

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JNANA SANGAMA”, Belagavi-590018



Mini Project Report on “wowFOOD -THE FOOD HEAVEN”

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR
THE AWARD OF DEGREE OF

**BACHELOR OF ENGINEERING IN
INFORMATION SCIENCE AND ENGINEERING**

SUBMITTED BY

NISHANT MANJUNATH HEGDE (1JB21IS073)

Under the Guidance of

Mr. Abhinand B. V
Assistant Professor,
Dept. of ISE, SJBIT
Bengaluru-60



**DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING
SJBIT INSTITUTE OF TECHNOLOGY**

BGS HEALTH AND EDUCATION CITY, KENGERI, BENGALURU-560060, KARNATAKA, INDIA.

2023-2024

|| Jai Sri Gurudev ||
Sri Adichunchanagiri Shikshana Trust ®

SJB INSTITUTE OF TECHNOLOGY

BGS Health & Education City, Kengeri, Bengaluru - 560 060

Department of Information Science & Engineering



CERTIFICATE

Certified that the Mini project work entitled “**wowFOOD -THE FOOD HEAVEN**” carried out by **NISHANT MANJUNATH HEGDE** bearing **USN 1JB21IS073** is a bonafide student of **SJB Institute of Technology** in partial fulfillment for 6th semester Mini Project in **INFORMATION SCIENCE AND ENGINEERING** of the **VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELAGAVI** during the academic year **2023-24**. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The mini project report has been approved as it satisfies the academic requirements in respect of Mini Project prescribed for the said degree.

Mr. Abhinand B. V
Assistant Professor
Dept. of ISE, SJBIT

Dr. Shashidhara HR
Professor & Head
Dept. of ISE, SJBIT

INTERNAL VIVA

Name of the Examiners	Signature with Date
1. _____	_____
2. _____	_____



ACKNOWLEDGEMENT



I would like to express my profound thanks to **His Divine Soul Padmabhushan Sri Sri Sri Dr. Balagangadharanatha MahaSwamiji** and His Holiness **Jagadguru Sri Sri Sri Dr. Nirmalanandanatha Swamiji** for providing me an opportunity to pursue my academics in this esteemed institution.

I would also like to express my profound thanks to **Revered Sri Sri Dr. Prakashnath Swamiji, Managing Director, SJB Institute of Technology**, for his continuous support in providing amenities to carry out this mini project in this admired institution.

I express our gratitude to **Dr. K. V. Mahendra Prashanth**, Principal, SJB Institute of Technology, for providing excellent facilities and academic ambience, which have helped me in satisfactory completion of mini project work.

I extend our sincere thanks to **Dr. Babu N V**, Academic Dean, SJB Institute of Technology for providing us constant support throughout the period of Mini Project.

I extend our sincere thanks to **Dr. Shashidhara H.R**, Professor & Head, Dept. of Information Science and Engineering, for providing us invaluable support throughout the period of Mini Project work.

I express my heartfelt gratitude to our guide, **Mr. Abhinand B. V**, Assistant Professor, Department of Information Science and Engineering and the Mini Project Coordinator **Mrs. Gayathri S**, Assistant Professor, Department of Information Science and Engineering for their valuable guidance, suggestions and encouragement during the entire period of our mini project work.

Finally, I take this opportunity to extend my earnest gratitude and respect to our parents, Teaching & Non-teaching staffs of the department, the library staff and all our friends, for their continuous support and encouragement.

NISHANT MANJUNATH HEGDE
(1JB21IS073)

ABSTRACT

The project explores the development and implementation of a web-based food ordering system using the Django framework. The report highlights the advantages of Django, such as rapid development, security features, scalability, and ORM support. It addresses common problems faced in the food service industry, including late deliveries, incorrect orders, and cold food delivery. The objectives of the project include enhancing organizational efficiency, improving customer satisfaction, and reducing food wastage. The scope of the project encompasses the comprehensive development, implementation, and deployment of the food ordering system. The report also discusses the system architecture, customer interface, admin interface, implementation details, testing procedures, and concludes with the successful creation of a robust and user-friendly platform that streamlines the food ordering process. The project demonstrates the effective use of modern web technologies to enhance operational efficiency and customer satisfaction in the restaurant industry.

