



A

PROJECT REPORT ON

“ONLINE FOOD ORDERING SYSTEM”

SUBMITTED TO

SHIVAJI UNIVERSITY, KOLHAPUR.

FOR THE PARTIAL FULFILMENT OF

BACHELOR OF COMPUTER SCIENCE

(B.Sc. - III SEMESTER-VI)

BY

MR. SHRIVARDHAN VINAYAK PATIL.

MR. SHRIVATSA SACHIN KORE.

UNDER THE GUIDANCE OF

PROF.A. C. KUMBHAR

THROUGH

THE PRINCIPAL

DR. N. L. KADAM

D. R. MANE MAHAVIDYALAYA, KAGAL

2024-2025



D. R. MANE MAHAVIDYALAYA, KAGAL

CERTIFICATE

This is to certify that **MR. SHRIVARDHAN VINAYAK PATIL** and **MR. SHRIVATSA SACHIN KORE** have satisfactorily completed the project work on "**ONLINE FOOD ORDERING SYSTEM**" under the guidance of **MR. A. C. KUMBHAR** For the partial fulfillment of B.Sc.-III Sem-VI submitted to "**Shivaji University, Kolhapur**" during the academic year 2024- 2025.

To the best of my knowledge and belief the matter presented by them are original and not copied from any source. Also this report has not been submitted earlier for the award of any Degree or Diploma of Shivaji University or any other university.

Place: Kagal

Date:

External Examiner

Mr. Arun. C. Kumbhar
(HOD)

External Examiner

Dr. N. L. Kadam
(Principal)



D. R. MANE MAHAVIDYALAYA, KAGAL.

GUIDE'S CERTIFICATE

This is to certify that **MR. SHRIVARDHAN VINAYAK PATIL** and **MR. SHRIVATSA SACHIN KORE** have satisfactorily completed the project work on "**ONLINE FOOD ORDERING SYSTEM**" for the partial fulfillment of B.Sc.-III Sem-VI submitted to "**Shivaji University, Kolhapur**" during the academic year 2024-2025 under my guidance.

To the best of my knowledge and belief the matter presented by them are original and not copied from any source. Also this report has not been submitted earlier for the award of any Degree or Diploma of Shivaji University.

Place: Kagal

MR. A. C. KUMBHAR

Date:

(project Guide)

DECLARATION

We hereby declare that the project report entitled "**ONLINE FOOD ORDERING SYSTEM**" submitted in partial fulfillment of the requirements for the degree B.Sc.-III Semester-VI of "**Shivaji University, Kolhapur**" is our original work and not submitted for the award of any other degree, diploma, fellowship or any other similar titles or prizes.

Date: / /2025

Place: -Kagal

Sr. No	Name	Sign
1.	Mr. Patil Shrivardhan Vinayak	
2.	Mr. Kore Shrivatsa Sachin	

ACKNOWLEDGEMENT

As per B.Sc.-III course curriculum our project is successfully completed. In this course of period our project guide **MR. A.C. KUMBHAR** helped us in every respect for making this project successful.

We are also thankful to our learning Centre **D. R. MANE MAHAVIDYALAYA, KAGAL** Principal **Dr. N. L. Kadam**, center co- coordinator **MR. A.C. KUMBHAR** for providing well equipped laboratory and also thankful to for his internal supervision during the course.

All our college and seniors gave us valuable suggestions to improve our application and inspired us to make the project successful. We express sincere gratitude for them.

Date:- / /2025

Place: - Kagal

Sr. No	Name	Sign
1.	Mr. Patil Shrivardhan Vinayak	
2.	Mr. Kore Shrivatsa Sachin	

INDEX

Sr. No	Particular	Page No.
1.	Introduction to project <ul style="list-style-type: none">• Introduction• Existing system• Need & Scope• Organization Profile	
2.	Proposed System <ul style="list-style-type: none">• Objectives• Requirement Engineering• Requirement Gathering	
3.	System analysis <ul style="list-style-type: none">• Feasibility study• Platform Selection	
4.	System Diagram <ul style="list-style-type: none">• DFD• ERD• System Requirement	
5.	System Requirements <ul style="list-style-type: none">• Hardware• Software	
6.	System Design <ul style="list-style-type: none">• Database Design• Input Design• Output Design	
7.	User Guideline <ul style="list-style-type: none">• Installation process	
8.	Source code	
9.	Outputs <ul style="list-style-type: none">• Input screen and reports	
10.	Conclusion and Suggestions <ul style="list-style-type: none">• Conclusion and suggestions• Future Enhancements• Bibliography	

INTRODUCTION TO PROJECT

INTRODUCTION

The online food ordering system is a user-friendly platform designed to streamline the process of ordering food. With the increasing demand for convenience and efficiency in today's fast-paced lifestyle, this system bridges the gap between customers and food stores, allowing them to interact seamlessly. Users can browse menus, select dishes, and place orders directly through the website , eliminating the need for physical visits or phone calls. This digital transformation ensures a more personalized and efficient dining experience.

