



PRESIDENCY COLLEGE
(AUTONOMOUS)
AFFILIATED TO BENGALURU CITY UNIVERSITY, APPROVED BY AICTE, DELHI & RECOGNISED BY THE GOVT. OF KARNATAKA
RE-ACCREDITED BY NAAC WITH 'A+' GRADE

A Project Report on

“ONLINE FOOD DELIVERY”

Submitted in Partial Fulfillment of the Requirements For the award of the degree

Master of Computer Applications

SUBMITTED BY

GEO SIMON

22P01091

Under the Guidance of

Ms.Veena S Badiger

PRESIDENCY COLLEGE

Kempapura, Hebbal, Bengaluru – 24

Re-accredited by NAAC with 'A+' Grade

DEPARTMENT OF COMPUTER APPLICATIONS



CERTIFICATE

This is to certify that **GEO SIMON** with Register No. **22P01091** has satisfactorily completed the fourth semester MCA Project titled “**ONLINE FOOD DELIVERY**“, as a partial fulfillment of the requirements for the award of the Degree in **Master of Computer Applications**, awarded by **Bengaluru City University**, during the Academic Year **2024**.

Project Guide : Ms.Veena S Badiger

Head of Department
(Department of computer Application)

Examiners

1. -----

2. -----

Reg No: :-----

Examination Center: -----

Date of the exam: -----



Declaration

The project titled “**ONLINE FOOD DELIVERY**” developed by me in the partial fulfillment for the award of Master of Computer Application. It is a systematic work carried by us under the guidance of Ms.Veena S Badiger, Assistant professor, Department of Computer Applications.

I, declare that this same project has not been submitted to any degree or diploma to the Bengaluru City University or any other Universities.

Name of the student: - GEO SIMON

Date:-

Signature



Acknowledgement

The development of software is generally bit complex and time consuming task. The goal of developing the project “**ONLINE FOOD DELIVERY**” could not be archived without the encouragements of kindly helpful and supportive people. Here by we convey our sincere thanks for all of them.

I take this opportunity to express my gratitude to people who had been instrumental in the successful completion of this project.

I am thankful to our management trustee for providing us an opportunity to work and complete the project successfully.

I wish to express my thanks to our **Principal Dr. Suchithra R Nair** for her support to the project work. I would like to acknowledge my gratitude to our HOD of Master of Computer Applications. **Dr.Alli** for her encouragement and support. Without their encouragement and guidance this project would not have materialized.

The guidance and support received from our Internal Guide Ms.Veena S Badiger, who contributed to this project, was vital for the success of the project. We are grateful for his constant support and help.

ONLINE FOOD DELIVERY

INDEX

SL.NO	DESCRIPTION	PAGE NO
01.	INTRODUCTION	1 - 2
02.	SOFTWARE REQUIREMENT ANALYSIS	3 - 5
03	SOFTWARE REQUIREMENT SPECIFICATION	6-10
04	SOFTWARE AND HARDWARE CONFIGURATION	11-12
05.	SOFTWARE PROFILE	13 - 43
06.	SYSTEM DESIGN	44– 49
6.1	DESIGN DIAGRAMS – CLASS, USE-CASE, DFD, ERD, SEQUENCE, SCHEMA, ACTIVITY, DEPLOYMENT	49 - 54
07.	SYSTEM TESTING	55- 58
7.1	SCREENSHOTS	59-70
09.	SYSTEM IMPLEMENTATION - CODING	71-113
10.	CONCLUSION and FUTURE ENHANCEMENTS	114 - 116
11.	BIBLIOGRAPHY	117-118



PRESIDENCY COLLEGE

(AUTONOMOUS)

AFFILIATED TO BENGALURU CITY UNIVERSITY, APPROVED BY AICTE, DELHI & RECOGNISED BY THE GOVT. OF KARNATAKA

RE-ACCREDITED BY NAAC WITH 'A+' GRADE

ONLINE FOOD DELIVERY

Introduction

ONLINE FOOD DELIVERY

1. INTRODUCTION

1.1 OBJECTIVE:

In today's fast-paced world, the food industry is undergoing a significant transformation with the advent of online platforms. The Online Food Ordering System emerges as a solution to meet the evolving needs of restaurants and customers alike. By leveraging the power of the internet, OFOS revolutionizes the way people discover, order, and enjoy food. This system caters to the diverse requirements of stakeholders within the food ecosystem. For restaurant owners and managers, it offers robust management tools to streamline operations, optimize menu offerings, and expand their digital presence. On the other hand, for customers, project provides a seamless and convenient way to explore culinary options, make informed decisions, and enjoy hassle-free ordering experiences. This paper delves into the architecture, functionalities, and benefits of the system. By examining its features such as user registration, restaurant search, menu browsing, and filtering options, we aim to highlight how the system enhances efficiency, accessibility, and satisfaction for both restaurants and customers. Additionally, we explore the technological advancements driving the adoption of such systems and the future prospects of online food ordering in the ever-evolving landscape of the food industry.

MAJOR MODULES:

Admin
User
Hotels
Delivery Boys

Functionalities of Modules

Admin can perform the following operations

Platform management oversight
Maintenance
Managing recruitment processes
Engaging with potential candidates with unparalleled efficiency.

Hotels can perform the following operations:

Registration and login
Update menu or Change menu



ONLINE FOOD DELIVERY

Add multiple Hotels
Raise complaints

Users can perform the following operations

Registration and login
Order food
View Hotels
Make payment
View and Update of complaints

Delivery Boys can perform the following operations

Delivery Boy Registration
Update Registered Details



Software Requirement Analysis

ONLINE FOOD DELIVERY

2. SOFTWARE REQUIREMENT ANALYSIS

System Analysis is a detailed study of the various operations performed by a system and their relationships within and outside of the system. Analysis begins when a developer begins a study of the program using existing system. During analysis, data are collected on the various files, decision points and transactions handled by the present system. The commonly used tools in the system are Data Flow Diagram, interviews, etc. Training, experience and common sense are required for collection of relevant information needed for the development of the system. The success of the system depends largely on how clearly the problem is defined, thoroughly investigated and properly carried out through the choice of solution.

2.1 EXISTING SYSTEM

The existing system of an online food delivery project typically comprises a combination of manual processes, disjointed platforms, and fragmented operations. In this traditional model, customers primarily place orders through phone calls or visiting restaurant websites directly. This method lacks the convenience and efficiency that modern consumers expect, often resulting in longer wait times and potential errors in order placement. For restaurants, managing orders from multiple channels, such as phone calls, walk-ins, and online orders, can be cumbersome and inefficient. Without a centralized system for order management, restaurant staff may struggle to keep track of incoming orders, leading to errors, delays, and potential customer dissatisfaction. Delivery logistics in the existing system are often handled manually or through third-party delivery services. This can lead to challenges in optimizing delivery routes, allocating drivers, and providing accurate delivery estimates to customers. Additionally, without real-time tracking capabilities, customers may experience uncertainty regarding the status and whereabouts of their orders.

2.2 PROPOSED SYSTEM

The proposed system for the online food delivery project aims to address the limitations of the existing system by introducing a comprehensive and integrated platform that streamlines the entire food ordering and delivery process. The proposed system leverages modern technology to enhance user experience, improve operational efficiency, and ensure seamless interactions between

ONLINE FOOD DELIVERY

customers, restaurants, and delivery personnel. At the core of the proposed system is a user-friendly and intuitive mobile and web application that allows customers to browse menus, place orders, track deliveries in real-time, and provide feedback on their experiences. The application offers a personalized experience, with features such as saved preferences, order history, and recommendations based on past orders and user preferences. For restaurants, the proposed system provides a centralized dashboard for managing menus, processing orders, and tracking sales and inventory in real-time. Restaurant owners can easily update menu items, prices, and availability, ensuring accuracy and consistency across all channels. Integration with the delivery management system allows restaurants to optimize delivery routes, assign drivers, and provide accurate delivery estimates to customers. Delivery logistics are streamlined through an intelligent routing and tracking system that optimizes delivery routes based on factors such as distance, traffic conditions, and delivery priorities. Delivery personnel receive real-time updates on order assignments, delivery addresses, and customer preferences, enabling them to provide efficient and timely service. Data management in the proposed system is centralized, with a robust database that stores menu information, customer profiles, order history, and transaction records. Advanced analytics capabilities provide insights into customer behavior, sales trends, and operational performance, empowering restaurants to make data-driven decisions and optimize their operations.

LITERATURE SURVEY

The application offers a personalized experience, with features such as saved preferences, order history, and recommendations based on past orders and user preferences. For restaurants, the proposed system provides a centralized dashboard for managing menus, processing orders, and tracking sales and inventory in real-time. Studies highlight the impact of mobile apps, user-friendly interfaces, and efficient logistics on customer satisfaction and service quality. Key research focuses on the role of AI and machine learning in optimizing delivery times, predicting demand, and enhancing personalization. The survey also addresses challenges like food quality control, delivery delays, and environmental concerns from packaging waste. The increasing importance of contactless delivery and the role of customer reviews in shaping choices are also examined.



Software Requirement Specifications

3. SOFTWARE REQUIREMENT SPECIFICATION

3.1 PYTHON

Python stands as a paramount figure in the realm of programming languages, revered for its unparalleled versatility, simplicity, and widespread adoption. Conceptualized by Guido van Rossum and unveiled in 1991, Python swiftly emerged as a cornerstone of modern software development, captivating programmers with its elegant syntax and pragmatic design principles. The hallmark of Python's design philosophy lies in its unwavering commitment to code readability, a trait accentuated using white space indentation to delineate code blocks—a departure from the cumbersome syntax characteristic of languages like C++ or Java. This emphasis on clarity empowers developers to express intricate concepts succinctly, rendering Python an ideal choice for projects ranging from small-scale scripts to large-scale enterprise applications. Beyond its aesthetic appeal, Python's strength lies in its support for an array of programming paradigms, including object-oriented, imperative, functional, and procedural styles. This versatility enables developers to adopt diverse approaches to problem-solving, adapting seamlessly to the unique requirements of each project. Moreover, Python's dynamic type system and automatic memory management alleviate the burden of manual memory allocation, liberating developers to focus on elegant crafting solutions rather than grappling with low-level memory concerns. Python's arsenal of features extends beyond its core language constructs to encompass an extensive standard library, replete with a plethora of pre-built modules catering to a myriad of domains. From web development frameworks like Django and Flask to data science tools such as NumPy and Pandas, Python's standard library serves as a treasure trove of resources for developers seeking to expedite development and harness the power of existing solutions. Additionally, Python's cross-platform compatibility ensures that code written in Python can seamlessly traverse various operating systems, Python continues to evolve through collaborative open-source endeavors and community-driven initiatives. The Python Software Foundation oversees the stewardship of C Python—the reference implementation of Python—ensuring its continual refinement and accessibility to developers worldwide. With its robust ecosystem, dynamic features, and unwavering commitment to simplicity and elegance, Python stands as a testament to the enduring power of a programming language designed with developers in mind. As technology evolves and

ONLINE FOOD DELIVERY

new challenges emerge, Python remains steadfast, poised to empower developers, and shape the future of software development for generations to come.

3.2 SQL

Microsoft SQL Server is a relational database management system developed by Microsoft. As a database server, it is a software product with the primary function of storing and retrieving data as requested by other software applications—which may run either on the same computer or on another computer across a network (including the Internet). Microsoft markets at least a dozen different editions of Microsoft SQL Server, aimed at different audiences and for workloads ranging from small single-machine applications to large Internet-facing applications with many concurrent users. Structured Query Language is a domain-specific language used in programming and designed for managing data held in a relational database management system (RDBMS), or for stream processing in a relational data stream management system (RDSMS). Originally based upon relational algebra and tuple relational calculus, SQL consists of a data definition language, data manipulation language, and data control language. The scope of SQL includes data insert, query, update and delete, schema creation and modification, and data access control. Although SQL is often described as, and to a great extent is, a declarative language (4GL), it also includes procedural elements. SQL was one of the first commercial languages for Edgar F. Codd's relational model, as described in his influential 1970 paper, "A Relational Model of Data for Large Shared Data Banks". Despite not entirely adhering to the relational model as described by Codd, it became the most widely used database language. SQL became a standard of the American National Standards Institute (ANSI) in 1986, and of the International Organization for Standardization (ISO) in 1987. Since then, the standard has been revised to include a larger set of features. Despite the existence of such standards, most SQL code is not completely portable among different database systems without adjustments.

4. HARDWARE SPECIFICATION

4.1 Server Infrastructure

Processor: Multi-core processor for handling concurrent user requests efficiently.

RAM: Sufficient memory to support the web application and database operations concurrently.

Storage: SSD storage for fast data access and retrieval.

Network Interface: Gigabit Ethernet interface for high-speed network connectivity.

ONLINE FOOD DELIVERY

Database Server

Database Management System: Compatible with Python Django ORM (e.g., PostgreSQL, MySQL, SQLite).

Storage: Adequate storage capacity for storing user data, job listings, and system logs.

RAM: Sufficient memory allocation for efficient database operations and query processing.

Backup Solution: Regular backup mechanism to prevent data loss in case of system failure.

4.3 Development Workstation

Processor: Multi-core processor for smooth development and testing workflows.

RAM: Sufficient memory for running the IDE (VSCode), web browser, and other development tools simultaneously.

Storage: SSD storage for fast code compilation and project loading times.

Operating System: Compatible with Windows, macOS, or Linux distributions for development.

4.4 Client Devices

- 1- Web Browser: Compatible with modern web browsers such as Chrome, Firefox, Safari, and Edge for accessing the online job portal.
- 2- Display: Support for various screen resolutions and sizes for optimal viewing experience.
- 3- Internet Connectivity: Reliable internet connection for accessing the portal from anywhere.

4.5 TOOLS

- Front End Tool: HTML, CSS, JavaScript: For building the user interface and client-side functionalities.
- React.js or Angular: Frameworks for developing dynamic and interactive web applications.
- Bootstrap or Materialize CSS: Front-end frameworks for responsive design and UI components.
- VSCode or WebStorm: Integrated development environments (IDEs) for front-end development.
- Back End Tool: Python Django: Back end framework for building web applications with a clean and pragmatic design.

ONLINE FOOD DELIVERY

- Django REST Framework: Toolkit for building Web API s in Django, facilitating communication between the front end and back end.
- PostgreSQL or MySQL: Relational database management systems for storing and managing application data. Redis: In-memory data store for caching and session management. VSCode or PyCharm: IDEs for Python development and debugging.
- Operating System: Linux (Ubuntu, CentOS): Stable and reliable operating system commonly used for web server hosting and development. Windows: Suitable for development environments, with support for various development tools and IDEs.



H/W and S/W Configurations

ONLINE FOOD DELIVERY

4. SOFTWARE AND HARDWARE CONFIGURATION

HARDWARE	
Processor	Intel Core i3
RAM	4GB
HDD	500GB
SOFTWARE	
CLIENT SIDE TECHNOLOGIES	HTML, CSS, JAVASCRIPT
SERVER SIDE TECHNOLOGIES	PYTHON DJANGO, DJANGO REST FRAMEWORK
DATABASE	SQLyog ULTIMATE
WEB SERVER	XAMPP CONTROL PANEL
IDE	VISUAL STUDIO CODE or PYCHARM
WEB AUTHORIZING TOOLS	PHOTOSHOP, FLASH



PRESIDENCY COLLEGE
(AUTONOMOUS)

AFFILIATED TO BENGALURU CITY UNIVERSITY, APPROVED BY AICTE, DELHI & RECOGNISED BY THE GOVT. OF KARNATAKA
RE-ACCREDITED BY NAAC WITH 'A+' GRADE

ONLINE FOOD DELIVERY

Software Profile

ONLINE FOOD DELIVERY

5. SOFTWARE PROFILE

5.1 ABOUT HTML/XHTML

HTML (Hyper Text Markup Language) :

HTML was created by Tim Berners-Lee at European Laboratory for Particle Physics (CERN) in late 1980's. Developed by the World Wide Web Consortium; HTML or the Hyper Text Markup Language, as its name suggests, is a markup language for Web pages. Today, the most important component of any web page is the text-based information that it contains. The markup tags of HTML define the structure of the text-based information of a web page. HTML tags are used to denote various text-based information of a web page as paragraphs, headings, links, bullet points etc. Various HTML tags can also be used to supplement the text with images, forms and other objects. HTML tags are for browsing; they are meant for interactions between humans and computers.

Hardware Interface:

Client side:

Processor: Intel Core i3.

RAM: 2GB.

Network Interface.

Server Side:

Processor: Intel Core i3.

RAM: 2GB.

Disk space: 2GB.

Software Interface

Client side:

Windows XP/ Vista/ Win 7.

Internet Explorer 6.0 or above

Network Interface.

Server Side:

Apache Tomcat Web Server 7.0.1.

Oracle 10g Database as Back End.

J2EE Framework.



ONLINE FOOD DELIVERY

HTML draft version timeline

October 1991

HTML Tags an informal CERN document listing eighteen HTML tags, was first mentioned in public.

June 1992

First informal draft of the HTML DTD with seven subsequent revisions (July 15, August 6, August 18, November 17, November 19, November 20, November 22)

November 1992

HTML DTD 1.1 (the first with a version number, based on RCS revisions, which start with 1.1 rather than 1.0), an informal draft

June 1993

Hypertext Markup Language- was published by the IETF IIR Working Group as an Internet-Draft (a rough proposal for a standard). It was replaced by a second version- one month later, followed by six further drafts published by IETF itself- that finally led to HTML 2.0 in RFC1866

November 1993

HTML+ was published by the IETF as an Internet-Draft and was a competing proposal to the Hypertext Markup Language draft. It expired in May 1994.

April 1995 (authored March 1995)

HTML 3.0- was proposed as a standard to the IETF, but the proposal expired five months later without further action. It included many of the capabilities that were in Raggett's HTML+ proposal, such as support for tables, text flow around figures and the display of

January 2008

ONLINE FOOD DELIVERY

HTML5 was published as a Working Draft by the W3C.

Although its syntax closely resembles that of SGML, HTML5 has abandoned any attempt to be an SGML application and has explicitly defined its own "html" serialization, in addition to an alternative XML-based XHTML5 serialization.

Characteristics: -

It is the language which can be easily understood and can be modified.

Effective presentations can be made with the HTML with the help of its all formatting tags.

It provides the more flexible way to design web pages along with the text.

Links can also be added to the web pages so it helps the readers to browse the information of their interest.

You can display HTML documents on any platforms such as Macintosh, Windows and Linux etc.

Graphics, videos and sounds can also be added to the web pages which give an extra attractive look to your web pages. Allows database integration with wide variety of applications.

Additional internet capabilities.

XHTML Versions: -

XHTML is a separate language that began as a reformulation of HTML 4.01 using XML 1.0. It continues to be developed:

XHTML 1.0, published January 26, 2000, as a W3C Recommendation, later revised and republished August 1, 2002. It offers the same three variations as HTML 4.0 and 4.01, reformulated in XML, with minor restrictions.

XHTML 1.1 published May 31, 2001, as a W3C Recommendation. It is based on XHTML 1.0 Strict, but includes minor changes, can be customized, is reformulated using modules from Modularization of XHTML, which was published April 10, 2001, as a W3C Recommendation.

XHTML 2.0 There is no XHTML 2.0 standard. XHTML 2.0 is only a draft document and it is inappropriate to cite this document as other than work in progress. XHTML 2.0 is incompatible with

ONLINE FOOD DELIVERY

XHTML 1.x and, therefore, would be more accurately characterized as an XHTML-inspired new language than an update to XHTML 1.x.