TABLE OF CONTENTS

Acknowledgement	i
Abstract	ii
Table of Contents	iii
List of Figures	iv

CHAPTER NO	TITLE	PAGE NO
1	Introduction	1-2
1.1	Introduction to Django	1
1.2	Advantages of Django	1
1.3	Problem Statement	2
1.4	Objectives	2
1.5	Scope	2
2	Software Requirements Specification	3
2.1	Hardware Requirements	3
2.2	Software Requirements	3
3	Software Design	4-6
3.1	System Architecture	5
3.2	Customer Interface	5
3.3	Admin Interface	6
4	Implementation	7
5	Testing	8
6	Snapshots	9-10
7	Conclusion	11
	References	12

LIST OF FIGURES

FIGURE NO	NAME	PAGE NO
4.1	Code snippet	7
5.1	Testing	8
6.1	Home page	9
6.2	Menu Page	9
6.3	ContactUs Page	9
6.4	Login Page	10
6.5	Django Administration Page	10
6.6	AboutUs Page	10

CHAPTER 1

INTRODUCTION

1.1 Introduction to Django

Django is an open-source Python-based, Web framework that follows the typical design patterns and Best Industry Standards. Purpose: Django makes it possible for developers to have web applications developed in a short span of time and with less number of code input. Architecture: Django is a framework that operates based on the Model-View-Template or the Model-View-Controller as it is commonly referred to. Database support: Allows for SQL-based support for different types of databases including PostgreSQL, MySQL, SQLite, and Oracle. Scalability: This one is useful to websites that may receive a lot of traffic into the web, page or system. Security: The browser therefore has a default security set up to counter ordinary dangers that are posed on the internet. REST framework: Teaches how to create Web APIs. It is also interesting to note that there is no strict necessity for the beginner to meet in order to work with Django – one just needs to have Python on his/her computer. From here, you can create a new project, define models, set up views and templates, start off the application too. If there is anything more specific that you would like to know about Django or How to get started creating a project with the technology please let me know.

1.2 Advantages of Django

- 1. Rapid development:** Django philosophy of “batteries included” offers many features as many of them come with the framework.
- 2. Don't Repeat Yourself (DRY) principle:** Reusability is also respected in Django since this framework is against redundancy.
- 3. Security:** Some of the weaknesses that are included are the SQL injections, cross site scripting, and click jacking among others.
- 4. Scalability:** Naturally built for traffic and you can integrate extra tiers to it in case you notice business is picking up.
- 5. ORM:** Enables the use of Python data code instead of SQL while being compatible with a number of back-ends.
- 6. Admin interface:** Website administrator interface that is automatically placed to control the content of the site.
- 7. URL routing:** This will make a neat, orderly URL structure that will be easier for the search engines to crawl as well as for the visitors to follow.
- 8. Template engine:** Convenient and potent for designing rich and changing HTML documents.
- 9. Authentication system:** Fast and strong user authentication and authorization as for built-in security means.
- 10. REST framework:** It has outstanding support for usage in constructing APIs.

- 11. Large community and ecosystem:** Huge documentation which is available online, third-party packages, and lots of community support.
- 12. Database migration:** Inbuilt customizable functionalities for the direct alteration of the database schema.
- 13. Internationalization:** Integrated capability for dealing with sites that are created in more than one language.
- 14. Testing support:** Contains a test environment that you can use for writing tests and then running them.

1.3 Problem Statement

Some of the major problems reported by the customers surveyed revolved

1. Around late deliveries because of network problem
2. Incorrect orders being delivered due to communication problem.
3. Orders not being delivered at all, rude customer service.
4. Cold food being delivered, and the driver requiring a lot of guidance to find the delivery location.
5. Sometimes a payment issue occurs.

1.4 Objectives

1. To enhance firm and organizational efficiency by introducing effective utilization use of technology in the daily operations to enhance provision of services to the customers.
2. As a way of being distinct in the kind of competition that is provided by the food service industry.
3. In order to allow customer place an order and be delivered something that is not on menu.
4. But in a bid to allow the customers to have a physical confirmation of the order that they placed.
5. In order to let customers decide the ingredients of food they want to order, some necessary elements have to be introduced.
6. An objective that contributes to sustainability and can be used in restaurants is- reducing food wastage.
7. To avoid placement of orders in the wrong place through confirmation by the sight..
8. Increase efficiency of the restaurant's employees.
9. Reduce paper works and increase the level of accuracy.
10. Improve some factors such as rate of service delivery, size of sale and satisfaction level of the customers.