One of the core objectives of this system is to enhance customer satisfaction by providing a hassle-free ordering process. The system allows users to explore a variety of fast-food based on preferences, or choices. Additionally, it ensures accurate order placement, reducing human errors often associated with manual processes. benefit from an organized order management system that improves operational efficiency

The system integrates features such as secure payment gateways and user login for security concerns. These functionalities not only provide convenience but also build trust and loyalty among users. User friendly design , further enhance the user experience, making it a preferred choice for modern consumers.

This online food ordering system also promotes digitalization and sustainability by reducing the need for printed menus and receipts. Customers can enjoy their favorite meals at the click of a button

In conclusion, the online food ordering system is a versatile and innovative solution for modern food service needs. It benefits customers by saving time and offering convenience while empowering to deliver better services and increase profitability. This system exemplifies how technology can transform traditional business models into efficient, customer-centric platforms.

EXISTING SYSTEM

EXISTING SYSTEM

These platforms allow customers to browse menus, place orders, and make payments online. However, restaurants often face high commission fees and limited control over their branding and customer interactions.

The orders section lets users to choose their favorite dishes. The contact submissions section helps manage customer inquiries, while the feedback section allows customers to share their experiences.

The dashboard is divided into several sections, including orders, contact submissions, and feedback submissions, each with its own unique features and functionality.

Overall, while online food ordering is widely available, many existing systems lack flexibility, security, and efficiency. A dedicated, well-optimized online food ordering system can improve restaurant operations, reduce dependency on third-party apps, and provide a seamless customer experience.

NEED AND SCOPE

Need:

With increasing demand for convenience, customers prefer online food ordering over traditional methods. Phone calls and walk-ins are time-consuming and error-prone. A dedicated system ensures a faster and smoother process.

Restaurants need an efficient system to manage orders, reduce errors, and handle peak hours. Automation improves efficiency, enhances customer satisfaction, and eases staff workload.

Digital payments and contactless delivery have made online ordering essential. A secure and user-friendly system ensures smooth transactions and boosts customer trust, helping businesses grow.

Scope:

The FoodHub project aims to provide a comprehensive platform that streamlines the food ordering process while enhancing user engagement through features like dish suggestions, feedback collection, and contact management.

The system allows users to easily place orders, suggest new dishes, provide feedback, and register seamlessly. It also provides an intuitive admin dashboard where administrators can manage orders, view user suggestions, analyze feedback, and respond to customer queries.

This system is scalable and can be further extended to include features such as online payment integration, and personalized recommendations based on user preferences. With a user-friendly interface and efficient backend management, FoodHub ensures a smoother experience.

ORGANIZATION PROFILE

INTRODUCTION :

FoodHub is a dynamic food ordering service that was founded 10 years ago by Mr. Ram Patil in Belgaum, India. Initially operating as an offline fast food ordering service, FoodHub quickly gained popularity due to its quick service and diverse menu offerings. Recognizing the evolving market trends and the growing demand for online services, the founder took a strategic step to digitize the business by launching a user-friendly web application.

Today, FoodHub stands as a reliable online food ordering platform in India, offering a wide range of fast food items and meal combos through a single, easy-to-navigate website. The platform aims to provide a seamless and convenient ordering experience for customers, combining speed, variety, and customer satisfaction at its core. With its digital transformation, FoodHub continues to grow and adapt, embracing technology to meet modern customer expectations and enhance service delivery.

PROPOSED SYSTEM

Proposed System

- A proposed system for a food ordering , FoodHub could include the following key features:
- Provides a user-friendly online platform for customers to browse, select, and order food easily.
- Allows users to create accounts, login securely.
- Facilitates a seamless food ordering system where users can choose items, customize orders, and place them in real-time.
- Includes an admin dashboard to manage orders, view feedback, handle contact submissions, and monitor registered users.
- Provides a dedicated "Your Taste" feature where customers can suggest new dishes to improve customer engagement.
- Supports contact and feedback forms for users to communicate their issues or suggestions.
- Ensures data validation and secure backend integration using Node.js and MySQL.
- Enables the admin to view and manage customer suggestions, feedback, and user details in one central location.
- Aims to provide a faster, more efficient alternative to the traditional offline food ordering process

Objective

- Provide a centralized food ordering platform where users can browse a variety of fast food options from multiple vendors.
- Enhance user convenience by offering a smooth ordering experience with secure payments.
- Ensure data security by protecting user information.
- Improve vendor accessibility by enabling multiple food vendors to list their products and manage their menus easily.
- Enable feedback and support by allowing users to review orders, rate vendors, and get customer assistance.
- Expand market reach by connecting food vendors with a wider audience through an online ordering system.

Requirement Gathering

Requirement Gathering is the process of generating a list of functional, technical and systematic requirements from several project stakeholders, such as clients, IT staff, product users or vendors. This list may likely include features, activities and tasks for a team to execute in order to achieve the goals of a project. Usually, there are two types of requirements to consider: functional and non-functional. Functional requirements include the information, interactions and processes that a client requests. Non-functional requirements are other technical and operational aspects of a project.

