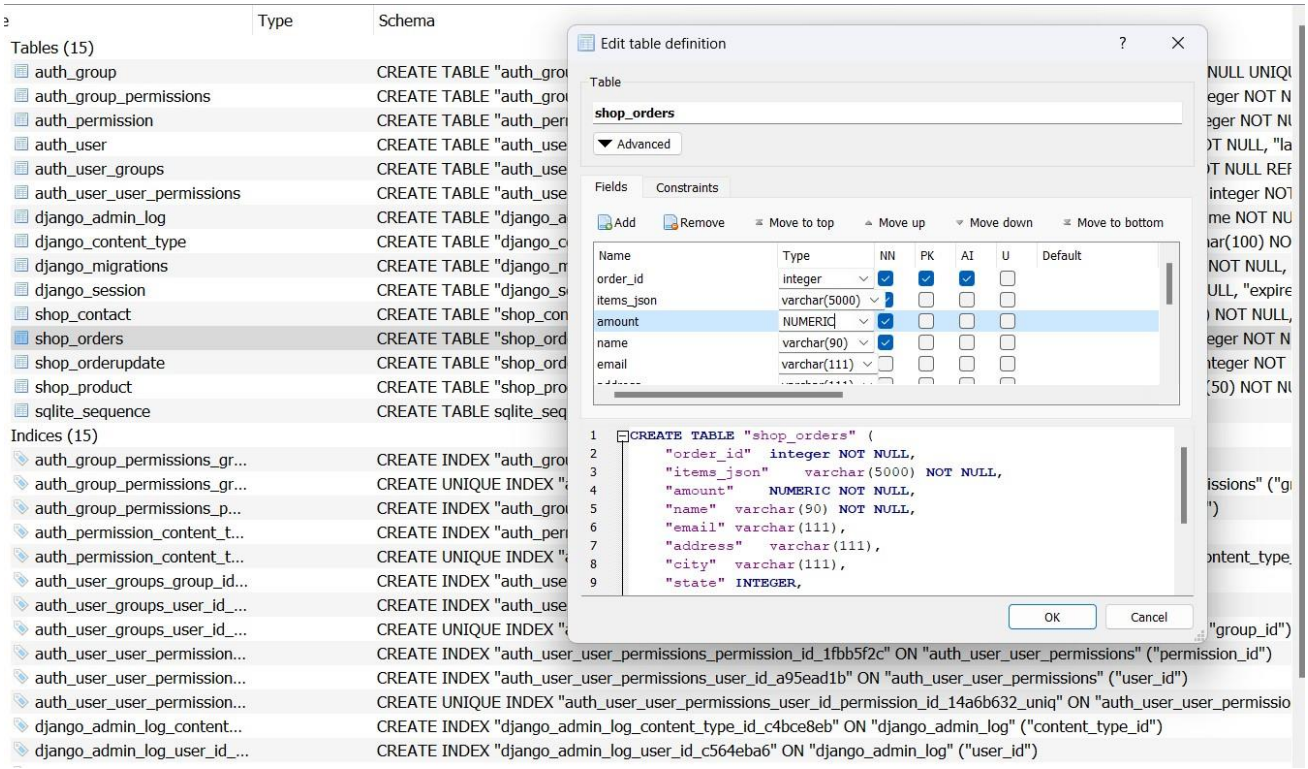


Fig.5.3 DB Browser Database

5.2 Existed System

In the online order website, there is only the option for ordering the selective food and will be delivered to their homes. In case if a family needs to have their food in a restaurant they need to go to the restaurant and having a very long queue, they keep on waiting for the service. There will be a delay in the time to get their order prepared. And the managers need to maintain waiters to keep the track of the customer orders, this includes the man effort in calculating the bills. As the food is ordered and delivered through the online ordering website all the restaurant operations are being delayed. In order to efficiently serve the customer without time delay and offer food at optimal prices ,that saves money for both the customers and for management. For maintaining the restaurants, we came up with the new website called Smart Order System.

5.3 Proposed System

Customers are directly ordering their food to their respective tables and we can also reduce the waiting time of the people by taking their orders directly to the table. And they can enjoy their food in the atmosphere of restaurant vibes. They can select and order their food from the table. The main purpose is to improve the performance of the restaurant by eradicating the daily paperwork. With this application the tasks would be performed in less amount of time and more efficiently. An additional benefit of this software is that during the rush hours the load can be balanced effectively, and restaurants would perform better than usual. In addition to this, human error that occurs when performing tasks manually is also minimized and presence of queues in the system to assign tasks to chefs can reduce congestion in the kitchen. The system would also result in reduction of labour which would result in the reduction of expenses of the restaurant.