1.5 Scope

The scope of this food ordering project encompasses the development, implementation, and deployment of a comprehensive web-based food ordering system using the Django framework.

CHAPTER 2

SOFTWARE REQUIREMENTS SPECIFICATION

There are no systems which can run without hardware and software requirements. So for any system in this world, the hardware and software requirements are the most basic necessity to work. For each and every system there will be different hardware and software requirements. So we shall see the particular requirement of our system. Software requirements concerned with portraying programming asset prerequisites and essentials that should be introduced on a computer to give best working of an application.

2.1 Hardware Requirements

Processor: Pentium core 2 duo and higher

RAM: 512MB

Hard Disk: 40GB

Monitor: Colour Monitor

2.1.1 Software's used

Operating System – Windows OS

2.1.2 Technologies used

HTML, CSS, JS, Django

2.2 Software Requirements

- OS: Windows 7 and higher
- Languages: DJANGO, HTML, CSS, JAVASCRIPT, BOOTSTRAP
- Software: VS CODE(any browser which supports XAMPP)

CHAPTER 3

SOFTWARE DESIGN

The Django-based food ordering system features secure user registration and authentication, enabling customers to browse, customize, and place orders from a categorized menu with real-time updates.

User Registration:

Username and Password: Require users to create a unique username and password.

Email Address: Request a valid email address for account verification and password recovery.

Name and Contact Information: Collect name, phone number, and address for delivery purposes.

Password Strength: Enforce strong password policies (e.g., length, complexity).

Terms and Conditions: Require users to agree to terms and conditions and privacy policies.

Authentication:

Login Credentials: Allow users to log in with their username and password.

Session Management: Implement secure session management to protect user data.

Password Recovery: Provide a secure password recovery process via email or phone.

Two-Factor Authentication (2FA): Offer optional 2FA for added security.

Logout: Ensure secure logout functionality to end user sessions.

Menu Management:

Menu Categories: Organize menu items into categories (e.g., appetizers, entrees, desserts) for easy navigation.

Menu Item Management: Allow admins to add, edit, and remove menu items, including descriptions, prices, and images.

Menu Item Variations: Support variations of menu items (e.g., size, flavor, special requests) with separate pricing.

Nutritional Information: Provide nutritional information (e.g., calories, allergens) for menu items.

Menu Item Availability: Manage availability of menu items (e.g., daily specials, seasonal offerings).

Pricing and Discounts: Support various pricing options (e.g., happy hour, promotions) and discounts (e.g., coupons, loyalty programs).

Menu Item Search: Implement a search function for menu items to facilitate easy ordering

Menu Item Recommendations: Offer personalized recommendations based on user preferences and order history.

Menu Item Reviews: Allow customers to leave reviews and ratings for menu items.

Menu Management Permissions: Control access to menu management features based on user roles (e.g., admin, manager, staff).

Menu Item Images: Support high-quality images for menu items to showcase dishes.

Menu Item Descriptions: Allow for detailed descriptions of menu items, including ingredients and preparation methods.

Order Placement:

Order Summary: Display a clear summary of the order, including menu items, quantities, and prices.

Payment Options: Offer various payment options (e.g., credit cards, online payment services, cash on delivery).

Delivery or Pickup Options: Allow customers to choose between delivery or pickup, with associated fees and estimated times.

Order Notes: Provide a field for customers to add special instructions or notes (e.g., food allergies, special requests).

Order Timing: Allow customers to choose a preferred delivery or pickup time, with options for ASAP, scheduled, or later.

Order Confirmation: Send a confirmation email or notification with order details and status updates.

Order Modification: Allow customers to modify their orders before processing, with clear instructions on how to do so.

Order Status Updates: Provide real-time updates on order status (e.g., preparation, delivery, completed).

Secure Payment Processing: Ensure secure payment processing with encryption and compliance with industry standards (e.g., PCI-DSS).

Loyalty Program Integration: Integrate loyalty programs to reward customers for repeat orders and loyalty.

3.1. System Architecture**3.1.1 Frontend (Client Side):**

Technologies: HTML, CSS, JavaScript, and Django templates for rendering dynamic content.