Consider starting the process of requirements gathering at the beginning of a project to help ensure effective planning and management. Here are some aspects to consider as you start requirements gathering:

Timeline:

This entails estimating the overall length and schedule of a project. This can help you effectively plan for its requirements over the course of an entire project, ensuring that you're prepared throughout its duration.

People:

Consider who you want to include in a project and what their various roles might be. Try to determine their individual requirements, as well as how they might collaborate with the entire team.

Goals:

Identifying primary goals early can help you determine a project's requirements and ensure that the project specifically focuses on executing these objectives.

Fact Finding Techniques

In order to gather analyze the relevant information following method or adopted.

1) Observation:-

Find hand information about various activities carried out can be studied through observation. This method used to solve how documents where handled, how processes where carried out and whether those stapes when actually followed thus current scenario was understood using observation.

2) Record Review:-

The study of records maintained to give clear idea as to how actual process carried also management working on whole.

3) Document Searching:-

Various documents which are handled these used formats where studied relation with system, information analyzed.

Requirement Analysis

Requirement analysis is a software engineering task that bridges the gap between system level requirements engineering & software design. Requirement engineering activities results in the specification of software's operational characteristics include software's interface with other system elements, and establish constraints that software must meet. Requirement analysis provides the software designer with representation of information, function, & behavior that can be translated to data, architectural, interface & component-level designs. Finally the requirement specification provides the developer & the customer with the means to access quality ones software.

Software requirement analysis may be divided into five area of efforts :

- Problem recognition
- Evaluation & synthesis
- Modeling
- Specification
- Review

Initially, the analyst studies the systems specification & the software project plan. It is important to understand software in a system context & to review the software the software scope that was used to generate planning estimate. Next communication for analysis must be established so that the problem recognition is ensured. The goal is recognition of the basic problem elements as perceived by customers or user.

Problem evaluation & solution synthesis is the next major area of effort for analysis. The analyst must all externally observable data objects, evaluate the flow & content of information, define & elaborate all software functions, understand software behavior in the context of events that affects the system, establish system interface characteristics, & uncover additional design constraints.

System Analysis

Feasibility Study

- **Technical Feasibility:**

At first it's necessary to check that the proposed system is technically feasible or not & to determine the technology and skill necessary to carry out the project. If they are not available then find out the solution to obtain them. Hardware is already available in the collage.

- **Social Feasibility:**

Although generally there is always resistance, initially to any change in the system is aimed at reliving the work load of the user to extent the system is going to facilitate user to perform operation like calculating salary amount and deduction, generation reports with less possible error.

Platform Selection

We have developed **FoodHub**, an online food ordering system, using **HTML, CSS, and JavaScript** for the front end and **(Node.js)** with Express.js , Xampp server as the back-end database. These technologies ensure a responsive user interface and efficient data management for seamless order processing.

A **database** is a structured collection of related data, while a **database management system (DBMS)** is software that enables users to create, store, and manage data efficiently. MySQL is used as the DBMS for **FoodHub** to handle user accounts, orders, payments, and vendor details.

A database allows defining, constructing, and manipulating data. **Defining** involves structuring the database, **constructing** stores the data, and **manipulating** enables retrieving and updating records as needed.

To design and maintain an efficient database, we must:

- Identify relevant data for the food ordering system.
- Define specific attributes like orders, customers, and vendors.
- Establish relationships between objects such as users, orders, and payments.

The designed database ensures efficient data storage, quick retrieval, and smooth operation of **FoodHub**, making online food ordering hassle-free.

Characteristics of Database Management System:

- It keeps complex relation between data.
- Keeps a tight control of data redundancy
- Enforces user defined rules to insure the integrity of table data
- Ensures that data can be shared across applications
- Enforces data access authorization

➤ Front End Tool [visual Studio IDE 2024-25]:

- Visual Studio is a powerful developer tool that you can use to complete the entire development cycle in one place. It is a comprehensive integrated development environment (IDE) that you can use to write, edit, debug, and build code, and then deploy your app. Beyond code editing and debugging, Visual Studio includes compilers, code completion tools, source control, extensions, and many more features to enhance every stage of the software development process.
- With the variety of features and languages support in Visual Studio, you can grow from writing your first "Hello World" program to developing and deploying apps. For example, build, debug, and test .NET and C++ apps, edit ASP.NET pages in the web designer view, develop cross-platform mobile and desktop apps with .NET, or build responsive Web UIs in C#.

➤ Backend- MySQL:

MySQL is an open-source relational database management system (RDBMS) that uses Structured Query Language (SQL) for managing and accessing data. The name “MySQL” combines “My” (the name of co-founder Michael Widenius’s daughter) and “SQL.” It organizes information into structured tables and supports relationships among them to efficiently handle large sets of data.