4. UML & DF DIAGRAMS

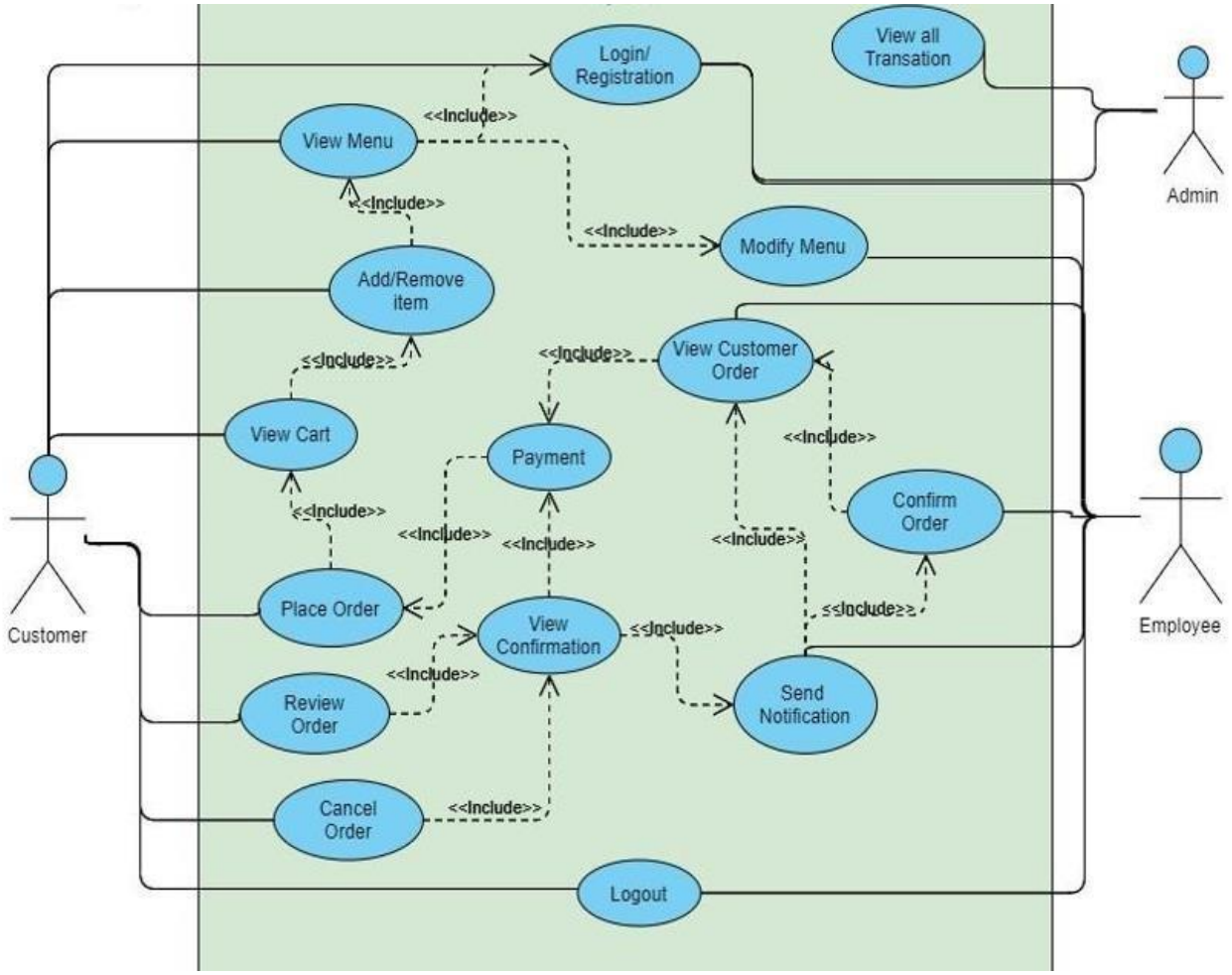


Fig.6.1 Use Case Diagram

1. In this use case diagram proposed have two actors, namely Customer and Admin.
2. Customers can access the website to place orders and payments while Admin can access the website to receive orders, payments and making order report.
3. Use case starts from login into the website with the customer's username and password, ordering food, making payment. Then Employee makes an order report based on customer order data.

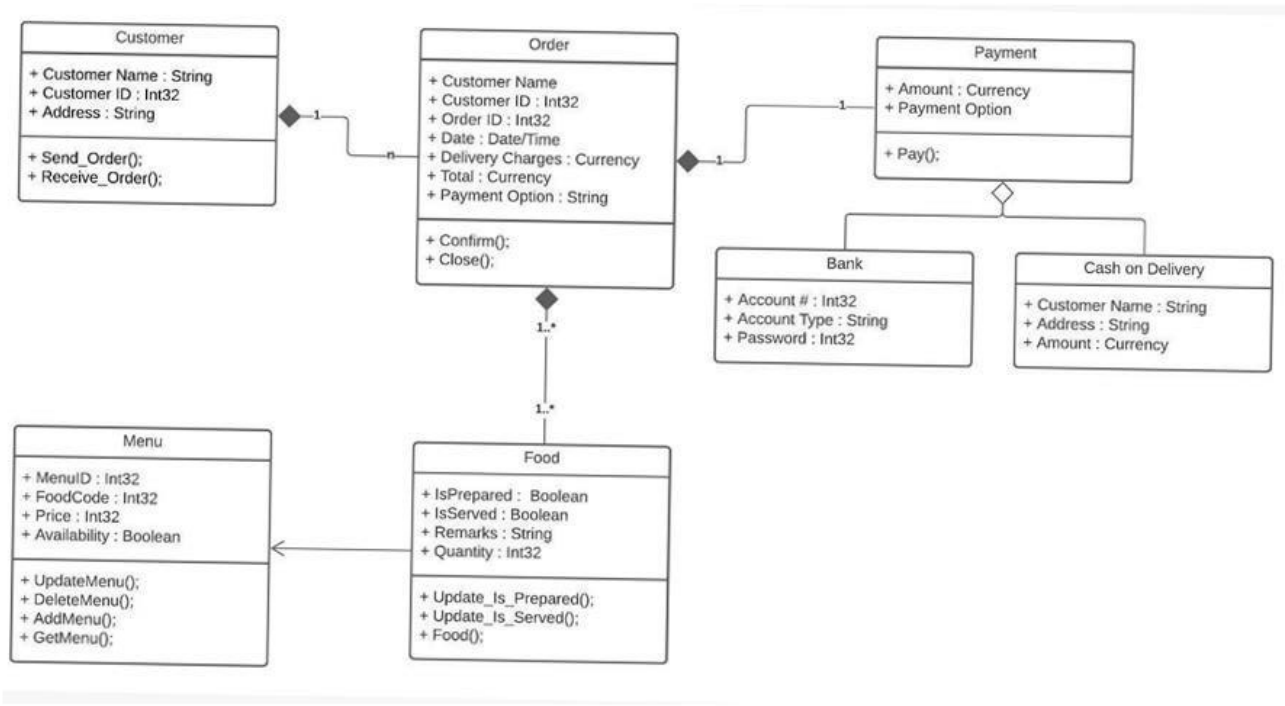


Fig.6.2. Class diagram

1. The image you sent is a diagram of a company's accounting system. It shows how the company processes orders, payments, and other financial transactions
2. The system starts with a customer placing an order. The order is then processed by the company's order processing system. This system generates an order ID and calculates the total amount of the order
3. The order is then sent to the customer for confirmation. Once the customer confirms the order, the company processes the payment. The payment can be made by bank transfer, credit card, or cash on delivery.