Responsibilities: Rendering the user interface, handling user interactions, making AJAX calls to the backend, and displaying data.

3.1.2 Backend (Server Side):

Technologies: Django for handling HTTP requests, processing business logic, interacting with the database, and returning responses to the frontend.

Responsibilities: Managing authentication, handling order processing, managing inventory, serving dynamic content, and API endpoints if needed.

3.2 Customer Interface

Menu: A visual representation of the restaurant's menu, including images, descriptions, and prices.

Search: A search bar to find specific menu items or cuisines.

Order Basket: A summary of the customer's selected items, with options to edit quantities or remove items.

Checkout: A secure payment process, including payment method selection and order review.

Order Tracking: Real-time updates on order status, including preparation and delivery times.

Account Management: Access to order history, loyalty programs, and account settings.

Reviews and Ratings: The ability to leave feedback and ratings for menu items and the overall experience.

Special Offers: Display of promotions, discounts, and special deals.

Restaurant Information: Contact details, address, and operating hours of the restaurant.

Help and Support: Access to customer support, FAQs, and contact forms.

Login/Registration: Option to log in or register for an account to save preferences and order history.

Payment Methods: Various payment options, such as credit cards, online payment services, or cash on delivery.

3.3 Admin Interface

Dashboard: A centralized overview of the website's performance, including sales, orders, and customer data.

Order Management: Ability to view, edit, and manage orders, including order status and customer information.

Menu Management: Tools to add, edit, and remove menu items, including pricing, descriptions, and images.

Restaurant Management: Management of restaurant settings, including contact information, operating hours, and delivery areas.

User Management: Management of customer accounts, including account settings, order history, and loyalty programs.

Payment Management: Management of payment options, including payment processors, commission rates, and transaction history.

Reporting and Analytics: Access to sales reports, customer behavior, and market trends to inform business decisions.

Content Management: Ability to edit website content, including text, images, and promotions.

Settings and Configuration: Management of website settings, including currency, language, and tax rates.

Staff Management: Management of staff accounts, including permissions, roles, and access levels.

Customer Support: Tools to manage customer inquiries, support tickets, and feedback.

Security and Backup: Management of website security, including backups, updates, and access controls.

CHAPTER 4

IMPLEMENTATION

Implementation is the stage in the project where the theoretical design is turned into a working system and is giving confidence on the new system for the users that it will work efficiently and effectively. It involves careful planning, investigation of the current system and its constraints on implementation, design of methods to achieve the changeover, an evaluation of change over methods.

Implementation is the most important phase. The most critical stage in achieving a successful new system is giving the users confidence that the new system will work and be effective. Any system developed should be secured and protected against possible hazards.

Component testing is a method where testing of each component in an application is done separately. Component testing is also known as module, unit or program testing. It finds the defects in the module and verifies the functioning of software. Our project is console-based, so all the implementation of our project is in console.

During the testing phase of the Django-based food ordering system, several important issues were identified and resolved, ensuring the application's robustness and reliability. The thorough testing process ensured that the system was ready for deployment, providing a reliable and secure platform for both customers and restaurant staff.

```
1  import json
2  from django.shortcuts import redirect, render
3  from django.http import HttpResponse
4  from food.models import Burger, Pizza, Order, Item
5  from .forms import NewUserForm
6  from django.views.decorators.csrf import csrf_exempt
7  from django.contrib.auth import authenticate, login, logout
8  from django.contrib import messages
9  import random
10
11 def randomOrderNumber(length):
12     sample = 'ABCDEFGHIIJ1234567890'
13     number0 = ''.join((random.choice(sample) for i in range(length)))
14     return number0
15
16 def index(request):
17     request.session.set_expiry(0)
18     ctx = {'active_link': 'index'}
19     return render(request, 'food/index.html', ctx)

```

```
1  {% extends 'food/base.html' %}
2  {% block title %}Home {% endblock %}
3  {% block content %}
4      <div class="home"></div>
5  {% endblock %}

```

Figure 4.1 : Code snippet

CHAPTER 5

TESTING

Testing is the process of evaluation a software item to detect differences between the given input and the expected output. Also, to assess the feature of a software item. Testing assesses the quality of the product. Software testing is a process that should be done during the development process. Unit Testing is a level of software testing where individual units/components of a software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output.