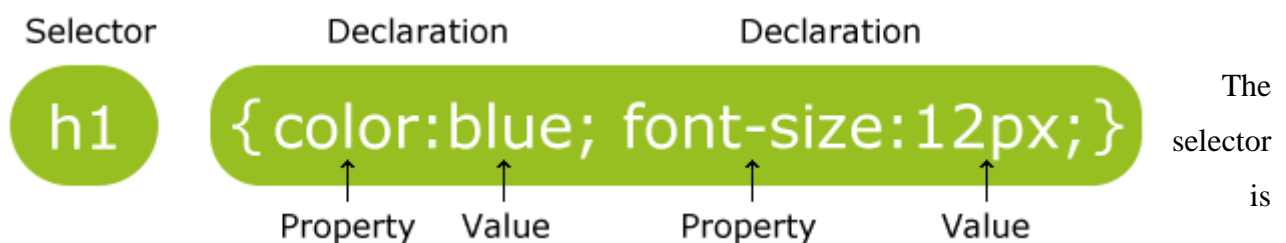
5.2 CASCADING STYLE SHEET



Cascading Style Sheets (CSS) is a style sheet language used to describe the presentation semantics (the look and formatting) of a document written in a markup language. It's most common application is to style web pages written in HTML and XHTML, but the language can also be applied to any kind of XML document, including plain XML, SVG and XUL.

CSS defines HOW HTML elements are to be displayed. Styles are normally saved in external .css files. External style sheets enable you to change the appearance and layout of all the pages in a Web site, just by editing one single file! The CSS files referenced in the HTML page.

A CSS rule has two main parts: a selector, and one or more declarations:



normally the HTML element you want to style. Each declaration consists of a property and a value. The property is the style attribute you want to change. Each property has a value.

Following example formats a paragraph in an HTML document

```
p
{
    color:red;
    text-align:center;
}
```

5.3 JAVASCRIPT



JavaScript was originally developed by Brendan Eich of Netscape under the name Mocha, which was later renamed to LiveScript, and finally to JavaScript.

JavaScript is a prototype-based, object-oriented Scripting Language that is dynamic, weakly typed and has first-class functions. It is also considered a functional programming language because it has closures and supports higher-order functions. JavaScript is primarily used in the form of client-side JavaScript, implemented as part of a web browser in order to provide enhanced user interfaces and dynamic websites.

The primary use of JavaScript is to write functions that are embedded in or included from HTML pages and that interact with the Document Object Model (DOM) of the page. Some simple examples of this usage are:

Opening or popping up a new window with programmatic control over the size, position, and attributes of the new window (e.g. whether the menus, toolbars, etc. are visible).

Validating input values of a web form to make sure that they are acceptable before being submitted to the server.

Changing images as the mouse cursor moves over them: This effect is often used to draw the user's attention to important links displayed as graphical elements.

EXAMPLE:

a) A simple recursive function:

```
function factorial(n)
{
    if (n === 0)
    {
        return 1;
    }
    return n * factorial(n - 1);
}
```


5.4 DJANGO



Django is a high-level, open-source web framework written in Python, designed to simplify the process of building dynamic web applications. It follows the Model-View-Template (MVT) architectural pattern, which separates data management, user interface, and application logic into distinct components. Django's primary goal is to encourage rapid development and ensure clean, maintainable code by following key principles like Don't Repeat Yourself (DRY) and Convention over Configuration.

Key Features of Django

Model-View-Template (MVT) Architecture

Built-in Admin Interface

Security Features

Scalability and Flexibility

URL Routing

Templating System

Extensible with Third-Party Packages

Testing Framework

Internationalization (i18n)

Django is a powerful framework that makes web development easier and faster, especially for content-heavy applications or projects requiring robust security features. Its “batteries-included” approach, along with an active community and a strong focus on best practices, makes it one of the most popular choices for web developers worldwide.



ONLINE FOOD DELIVERY

Characteristics of Django

- “Batteries-Included” Framework
- Rapid Development
- Security-Focused
- Secure Password Storage
- Scalability
- Versatile and Flexible
- Object-Relational Mapper (ORM)
- Admin Interface
- URL Routing
- Templating Engine
- Middleware
- Internationalization (i18n)

5.5 Django REST framework



Django REST Framework (DRF) is a powerful and flexible toolkit for building Web APIs in Django. It allows developers to easily build RESTful APIs (Representational State Transfer), which are used to interact with data over the web using standard HTTP methods (GET, POST, PUT, DELETE).

DRF is built on top of Django and is one of the most popular tools for creating APIs. It provides abstractions to handle common API tasks like serialization, authentication, permissions, request parsing, and response rendering, making it easier to create maintainable and salable APIs.

Characteristics Of Django REST framework

Ease of Serialization

Authentication and Permissions

Throttling

Pagination

Filtering, Searching, and Ordering

Browsable API

Content Negotiation

ONLINE FOOD DELIVERY

5.6 XAMPP (Web Server)



XAMPP is a free and open-source cross-platform web server solution stack package developed by Apache Friends. It consists of the Apache HTTP Server, MariaDB (a fork of MySQL), and interpreters for scripts written in PHP and Perl. The name "XAMPP" is an acronym where:

- X stands for cross-platform (as it can run on various operating systems, such as Windows, Linux, and macOS).
- A stands for Apache, the web server.
- M stands for MariaDB, the database system.
- P stands for PHP, a server-side scripting language.
- P stands for Perl, another scripting language.

Components of XAMPP:

Apache HTTP Server: The web server component. It allows you to serve web pages over the internet or intranet. It's one of the most widely used web server solutions.

MariaDB: Previously, XAMPP included MySQL as its database management system, but now it uses MariaDB. MariaDB is fully compatible with MySQL, and it is often used to manage databases for dynamic websites and applications.

PHP: PHP is a server-side scripting language designed for web development. With XAMPP, you can develop and test PHP-based applications locally before deploying them to a live server.

Perl: Perl is another programming language supported by XAMPP, often used for system administration, web development, and network programming.

ONLINE FOOD DELIVERY

Apache HTTP Server, commonly known as Apache, is an open-source and widely-used web server software maintained by the Apache Software Foundation. It enables computers to serve web pages to users over the internet or an intranet, making it one of the most critical technologies for web hosting.

Key Features:

Cross-Platform: Apache works on various operating systems, including Windows, Linux, Unix, and macOS.

Modular Architecture: Apache has a modular structure, allowing administrators to customize its functionality. Modules can be loaded or unloaded to add features like URL rewriting, SSL support, or dynamic content handling.

Virtual Hosting: Apache supports both IP-based and name-based virtual hosting, allowing multiple websites to be hosted on a single server.

Support for Scripting Languages: Apache can serve static content (like HTML pages), but it also supports dynamic content through modules like `mod_php` (for PHP), `mod_perl` (for Perl), and `mod_proxy` (for reverse proxy functionality).

Security Features: Apache offers various security modules, including SSL/TLS encryption with `mod_ssl`, authentication, and access control. It also supports HTTPS to secure data transfer.

Custom Configuration: Configuration of Apache is done through text files, mainly `httpd.conf` and `.htaccess` files. These allow customization of nearly every aspect of server behavior, including URL redirection, caching, and file permissions.

Performance: Apache supports load balancing and caching mechanisms, helping to optimize server performance and distribute traffic efficiently across multiple servers.

ONLINE FOOD DELIVERY

Community Support: As an open-source project, Apache has a vast community of users and contributors, which ensures frequent updates, patches, and a wide array of online resources for troubleshooting and learning.

How It Works:

Apache receives requests from clients (typically browsers) and processes them, returning the requested files (like HTML, images, or scripts) or dynamic content from the server. It operates based on a request-response model, with support for the HTTP/1.1, HTTP/2, and HTTPS protocols.

Use Cases:

Web Hosting: Apache is used by individuals and organizations worldwide to host websites and applications.

Local Development: Many developers use Apache for setting up local development environments.

Load Balancing: With its ability to distribute traffic, Apache is used to ensure high availability and load distribution for large websites.

Advantages:

Highly Configurable: Through its modules and configuration files, Apache allows extensive customization.

Reliable and Stable: Apache has been around since 1995 and is known for its stability and robustness.

Wide Support for Various Technologies: It integrates seamlessly with popular technologies like PHP, Python, Perl, and more.

ONLINE FOOD DELIVERY

Disadvantages:

Resource-Intensive: Compared to lightweight servers like Nginx, Apache can consume more system resources.

Performance in High Traffic: Apache may struggle with high-traffic websites compared to other web servers designed for speed, such as Nginx.

Conclusion:

Apache HTTP Server is one of the most popular and versatile web servers in use today. Its modular nature, cross-platform compatibility, and wide range of features make it suitable for everything from personal websites to large-scale enterprise applications.

MariaDB is an open-source relational database management system (RDBMS) that is a fork of MySQL. It was created by the original developers of MySQL after Oracle Corporation acquired MySQL, with concerns that MySQL's development might become less community-driven under Oracle's ownership. MariaDB aims to remain fully open-source and compatible with MySQL, while offering enhancements in performance, scalability, and new features.

Key Features:

Compatibility with MySQL:

MariaDB maintains backward compatibility with MySQL. Most MySQL applications can run on MariaDB without modification.

It uses the same SQL syntax, libraries, APIs, and data files as MySQL, making migration seamless.

Storage Engines:

ONLINE FOOD DELIVERY

MariaDB offers a variety of storage engines, including InnoDB (for transactional support) and MyISAM (for performance in read-heavy operations).

Additional engines like Aria (designed for crash recovery) and ColumnStore (for analytical workloads) expand MariaDB's capabilities.

Improved Performance:

MariaDB has several optimizations that improve query performance over MySQL, including faster replication, better thread handling, and optimization for large-scale environments.

It supports advanced caching mechanisms and query optimizations, improving execution speeds for complex queries.

Security Enhancements:

- MariaDB includes advanced security features such as role-based access control (RBAC), encryption at the column-level, and support for SSL/TLS encryption for data in transit.
- Enhanced user account management, including more sophisticated password policies, is also a notable feature.

Scalability:

MariaDB supports master-slave replication and Galera Cluster technology for multi-master replication. This allows for high availability and scalability across multiple servers.

It also includes thread pool management, which enables it to efficiently handle thousands of concurrent connections, making it ideal for large-scale applications.

Open Source:

Unlike MySQL (which has a commercial version), MariaDB remains 100% open-source. It is developed and maintained by a global community of contributors.

ONLINE FOOD DELIVERY

New Features:

MariaDB continues to introduce new features that aren't available in MySQL, such as JSON functions, virtual columns, and enhanced SQL syntax for more complex queries.

How MariaDB Works:

MariaDB, like other relational databases, stores data in structured tables with rows and columns. It uses Structured Query Language (SQL) to manage and retrieve data from these tables. You can perform a wide range of database operations, from basic queries to complex transactions, using MariaDB.

Use Cases:

Web Applications: It powers a variety of dynamic websites, from small blogs to large-scale applications.

Data Warehousing: With features like the ColumnStore engine, MariaDB is used in analytical applications and big data processing.

Enterprise Applications: Due to its scalability and performance, MariaDB is commonly used in enterprise environments for business-critical applications.

Advantages:

Free and Open-Source: There are no commercial versions, making it cost-effective for both small businesses and large enterprises.

Strong Community Support: MariaDB is backed by a large and active open-source community.

ONLINE FOOD DELIVERY

MySQL Compatibility: Existing MySQL applications can be easily migrated to MariaDB with minimal or no changes.

Enhanced Features: MariaDB introduces new features and performance improvements regularly, which gives it an edge over MySQL in some areas.

Disadvantages:

Potential for Divergence: While MariaDB is currently compatible with MySQL, over time, it may diverge as both projects evolve independently, leading to possible incompatibilities.

Adoption: Although growing, MariaDB's market share is still smaller than MySQL, meaning some organizations or hosting services may prefer MySQL.

Conclusion:

MariaDB is a powerful, flexible, and open-source alternative to MySQL, offering improvements in performance, scalability, and security. Its compatibility with MySQL makes it an easy choice for those looking to switch or start with an open-source database solution while benefiting from continued innovation and support from the community.

Additional Features:

phpMyAdmin: A tool included in XAMPP for managing MariaDB databases via a web interface. It allows you to perform tasks such as creating and modifying databases, tables, fields, and rows.

OpenSSL: A toolkit for the Secure Sockets Layer (SSL) and Transport Layer Security (TLS) protocols, enabling encrypted communication.

ONLINE FOOD DELIVERY

FileZilla FTP Server: XAMPP includes the FileZilla FTP server, which allows you to transfer files using FTP (File Transfer Protocol).

Mercury Mail: XAMPP includes the Mercury Mail Transport System, which allows you to set up and test email-related functionalities on your local machine.

Tomcat: XAMPP supports Java-based web applications through Apache Tomcat.

Uses of XAMPP:

Web Development: XAMPP allows developers to simulate a live web server on their local machine. It is commonly used for the development and testing of websites before launching them on a production server.

Testing Environments: It provides an easy way to create local server environments for testing PHP and MySQL applications without requiring an active internet connection.

Learning: XAMPP is useful for beginners to practice developing PHP applications and managing databases since it provides an all-in-one solution for creating a local development server.

Advantages:

Cross-Platform Compatibility: It works on multiple operating systems (Windows, Linux, macOS).

Ease of Installation: XAMPP comes pre-packaged with everything you need to run a local server, significantly simplifying the setup process.

Lightweight: It's designed to run efficiently on personal computers.

Open Source: It's free and open-source, making it accessible for everyone, from beginner developers to professionals.

ONLINE FOOD DELIVERY

How to Use XAMPP:

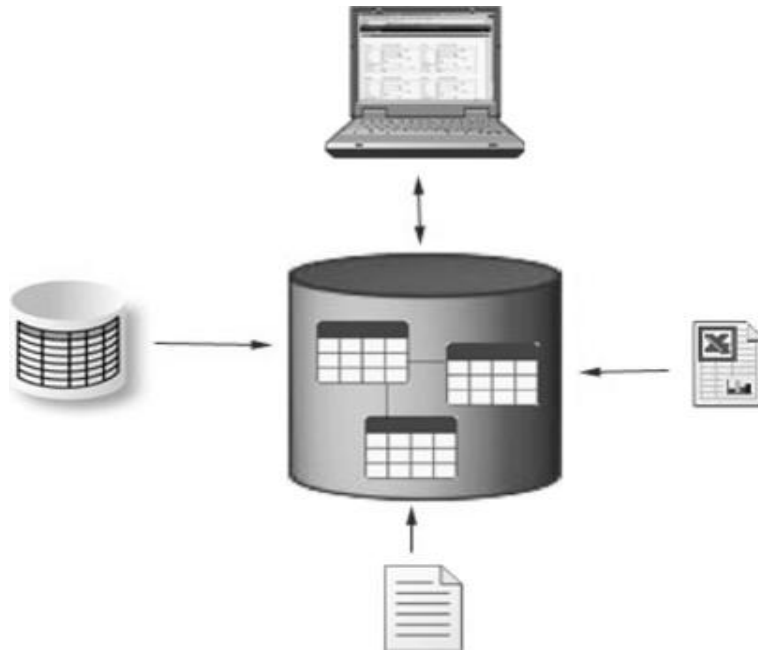
- **Installation:** Download the appropriate version of XAMPP for your operating system from the Apache Friends website.
- **Starting the Server:** After installation, you can start/stop various services (like Apache, MariaDB) via the XAMPP control panel.
- **Deploying Applications:** You can place your PHP applications in the ht docs folder of the XAMPP installation directory. From there, you can access your web projects via a browser by entering local host followed by the project folder name.

Disadvantages:Security Issues: XAMPP is not recommended for production environments because its default settings are not secure. It's meant for local development environments only.

Resource Intensive: While it's relatively lightweight, running XAMPP on older or less powerful machines can sometimes cause performance issues.

Conclusion:XAMPP is an excellent tool for developers looking to set up a local server environment with minimal effort. It's widely used for testing and developing dynamic websites, especially for those using PHP and MySQL/MariaDB. However, it's designed specifically for development purposes and not for production use due to potential security risks.

5.7 DATABASE MANAGEMENT SYSTEM



A database management system (DBMS) is a collection of programs that enables users to create and maintain a database. This is a software system that allows access to the data contained in the database. The primary goal of a DBMS is to provide an environment that is both convenient and efficient to use in storing and retrieving database information.

Functions of DBMS:-

DBMS is a general purpose software system that performs the following functions:-

Defining a database.

Constructing the database.

Manipulating the database.

Sharing database among various users.

Protecting the database.

Maintaining a database.

ONLINE FOOD DELIVERY

DBMS Architecture:-

DBMS architecture consists of three levels known as Three Schema Architecture. It is convenient tool with which the user can visualize the schema levels in a database system. It contains of the following three schemas:-

The Internal level: - This contains of an internal schema, which describes the physical storage structure of the database. It is the lowest level of abstraction. It does not hide the storage details. It contains the definition of the stored record, the method of representing the data fields and the access aids used. This internal schema uses a physical data model and describes the complete details of data storage and access paths for the database. It is also called the physical schema.

The Conceptual level: - This has a conceptual schema, which describes the structures of a database for a group of users. This schema hides the storage details from the user and it includes description of entities, data types, relationships, user operations and constraints. The description of data at this level is in a format independent of its physical representation.

The External level: - This has a number of external schemas or user views. Each external schema describes the part of the database that a particular user group is interested in and hides all the other details from this group. This is at a highest level of database absorption.

ONLINE FOOD DELIVERY

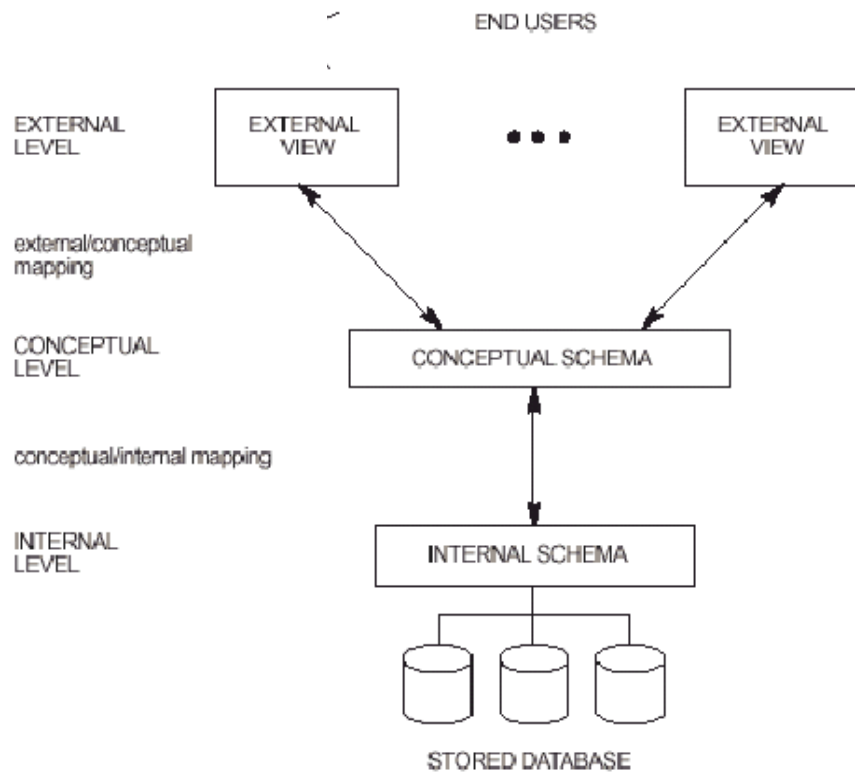


Fig: - Three schema DBMS architecture

Client-Server Architecture: -

The client/server architecture was developed to deal with computing environments in which a large number of PCs, workstations, file servers, printers, database servers, Web servers and other equipment are connected via a network. There are two client/server architecture:-

Two-tier

Three-tier

N-tier

Two-Tier Client/Server Architecture: -

In two tier architecture, the software components are distributed over two systems: the client and the server. This architecture has two forms as: logical two-tier and physical two-tier.