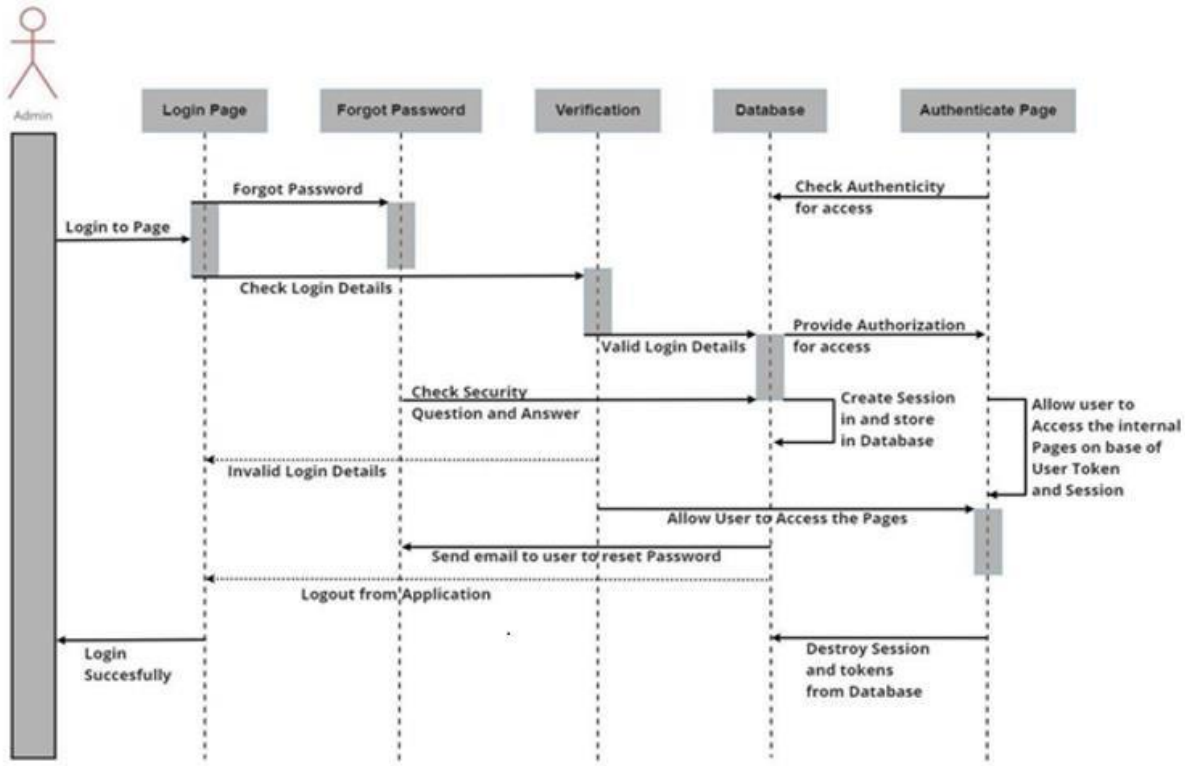


Fig 6.3 Admin Sequence Diagram of Login Page

The image you sent is a sequence diagram of a food order system. It shows the steps involved in placing, processing, and delivering an order. The diagram also shows the interactions between the different actors involved in the system, namely the customer, the restaurant, and the customer service.

1. The customer places an order with the restaurant.
2. The restaurant confirms the order with the customer.
3. The customer confirms payment.
4. The restaurant prepares the order.
5. The kitchen notifies the restaurant that the order is ready.