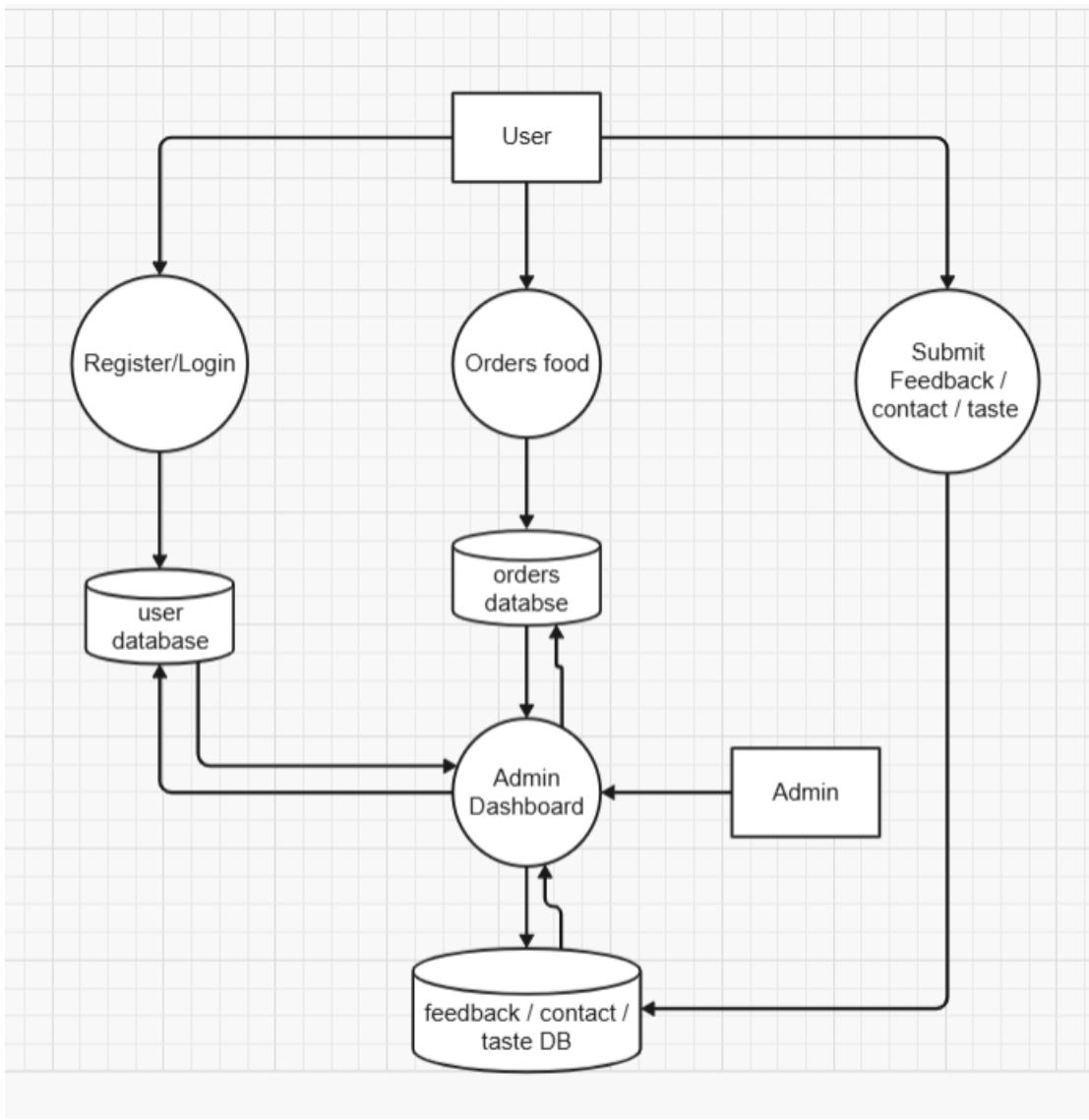
As an RDBMS, MySQL works closely with the operating system to store, retrieve, and manage data. It also offers features for data security, user access control, data integrity checks, and backup support. Developers use SQL commands to create, update, and retrieve data, making MySQL highly suitable for web-based and enterprise-level applications.

MySQL is available under the GNU General Public License as well as commercial licenses. Originally developed by MySQL AB (a Swedish company), it was later acquired by Sun Microsystems, and eventually by Oracle Corporation. After Oracle’s acquisition, the original developers created MariaDB, a fork of MySQL, to maintain its open-source nature.