ONLINE FOOD DELIVERY

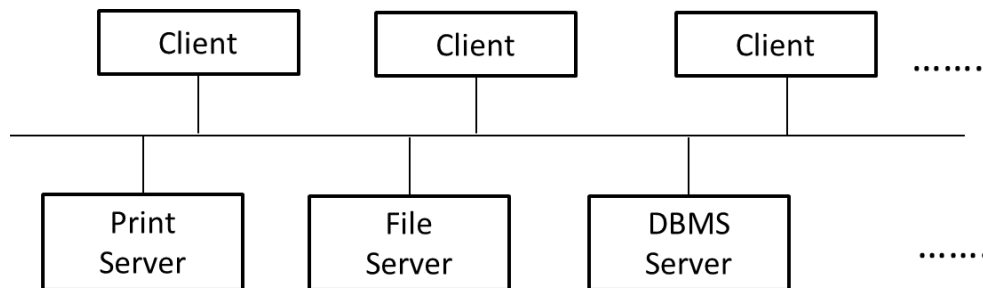


Fig: - Logical two-tier client/server architecture

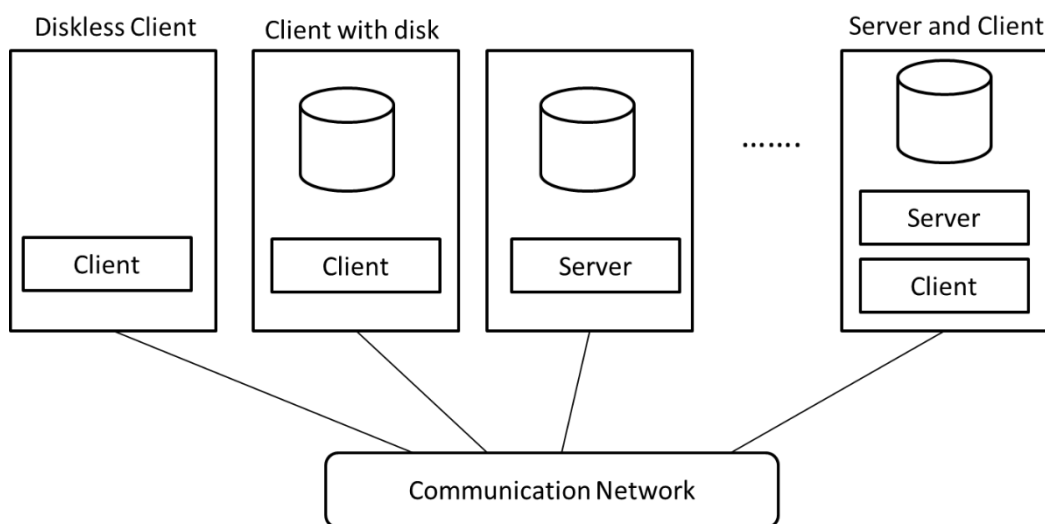


Fig: - Physical two-tier client/server architecture

The above figure shows the physical client/server architecture. Some machines like diskless workstations, or workstations/PCs with disks that have only client software installed would be only client sites. Other machines would be dedicated servers. Some other machines would have both client and server functionality. In relational DBMS, user interface and application programs can run at client side. The query and transactions functionality are included on the server side. A client program can send query and transaction requests using the ODBC API which are then processed at the server site. The query results are sending back to the client program which can process or display the results needed.

ONLINE FOOD DELIVERY

Three-Tier Client/Server Architecture: -

With the emergence of World Wide Web, many web applications use the three-tier architecture. There is an intermediate layer between the client and the database server called the application server or the web server. The web server plays the intermediary role by storing business rules that are used to access data from the database server. It checks the client's credentials before forwarding a request to the database server. The intermediate server accepts requests from the client, process the request and sends database commands to the database server, and then acts as a conduit for passing processed data from the database server to the clients, where it may be processed further.



Fig: - Logical three-tier client/server architecture

N-Tier Architecture: -

In N-Tier architecture, the middle tier is allowed to have multiple application objects rather than a single application. Each of these application objects must have a well-defined interface which allows them to contact and communication with one another. An interface actually brings an idea of contract. That is, each object states through its interface that it will accept certain parameters and return a specific set of results.

Application objects use their interfaces to do business processing. With and N-Tier architecture, one can have multiple applications using a common set of business objects across an organization. This promotes the standardization of business practices by creating a single set of business functions for the entire organization to access. If a particular business rule changes, then changes have to be made to only the business object and if need, to its interface also.

ONLINE FOOD DELIVERY

Normalization: - Normalization is a process during which unsatisfactory relation schemas are decomposed by breaking up their attributes into smaller relation schemas that possess desirable properties. Normalization of data can be looked upon as a process of analyzing the given relation schemas based on their functional dependencies and primary keys to achieve the desirable properties of minimizing redundancy, insertion, deletion and update anomalies.

The different Normal Forms present in DBMS are: -

First Normal Form: - It states that the domains of attributes must include only atomic values and that the value of any attribute in a tuple must be single value from the domain of that attribute. It disallows a set of values, a tuple of values, or a combination of both as an attribute value for a single tuple.

Second Normal Form: - A relation is said to be in Second Normal Form if it is in 1NF and non-key attributes are fully functionally dependent on the key attribute(s). If the key has more than one attribute (composite key) then no non-key attributes should be functionally dependent upon a part of the key attributes.

Third Normal Form: - A relation is said to be in Third Normal Form if it is in 2NF and no non-prime attributes of relation R is transitively dependent on the primary key.

Fourth Normal Form: - Under this, a record type should not contain two or more independent multi-valued facts about an entity. In addition, the record must satisfy third normal form.

Fifth Normal Form: - It also deals with multi-valued facts. Here, the record must satisfy the fourth normal form.

Boyce Codd Normal Form: - It is a [normal form](#) used in [database normalization](#). It is a slightly stronger version of the [third normal form](#) (3NF). A table is in Boyce–Codd normal form [if and only if](#) for

ONLINE FOOD DELIVERY

every one of its [nontrivial dependencies](#) $X \rightarrow Y$, X is a [super key](#)—that is, X is either a [candidate key](#) or a superset thereof

SQLyog ULTIMATE



SQLyog Ultimate is a comprehensive graphical user interface (GUI) tool for managing and interacting with **MySQL** and **MariaDB** databases. Developed by **Webyog**, it provides a wide range of features designed to simplify database administration, querying, development, and performance optimization. SQLyog is particularly popular among developers, database administrators, and data analysts due to its ease of use and robust feature set. SQLyog Ultimate is the premium, full-featured edition of the SQLyog toolset.

Below is a detailed summary of SQLyog Ultimate

1. Overview of SQLyog Ultimate

Type: SQLyog Ultimate is a **desktop-based MySQL/MariaDB management tool**.

Platform: It runs on **Windows** (but can manage databases hosted on remote Linux servers) and integrates seamlessly with MySQL and MariaDB databases.

Primary Functionality: It is used to administer, monitor, and optimize MySQL/MariaDB databases, execute queries, design database schemas, and perform data migrations.

ONLINE FOOD DELIVERY

Audience: Ideal for **database administrators (DBAs)**, **developers**, and **analysts** who require powerful MySQL management and development capabilities.

Key Features of SQLyog Ultimate

a. Visual Query Builder

SQLyog Ultimate includes a **drag-and-drop query builder** that allows users to visually create SQL queries without needing in-depth SQL knowledge. This feature helps to quickly generate SELECT, INSERT, UPDATE, and DELETE statements.

It supports **joins**, **sub-queries**, and **conditions**.

b. Advanced Query Execution and Management

Intelli-Sense: As you type SQL queries, SQLyog offers auto-completion and suggestions, which helps reduce errors and improve productivity.

Multiple Query Tabs: Users can execute and manage multiple SQL queries in different tabs within the same session.

Query Profiler: Provides **query profiling** and optimization by highlighting slow queries, allowing you to analyze and tune SQL queries for better performance.

Execution History: Allows you to view, save, and reload previous SQL queries for future use.

c. Schema and Database Design

SQLyog provides a **schema designer** with a visual interface to design database tables, indexes, and relationships.

You can easily manage database schema changes with the **Database Synchronization Wizard**, which compares the schema of two databases and generates scripts to bring them into sync.

ONLINE FOOD DELIVERY

Reverse Engineering: Allows you to reverse-engineer an existing database to create a visual representation (ER diagram) of its structure.

d. Data Management

- **Import/Export Data:** SQLyog offers powerful tools to import data from multiple sources like CSV, Excel, or other database formats, and export data to a variety of formats.
- **Scheduled Data Synchronization:** The **Data Sync Tool** lets you synchronize data between two databases, either locally or on a remote server. You can automate these syncs on a schedule.
- **Scheduled Backups:** Automate the backup process of your databases, allowing for easy restoration in the event of data loss.
- **Data Migration:** SQLyog supports migration from other database platforms like **Microsoft Access, SQL Server, PostgreSQL, or SQLite.**
- **e. Database Monitoring and Management**
- **Real-Time Monitoring:** SQLyog Ultimate includes tools for monitoring the performance of MySQL databases, showing important performance metrics like server health, open connections, query statistics, and more.
- **Process Manager:** Allows you to view and manage server processes and **kill problematic queries** directly from the GUI.

User Management: Provides an easy way to manage MySQL users and permissions, including creating new users, modifying existing users, and managing roles and privileges.

Replication Monitoring: SQLyog monitors **MySQL replication** environments, showing the status of replication servers and alerts for replication errors.

ONLINE FOOD DELIVERY

Scheduled Jobs: SQLyog Ultimate can run jobs automatically on a scheduled basis. You can set up scripts to run regularly for backups, data synchronization, query execution, etc.

SSH and HTTP Tunneling: It provides secure tunneling methods like **SSH** and **HTTP** for remote database management, ensuring secure connectivity even across the internet.

Visual Data Comparison: The **Visual Data Compare Tool** allows users to compare and synchronize data between different databases or tables visually. This feature is crucial for **data consistency** across environments.

g. Migration and Backup Tools

SQLyog Ultimate includes built-in tools for **database migration** from other platforms to MySQL/MariaDB.

Scheduled Backup: You can schedule regular backups of databases and restore them easily when needed.

h. Performance Optimization

Query Profiler: SQLyog includes an advanced query profiler that helps analyze and optimize SQL queries. It identifies slow-running queries and suggests optimizations.

Explain Plan: The tool uses MySQL's **EXPLAIN** command to analyze the execution plan for queries, giving insight into how MySQL handles your queries and helping to optimize database performance.

Index Management: Provides tools for creating, modifying, and deleting indexes to improve query performance.

ONLINE FOOD DELIVERY

i. Report Generation

SQLyog can generate **customized reports** using query results and export these reports in formats like CSV, HTML, or XML. This feature is useful for data analysis and business reporting.

j. Enterprise-Level Features

SQLyog Ultimate also supports advanced features suitable for **large enterprise environments**:

Sharding: SQLyog supports database sharding for better scalability.

Partitioning: Manage large datasets efficiently by partitioning tables.

Audit Log Analysis: Tracks all database activities and provides auditing capabilities for security and compliance.

3. Security Features

Secure Connections: SQLyog provides support for encrypted connections via **SSL** and **SSH Tunneling**, ensuring safe access to databases over insecure networks.

User Authentication: Manage user roles and permissions directly from the GUI, including password management and privilege assignment.

Data Masking: For sensitive data, SQLyog provides tools to mask and anonymize data in non-production environments.

4. User Interface and Ease of Use

SQLyog Ultimate is known for its **clean and intuitive interface**. Users can manage databases using point-and-click operations without needing to write complex SQL commands.

ONLINE FOOD DELIVERY

The **tabbed interface** enables easy management of multiple connections and query tabs, improving workflow efficiency.

Drag-and-Drop Features: Many operations, such as table design and query building, support drag-and-drop actions, making it easy for users to build queries and design databases visually.

5. Integration with Other Tools

SQLyog can be easily integrated with **version control systems** for database versioning and auditing.

It also supports **integration with third-party tools** like monitoring platforms, analytics tools, and reporting software, expanding its utility in enterprise environments.

6. Licensing and Pricing

- SQLyog Ultimate is a **commercial product**, which requires a license. It is typically sold as a one-time purchase or via subscription, depending on the version or package.
- It offers different pricing tiers depending on whether it's being used for personal, small business, or enterprise purposes.
- A free version, **SQLyog Community Edition**, exists with limited features, but the Ultimate version includes the full range of advanced capabilities.

7. Use Cases

Database Administration: SQLyog Ultimate is ideal for managing MySQL and MariaDB databases in production environments. **Development:** Its tools for query building, schema design, and debugging make it a favorite among developers working with MySQL/MariaDB. **Backup and Recovery:** SQLyog is widely used for database backup, recovery, and data migration between different database platforms. **Performance Optimization:** DBAs use SQLyog to monitor and optimize database performance by identifying and troubleshooting slow queries and poorly optimized indexes.

ONLINE FOOD DELIVERY

8. Alternatives to SQLyog Ultimate

MySQL Workbench: Another popular free MySQL management tool that offers similar functionality.**HeidiSQL:** A lightweight, free tool for managing MySQL databases.

Toad for MySQL: A commercial product offering MySQL database management features with a focus on performance tuning.**Navicat:** A commercial database management tool with broader support for databases like MySQL, PostgreSQL, SQLite, etc.

Conclusion:SQLyog Ultimate is a powerful and feature-rich MySQL/MariaDB database management tool that caters to both beginners and advanced users. Its GUI makes database administration simpler, and it offers a comprehensive range of tools for database querying, optimization, synchronization, and security. With advanced features like query profiling, real-time monitoring, and secure tunneling, SQLyog Ultimate is a preferred tool for many database administrators and developers looking for an all-in-one MySQL management solution.



PRESIDENCY COLLEGE
(AUTONOMOUS)

AFFILIATED TO BENGALURU CITY UNIVERSITY, APPROVED BY AICTE, DELHI & RECOGNISED BY THE GOVT. OF KARNATAKA
RE-ACCREDITED BY NAAC WITH 'A+' GRADE

ONLINE FOOD DELIVERY

System Design

ONLINE FOOD DELIVERY

6. SYSTEM DESIGN

6.1 DATABASE SCHEMA DESIGN:

Create Table Login

```
CREATE TABLE `foodapp_login` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `username` varchar(225) NOT NULL,  
  `password` varchar(225) NOT NULL,  
  `usertype` varchar(225) NOT NULL,  
  PRIMARY KEY (`id`)  
) ENGINE=MyISAM AUTO_INCREMENT=32 DEFAULT CHARSET=latin1 COLLATE=latin  
1_swedish_ci
```

Create Table Menu

```
CREATE TABLE `foodapp_menu` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `food_name` varchar(225) NOT NULL,  
  `image` varchar(225) NOT NULL,  
  `price` varchar(225) NOT NULL,  
  `quantity` varchar(225) NOT NULL,  
  `status` varchar(225) NOT NULL,  
  `category_id` int(11) NOT NULL,  
  `hotel_id` int(11) NOT NULL,  
  PRIMARY KEY (`id`),  
  KEY `foodapp_menu_category_id_a295d627` (`category_id`),  
  KEY `foodapp_menu_hotel_id_2b374b1c` (`hotel_id`)
```

ONLINE FOOD DELIVERY

```
) ENGINE=MyISAM AUTO_INCREMENT=24 DEFAULT CHARSET=latin1 COLLATE=latin1  
_swedish_ci
```

create table smartcitizen

Create Table Passanger

```
CREATE TABLE `foodapp_passenger` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `fname` varchar(225) NOT NULL,  
  `lname` varchar(225) NOT NULL,  
  `hname` varchar(225) NOT NULL,  
  `place` varchar(225) NOT NULL,  
  `pincode` varchar(225) NOT NULL,  
  `gender` varchar(225) NOT NULL,  
  `age` varchar(225) NOT NULL,  
  `phone` varchar(225) NOT NULL,  
  `email` varchar(225) NOT NULL,  
  `login_id` int(11) NOT NULL,  
  PRIMARY KEY (`id`),  
  KEY `foodapp_passenger_login_id_37c4f24a` (`login_id`)  
) ENGINE=MyISAM AUTO_INCREMENT=8 DEFAULT CHARSET=latin1 COLLATE=latin1  
_swedish_ci
```

Create table Payment

```
CREATE TABLE `foodapp_payment` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,
```

ONLINE FOOD DELIVERY

```
`booking_id` int(11) NOT NULL,  
`tdate` varchar(225) NOT NULL,  
PRIMARY KEY (`id`)  
) ENGINE=MyISAM AUTO_INCREMENT=4 DEFAULT CHARSET=latin1 COLLATE=latin1  
_swedish_ci
```

Create Table Reported Hotel

```
CREATE TABLE `foodapp_reported_hotel` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `reason_details` varchar(225) NOT NULL,  
  `date_time` varchar(225) NOT NULL,  
  `hotel_id` int(11) NOT NULL,  
  `passenger_id` int(11) NOT NULL,  
  `status` varchar(11) NOT NULL,  
  PRIMARY KEY (`id`),  
  KEY `foodapp_reported_hotel_hotel_id_8cfb5771` (`hotel_id`),  
  KEY `foodapp_reported_hotel_passenger_id_55929eb3` (`passenger_id`)  
) ENGINE=MyISAM AUTO_INCREMENT=6 DEFAULT CHARSET=latin1 COLLATE=latin1  
_swedish_ci
```

Create Table Hotel Feedback

```
CREATE TABLE `foodapp_rating` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `rate` varchar(225) NOT NULL,  
  `review` varchar(225) NOT NULL,  
  `date_time` varchar(225) NOT NULL,
```

ONLINE FOOD DELIVERY

```
`hotel_id` int(11) NOT NULL,
`passenger_id` int(11) NOT NULL,
PRIMARY KEY (`id`),
KEY `foodapp_rating_hotel_id_d6100a38` (`hotel_id`),
KEY `foodapp_rating_passenger_id_edb818b2` (`passenger_id`)
) ENGINE=MyISAM AUTO_INCREMENT=5 DEFAULT CHARSET=latin1 COLLATE=latin1_swedish_ci
```

6.2 DATA DICTIONARY

Primary Key Constraints Used:

Primary Key Name	Table Name	Column Name
Hotelid	Hotellogin	hotel_id
Passangerid	Passangerlogin	passanger_id
Bookingid	Payment	Booking_id
Passangerid, Bookingid	Rating	id
Hotelid	ReportedHotel	Hotelid
Menuid	HotelMenu	Menuid
DeliveryBoyid	HotelDeliveryboy	Deliveryboyid
Reportedid	ReportedHotel	ReportedID
UserId	HotelFeedback	Feedback

ONLINE FOOD DELIVERY

Sequences Used

create sequence hotelidseq increment by 1 start with 101 nocycle;
create sequence useridseq increment by 1 start with 101 nocycle;
create sequence deliveryboyidseq increment by 1 start with 101 nocycle;
create sequence menuidseq increment by 1 start with 101 nocycle;
create sequence idseq increment by 1 start with 101 nocycle;
create sequence reportedidseq increment by 1 start with 101 nocycle;
create sequence feedbackidseq increment by 1 start with 101 nocycle;

6.3 DATA FLOW DIAGRAM

DATA FLOW DIAGRAM

Data Flow diagram:

A data-flow diagram (DFD) is a graphical representation of the "flow" of data through an information system. DFDs can also be used for the visualization of data processing (structured design).