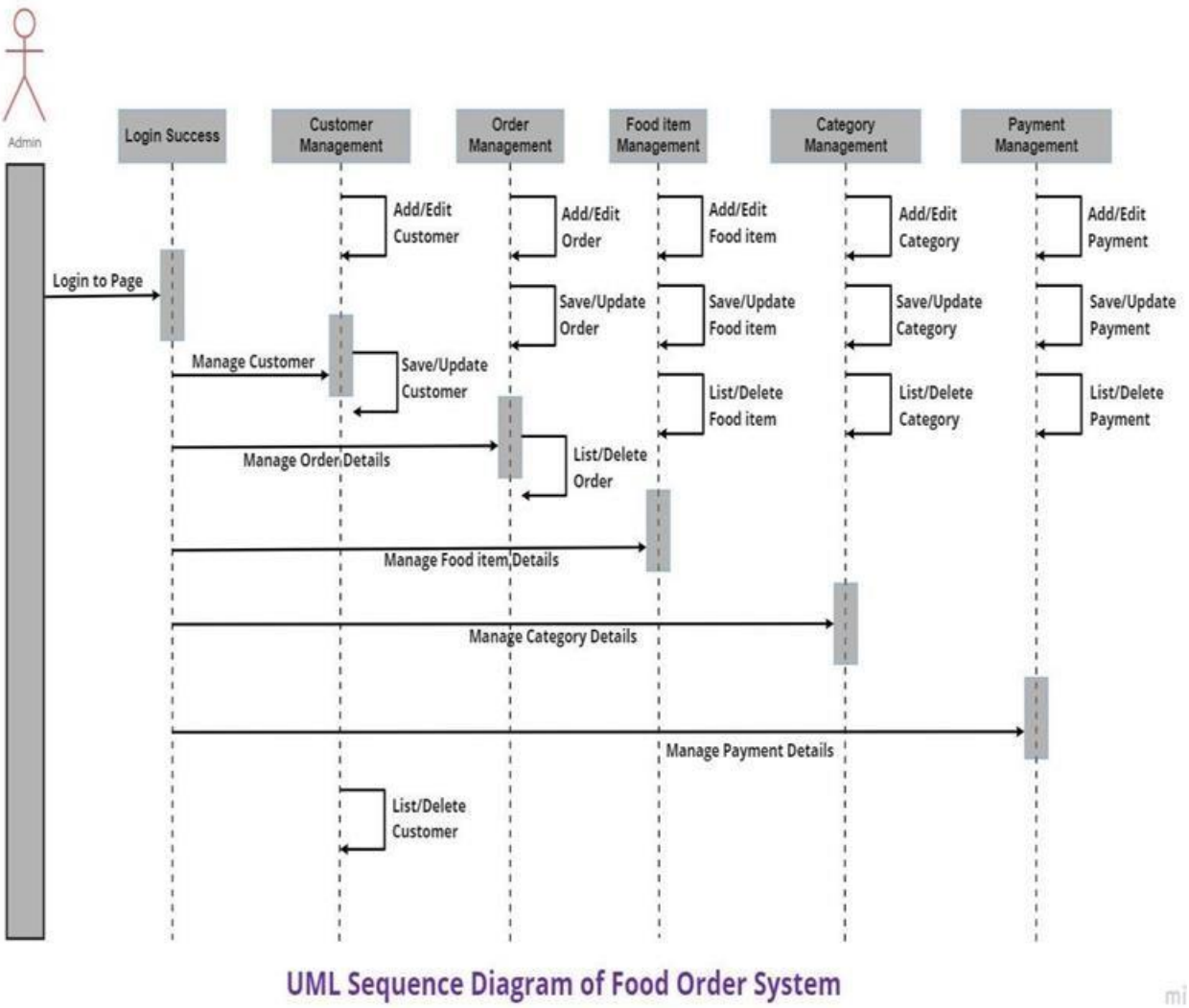


Fig 6.4 Sequence Diagram of Admin

The image you sent contains a sequence diagram for a food order system. It shows the steps involved in placing, processing, and delivering an order, as well as the interactions between the different actors involved in the system, namely the customer, the restaurant, and the delivery service.

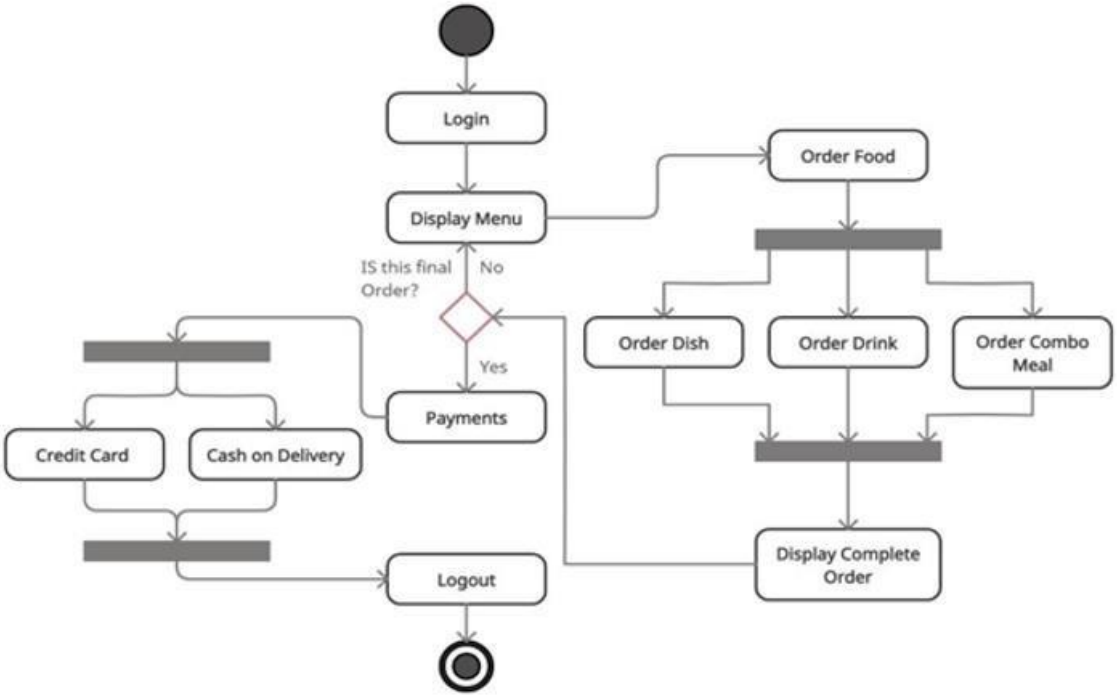


Fig 6.5 Activity Diagram of SOS

The image you sent is an activity diagram of a food ordering system. It shows the different activities involved in placing and processing an order, and the different flows that can occur depending on the user's choices.

1. The system starts by displaying the menu to the user.
2. The user selects the dish they want to order.
3. The system adds the dish to the user's order.
4. The user confirms their order.
5. The system processes the payment.
6. The system displays the order confirmation to the user.
7. The user confirms the order.
8. The system ends the process.

5.CODE & RESULT

```python
def checkout(request):
    if request.method == "POST":
        items_json = request.POST.get('itemsJson', '')
        user_id = request.POST.get('user_id', '')
        name = request.POST.get('name', '')
        amount = request.POST.get('amount', '')
        phone = request.POST.get('phone', '')
        order = Orders(items_json=items_json, userId=user_id, name=name, phone=phone, amount=amount)
        order.save()
        update = OrderUpdate(order_id=order.order_id, update_desc="The Order has been Placed")
        update.save()
        thank = True
        id = order.order_id
        if 'onlinePay' in request.POST:
            # Request paytm to transfer the amount to your account after payment by user
            dict = {

                'MID': 'WorldP64425807474247',  # Your-Merchant-Id-Here
                'ORDER_ID': str(order.order_id),
                'TXN_AMOUNT': str(amount),
                'CUST_ID': str(user_id),
                'INDUSTRY_TYPE_ID': 'Retail',
                'WEBSITE': 'WEBSTAGING',
                'CHANNEL_ID': 'WEB',
                'CALLBACK_URL': 'http://127.0.0.1:8000/shop/handlerequest/',
```

Fig 7.1 Checkout Code

The image you sent is a diagram of a Paytm paym ent gateway. It shows how the customer, the merchant, and the payment gateway interact with each other to process a payment.