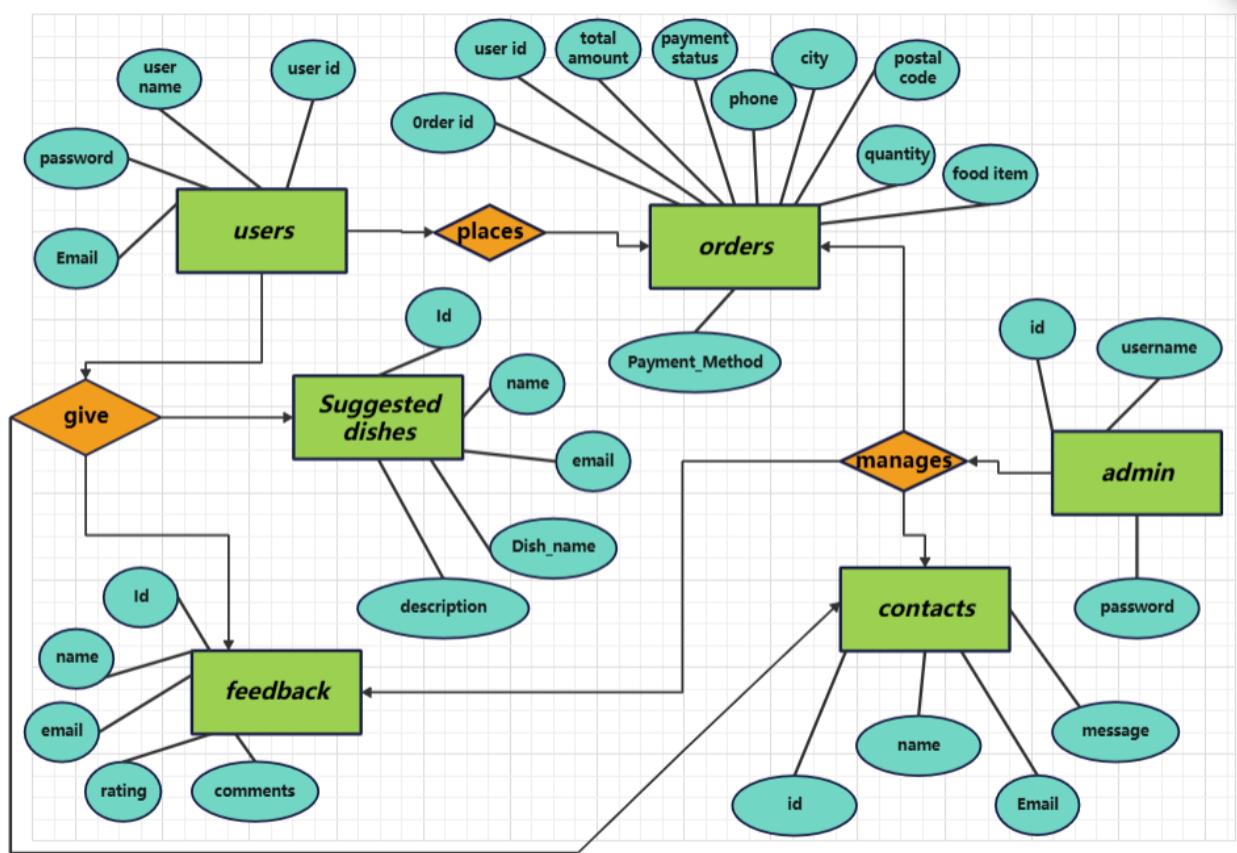
MySQL is an integral part of the LAMP stack—Linux, Apache, MySQL, and PHP/Perl/Python—used widely in web development. It is compatible with many programming languages and frameworks, making it a reliable choice for developers building dynamic and scalable applications.

System Diagram

DFD



ERD



Software Testing

System Testing

Testing presents an interesting anomaly for the software engineer during earlier software activities, the engineer attempts to build software from an abstract concept to a tangible product. The engineer creates a serial of test cases that are intended to "demolish" the software that has been built testing is conducted at various level of project

Module Testing

After doing unit testing, whole system is tested whether it is working properly or not. The screen together works as per requirement without effecting working of controls. In this we made sure that all the fields that should not be left blank or little by end-user or not.

Integrated System Testing

Validation Testing:-

At the culmination of integration testing, software is completely assembled as a package, interfacing errors have been uncovered and corrected and a final series of software tests-validation testing-may begin. Validation succeeds when software function in a manner that can be reasonable expected by the customer

System Testing:-

Software is incorporated with other system element (i.e. Hardware, People, information) and a series of system integration and validation tests are not conducted solely by software engineers. However, steps taken during software design and testing can greatly improve the probability of successful integration in the larger system

SYSTEM REQUIREMENTS

➤ HARDWARE REQUIREMENT :

The For Client/User System:

- Processor: Intel Core i3 or above
- RAM: Minimum 4 GB
- Hard Disk: At least 250 GB
- Display: 1024x768 resolution or higher
- Input Devices: Keyboard and Mouse
- Internet Connection: Required for online access

For Server System (Development/Hosting):

- Processor: Intel Core i5 or higher
- RAM: Minimum 8 GB
- Hard Disk: At least 500 GB
- Internet: High-speed connection for handling multiple requests
- Other: Backup power source recommended (UPS)

➤ SOFTWARE REQUIREMENT :

Frontend Technologies:

- HTML5, CSS3, JavaScript
- Web Browser: Google Chrome, Mozilla Firefox, or Microsoft Edge

Backend Technologies:

- Node.js (Express framework)
- Database: MySQL 8.0 CE
- Validation Library: Joi (for input validation)

Development Tools:

- Visual Studio Code or any preferred code editor
- XAMPP / MySQL Workbench (For managing MySQL database)
- Postman (For API testing)
- Git (For version control, optional)

Operating System:

- Windows 10/11, Linux, or macOS

System Design

Database Design

Admin

Server: 127.0.0.1 » Database: food_ordering » Table: admin

The screenshot shows the MySQL Workbench interface with the following details:

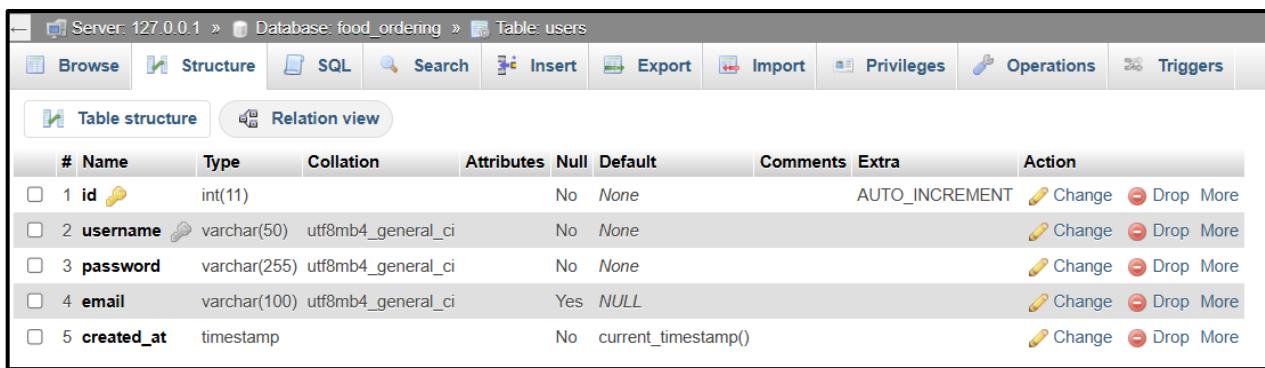
- Server: 127.0.0.1
- Database: food_ordering
- Table: admin

The toolbar at the top includes: Browse, Structure, SQL, Search, Insert, Export, Import, Privileges, Operations, and Triggers.

The "Table structure" tab is selected, showing the following table definition:

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	username	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
3	password	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More

Users



The screenshot shows the 'Structure' tab of the MySQL Workbench interface for the 'users' table. The table has five columns: 'id', 'username', 'password', 'email', and 'created_at'. The 'id' column is defined as an int(11) type with a primary key icon, set to not null with a default value of 'None'. It also has an 'AUTO_INCREMENT' attribute and includes 'Change', 'Drop', and 'More' action buttons. The other four columns ('username', 'password', 'email', 'created_at') are defined as varchar types with specific lengths and collations (utf8mb4_general_ci). The 'username' and 'password' columns are not null, while 'email' is nullable ('Yes') with a default value of 'NULL'. The 'created_at' column is not null and uses the 'current_timestamp()' function as its default value. Action buttons for 'Change', 'Drop', and 'More' are available for each of these columns as well.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id 	int(11)			No	None		AUTO_INCREMENT	 Change  Drop  More
2	username 	varchar(50)	utf8mb4_general_ci		No	None			 Change  Drop  More
3	password	varchar(255)	utf8mb4_general_ci		No	None			 Change  Drop  More
4	email	varchar(100)	utf8mb4_general_ci		Yes	NULL			 Change  Drop  More
5	created_at	timestamp			No	current_timestamp()			 Change  Drop  More

Contacts

Server: 127.0.0.1 » Database: food_ordering » Table: contacts

Browse Structure SQL Search Insert Export Import Privileges Operations Triggers

Table structure Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	name	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
3	email	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
4	message	text	utf8mb4_general_ci		No	None			Change Drop More
5	submitted_at	timestamp			No	current_timestamp()			Change Drop More

Check all With selected: Browse Change Drop Primary Unique Index Spatial Fulltext

Feedback

The screenshot shows the MySQL Workbench interface for the 'feedback' table in the 'food_ordering' database. The table has seven columns: id, name, email, rating, comments, created_at, and submitted_at. The 'id' column is defined as an int(11) with AUTO_INCREMENT, while the other columns are varchar or timestamp types. The 'rating' column is nullable (Yes), and the 'comments' column is of type text.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id 🛡	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	name	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
3	email	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
4	rating	varchar(50)	utf8mb4_general_ci		Yes	NULL			Change Drop More
5	comments	text	utf8mb4_general_ci		No	None			Change Drop More
6	created_at	timestamp			No	current_timestamp()			Change Drop More
7	submitted_at	timestamp			No	current_timestamp()			Change Drop More

With selected: Check all **With selected:** Browse Change Drop Primary Unique Index Spatial Fulltext

Orders

Server: 127.0.0.1 » Database: food_ordering » Table: orders

Browse Structure SQL Search Insert Export Import Privileges Operations Triggers

Table structure Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id ⚒	int(11)			No	None		AUTO_INCREMENT	More
2	name	varchar(255)	utf8mb4_general_ci		No	None			More
3	phone	varchar(20)	utf8mb4_general_ci		No	None			More
4	email	varchar(255)	utf8mb4_general_ci		No	None			More
5	address	text	utf8mb4_general_ci		No	None			More
6	city	varchar(100)	utf8mb4_general_ci		No	None			More
7	postal_code	varchar(20)	utf8mb4_general_ci		No	None			More
8	country	varchar(100)	utf8mb4_general_ci		No	None			More
9	food_item	varchar(255)	utf8mb4_general_ci		No	None			More
10	quantity	int(11)			No	None			More
11	amount	varchar(20)	utf8mb4_general_ci		No	None			More
12	order_date	date			No	None			More
13	special_instructions	text	utf8mb4_general_ci		Yes	NULL			More
14	payment_method	varchar(50)	utf8mb4_general_ci		No	None			More
15	created_at	timestamp			No	current_timestamp()			More