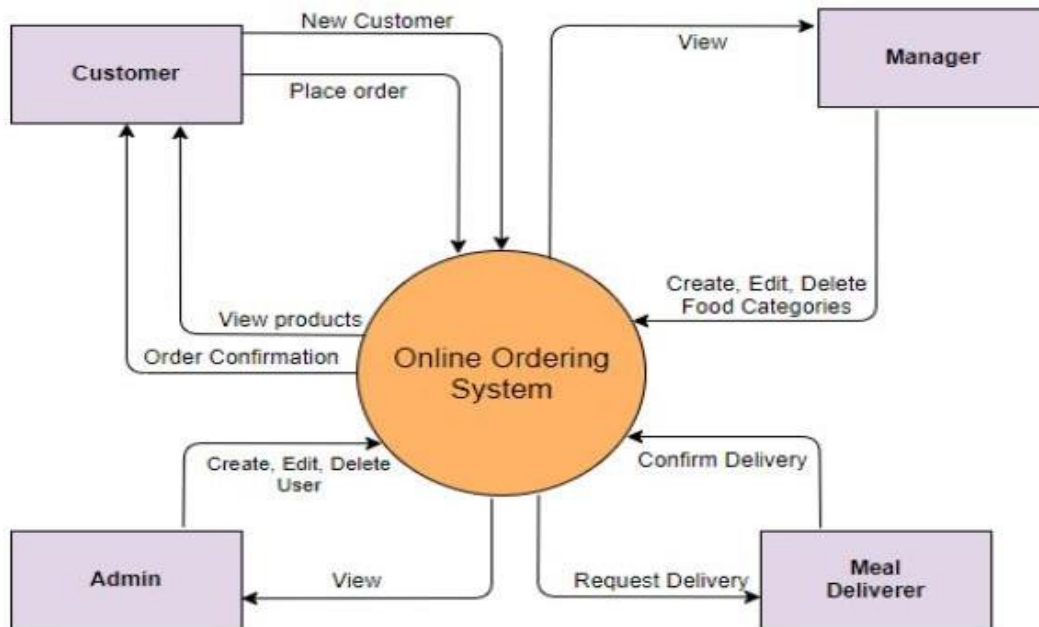
On a DFD, data items flow from an external data source or an internal data store to an internal data store or an external data sink, via an internal process.

A DFD provides no information about the timing or ordering of processes, or about whether processes will operate in sequence or in parallel. It is therefore quite different from a flowchart, which shows the flow of control through an algorithm, allowing a reader to determine what operations will be performed, in what order, and under what circumstances, but not what kinds of data will be input to

ONLINE FOOD DELIVERY

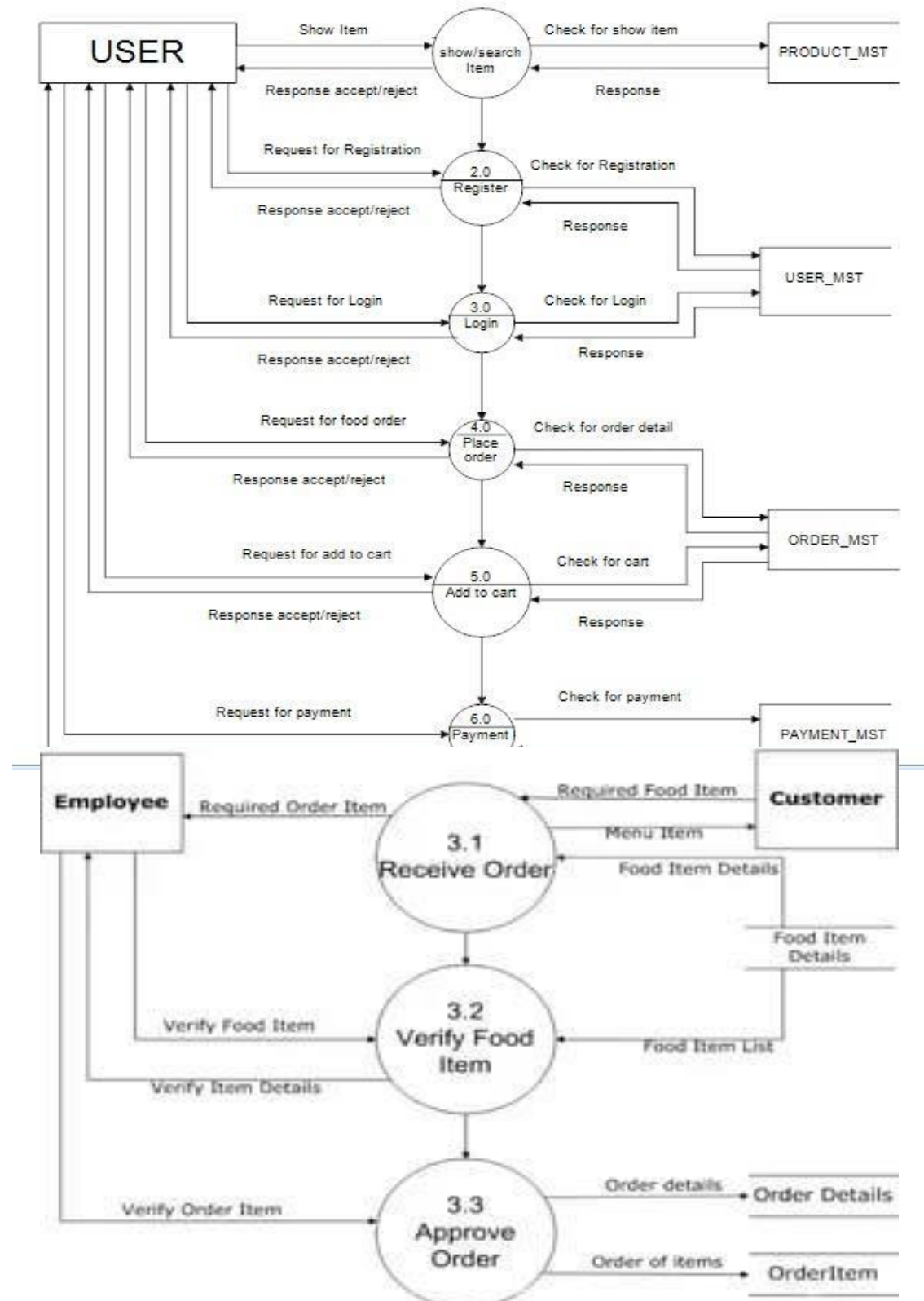
and output from the system, nor where the data will come from and go to, nor where the data will be stored.

CONTEXT LEVEL:



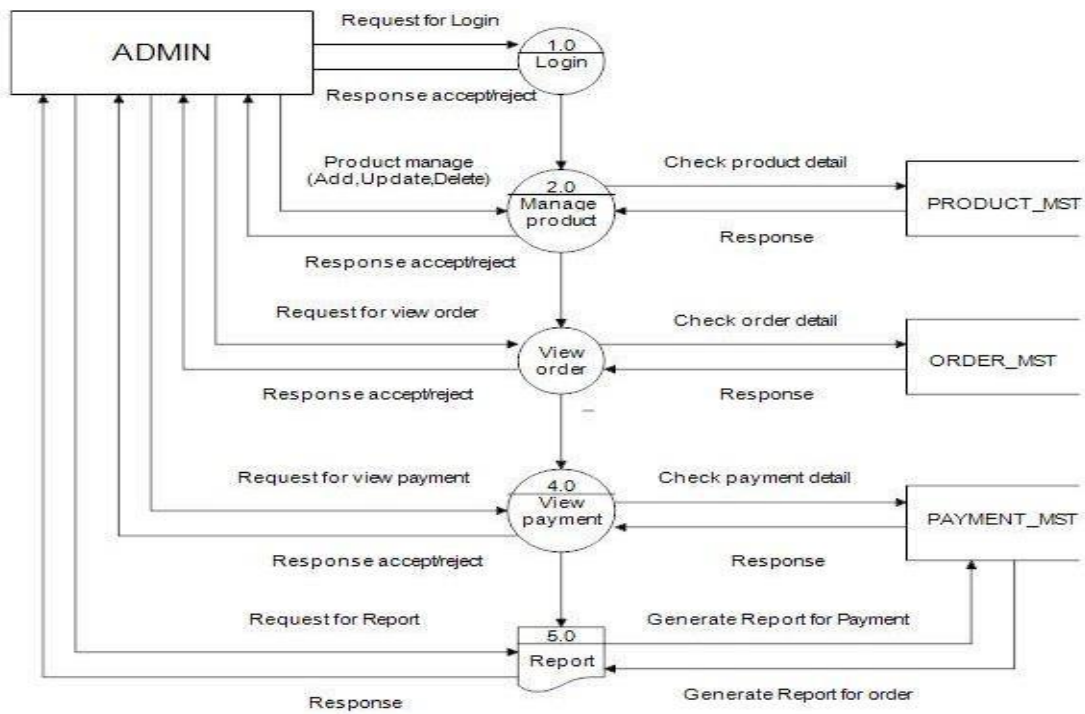
ONLINE FOOD DELIVERY

LEVEL 1:

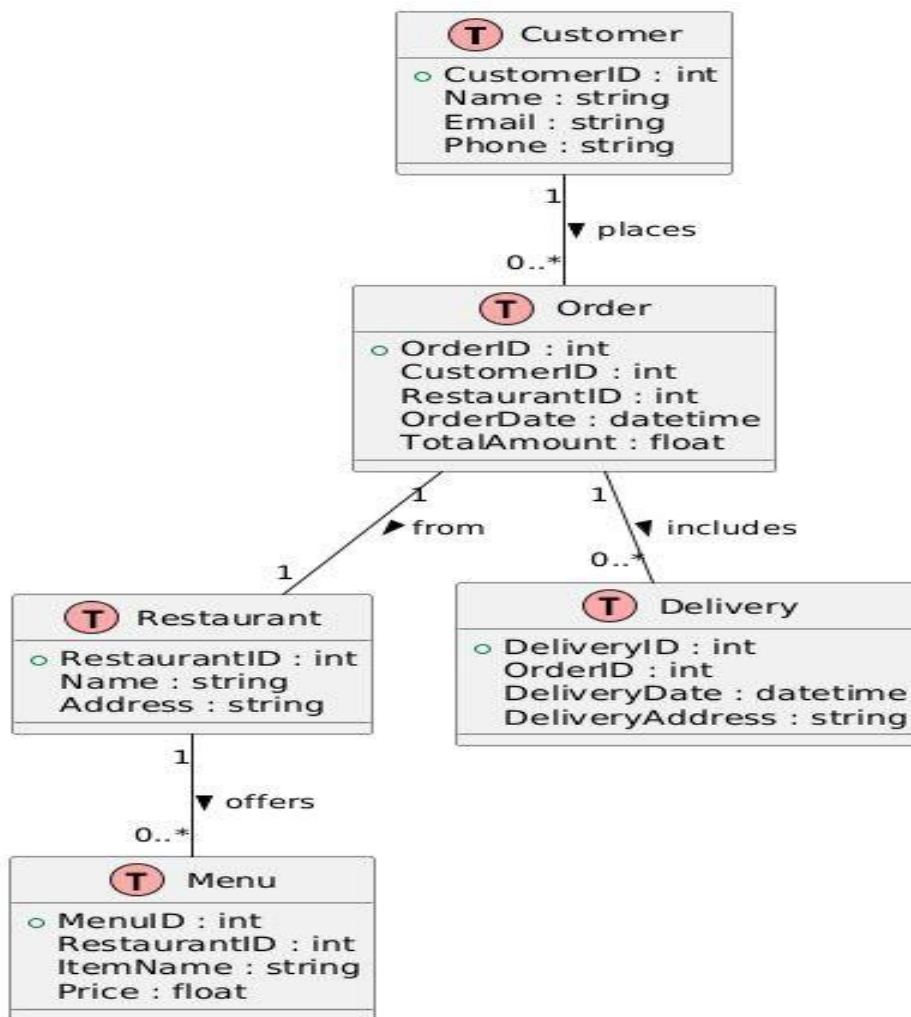


ONLINE FOOD DELIVERY

LEVEL 2:

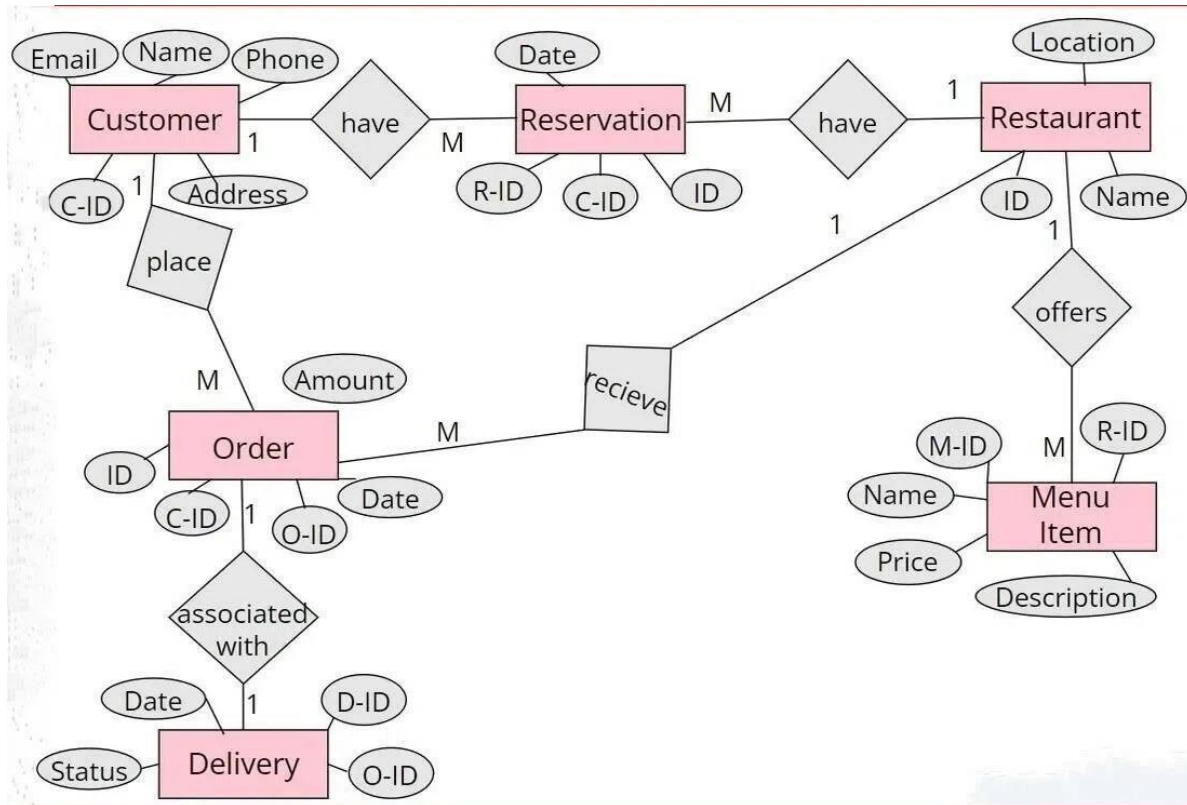


ONLINE FOOD DELIVERY SCHEMA DIAGRAM:



ONLINE FOOD DELIVERY

6.4 ER DIAGRAM





System Testing

ONLINE FOOD DELIVERY

7. TESTING

Testing is an indispensable phase in the software development lifecycle, encompassing a series of systematic processes and methodologies aimed at evaluating the quality, functionality, and performance of a software product. It serves as a critical means of identifying defects, errors, and discrepancies within the codebase, thereby ensuring that the software meets specified requirements and performs as expected under various conditions. At its core, testing involves the execution of predefined test cases, scenarios, and scripts designed to validate different aspects of the software, including its functionality, usability, security, and scalability. Through meticulous testing efforts, developers and quality assurance (QA) professionals aim to mitigate risks, enhance user experience, and maximize the reliability and robustness of the software. The testing process typically begins with the formulation of a comprehensive test plan, which outlines the testing strategy, objectives, scope, and resources required for validating the software. Test cases are then developed based on the test plan, covering a wide range of scenarios and use cases to ensure thorough coverage of the software's functionalities.

TESTING TECHNIQUES

Unit testing

Integration Testing

Validation Testing

System Testing

Output Testing

User Acceptance Testing

UNIT TESTING

It is the first level of testing. Each module is tested individually and focus is given for finding errors limited to each individual module and correcting them. The different modules of the system are tested

ONLINE FOOD DELIVERY

individually and corrected all errors. Each module is focused to work satisfactorily with regard to the expected output from the module. Validation checks for fields are also done here. Each process was

done individually and tested separately. Errors discovered were corrected. After unit testing, modules were integrated to form the complete system.

INTEGRATION TESTING

The tested modules are combined into subsystems and these are tested again. It is the second- level of testing. When modules are integrated, problems can arise at boundaries like incompatibility of data type of value being passed across the interface or some unexpected problems which appeared only after integration. Integration test was carried out while each module was integrated, interfaces were tested and corrected errors.

VALIDATION TESTING

For each input forms validation testing are done to ensure that only allowed values will be entered. Entering incorrect values does the validation testing and it is checked whether the errors are being considered. Incorrect values are to be discarded. The errors are rectified.

SYSTEM TESTING

System testing was performed to verify that all system elements have been properly integrated and perform allocated function. Security testing was done to check the security mechanisms built into the system, which will protect it from improper penetration, performance testing was done to test the runtime performance of the software. For user acceptance testing the system was given to the end user to use.

OUTPUT TESTING

After performing the validation testing, the next step is the output testing of the enhanced system. No system could be useful if it does not produce the required output in the required format. The

ONLINE FOOD DELIVERY

outputs generated or the displayed by the system are tested by asking the users about the format required by them.

USER ACCEPTANCE TESTING

User acceptance testing comprises a completed and successful end-to-end system test with review of the results by one or more users with specific knowledge. Users may apply a variety

of validation techniques. For example: generate a report from the new system and compare the results with the same report from the current system, data inspection and others. Prior to this testing the system was delivered to the clients along with the steps for implementation. In user acceptance testing the system was run in the client environment by system users. The users were allowed to test the system and raise any issues for a specific period after the system is assumed

7.3 SAMPLE TEST CASES DONE

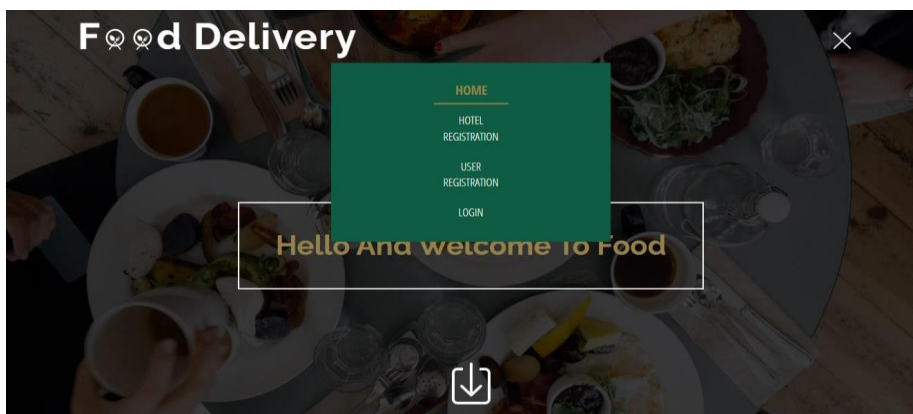
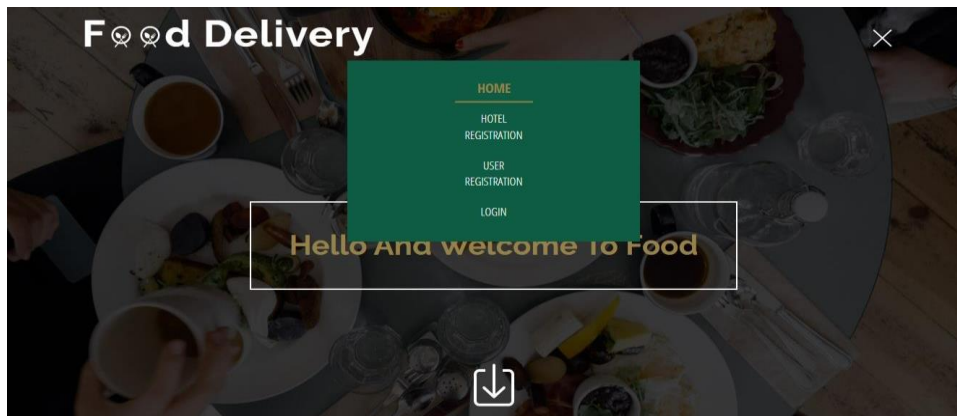
Sl.No.	Test Case Done	Action taken
1	Invalid user credentials entered	Redirected to error page. Customer of Admin main page not shown.
2	Incomplete form entry in the registration and few other pages.	Users are prompted to complete the page to proceed next.
3	Illegal data entry. E.g., experience entered in negative values. Phone numbers out of range, etc.,	All such errors are captured and prompted accordingly so that users will provide only the legal input to the system.
4	Invalid Complaint / User ID.	Captured and prompted the user to enter only the valid ID.
5	Incorrect Hotel entry	Error message notified to the user for entering the correct HotelID.