1. The customer places an order with the merchant.
2. The merchant initiates the payment process by sending the customer's payment information to the payment gateway.
3. The payment gateway authenticates the customer's payment.
4. The customer confirms the payment.
5. The payment gateway notifies the merchant that the payment has been successful.
6. The merchant fulfils the order.

```python
def index(request):
    allProds = []
    catprods = Product.objects.values('category', 'id')
    cats = {item['category'] for item in catprods}
    for cat in cats:
        prod = Product.objects.filter(category=cat)
        n = len(prod)
        nSlides = n // 4 + ceil((n / 4) - (n // 4))
        allProds.append([prod, range(1, nSlides), nSlides])
    d = {'allProds': allProds}
    return render(request, 'shop/index.html', d)
```

Fig 7.2 View all Products Code

The code first gets all products from the database. Then, it creates a set of all category names.For each category name, it filters the products to only include products in that category.Finally, it creates a list of all products, grouped by category.

```python
def about(request):
    return render(request, 'shop/about.html')

def orderView(request):
    if request.user.is_authenticated:
        current_user = request.user
        orderHistory = Orders.objects.filter(userId=current_user.id)
        if len(orderHistory) == 0:
            messages.info(request, "You have not ordered.")
            return render(request, 'shop/orderView.html')
        return render(request, 'shop/orderView.html', {'orderHistory': orderHistory})
    return render(request, 'shop/orderView.html')


def searchMatch(query, item):
    if query in item.desc.lower() or query in item.product_name.lower() or query in item.category.lower()
        return True
    else:
        return False

def contact(request):
    thank = False
    if request.method == "POST":
        name = request.POST.get('name', '')
        email = request.POST.get('email', '')
        phone = request.POST.get('phone', '')
        desc = request.POST.get('desc', '')
        contact = Contact(name=name, email=email, phone=phone, desc=desc)
        contact.save()
        thank = True
        return render(request, 'shop/contact.html', {'thank': thank})
    return render(request, 'shop/contact.html', {'thank': thank})
```

Fig 7.3 Order List Code

The code first gets all products from the database. Then, it creates a dictionary to store the products, grouped by category. For each product, it looks up the category name and adds the product to the corresponding list of products in the dictionary. Finally, it returns the dictionary.

Fig 7.4 Contact Page Code

This contact form can be used by people to contact the website owner for a variety of reasons, such as to ask questions, provide feedback, or report a problem.To use the contact form, simply enter your name, email address, phone number, and message in the appropriate fields and click the submit button. Your message will be sent to the website owner, who will respond to you as soon as possible.