Check all With selected: Browse Primary Unique Index Spatial Fulltext

Suggested Dishes

The screenshot shows the MySQL Workbench interface for the 'suggested_dishes' table in the 'food_ordering' database. The table structure is as follows:

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	name	varchar(100)	utf8mb4_general_ci		Yes	NULL			Change Drop More
3	email	varchar(100)	utf8mb4_general_ci		Yes	NULL			Change Drop More
4	dish_name	varchar(100)	utf8mb4_general_ci		Yes	NULL			Change Drop More
5	description	text	utf8mb4_general_ci		Yes	NULL			Change Drop More
6	submitted_at	timestamp			No	current_timestamp()			Change Drop More

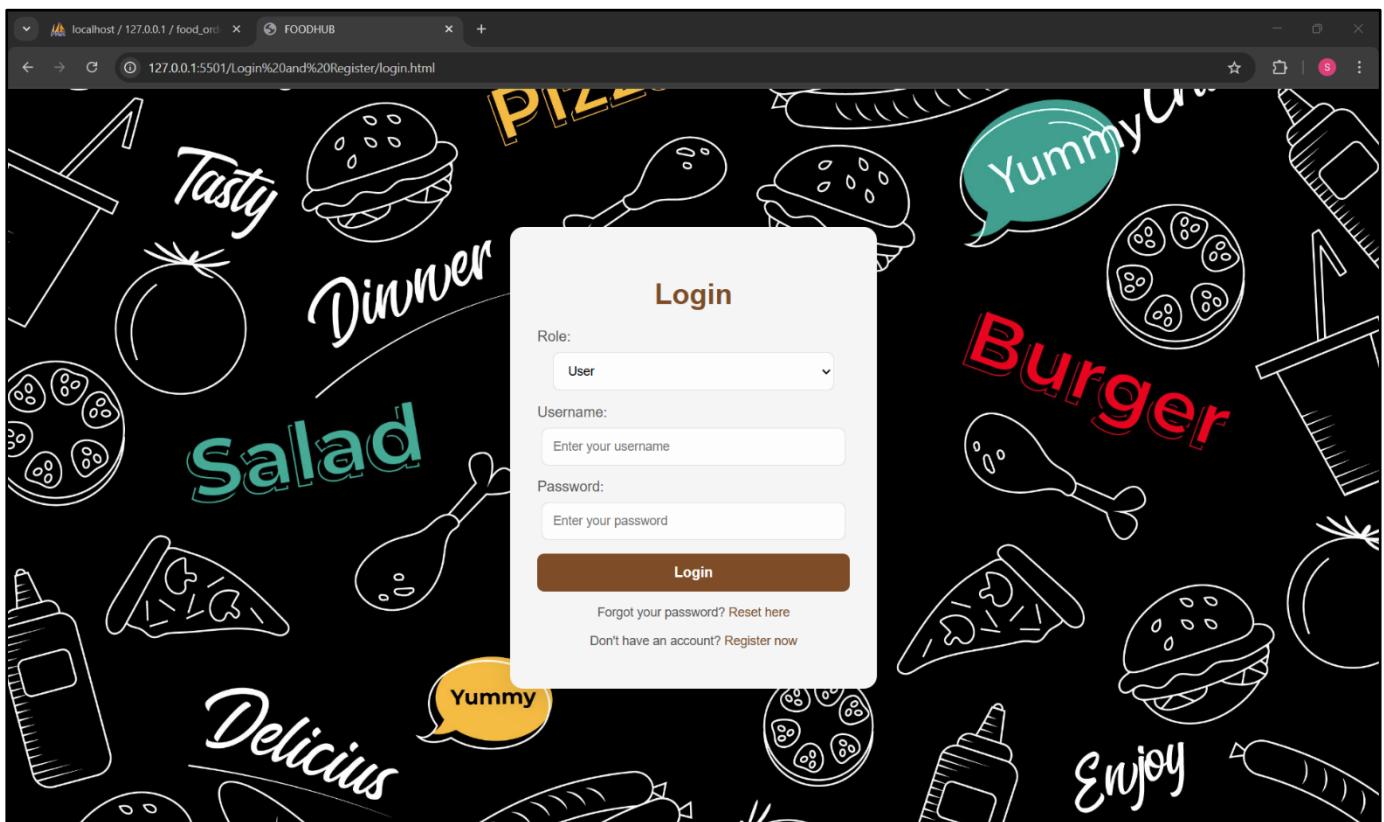
Below the table, there are buttons for 'Check all', 'With selected:', and various actions like 'Browse', 'Change', 'Drop', 'Primary', 'Unique', 'Index', 'Spatial', and 'Fulltext'. At the bottom, there are buttons for 'Print', 'Propose table structure', 'Move columns', 'Normalize', and a search bar with 'Add 1' and 'column(s) after submitted_at'.