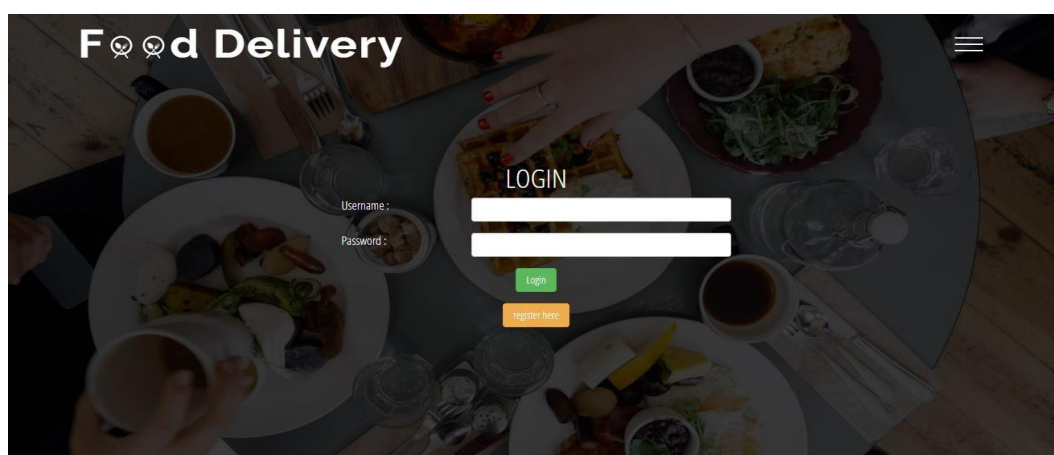
Screenshots

ONLINE FOOD DELIVERY

8. SCREENSHOTS

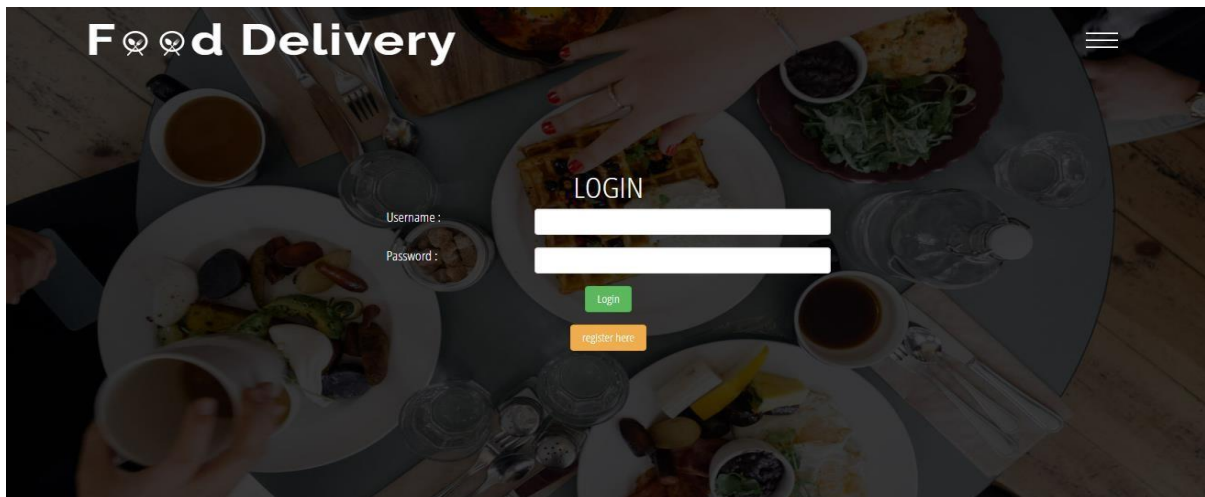


Login Page:



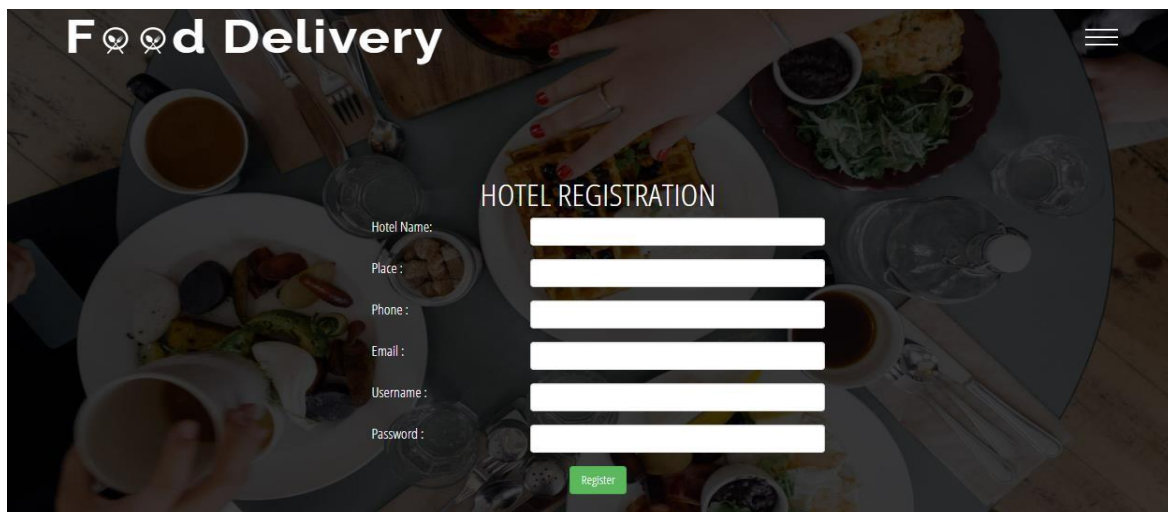
ONLINE FOOD DELIVERY

Hotel Login



The screenshot shows the 'Food Delivery' app interface. At the top left is the app logo 'Food Delivery' with a fork and knife icon. At the top right is a hamburger menu icon. The background is a dark, semi-transparent overlay on a photo of a dining table with food. In the center, the word 'LOGIN' is displayed. Below it are two input fields: 'Username:' and 'Password:'. Below the password field are two buttons: a green 'Login' button and an orange 'register here' button.

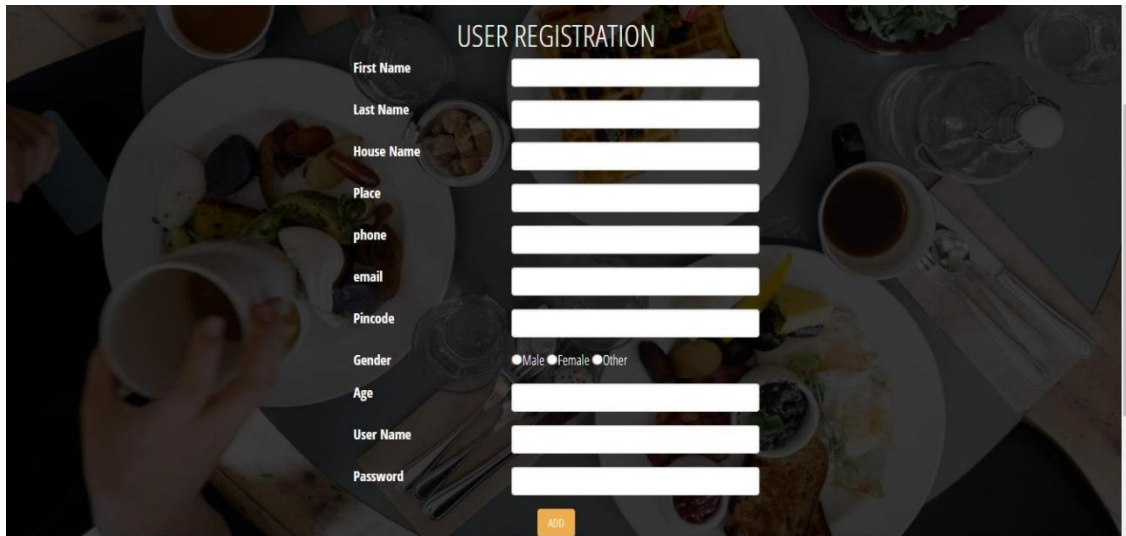
Hotel Registration Page:



The screenshot shows the 'Food Delivery' app interface for hotel registration. At the top left is the app logo 'Food Delivery' with a fork and knife icon. At the top right is a hamburger menu icon. The background is a dark, semi-transparent overlay on a photo of a dining table with food. In the center, the words 'HOTEL REGISTRATION' are displayed. Below it are six input fields: 'Hotel Name:', 'Place:', 'Phone:', 'Email:', 'Username:', and 'Password:'. Below the password field is a green 'Register' button.

ONLINE FOOD DELIVERY

User Registration Page:



USER REGISTRATION

First Name

Last Name

House Name

Place

phone

email

Pincode

Gender ☐ Male ☐ Female ☐ Other

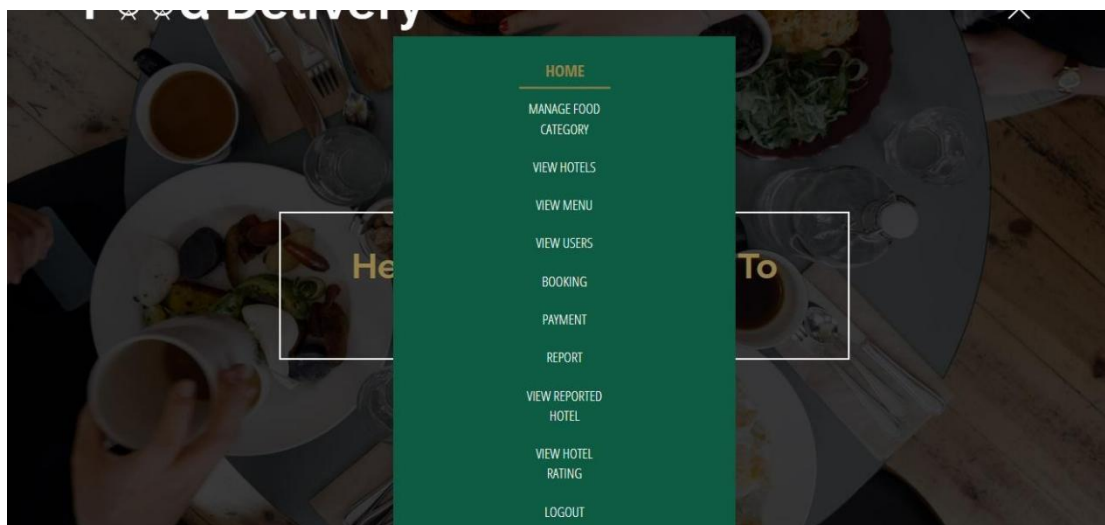
Age

User Name

Password

ADD

Admin Side:



ONLINE FOOD DELIVERY

Food Category :

Add Category

Add

Category

Sl. No	Category	Remove	Update
1	chicken	Remove	Update
2	fried	Remove	Update
3	fry	Remove	Update
4	biriyani	Remove	Update
5	Juices	Remove	Update
6	shakes	Remove	Update
7	breakfast	Remove	Update
8	lunch	Remove	Update

Food Delivery

Search hotel...

Hotels

Hotel Name	Phone	Email	Status	Action
coffeee house	9867548692	coffee@gmail.com	active	Accept Reject
cafe12	7865675467	cafe12@gmail.com	accept	Accept Reject
cafe11	1234567890	cafe11@gmail.com	pending	Accept Reject

© 2021 Food_delivery . All Rights Reserved | Design by Food_delivery

Food Delivery

VIEW USER

First Nmae	Last Name	House Name	Place	Pin code	Gender	Age	Phone	Email
mini	m	mini house	kochi	678999	female	26	6998759985	mini@gmail.com
arya	p v	aryasss home	kannur	678999	female	22	9867548692	aryaa@gmail.com
aushin	benison	house	kakkanad	680006	male	22	8967452389	aushin@gmail.com
ram	anand	ram nilayalam	aluva	689015	male	36	7865675467	ram@gmail.com
reenu	sachin	house no.36, bhavan	vytila	680007	female	23	7865675467	reenu@gmail.com

ONLINE FOOD DELIVERY

Booking

Username	Product	Quantity	Amount
minim	vggw	5	0
minim	vggw	2	1700
minim	vggw	5	4250
minim	vggw	1	850
minim	vggw	2	1700
aushinbenson	vggw	2	1700
aushinbenson	hyderabadi biriyani chicken	1	200
aushinbenson	fresh lime	1	25
aushinbenson	hyderabadi biriyani chicken	1	200

Food Delivery

Payment

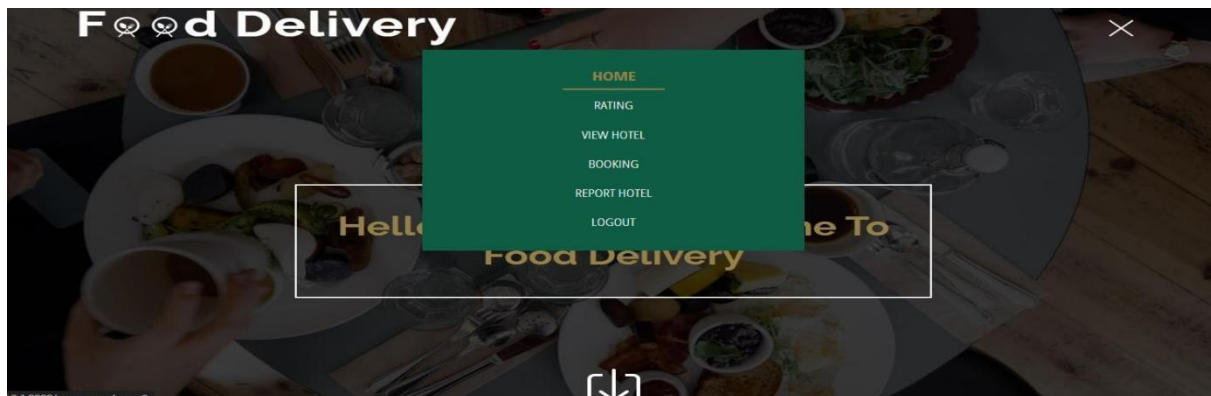
Username	Amount
aushin benson	1700
aushin benson	200
aushin benson	25
aushin benson	200

ONLINE FOOD DELIVERY

PURCHASE REPORT					
print					
Food_name	User Name	total_price	Quantity	Date_time	Admin Revenu
vgvgvv	aushin	1700	2	2024-03-27	10
hyderabad biriyani chicken	aushin	200	1	2024-03-27	10
fresh lime	aushin	25	1	2024-03-27	10
hyderabad biriyani chicken	aushin	200	1	2024-03-28	10

© 2021 Food_delivery . All Rights Reserved | Design by Food_delivery

User Side






Hotels				
Hotel Name	Phone	Email	Status	Action
coffee house	9867548692	coffee@gmail.com	active	Menu And Price
cafe12	7865675467	cafe12@gmail.com	accept	Menu And Price
cafe11	1234567890	cafe11@gmail.com	pending	Menu And Price

© 2021 Food_delivery . All Rights Reserved | Design by Food_delivery

ONLINE FOOD DELIVERY

Food Delivery

Menu

sl no	category	food name	image	price	quantity	
1	biriyani	hyderabadi biriyani chicken		200	1 kg	Order now
2	biriyani	malabari dum biriyani		170	2 kg	Order now
3	juices	fresh lime		25	1	Order now

Food Delivery

BOOKINGS

User Name	Food Name	Price	Quantity	Date	Status	Action
auslin benson	vgygy	1700	2	2024-03-27	Picked Up	
auslin benson	hyderabadi biriyani chicken	200	1	2024-03-27	Delivery Completed	
auslin benson	fresh lime	25	1	2024-03-27	Delivery Completed	
auslin benson	hyderabadi biriyani chicken	200	1	2024-03-28	paid	

© 2021 Food_delivery . All Rights Reserved | Design by Food_delivery

Food Delivery

Report Hotel

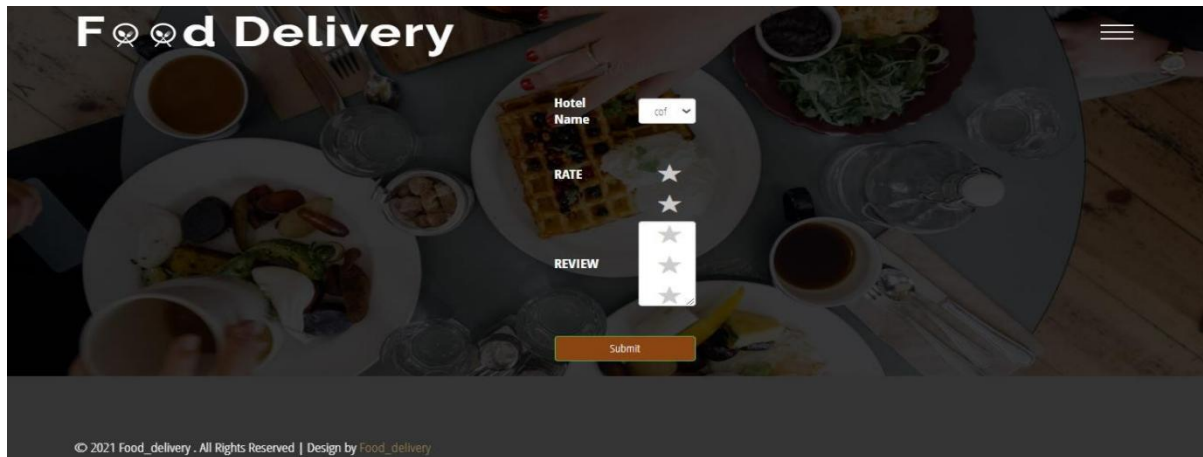
Reason

Hotel

[Add](#)