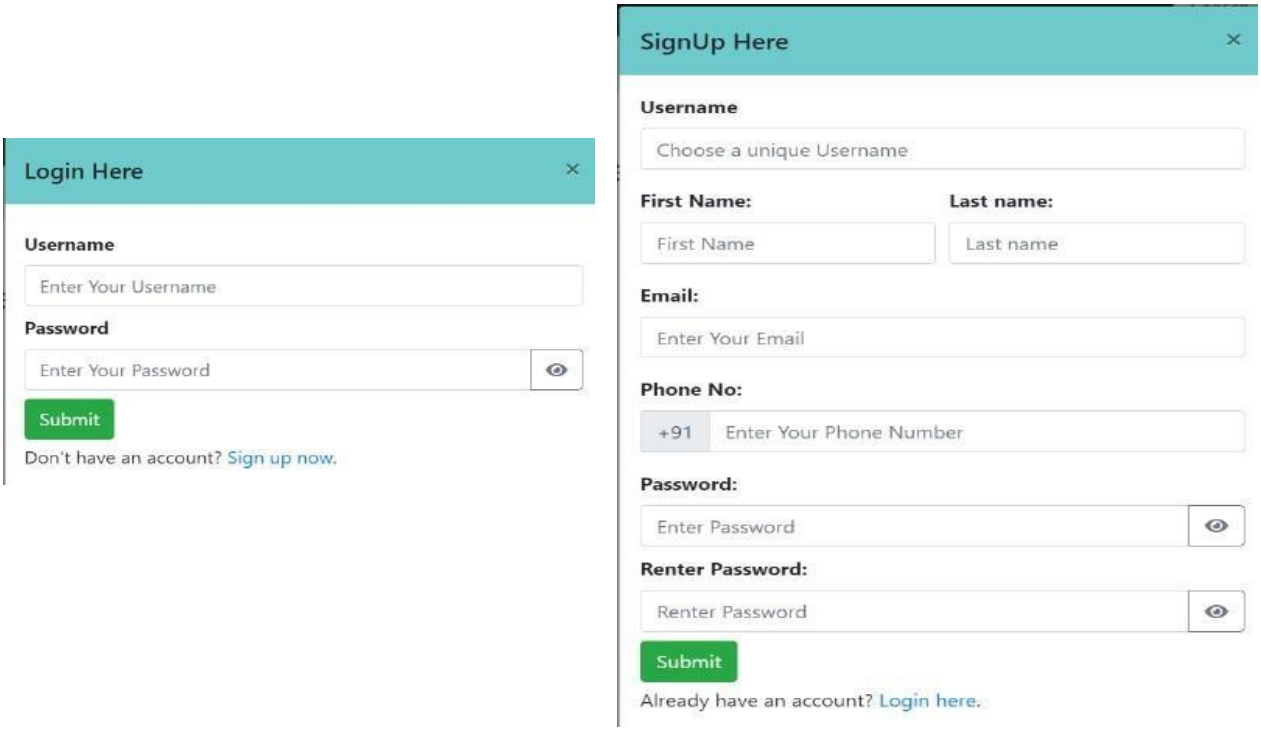


Fig 7.5 Login & Signup Module

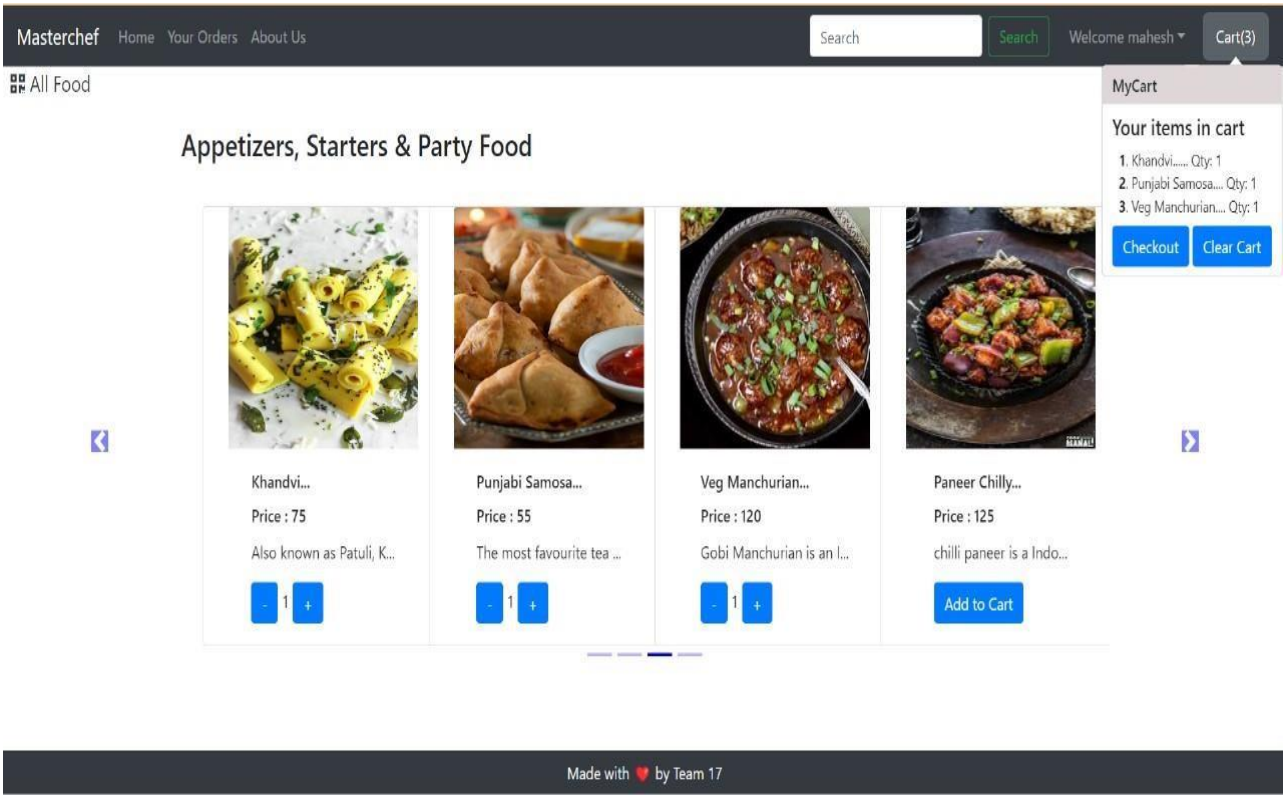


Fig 7.6 Home Page

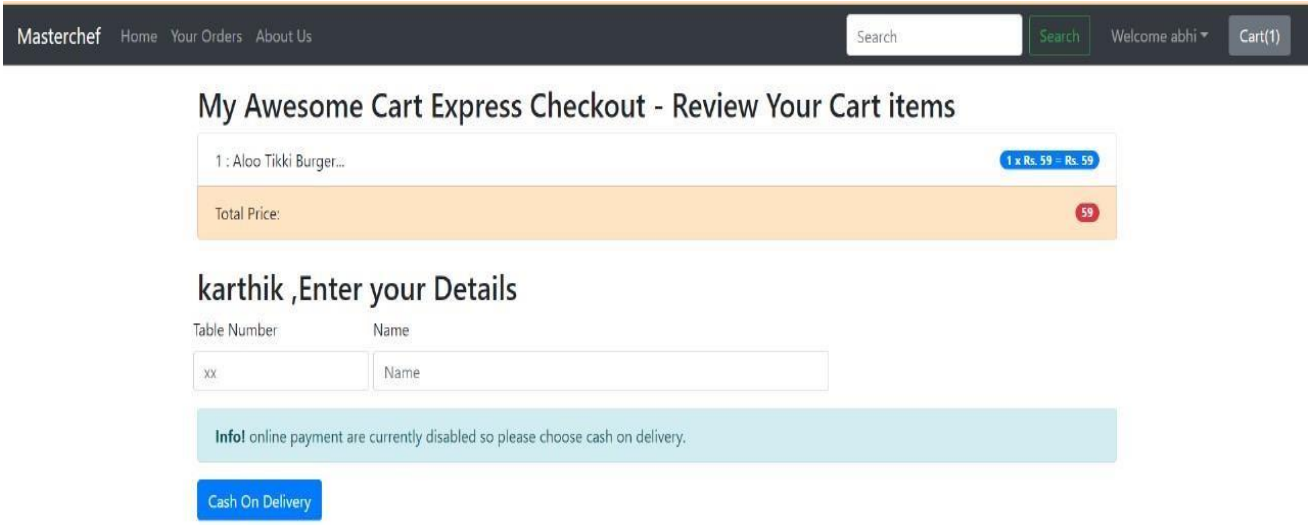


Fig 7.7Checkout Page

Fig 7.8 Confirm Order

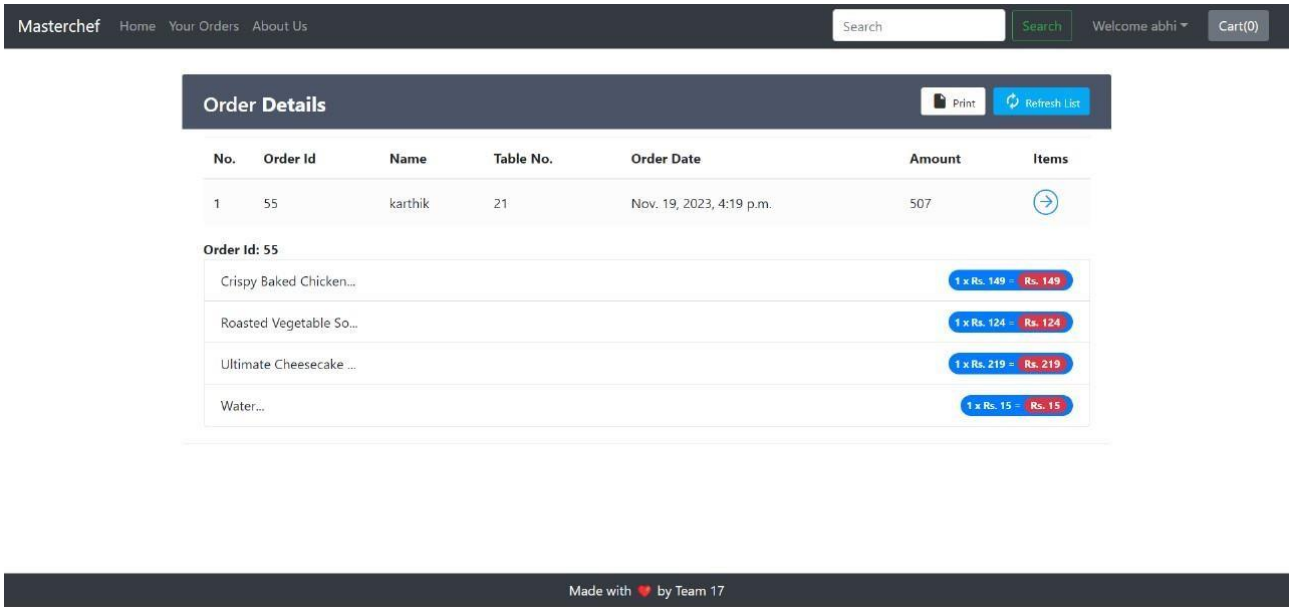


Fig 7.9 View Order Page

Fig 7.10 Search bar

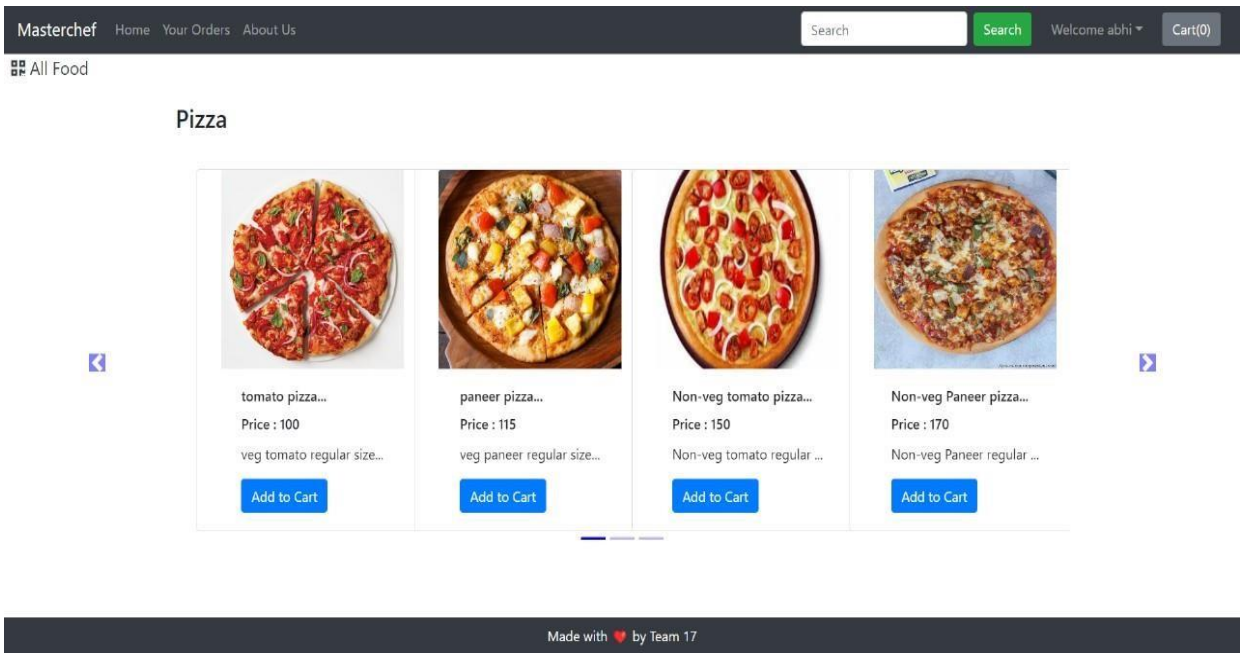


Fig 7.11 Search Page

Fig 7.12 Admin Login Page

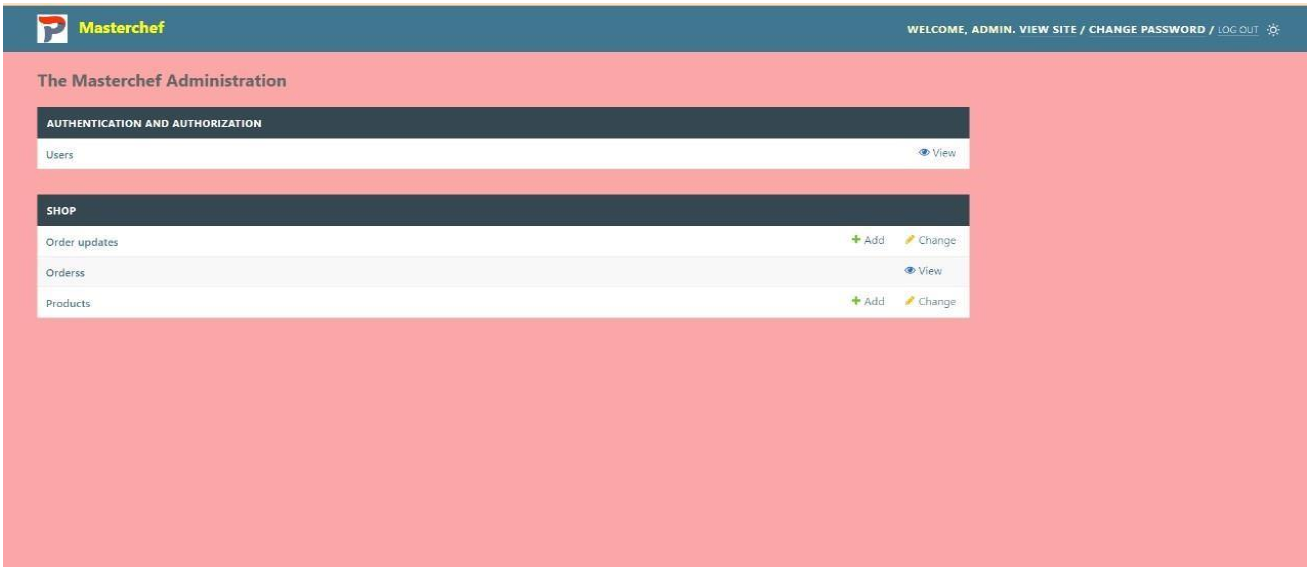


Fig 7.13 Admin Home Page

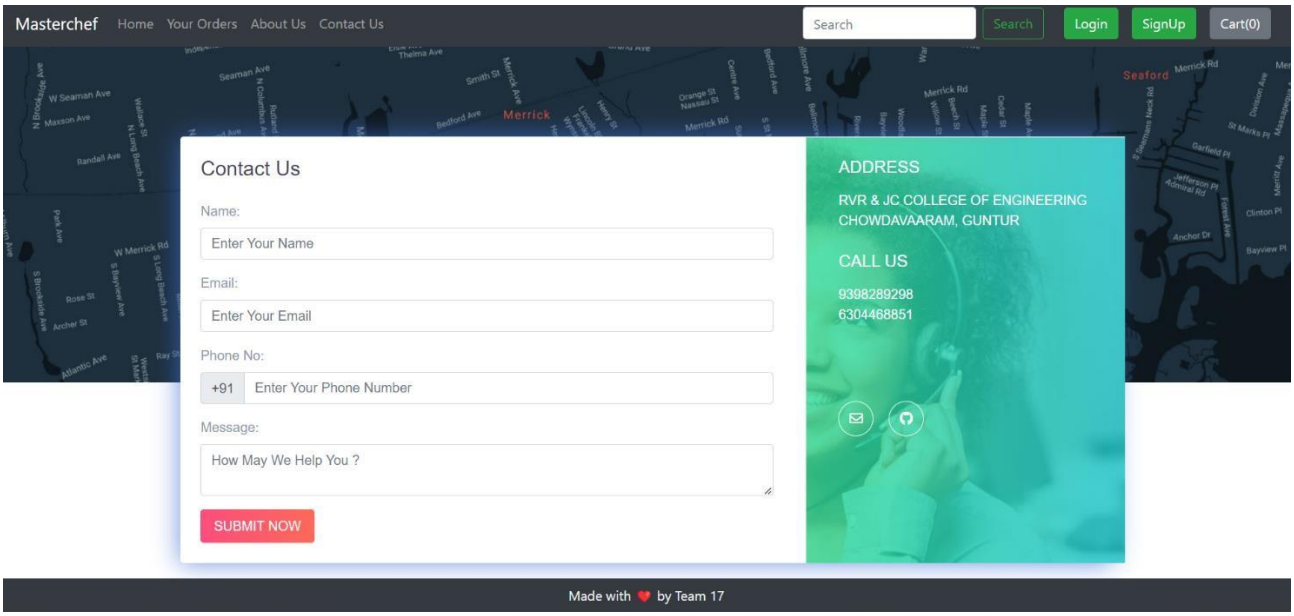


Fig 7.14 Contact Us Page

6.CONCLUSION & FUTURE ENHANCEMENTS

After reviewing our work, the conclusion is that after many adjustments the system works. As good as it is now, there can still be made many adjustments/improvements. However in the time was given that three persons can work on this project, the overall results are satisfactory in our opinion. The report covers the entire course of the project and results are there were needed. The first weeks the work progressed slower than expected, then the pace was increased to finish on time.