Component testing is a method where testing of each component in an application is done separately. Component testing is also known as module, unit or program testing. It finds the defects in the module and verifies the functioning of software. Our project is console-based, so all the implementation of our project is in console.

```
1 from django.test import TestCase
2 from django.urls import reverse
3
4 from food.models import Pizza
5
6 # Create your tests here.
7 class homePageTestCase(TestCase):
8     def test_home_page(self):
9         response = self.client.get(reverse('index'))
10        self.assertEqual(response.status_code, 200)
11
12 class PizzaTestCase(TestCase):
13     def test_newPizza_added(self):
14         numpizza = Pizza.objects.count()
15         Pizza.objects.create(name='pizza5', priceM=6, priceL=8, pimg= "someUrl")
16         self.assertEqual(Pizza.objects.count(), numpizza+1)
```

```
PS C:\Users\nisha\Desktop\django> python manage.py test
Found 2 test(s).
Creating test database for alias 'default'...
System check identified no issues (0 silenced).
..
-----
Ran 2 tests in 0.570s

OK
Destroying test database for alias 'default'...
PS C:\Users\nisha\Desktop\django> █
```

Figure 5.1 : Testing

CHAPTER 6

SNAPSHOTS

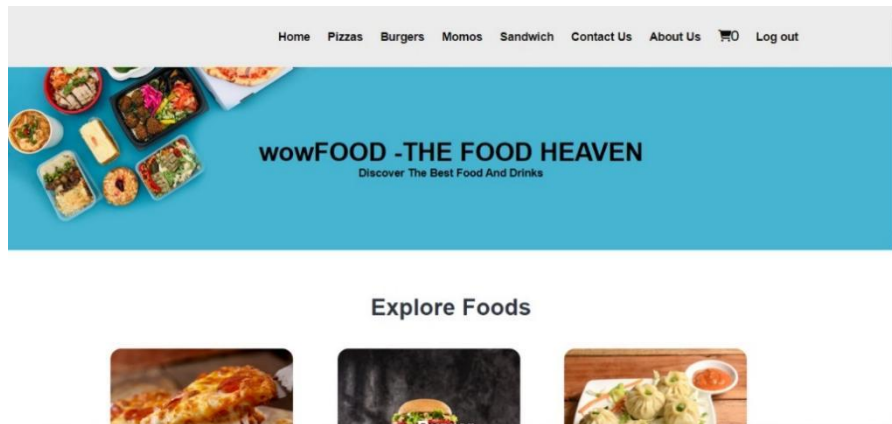


Figure 6.1: Home page

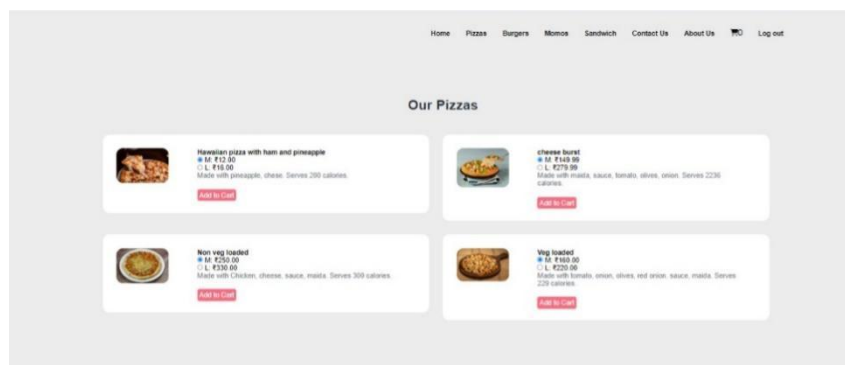


Figure 6.2: Menu Page

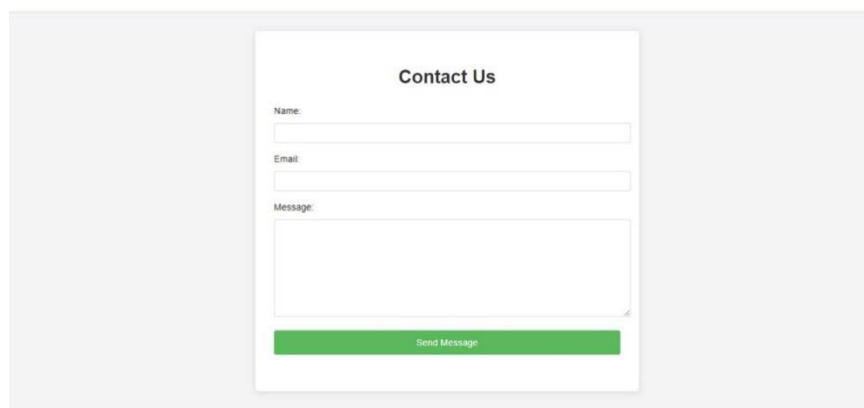
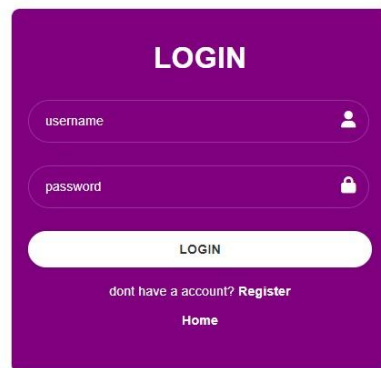
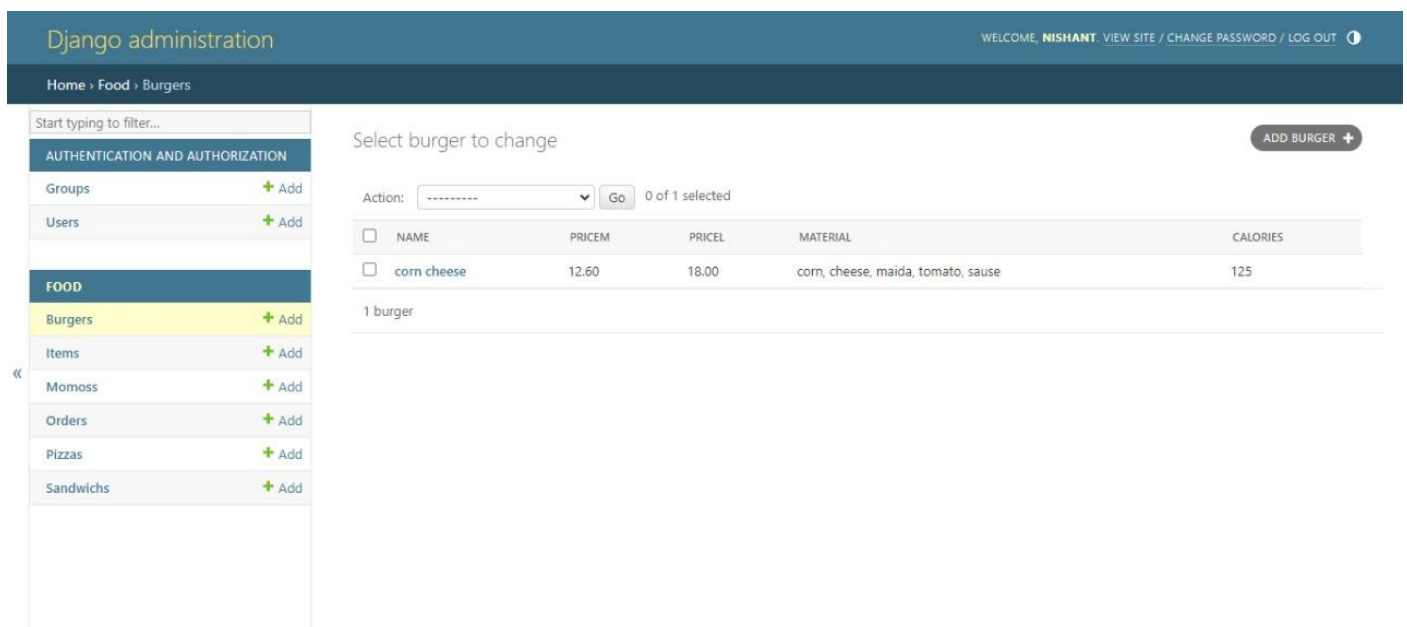