© 2021 Food_delivery . All Rights Reserved | Design by Food_delivery

ONLINE FOOD DELIVERY



Food Delivery

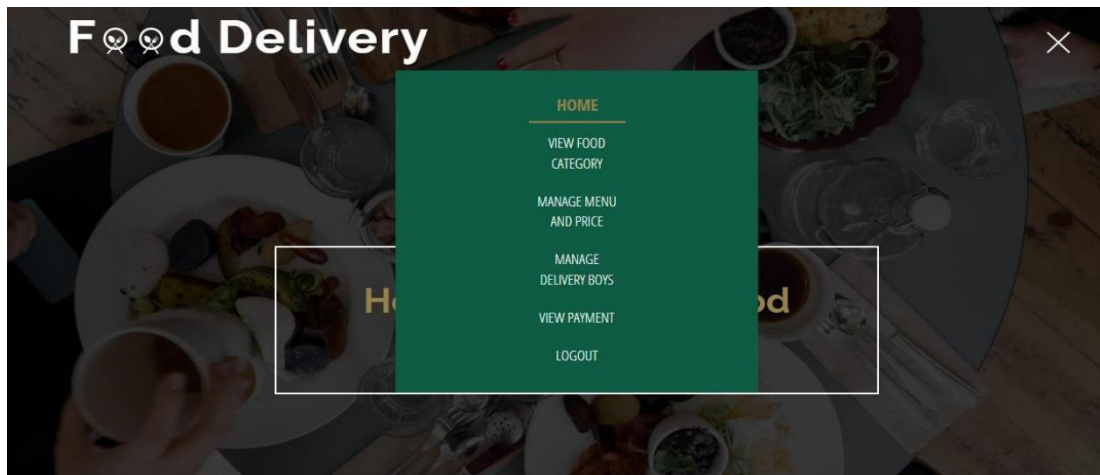
Hotel Name:

RATE: ★ ★ ★

REVIEW: ★ ★ ★

© 2021 Food_delivery . All Rights Reserved | Design by Food_delivery

HOTEL SIDE



Food Delivery

HOME

VIEW FOOD CATEGORY

MANAGE MENU AND PRICE

MANAGE DELIVERY BOYS

VIEW PAYMENT

LOGOUT

FOOD CATEGORY	
Sl No.	Category Name
1	chicken
2	fried
3	fry
4	biriyani
5	juices
6	shakes
7	breakfast
8	lunch
9	noodles
10	hot beverages
11	vegetable curry
12	beef items
13	chicken items

ONLINE FOOD DELIVERY

Food Delivery

BOOKINGS

Passenger Name	Food Name	Price	Quantity	Total Amount	Date	Action
aushin	hyderabadi biriyani chicken	200	1	200	2024-03-27	View User Details
aushin	hyderabadi biriyani chicken	200	1	200	2024-03-28	View User Details View Payment

Add Menu

category:

food name:



image: No file chosen

price:

quantity:

[Add](#)

Menu

sl no	category	food name	image	price	quantity	
1	biriyani	hyderabadi biriyani chicken		200	1 kg	Update Remove Bookings
2	biriyani	malabari dum biriyani		170	2 kg	Update Remove Bookings

ONLINE FOOD DELIVERY

Food Delivery

Add Delivery Boys

First Name

Last Name

House Name

Place

phone

email

User Name

Password

ADD

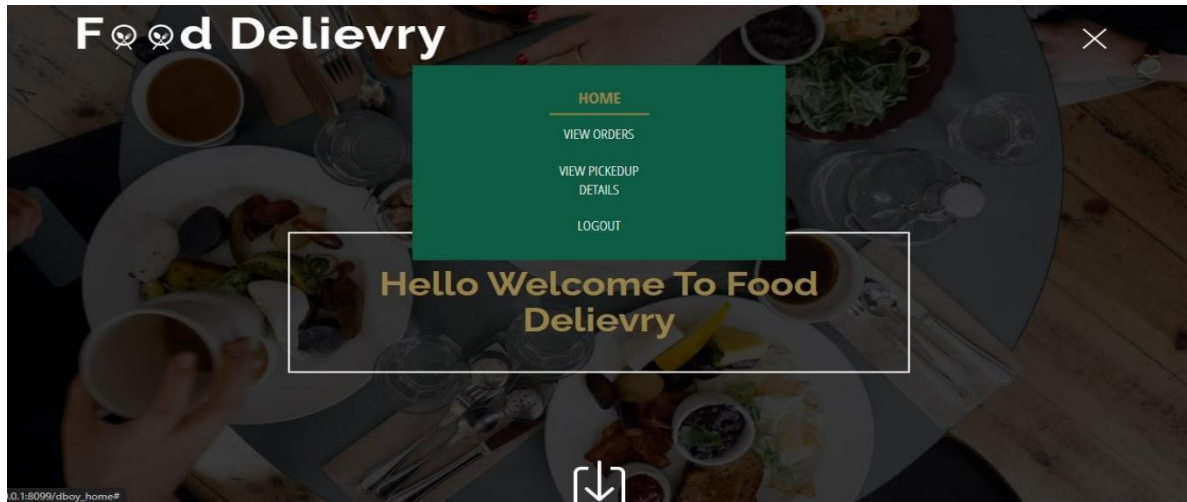
Food Delivery

PAYMENT DETAILS

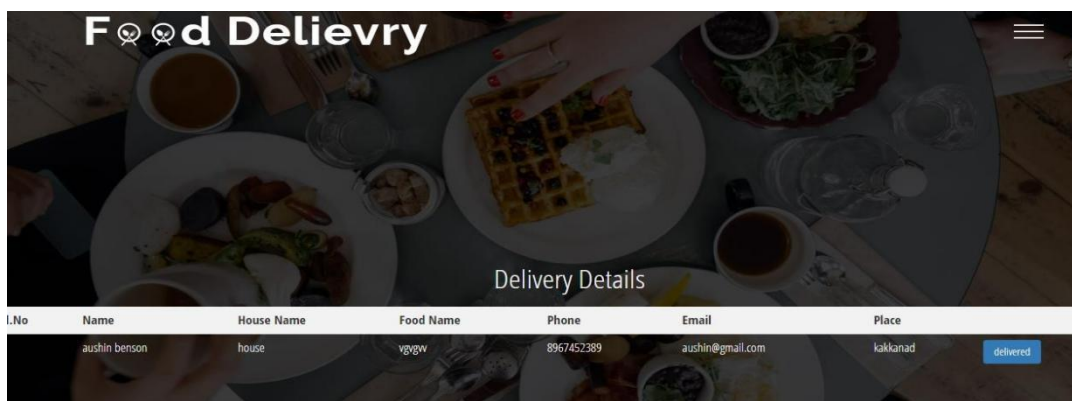
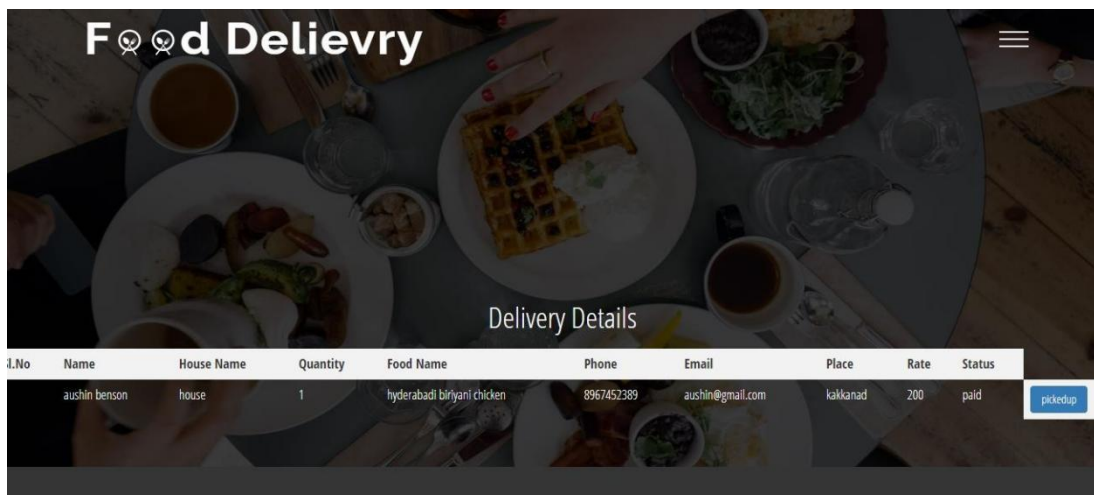
Sl no	User Name	Food Name	Price	Quantity	Date
1	aushin benson	hyderabadi biriyani chicken	200	1	2024-03-27
2	aushin benson	fresh lime	25	1	2024-03-27
3	aushin benson	hyderabadi biriyani chicken	200	1	2024-03-28

© 2021 Food_delivery . All Rights Reserved | Design by Food_delivery

ONLINE FOOD DELIVERY



DELIVERY BOY SIDE





Coding

ONLINE FOOD DELIVERY

Admin Home

```
<!-- start search-->

<div class="banner">

    <p class="animated wow fadeInLeft" data-wow-duration="1000ms" data-wow-
delay="500ms">Sed ut perspiciatis unde omnis iste natus.</p>

    <label></label>

    <h4 class="animated wow fadeInTop" data-wow-duration="1000ms" data-wow-
delay="500ms">Hello Admin Welcome To Food Delivery</h4>

    <a class="scroll down" href="#content-down"></a>

</div>

</div>

<!--content-->

{% include "footer.html" %}
```

Admin Manage

```
{% include "adminheader.html" %}

<!-- start search-->

<div class="banner">

<center>

    <form action="" method='post'>

{% if up %}

    <h1 style="color: #fff;">Manage Schedule</h1>

    <table class="table" style="width: 500px;color: #fff;">
```

ONLINE FOOD DELIVERY

<tr>

<td>ArrivalTime :</td>

<td><input type="time" name="at" class="form-control" required
value="{{ up.arrivaltime }}"></td>

</tr>

<tr>

<td>DepartureTime</td>

<td><input type="time" name="dt" class="form-control" required
value="{{ up.departuretime }}"></td>

</tr>

<tr>

<td>stop station :</td>

<td>

<select name="stopstation" class="form-control" required>

<option>Select</option>

{% for row in q %}

<option value="{{ row.id }}">{{ row.sname }}</option>

{% endfor %}

</td>

</tr>

<tr>

<td colspan="2" align="center"><input type="submit" name="submit" value="Add"
class="btn btn-success"></td>

</tr>

</table>

{% else %}

<h1 style="color: #fff;">Manage Schedule</h1>

ONLINE FOOD DELIVERY

```
<table class="table" style="width: 500px;color: #fff;">

<tr>

<td>ArrivalTime :</td>

<td><input type="time" name="at" class="form-control" required></td>

</tr>

<tr>

<td>DepartureTime</td>

<td><input type="time" name="dt" class="form-control" required></td>

</tr>

<tr>

<td>stop station :</td>

<td>

<select name="stopstation" class="form-control" required>

<option>Select</option>

{% for row in q %}

<option value="{{ row.id }}">{{ row.sname }}</option>

{% endfor %}

</td>

</tr>

<tr>

<td colspan="2" align="center"><input type="submit" name="submit" value="Add"
class="btn btn-success"></td>

</tr>

</table>

{% endif %}

</center>

<center>
```


ONLINE FOOD DELIVERY

```
<table class="table" style="width: 700px;color: #fff;">
  <h1 align="center" style="color: #fff;">Category</h1>
  <tr>
    <th>Sl. No</th>
    <th>ArrivalTime</th>
    <th>DepartureTime</th>
    <th>StopStation</th>
    <th>Train</th>
  </tr>
  {% for row in qry1 %}
  <tr>
    <td>{{ forloop.counter }}</td>
    <td>{{ row.arrivaltime }}</td>
    <td>{{ row.departuretime }}</td>
    <td>{{ row.stopstation_id.sname }}</td>
    <td>{{ row.train_id.train_name }}</td>
    <td><a class="btn btn-danger" href="/delete_schedule/{{ row.id }}">Remove</a></td>
    <td><a class="btn btn-danger" href="/update_schedule/{{ row.id }}">Update</a></td>
  </tr>
  {% endfor %}
</table>
</form>
</center>
</div>
</div>
<!--content-->
{% include "footer.html" %}
```

Admin Manage Food

ONLINE FOOD DELIVERY

```
{% include "adminheader.html" %}
```

```
<!-- start search-->
```

```
<div class="banner">
```

```
<center>
```

```
<form action="" method='post'>
```

```
<h1 style="color: #fff;">Add Category</h1>
```

```
{% if up%}
```

```
<table class="table" style="width: 500px;color: #fff;">
```

```
<!-- <tr>
```

```
<td>Food Category :</td>
```

```
<td><input type="text" name="category" class="form-control" required  
value="{ { up.category} } "></td>
```

```
</tr> -->
```

```
<tr>
```

```
<td colspan="2" align="center"><input type="submit" name="submit" value="Add"  
class="btn btn-success"></td>
```

```
</tr>
```

```
</table>
```

```
{% else %}
```

```
<table class="table" style="width: 500px;color: #fff;">
```

```
<tr>
```

```
<td>Food Category :</td>
```

```
<td><input type="text" name="category" class="form-control" required></td>
```

```
</tr>
```

```
<tr>
```

```
<td colspan="2" align="center"><input type="submit" name="submit" value="Add"  
class="btn btn-success"></td>
```

ONLINE FOOD DELIVERY

```

</tr>

</table>

{% endif %}

</center>

<center>

<table class="table" style="width: 700px;color: #fff;">

  <h1 align="center" style="color: #fff;">Category</h1>

  <tr>

    <th>Sl. No</th>

    <th>Category</th>

  </tr>

  {% for row in qry1 %}

    <tr>

      <td>{{ forloop.counter }}</td>

      <td>{{ row.category }}</td>

      <td><a class="btn btn-danger" href="/delete_category/{{ row.id }}">Remove</a></td>

      <td><a class="btn btn-danger" href="/update_category/{{ row.id }}">update</a></td>

    </tr>

  {% endfor %}

</table>

</form>

</center>

</div>

</div>

<!--content-->

{% include "footer.html" %}

```

ONLINE FOOD DELIVERY

Admin Purchase Report

```
{% include 'adminheader.html' %}

<div style="margin-bottom: 1em; background:url('/static/assets/img/inter.jpg') ; height: 300px;
width: 100%; background-size: cover;" >

</div>

<!DOCTYPE html>

<html>

<head>

<title>tsánta Store</title>

<!-- for-mobile-apps -->

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

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<meta name="keywords" content="Best Store Responsive web template, Bootstrap Web
Templates, Flat Web Templates, Android Compatible web template,
Smartphone Compatible web template, free webdesigns for Nokia, Samsung, LG, SonyEricsson,
Motorola web design" />

<script type="application/x-javascript"> addEventListener("load", function() {
setTimeout(hideURLbar, 0); }, false);

    function hideURLbar(){ window.scrollTo(0,1); } </script>

<!-- //for-mobile-apps -->

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

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

<!-- js -->

<script src="/static/assets/js/jquery.min.js"></script>

<!-- //js -->

<!-- cart -->

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

<!-- cart -->

<!-- for bootstrap working -->
```

ONLINE FOOD DELIVERY

```
<script type="text/javascript" src="/static/assets/js/bootstrap-3.1.1.min.js"></script>

<!-- //for bootstrap working -->

<link
href="//fonts.googleapis.com/css?family=Open+Sans:400,300,300italic,400italic,600,600italic,700,700italic,800,800italic" rel="stylesheet" type="text/css">

<link
href="//fonts.googleapis.com/css?family=Lato:400,100,100italic,300,300italic,400italic,700,700italic,900,900italic" rel="stylesheet" type="text/css">

<!-- timer -->

<link rel="stylesheet" href="/static/assets/css/jquery.countdown.css" />

<!-- //timer -->

<!-- animation-effect -->

<link href="/static/assets/css/animate.min.css" rel="stylesheet">

<script src="/static/assets/js/wow.min.js"></script>

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

<!-- //animation-effect -->

</head>

<body>

<style type="text/css">
th,td{
padding:30px;
}</style><center>

<div id="printsec">

<center>

<link rel="stylesheet" href="/static/assets/assets/css/style-starter.css">

<font style="font-family:times new roman">

<!-- <br><br> -->
```

ONLINE FOOD DELIVERY

<p>

<!-- 0926k 4th block,

Attingal,

Trivandrum ,

 -->

</p> <hr>

<!-- Section Title -->

<div class="container section-title" data-aos="fade-up">

<h2 class="table table-striped" style="color: rgb(255, 255, 255);width: 1000px;">PURCHASE
REPORT</h2>

</div><!-- End Section Title -->

<!-- <h1>View Sales</h1> -->

<table class="table table-secondary" style="width: 1200px;color: black
!important;background:white !important;opacity: 0.85;">

<tr style="background-color: rgb(255, 255, 255);">

<!--

<h1>Customer Details</h1>

 -->

<h2>print</h2>

<form method="post">

<table class="table table-secondary">

<tr style="background-color: rgb(43, 46, 27);">

<th>Food_name</th>

<th>User Name</th>

<th>total_price</th>

<th>Quantity</th>

<th>Date_time</th>

<!-- <th style="background: rgb(185, 198, 130) !important;border: 1px solid rgb(33, 34,
35) !important;">Product Name</th>

ONLINE FOOD DELIVERY

```
<th style="background: rgb(185, 198, 130) !important;border: 1px solid rgb(33, 34, 35)
!important;">Product Image</th>
```

```
<th style="background: rgb(185, 198, 130) !important;border: 1px solid rgb(33, 34, 35)
!important;">Qutantity</th>
```

```
<th style="background: rgb(185, 198, 130) !important;border: 1px solid rgb(33, 34, 35)
!important;">Total</th> -->
```

```
</tr>
```

```
{% for r in p %}
```

```
<tr>
```

```
<td>{{r.menu.food_name }}</td>
```

```
<td>{{r.passenger.fname }}</td>
```

```
<td>{{ r.price }}</td>
```

```
<td>{{ r.qty }}</td>
```

```
<td>{{ r.date_time }}</td>
```

```
</tr>
```

```
{%endfor%}
```

```
</table>
```

```
</form>
```

```
</div>
```

```
</center>
```

```
<style type="text/css">
```

```
@media print{
```

```
body,body *{
```

```
visibility: hidden;
```

```
}
```

```
#printsec,#printsec *{
```

ONLINE FOOD DELIVERY

```
visibility: visible !important; }

#trp{
    background: red !important;
}

#print{
    display: none;
}

#formsec{
    display: none;
}

#header{
    display: none !important;
}

}

/* Style the outer table */
.table-secondary {
    width: 1200px;
    color: black !important;
    background: white !important;
    opacity: 0.85;
}

/* Style the header row */
tr[style="background-color: rgb(207, 227, 137);"] {
    background-color: rgb(207, 227, 137);
}

/* Style the "Print" button */
#print {
    background-color: rgb(40, 161, 167); /* Green background color */
```


ONLINE FOOD DELIVERY

```
color: #fff; /* White text color */

padding: 5px 10px;

border: none;

cursor: pointer;

}

/* Style the inner table */

.table-striped {

    width: 90%;

    border: 1px solid black;

}

/* Style table headers */

th {

    background: rgb(251, 251, 249) !important;

    border: 1px solid rgb(33, 34, 35) !important;

}

/* Style table cells */

td {

    background: rgb(255, 255, 255) !important;

    border: 1px solid rgb(33, 34, 35) !important;

}

</style>

{% include 'footer.html' %}
```

Admin Hotel Rating

```
<!-- start search-->

<div class="banner">

<center>

<form>
```