For customers, web-based ordering system can make it easier to order food without having waiting time so that customers can save time and costs. For admin, they can serve customers optimally in ordering their food and making the order report easier. Payment methods can also be done by customers through a system that is available on the web to facilitate customers in paying for their orders.

In future we would like to solve all draw backs and we want to add some features like making prediction using data analysis of the restaurant that how much ingredients the restaurant needs for a specific day. Also, will try to add more payment options and discount facilities.

The following section describes the work that will be implemented with future releases of the software.

1) Customize orders: Allow customers to customize food orders.
2) Enhance User Interface by adding more user interactive features. Provide Deals and promotional Offer details to home page. Provide Recipes of the Week/Day to Home Page.
3) Payment Options: Add different payment options such as Phone pe, Paytm, Google pay, Gift Cards etc. Allow to save payment details for future use.
4) Allow to process an order as a Guest
5) Order Process Estimate: Provide customer a visual graphical order status.
6) Restaurant Locator: Allow to find and choose a nearby restaurant.

7.REFERENCE / BIBLIOGRAPHY

Following links and websites were referred during the development of this project:

1. IEEE. IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications.

2. https://getbootstrap.com/

3. https://www.djangoproject.com/

4. https://www.w3schools.com/

5. http://stackoverflow.com/

6. https://codewithharry.com/

Books were referred during the running and implementing the project:

1. Web Development with Django by Ben Shaw and Saurabh Badhwar

2. The Bootstrap 4 Quick Start by Jacob Lett

3. Object Oriented Analysis and Design 2005 by Mike O'docherty.

4. Visual Modeling with Rational Rose 2002 by Teny Quatran.

7. The elements of UML style by Scott W.Ambler.

8. The Unified Modeling Language User Guide by Grady Booch, James Rumbaugh.