Figure 6.3: ContactUs Page



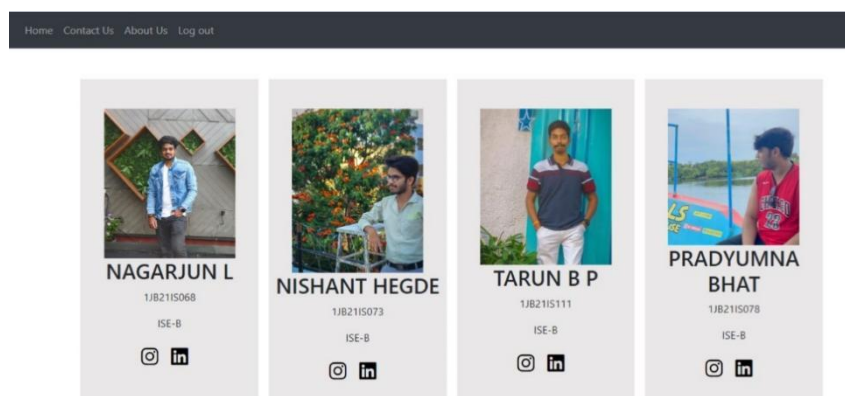
A purple login form with the title "LOGIN" at the top. It contains two input fields: "username" with a person icon and "password" with a lock icon. Below these is a white "LOGIN" button. At the bottom, there is a link "dont have a account? Register" and a "Home" link.

Figure 6.4: Login Page



The Django administration interface for the "wowFOOD" application. The top navigation bar shows "Django administration" and a welcome message for "NISHANT". The left sidebar lists various models: AUTHENTICATION AND AUTHORIZATION (Groups, Users), FOOD (Burgers, Items, Momoss, Orders, Pizzas, Sandwichs), and a search bar. The main content area shows the "Burgers" model with a table of existing burgers. The table has columns for NAME, PRICEM, PRICE, MATERIAL, and CALORIES. One burger is listed: "corn cheese" with a price of 12.60 and 18.00, made of corn, cheese, maida, tomato, and sauce, containing 125 calories. There is an "ADD BURGER" button and a "Select burger to change" section with an action dropdown and a "Go" button.

Figure 6.5: Django Administration Page



The "About Us" page features a dark navigation bar with links: Home, Contact Us, About Us, and Log out. Below the navigation bar, there are four vertical cards, each featuring a photo of a team member, their name, ID, and social media links (Instagram and LinkedIn). The team members are: NAGARJUN L (ID: 1JB21IS068), NISHANT HEGDE (ID: 1JB21IS073), TARUN B P (ID: 1JB21IS111), and PRADYUMNA BHAT (ID: 1JB21IS078).

Figure 6.6: AboutUs Page

CHAPTER 7

CONCLUSION

The development and testing of the Django-based food ordering system have resulted in a robust, secure, and user-friendly platform that addresses the inefficiencies of traditional food ordering methods. By leveraging Django's powerful features and adhering to best practices in web development, we have created a comprehensive solution that streamlines the ordering process for customers while providing restaurant staff with efficient tools for managing menu items, orders, and inventory. The testing phase helped identify and resolve critical issues, ensuring that the system performs reliably under various conditions and offers a seamless experience for users. With its scalable architecture and extensible design, the system is well-prepared for future enhancements and expansions, such as integrating mobile applications and advanced analytics. This project demonstrates the effective use of modern web to improving the operational efficiency and customer satisfaction in the restaurant industry.

REFERENCES

- [1] Adrian Holovaty, Jacob Kaplan Moss, The Definitive Guide to Django: Web Development Done Right, Second Edition, Springer-Verlag Berlin and Heidelberg GmbH & Co. KG Publishers, 2009
- [2] Jonathan Hayward, Django Java Script Integration: AJAX and jQuery, First Edition, Pack Publishing, 2011
- [3] <https://www.geeksforgeeks.org/>
- [4] <https://docs.python-guide.org/dev/virtualenvs/>
- [5] <https://docs.djangoproject.com/en/3.1/ref/templates/language/>
- [6] <https://docs.djangoproject.com/en/3.1/topics/db/models/>
- [7] <https://docs.djangoproject.com/en/3.1/topics/migrations/>
- [8] <https://docs.djangoproject.com/en/3.1/ref/contrib/admin/>
- [9] <https://docs.djangoproject.com/en/3.0/ref/templates/builtins/>