ONLINE FOOD DELIVERY

<h1 style="color: #fff;">VIEW HOTEL RATING</h1>

<table class="table" style="width: 970px;color: #fff;">

<tr>

<th>Hotel Name</th>

<th>Rate</th>

<th>Review</th>

<th>Date</th>

<th>Passenger</th>

</tr>

{ % for row in qry1 % }

<tr>

<td>{{ row.hotel.hotel_name }}</td>

<td>{{ row.rate }}</td>

<td>{{ row.review }}</td>

<td>{{ row.date_time }}</td>

<td>{{ row.passenger.fname }} {{ row.passenger.lname }}</td>

</tr>

</table>

{ % endfor % }

</form>

</center>

</div>

</div>

<!--content-->

Admin Hotel View

<div class="banner">

<center>

<form method="post" id="hotelSearchForm">

ONLINE FOOD DELIVERY

```
{% csrf_token %}

<input type="text" id="hotelSearchInput" placeholder="Search hotel...">

</form>

<table class="table" style="width: 980px;color: #fff;">

  <h1 style="color: #fff;">Hotels</h1>

  <tr>

    <th>Hotel Name</th>

    <th>Phone</th>

    <th>Email</th>

    <th>Status</th>

    <th>Action</th>

  </tr>

  {% for hotel in h %}

  <tr class="hotelRow">

    <td>{{ hotel.hotel_name }}</td>

    <td>{{ hotel.phone }}</td>

    <td>{{ hotel.email }}</td>

    <td>{{ hotel.status }}</td>

    <td>

      <a href="/admin_view_hotels_on_station_accept/{{ hotel.login_id }}">Accept</a>

      <a href="/admin_view_hotels_on_station_reject/{{ hotel.login_id }}">Reject</a>

    </td>

  </tr>

  {% endfor %}

</table>

</center>

</div>

<script>
```

ONLINE FOOD DELIVERY

```
document.addEventListener('DOMContentLoaded', function() {

    document.getElementById('hotelSearchInput').addEventListener('input', function() {

        const searchQuery = this.value.toLowerCase();

        const hotelRows = document.querySelectorAll('.hotelRow');

        hotelRows.forEach(function(row) {

            const hotelName = row.querySelector('td:first-child').textContent.toLowerCase();

            if (hotelName.includes(searchQuery)) {

                row.style.display = "";

            } else {

                row.style.display = 'none';

            }

        });

    });

});

</script>
```

Admin Menu View

```
<!-- start search-->

<div class="banner">

<center>

<form method="post">

<h1 style="color: #fff;">MENU</h1>

<table class="table" style="width: 980px;color: #fff;"> <tr>

    <th>Hotel Name</th>

    <th>Category Name</th>

    <th>Food Name </th>

    <th>Image</th>

    <th>Price</th>

    <th>Quantity</th> </tr>
```

ONLINE FOOD DELIVERY

{ % for row in qry1 % }

<tr>

<td>{{ row.hotel.hotel_name }}</td>

<td>{{ row.category.category }}</td>

<td>{{ row.food_name }}</td>

<td></td>

<td>{{ row.price }}</td>

<td>{{ row.quantity }}</td>

<td></td>

{ % endfor % }

</tr> </table>

</form>

</center></div>

</div>

<!--content-->

Admin Passanger View

<!-- start search-->

<div class="banner">

<center>

<form>

<h1 style="color: #fff;">VIEW USER</h1>

<table class="table" style="color: #fff;width: 1100px;">

<tr>

<th>First Nmae</th>

<th>Last Name</th>

<th>House Name</th>

<th>Place</th>

ONLINE FOOD DELIVERY

```

<th>Pin code</th>

<th>Gender</th>

<th>Age</th>

<th>Phone</th>

<th>Email</th>

</tr>

{ % for row in p % }

<tr>

<td>{ {row.fname} }</td>

<td>{ {row.lname} }</td>

<td>{ {row.hname} }</td>

<td>{ {row.place} }</td>

<td>{ {row.pincode} }</td>

<td>{ {row.gender} }</td>

<td>{ {row.age} }</td>

<td>{ {row.phone} }</td>

<td>{ {row.email} }</td>

</tr>

{ % endfor % }

</table>

</form>

</center>

</div>

</div>

<!--content-->

```

Admin Reported View

ONLINE FOOD DELIVERY

```
<div class="banner">
  <center>
    <form>
      <h1 style="color: #fff;">REPORTED HOTELS</h1>
      <table class="table" style="width: 980px;color: #fff;">
        <tr>
          <th>Hotel Name</th>
          <th>Passenger reported</th>
          <th>Reason</th>
          <th>Date</th>
          <th>Action</th>
        </tr>
        {% for row in qry1 %}
          <tr>
            <td>{{ row.hotel.hotel_name }}</td>
            <td>{{ row.passenger.fname }}</td>
            <td>{{ row.reason_details }}</td>
            <td>{{ row.date_time }}</td>
            {% if row.status == "Block" %}
              <td><a class="btn btn-warning" href="/update_hotel_unblocked/{{ row.hotel_id }}">Unblock</a></td>
            {% else %}
              <td><a class="btn btn-danger" href="/update_hotel_blocked/{{ row.hotel_id }}">Block</a></td>
            {% endif %}
          </tr>
        {% endfor %}
      </table>
```

ONLINE FOOD DELIVERY

</form>

</center>

</div>

Admin Booking View

<center>

<h1 style="font-family: 'serif'; font-weight: normal;color: #fff;" >Booking</h1>

<div class="container mt-5">

<table class="table table-bordered" style="color: #fff;">

<thead class="thead-dark">

<tr>

<th scope="col" style="padding: 10px; background-color: #333;">Username</th>

<th scope="col" style="padding: 10px; background-color: #333;">Product</th>

<th scope="col" style="padding: 10px; background-color: #333;">Quantity</th>

<th scope="col" style="padding: 10px; background-color: #333;">Amount</th>

</tr>

</thead>

<tbody>

{% for i in qry1 %}

<tr>

<td style="padding: 10px; background-color: #333;">{{ i.passenger.fname }} {{ i.passenger.lname }}</td>

<td style="padding: 10px; background-color: #333;">{{ i.menu.food_name }}</td>

<td style="padding: 10px; background-color: #333;">{{ i.qty }}</td>

<td style="padding: 10px; background-color: #333;">{{ i.price }}</td>

</tr>

{% endfor %}

</tbody>

ONLINE FOOD DELIVERY

</table>

</div>

</center>

Admin Payment view

<center>

<h1 style="font-family: 'serif'; font-weight: normal;color: #fff">Payment</h1>

<div class="container mt-5">

<table class="table table-bordered" style="color: #fff;">

<thead class="thead-dark">

<tr>

<th scope="col" style="padding: 10px; background-color: #333;">Username</th>

<th scope="col" style="padding: 10px; background-color: #333;">paymentfor</th>

<th scope="col" style="padding: 10px; background-color: #333;">Amount</th>

</tr>

</thead>

<tbody>

{ % for i in qry1 % }

<tr>

<td style="padding: 10px; background-color: #333;">{{ i.booking.passenger.fname }} {{ i.booking.passenger.lname }}</td>

<td style="padding: 10px; background-color: #333;">{{ i.payment_for }}</td>

<td style="padding: 10px; background-color: #333;">{{ i.booking.price }}</td>

</tr>

{ % endfor % }

</tbody>

</table>

DB Boy Home

ONLINE FOOD DELIVERY

```
{% include 'dboyheader.html' %}

<!-- start search-->

<div class="banner">

    <p class="animated wow fadeInLeft" data-wow-duration="1000ms" data-wow-
delay="500ms">Sed ut perspiciatis unde omnis iste natus.</p>

    <label></label>

    <h4 class="animated wow fadeInTop" data-wow-duration="1000ms" data-wow-
delay="500ms">Hello Welcome To Food Delievry</h4>

    <a class="scroll down" href="#content-down"></a>

</div>

</div>

<!--content-->

{% include 'footer.html' %}
```

Delivery Boy Orders View

```
<center>

    <div style="width: 100%;height: 300px;background:url('/static/images/veg2.jpg');background-
size: cover;background-position: center;">

    </div>

    <h1 style="color: white;">Delivery Details</h1><br>

    <form>

        <table class="table" style="width: 100%; border-collapse: collapse;">

            <tr>

                <th style="background-color: #f2f2f2; color: #333;">Sl.No</th>

                <th style="background-color: #f2f2f2; color: #333;">Name</th>

                <th style="background-color: #f2f2f2; color: #333;">House Name</th>

                <th style="background-color: #f2f2f2; color: #333;">Quantity</th>

                <th style="background-color: #f2f2f2; color: #333;">Food Name</th>

                <th style="background-color: #f2f2f2; color: #333;">Phone</th>
```

ONLINE FOOD DELIVERY

```

<th style="background-color: #f2f2f2; color: #333;">Email</th>
<th style="background-color: #f2f2f2; color: #333;">Place</th>
<th style="background-color: #f2f2f2; color: #333;">Rate</th>
<th style="background-color: #f2f2f2; color: #333;">Status</th>
</tr>
{% for i in res %}
<tr>
<td style="color: white;">{{ forloop.counter }}</td>
<td style="color: white;">{{ i.passenger.fname }} {{ i.passenger.lname }}</td>

<td style="color: white;">{{ i.passenger.hname }}</td>
<td style="color: white;">{{ i.qty }}</td>
<td style="color: white;">{{ i.menu.food_name }}</td>
<td style="color: white;">{{ i.passenger.phone }}</td>
<td style="color: white;">{{ i.passenger.email }}</td>
<td style="color: white;">{{ i.passenger.place }}</td>
<td style="color: white;">{{ i.price }}</td>
<td style="color: white;">{{ i.status }}</td>
{% if i.status == "paid" %}
<td style="background-color: #f2f2f2; color: #333;"><a class="btn btn-primary"
href="?action=pickedup&bid={{ i.id }}">pickedup</a></td>
{% endif %}
</tr>
{% endfor %}
</table>
</form><br><br>
</center>

```

Delivery Boy Pickup

ONLINE FOOD DELIVERY

<center>

<div style="width: 100%;height: 300px;background:url('/static/images/veg2.jpg');background-size: cover;background-position: center;">

</div>

<h1 style="color: #f2f2f2;">Delivery Details</h1>

<form>

<table class="table" style="width: 100%; border-collapse: collapse;"> <tr>

<th style="background-color: #f2f2f2; color: #333;">Sl.No</th>

<th style="background-color: #f2f2f2; color: #333;">Name</th>

<th style="background-color: #f2f2f2; color: #333;">House Name</th>

<th style="background-color: #f2f2f2; color: #333;">Food Name</th>

<th style="background-color: #f2f2f2; color: #333;">Phone</th>

<th style="background-color: #f2f2f2; color: #333;">Email</th>

<th style="background-color: #f2f2f2; color: #333;">Place</th>

<th style="background-color: #f2f2f2; color: #333;"></th>

<th style="background-color: #f2f2f2; color: #333;"></th>

</tr>

{ % for i in res % }

<tr>

<td style="color: white;">{{ forloop.counter }}</td>

<td style="color: white;">{{ i.passenger.fname }} {{ i.passenger.lname }}</td>

<td style="color: white;">{{ i.passenger.hname }}</td>

<td style="color: white;">{{ i.menu.food_name }}</td>

<td style="color: white;">{{ i.passenger.phone }}</td>

<td style="color: white;">{{ i.passenger.email }}</td>

<td style="color: white;">{{ i.passenger.place }}</td>

{ % if i.status == "Picked Up" % }

<td style="color: white;">

ONLINE FOOD DELIVERY

```
<a class="btn btn-primary" href="?action=deliver&bid={{ i.id }}">delivered</a>
</td>
{% endif %}
</tr>
{% endfor %}
</table>
</form><br><br> </center>
```

Food Report

```
<div style="margin-bottom: 1em; background:url('/static/assets/img/inter.jpg') ; height: 300px;
width: 100%; background-size: cover;" >
</div>
<!DOCTYPE html>
<html>
<head>
<title>tsánta Store</title>
<!-- for-mobile-apps -->
<meta name="viewport" content="width=device-width, initial-scale=1">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<meta name="keywords" content="Best Store Responsive web template, Bootstrap Web
Templates, Flat Web Templates, Android Compatible web template,
Smartphone Compatible web template, free webdesigns for Nokia, Samsung, LG, SonyEricsson,
Motorola web design" />
<script type="application/x-javascript"> addEventListener("load", function() {
setTimeout(hideURLbar, 0); }, false);
function hideURLbar(){ window.scrollTo(0,1); } </script>
<!-- //for-mobile-apps -->
<link href="/static/assets/css/bootstrap.css" rel="stylesheet" type="text/css" media="all" />
<link href="/static/assets/css/style.css" rel="stylesheet" type="text/css" media="all" />
<!-- js -->
```

ONLINE FOOD DELIVERY

```
<script src="/static/assets/js/jquery.min.js"></script>

<!-- //js -->

<!-- cart -->

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

<!-- cart -->

<!-- for bootstrap working -->

<script type="text/javascript" src="/static/assets/js/bootstrap-3.1.1.min.js"></script>

<!-- //for bootstrap working --><link
href="//fonts.googleapis.com/css?family=Open+Sans:400,300,300italic,400italic,600,600italic,700,
700italic,800,800italic" rel='stylesheet' type='text/css'>

<link
href="//fonts.googleapis.com/css?family=Lato:400,100,100italic,300,300italic,400italic,700,700itali
c,900,900italic" rel='stylesheet' type='text/css'>

<!-- timer -->

<link rel="stylesheet" href="/static/assets/css/jquery.countdown.css" />

<!-- //timer -->

<!-- animation-effect -->

<link href="/static/assets/css/animate.min.css" rel="stylesheet">

<script src="/static/assets/js/wow.min.js"></script>

<script>

new WOW().init();

</script>

<!-- //animation-effect -->

</head>

<body>

<style>

th,td{

padding:30px;

}
```

ONLINE FOOD DELIVERY

```
</style>

<center>

<div id="printsec">

  <center>

    <link rel="stylesheet" href="/static/assets/assets/css/style-starter.css">

    <font style="font-family:times new roman">

      <!-- <br><br> -->

      <p>

        <!-- 0926k 4th block,

        Attingal,

        Trivandrum ,

          <br><br> --

        </p> <hr>

        <!-- Section Title -->

        <div class="container section-title" data-aos="fade-up">

          <h2 class="table table-striped" style="color: rgb(255, 255, 255);width: 1000px;">HOTEL
REPORT</h2>

        </div><!-- End Section Title -->

        <!-- <h1>View Sales</h1> -->

        <table class="table table-secondary" style="width: 1200px;color: black
!important;background:white !important;opacity: 0.85;">

        <tr style="background-color: rgb(255, 255, 255);">

        <!--

        <br><h1>Customer Details</h1><br><br> -->

        <h2><a class="btn btn-success" id="print" onclick="print()" >print</a></h2>

        <form method="post">

          { % csrf_token % }

          <label for="category_id">Select Category:</label>

          <select id="category_id" name="category_id">
```


ONLINE FOOD DELIVERY

```

<option value="">All</option>
{% for category in q %}
<option value="{{ category.id }}">{{ category.category }}</option>
{% endfor %}
</select>
<button type="submit">Search</button>
</form>
<table class="table table-secondary">
<tr style="background-color: rgb(43, 46, 27);">
<th>Food Name</th>
<th>Image</th>
<th>Quantity</th>
<th>Date_time</th>
<th>Price</th>
<th>Total</th>
<th colspan="5"></th>
</tr>
{% for r in v %}
<tr>
<td>{{ r.menu.food_name }}</td>
<td></td>
<td>{{ r.qty }}</td>
<td>{{ r.date_time }}</td>
<td>{{ r.price }}</td>
<td>{{ r.price }}</td> <!-- Displaying individual prices -->
</tr>
{% endfor %}

```

ONLINE FOOD DELIVERY

```
<!-- Total Row -->
```

```
<tr>
```

```
<td colspan="4"></td>
```

```
<td>Total:</td>
```

```
<!-- Total Price Calculation -->
```

```
<td>{ { total_price } }</td>
```

```
<td></td>
```

```
</tr>
```

```
</table>
```

```
</div>
```

```
</center>
```

Hotel Home

```
<!-- start search-->
```

```
<div class="banner">
```

```
<p class="animated wow fadeInLeft" data-wow-duration="1000ms" data-wow-  
delay="500ms">Sed ut perspiciatis unde omnis iste natus.</p>
```

```
<label></label>
```

```
<h4 class="animated wow fadeInTop" data-wow-duration="1000ms" data-wow-  
delay="500ms">Hello { { up.hotel_name } } Welcome To Food Delivery</h4>
```

```
<a class="scroll down" href="#content-down"></a>
```

```
</div>
```

```
</div>
```

Hotel Assign DB

```
<!-- start search-->
```

```
<div class="banner">
```

```
<center>
```

```
<form method="post">
```

```
{ % if data['assign_details'] % }
```

ONLINE FOOD DELIVERY

```
<table class="table" style="width: 700px;color: #fff;">
```

```
<h2 style="color: #fff;">Assign Details</h2>
```

```
<tr>
```

```
<th>Delivery Boy</th>
```

```
<th>Phone</th>
```

```
<th>Email</th>
```

```
</tr>
```

```
{% for row in ad %}
```

```
<tr>
```

```
<td>{{ row.d_name }}</td>
```

```
<td>{{ row.phone }}</td>
```

```
<td>{{ row.email }}</td>
```

```
</tr>
```

```
{% endfor %}
```

```
</table>
```

```
{% else %}
```

```
<h1 style="color: #fff;">DELIVERY BOYS</h1>
```

```
<table class="table" style="width: 760px;color: #fff;">
```

```
<tr>
```

```
<td>
```

```
<select name="name" class="form-control" required>
```

```
<option>Select</option>
```

```
{% for row in db %}
```

```
<option value="{{ row.id }}">{{ row.name }}</option>
```

```
{% endfor %}
```

```
</select>
```

```
</td>
```

ONLINE FOOD DELIVERY

```
<td colspan="2" align="center"><input type="submit" name="submit" value="assign"
class="btn btn-warning"></td>
```

```
</tr>
```

```
</table>
```

```
{% endif %}
```

```
</form>
```

```
</center>
```

```
</div>
```

```
</div>
```

```
<!--content-->
```

Hotel Food Availability

```
<center>
```

```
<form method="post">
```

```
{% if data['stocks'] %}
```

```
<table>
```

```
<tr>
```

```
<td>Quantity</td>
```

```
<td><input type="number" name="stock" class="form-control" required></td>
```

```
</tr>
```

```
<tr>
```

```
<td><input type="submit" name="submitss" class="form-control" value="Update"
class="btn btn-warning"></td>
```

```
</tr>
```

```
</table>
```

```
{% else %}
```

```
<h1>manage food availability</h1>
```

```
<table>
```

```
<h1>menu</h1>
```

```
<tr>
```

ONLINE FOOD DELIVERY

```

<th>Sl no</th>
<th>Food Name</th>
<th>Price</th>
<th>Quantity</th>
</tr>
{% for row in data['menu'] %}
<tr>
<td>{{ loop.index }}</td>
<td>{{ row['food_name'] }}</td>
<td>{{ row['price'] }}</td>
<td>{{ row['quantity'] }}</td>
<td><a class="btn btn-primary"
href="?action=mangestock&mid={{ row['menu_id'] }}">MANAGE STOCK</a></td>

</tr>
{% endfor %}
</table>
{% endif %}

```

</form>

Hotel Manage Menu & Price

```

<div class="banner">
<center>
<form method='post' enctype="multipart/form-data">
{% if up %}
<h1 style="color: #fff;">Update Menu</h1>
<table class="table" style="width: 500px;color: #fff;">
<tr> <th>category :</th>

```

ONLINE FOOD DELIVERY

```

<td><input type="text" class="form-control" readonly name="category"
value="{{ up.category }}" required></td>

</tr><tr>

<th>food name :</th> <td><input type="text" class="form-control" name="food_name"
value="{{ up.food_name }}" required></td>

</tr>

<tr>

<th>Image</th>

<td>



<input type="file" name="image" id="" accept="image/*" required class="form-control"
value="{{ up.image }}"></td>

</tr> <tr> <th>price :</th>

<td><input type="text" class="form-control" name="price" value="{{ up.price }}"
required></td>

</tr><tr> <th>quantity :</th>

<td><input type="text" name="quantity" class="form-control" value="{{ up.quantity }}"
required></td>

</tr>

<td colspan="2" align="center"><input type="submit" name="submitss" value="Update"
class="btn btn-warning"></td>

</tr>

</table>

{% else %}

<h1 style="color: #fff;">Add Menu</h1>

<table class="table" style="width: 500px;color: #fff;"> <tr>

<th>category</th> <td>

<select name="category" class="form-control" required>

<option>Select</option>

```

ONLINE FOOD DELIVERY

```
{% for row in q %}

<option value="{{ row.id }}">{{ row.category }}</option>

{% endfor %}

</select> </td> </tr>

<tr><th>food name :</th>

<td><input type="text" name="food_name" class="form-control" required></td>

</tr>

<tr> <th>image :</th>

<td><input type="file" name="image" class="form-control" required></td>

</tr>

<tr> <th>price :</th>

<td><input type="text" name="price" class="form-control" required></td></tr>

<tr> <th>quantity :</th>

<td><input type="text" name="quantity" class="form-control" required></td>

</tr><tr>

<td colspan="2" align="center"><input type="submit" name="submit" value="Add"
class="btn btn-success"></td>

</tr>

</table>

{% endif %}

<table class="table" style="width: 1000px;color: #fff;">

<h1 align="center" style="color: #fff;">Menu</h1>

<tr>

<th>sl no</th>

<th>category</th>

<th>food name</th>

<th>image</th>

<th>price</th>
```


ONLINE FOOD DELIVERY

```

<th>quantity</th>
</tr>
{% for row in qry1 %} <tr>
    <td>{{forloop.counter}}</td>
    <td>{{row.category.category}}</td>
    <td>{{row.food_name}}</td>
    <td></td>
    <td>{{row.price}}</td>
    <td>{{row.quantity}}</td>
    <td><a class="btn btn-warning"
href="/update_hotel_manage_menu_and_price/{{row.id}}">Update</a>
    <a class="btn btn-danger"
href="/delete_hotel_manage_menu_and_price/{{row.id}}">Remove</a></td>
    <td><a class="btn btn-primary"
href="/hotel_view_booking_and_accept/{{row.id}}">Bookings</a></td> </tr>
{% endfor %} </table></form></center> </div></div>

```

Hotel DB View

```

<center><form method="post">
    {% if up %}
    <h1 style="color: wheat;">Update menu</h1>
    <table class="table" style="width: 500px;color: #fff;">
    <tr>
        <th>first name :</th>
        <td><input type="text" name="fname" value="{{up.fname}}" class="form-control"
required></td>
    </tr> <tr>
        <th>last name :</th>
        <td><input type="text" name="lname" value="{{up.lname}}" class="form-control"
required></td>

```

ONLINE FOOD DELIVERY

```

</tr> <tr> <th>house name :</th>

<td><input type="text" name="hname" value="{{ up.hname }}" class="form-control"
required></td>

</tr> <tr>

<th>place :</th>

<td><input type="text" name="place" value="{{ up.place }}" class="form-control"
required></td>

</tr> <tr>

<th>phone :</th>

<td><input type="text" name="phone" value="{{ up.phone }}" class="form-
control" required></td>

</tr><tr><th>email :</th>

<td><input type="email" name="email" value="{{ up.email }}" class="form-control"
required></td>

</tr><td colspan="2" align="center"><input type="submit" name="submitss" value="Update"
required class="btn btn-success"></td>

</tr> </table>

{ % else % }

<h1 style="color: wheat;">Add Delivery Boys</h1>

<table class="table" style="width: 500px;color: #fff;">

<tr> <th>First Name</th>

<td><input type="text" name="fname" pattern="[a-zA-Z,]{30}" required class="form-
control"></td>

</tr> <tr>

<th>Last Name</th>

<td><input type="text" name="lname" pattern="[a-zA-Z,]{30}" required class="form-
control"></td>

</tr><tr> <th>House Name</th>

<td><input type="text" name="hname" pattern="[a-zA-Z,]{30}" required class="form-
control"></td>

```

ONLINE FOOD DELIVERY

```

</tr> <tr><th>Place</th>

<td><input type="text" name="place" pattern="[a-zA-Z,]{30}" required class="form-
control"></td>

</tr><tr>

<th>phone</th>

<td><input type="text" name="phone" pattern="[0-9]{10}" required class="form-
control"></td>

</tr> <tr> <th>email</th>

<td><input type="text" name="email" required class="form-control" pattern="[a-zA-Z0-
9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}"></td>

</tr>      </tr>      <tr>

<th>User Name</th>

<td><input type="text" name="username" value="" class="form-control" required
pattern="[0-9a-zA-Z]{0,30}"></td>

</tr><tr> <!-- <th>Password</th>

<td><input type="Password" name="password" value="" class="form-control"
pattern="(?=.*\d)(?=.*[a-z])(?=.*[A-Z]).{5,}" title="Must contain at least one number and one
uppercase and lowercase letter, and at least 5 or more characters" required> -->

<th>Password</th>

<td><input type="Password" name="password" value="" class="form-control"
required pattern="(?=.*\d)(?=.*[a-z])(?=.*[A-Z]).{5,}" title="Must contain at least one number and
one uppercase and lowercase letter, and at least 5 or more characters">

</td></tr> <tr>

<td colspan="2" align="center"><input type="submit" name="submit" value="ADD"
class="btn btn-warning"></td>

</tr>

</table> <h1 style="color: wheat;"> Delivery Boys</h1> <table class="table"
style="width: 500px;color: #fff;">

<th>Last Name</th><th>House Name</th>      <th>Place</th>

<th>Phone</th>      <th>Email</th>      </tr> <tr>

<td>{{ forloop.counter }}</td><td>{{ row.fname }}</td><td>{{ row.lname }}</td>

```

ONLINE FOOD DELIVERY

```

<td>{{row.hname}}</td><td>{{row.place}}</td>
<td>{{row.phone}}</td>
<td>{{row.email}}</td>
<td><a href="/update_dboy_register/{{row.id}}" class="btn btn-danger">Update </a>
<br>
<a href="/delete_dboy_register/{{row.id}}" class="btn btn-
success">Remove</a></td>
</tr>{% endfor %}
</table> </form></center>

```

Hotel Registration

```

<div class="banner">
<center>
<form method="post"><h1 style="color: #fff;">HOTEL
REGISTRATION</h1>
<table class="table" style="width: 500px;color: #fff;">
<tr>
<td>Hotel Name:</td>
<td><input type="text" name="hname" pattern="[a-zA-
Z,]{30}" class="form-control" required></td>
</tr>
<tr>
<td>Place :</td>
<td><input
type="text" name="stations" pattern="[a-zA-Z,]{30}" class="form-control" required></td>
</tr>
<tr>
<td>Phone :</td>
<td><input type="numbers" name="phone" class="form-control" pattern="[0-9]{10}"
title="enter a valid phone number" required></td>
</tr>
<tr>
<td>Email :</td>
<td><input type="Email" name="email" class="form-control" required pattern="[a-z0-
9._%+-]+@[a-z0-9.-]+\.[a-z]{2,4}"></td>
</tr>
<tr>
<td>Username <td><input type="text"
name="username" class="form-control" required pattern="[0-9a-zA-Z]{0,30}"></td>
</tr>
<tr>
<td>Password :</td>
<td><input type="Password" name="password" class="form-control"
required pattern="(?!.*\d)(?!.*[a-z])(?!.*[A-Z]).{5,}" title="Must contain at least one number and
one uppercase and lowercase letter, and at least 5 or more characters"></td>
</tr>
<tr>
<td colspan="2" align="center"><input type="submit" name="submit"
value="Register" class="btn btn-success"></td>
</tr>
</table>
</form>
</center>
</div>

```

Hotel Booking View



ONLINE FOOD DELIVERY

```
<div class="banner"><center><form>      <h1 style="color: #fff;">BOOKINGS</h1>

<table class="table" style="width: 1000px;color: #fff;"> <tr>

    <th>Passenger Name</th><th>Food Name</th><th>Price</th>

<th>Quantity</th>          <th>Total Amount</th>          <th>Date</th>

    <!-- <th>Status</th> -->          <th>Action</th>          <tr>

    <td>{{ row.passenger.fname }}</td><td>{{ row.menu.food_name }}</td>

    <td>{{ row.menu.food_name }}</td><td>{{ row.price }}</td>          <td>{{ row.qty
}}</td>          <td>{{ row.price }}</td>          <td>{{ row.date_time }}          <!--
<td>{{ row.b_status }}</td> -->

    <td><a class="btn btn-primary" href="/passenger_details/{{ row.passenger_id
}}">View User Details</a></td>

    {% if row.status == 'pending' % }

    <td><a class="btn btn-success" href="/hotel_view_booking_and_accepts/{{ row.id
}}">Accept</a></td>

    <td><a class="btn btn-danger" href="/hotel_view_booking_and_reject/{{ row.id
}}">Reject</a></td>

    {% endif % }

    {% if row.status == 'paid' % }

    <!-- <td><a class="btn btn-primary" href="/hotel_assign_delivery_boy/{{
row.passenger_id }}/{{ row.booking_id }}">Assign Delivery Boy</a></td> -->

    <td><a class="btn btn-primary" href="/hotel_view_payment">View
Payment</a></td>

    {% endif % }

</tr></table>      </form>      </center></div>
```

Hotel View Payment

```
<div class="banner"></div>      <form> <h1 style="color: #fff;">PAYMENT DETAILS</h1>

<table class="table" style="width: 900px;color: #fff;">

    <tr> <th>Sl no</th>      <th>Passenger Name</th>          <th>Price</th>

    <th>Quantity</th><th>Date</th>          </tr>
```

ONLINE FOOD DELIVERY

```
<tr><td>{{ forloop.counter }}</td> <td>{{ payment.booking.passenger.fname }} {{
payment.booking.passenger.lname }}</td>

<td>{{ payment.booking.menu.food_name }}</td>

<td>{{ payment.booking.menu.price }}</td>

<td>{{ payment.booking.qty }}</td>

<td>{{ payment.tdate }}</td>

</tr>

{% endfor %} </table> </form> </center></div>
```

User Reg

```
<!-- start search--> <div class="banner"> <center> <form method="post">

<h1 style="color: #fff;">USER REGISTRATION</h1>

<table class="table" style="width: 500px;color: #fff;">

<tr> <th>First Name</th>

<td><input type="text" name="fname" pattern="[a-zA-Z,]{30}" required class="form-
control"></td></tr> <tr><th>Last Name</th>

<td><input type="text" name="lname" pattern="[a-zA-Z,]{30}" required class="form-
control"></td></tr> <tr> <th>House Name</th>

<td><input type="text" name="hname" pattern="[a-zA-Z,]{30}" required class="form-
control"></td>

</tr> <tr><th>Place</th>

<td><input type="text" name="place" pattern="[a-zA-Z,]{30}" required class="form-
control"></td></tr> <tr> <th>phone</th><td><input type="text" name="phone" pattern="[0-
9]{10}" required class="form-control"></td>

</tr> <tr> <th>email</th>

<td><input type="text" name="email" required class="form-control" pattern="[a-zA-Z0-
9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}"></td>

</tr><tr><th>Pincode</th>

<td><input type="text" name="pin" pattern="[0-9]{6}" required class="form-control"></td>

</tr> <tr> <th>Gender</th> <td>

<input type="radio" name="gen" id="" value="male">Male
```

ONLINE FOOD DELIVERY

```

<input type="radio" name="gen" id="" value="female">Female
<input type="radio" name="gen" value="other">Other
</td> </tr> <tr><th>Age</th>
<td><input type="text" name="age" required class="form-control"></td>
</tr> </tr> <tr> <th>User Name</th>
<td><input type="text" name="username" value="" class="form-control" required></td>
</tr> <tr> <!-- <th>Password</th>
<td><input type="Password" name="password" value="" class="form-control"
pattern="(?!.*\d)(?!.*[a-z])(?!.*[A-Z]).{5,}" title="Must contain at least one number and one
uppercase and lowercase letter, and at least 5 or more characters" required> -->
<th>Password</th>
<td><input type="Password" name="password" value="" class="form-control"
required pattern="(?!.*\d)(?!.*[a-z])(?!.*[A-Z]).{5,}" title="Must contain at least one number and
one uppercase and lowercase letter, and at least 5 or more characters">
</td> </tr><tr>
<td colspan="2" align="center"><input type="submit" name="submit" value="ADD"
class="btn btn-warning"></td>
</tr></table></form></center>
<!-- <a class="scroll down" href="#content-down"></a> --></div></div><!--content-->

```

User Booking

```

div class="banner"> <center><form>
<h1 style="color: #fff;">BOOKINGS</h1>
<table class="table" style="width: 1000px;color: #fff;"><tr>
<th>Passenger Name</th>
<th>Food Name</th><th>Price</th> <th>Quantity</th><th>Date</th>
<th>Status</th> <th>Action</th>
</tr> { % for booking in q % }
<tr><td>{{ booking.passenger.fname }} {{ booking.passenger.lname }}</td>
<td>{{ booking.menu.food_name }}</td> <td>{{ booking.price }}</td>

```


ONLINE FOOD DELIVERY

```
<td>{{ booking.qty }}</td> <td>{{ booking.date_time }}</td> <td>{{ booking.status }}</td> {% if booking.status == "accept" % }
```

```
<td><a href="/payment/{{ booking.id }}/{{ booking.price }}">Make  
Payment</a></td>{% endif % }
```

```
</tr>{% endfor % } </table></form></center></div>
```

User Payment

```
<!-- <h1 align="center">Payment Options</h1> -->
```

```
<center>
```

```
<style type="text/css">
```

```
td{background-color: transparent;font-weight: 2px;color: rgb(255, 255, 255)}
```

```
hr{border-color: orange}
```

```
#b { border: 1px solid grey; padding: 10px;
```

```
}</style><br><div align="center">
```

```
<center><div align="center"><h2 style="color: rgb(255, 255, 255);"> Payment </h2><br><table>
```

```
<table style="width: 500px; border-radius: 5px;" class="table table-borderless" id="b">
```

```
<tr> <td>PAYMENT DETAILS</td>
```

```
<td colspan="2" align="right"></td>
```

```
</tr> <tr><td colspan="2"> <small>CARD NUMBER</small><br>
```

```
<input type="text" placeholder="CARD NUMBER" class="form-control" required  
pattern="[0-9]{16}" maxlength="16" title="Enter 16 digit CARD NUMBER number">
```

```
</td>
```

```
<td > <small>CVV</small><br>
```

```
<input type="text" placeholder="CVV" class="form-control" required pattern="[0-9]{3}"  
maxlength="3" title="Enter 3 digit CV number">
```

```
</td>
```

```
<td> <small>EXPIRATION DATE</small><br>
```

```
<input type="date" placeholder="MM/YY" class="form-control" required  
pattern="[0-9,/]{5}" name="edate" title="Enter month and year">
```

```
</td>
```

ONLINE FOOD DELIVERY

```

<td colspan="2"> <small>CARD HOLDER</small><br>
    <input type="text" placeholder="Name on card" class="form-control" data-
valid='only-text' required >
</td> <td colspan="2" >
    <!-- <h6>Service Charge : {{pr}}</h6> -->          <h6>Amount : {{total}}</h6>

</td>          <td colspan="2" ><!-- <h6>Service Charge : {{pr}}</h6> -->
<h6>*this includes service charge for the application</h6>
</td>
> <td colspan="2" ><h6 style="color: rgb(0, 0, 0);font-size: 1em;"> Total :5000</h6> </td>
    </tr> -->    <input type="submit" value="PAY" class="btn btn-success" style="width:
100%" name="pay"></td>
</table></div></form></div></section></center>

```



Conclusion and Future Enhancements

10. CONCLUSION AND FUTURE ENHANCEMENTS

CONCLUSION:

In envisioning the future enhancements of the online food delivery project, a multitude of innovative features and advancements come to mind, all aimed at elevating user experience, operational efficiency, and market competitiveness. One potential enhancement involves the integration of personalized recommendation systems powered by advanced machine learning algorithms. By analyzing user preferences, order history, and browsing behavior, the platform can provide tailored food recommendations, enhancing customer engagement and satisfaction by offering suggestions aligned with individual tastes and preferences. This feature not only simplifies the ordering process for users but also promotes exploration of new cuisines and dishes, ultimately driving increased order frequency and customer loyalty.

FUTURE ENHANCEMENTS:

Personalized Recommendations: Implementing advanced machine learning algorithms to analyze user preferences, order history, and browsing behavior to provide personalized food recommendations. This feature would enhance customer engagement and satisfaction by offering tailored suggestions based on individual tastes and preferences.

Augmented Reality Menu Viewing: Introducing augmented reality (AR) technology to allow customers to visualize menu items in a virtual environment before placing their orders. This immersive experience would enable customers to make more informed decisions and enhance their overall dining experience.

Voice-Activated Ordering: Integrating voice recognition technology to enable customers to place orders using voice commands via virtual assistants or smart speakers. This hands-free ordering capability would offer convenience and accessibility, particularly for users with disabilities or those multitasking.

ONLINE FOOD DELIVERY

Drone Delivery Services: Exploring the feasibility of drone delivery services for faster and more efficient order fulfillment, especially in densely populated urban areas. Drone delivery could significantly reduce delivery times and costs while offering a unique and futuristic delivery experience to customers.

Blockchain-Based Food Traceability: Implementing blockchain technology to enhance food traceability and transparency throughout the supply chain. By recording and tracking every step of the food delivery process on a decentralized ledger, customers can have confidence in the origin, quality, and safety of their food orders.

Subscription-Based Meal Plans: Introducing subscription-based meal plans that offer customers the convenience of recurring orders for their favorite meals at discounted prices. This feature would encourage customer loyalty and repeat business while providing predictable revenue streams for restaurants.

Virtual Kitchen Collaborations: Partnering with virtual kitchens or ghost kitchens to expand the range of cuisines and dining options available on the platform. Virtual kitchen collaborations would enable restaurants to experiment with new concepts and reach a wider audience without the need for physical storefronts.

Gamification and Rewards Programs: Implementing gamification elements and rewards programs to incentivize customer engagement and loyalty. Features such as challenges, achievements, and loyalty points could encourage customers to interact more frequently with the platform and earn rewards for their loyalty.

Sustainability Initiatives: Introducing sustainability initiatives such as eco-friendly packaging options, carbon offset programs, and partnerships with local farmers and suppliers to promote sustainable and environmentally conscious practices throughout the food delivery process.

Integration with Emerging Technologies: Continuously exploring and integrating emerging technologies such as artificial intelligence, Internet of Things (IoT), and 5G connectivity to further enhance the platform's capabilities, improve operational efficiency, and stay ahead of market trends.



PRESIDENCY COLLEGE

(AUTONOMOUS)

AFFILIATED TO BENGALURU CITY UNIVERSITY, APPROVED BY AICTE, DELHI & RECOGNISED BY THE GOVT. OF KARNATAKA

RE-ACCREDITED BY NAAC WITH 'A+' GRADE

ONLINE FOOD DELIVERY

Bibliography



ONLINE FOOD DELIVERY

11. BIBLIOGRAPHY

Books Referred:

Software Engineering by Roger Pressman

Database Systems by Abraham Silberschatz

Sqlyog Community - GitHub

Websites referred:

JavaScript <https://javascript.info/>

HTML <https://html.com/>

Bootstrap <https://getbootstrap.com/>

CSS <https://www.w3schools.com/css/>

MySQL <https://www.javatpoint.com/mysql-tutorial>

PYTHON <https://www.python.org>

<https://www.kaggle.com/>