Form Design

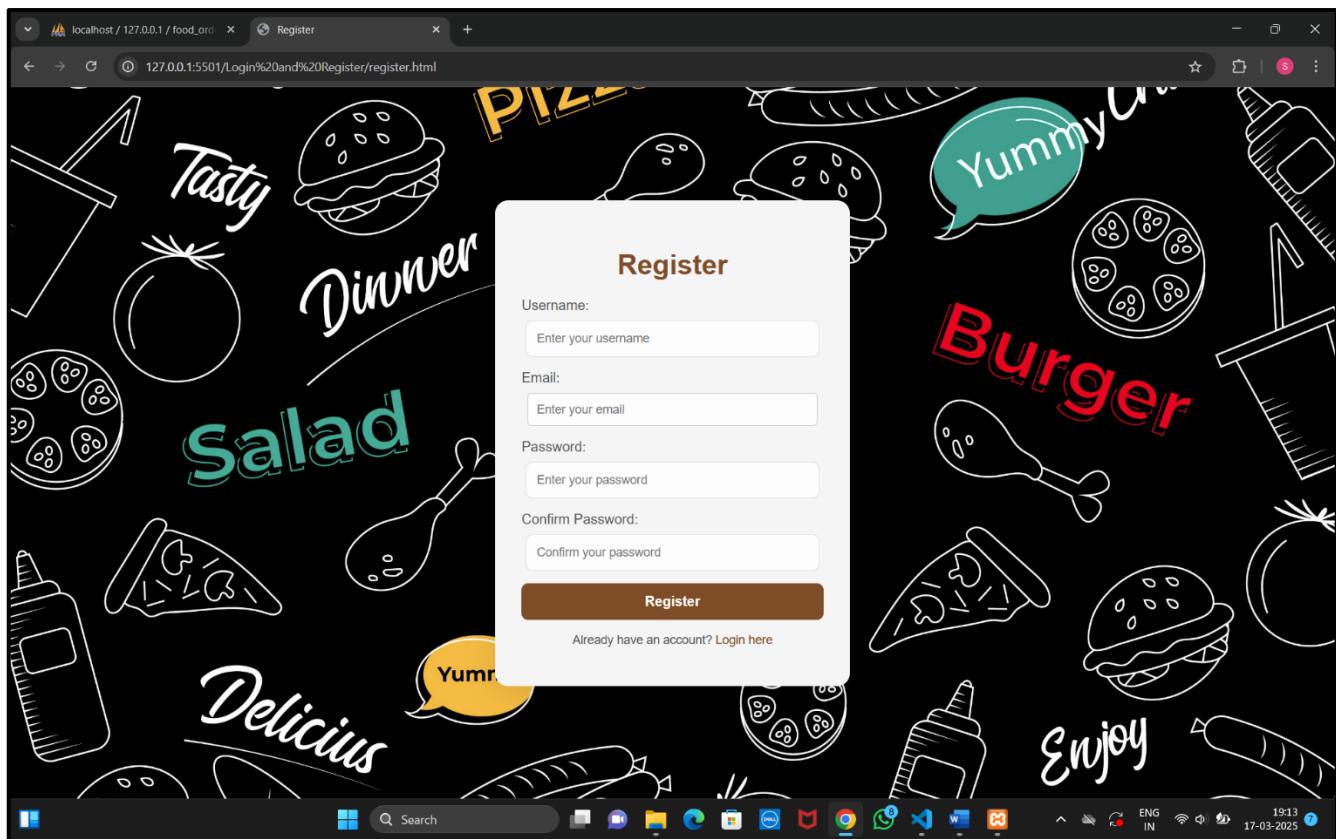
Home

The screenshot shows a web browser window with two tabs open. The active tab is titled 'FOODHUB' and displays the URL '127.0.0.1:5501/Main-web/index.html'. The page itself has a yellow header bar with the 'FoodHub' logo on the left and navigation links for 'LOGIN', 'HOME', 'ORDER', 'CONTACT', 'FEEDBACK', and 'YOUR TASTE' on the right. Below the header is a large image of a meal consisting of French fries, fried chicken nuggets, and a fried egg on a yellow plate. To the right of the image, there is promotional text: 'Craving Something Tasty? Get Delicious Meals Delivered with Ease!' followed by a smaller line 'Satisfy Your Hunger with Just a Few Clicks'. A prominent 'Order now' button is centered below the promotional text. The bottom of the screen shows a Windows taskbar with various pinned icons and system status indicators.

Login



Register



Order

Document 127.0.0.1:5501/Main-web/order.html

FoodHub ORDER

Pizza ₹399/- Delicious cheesy pizza with fresh toppings. Order Now

Burger ₹199/- Juicy beef patty with crispy lettuce and cheese. Order Now

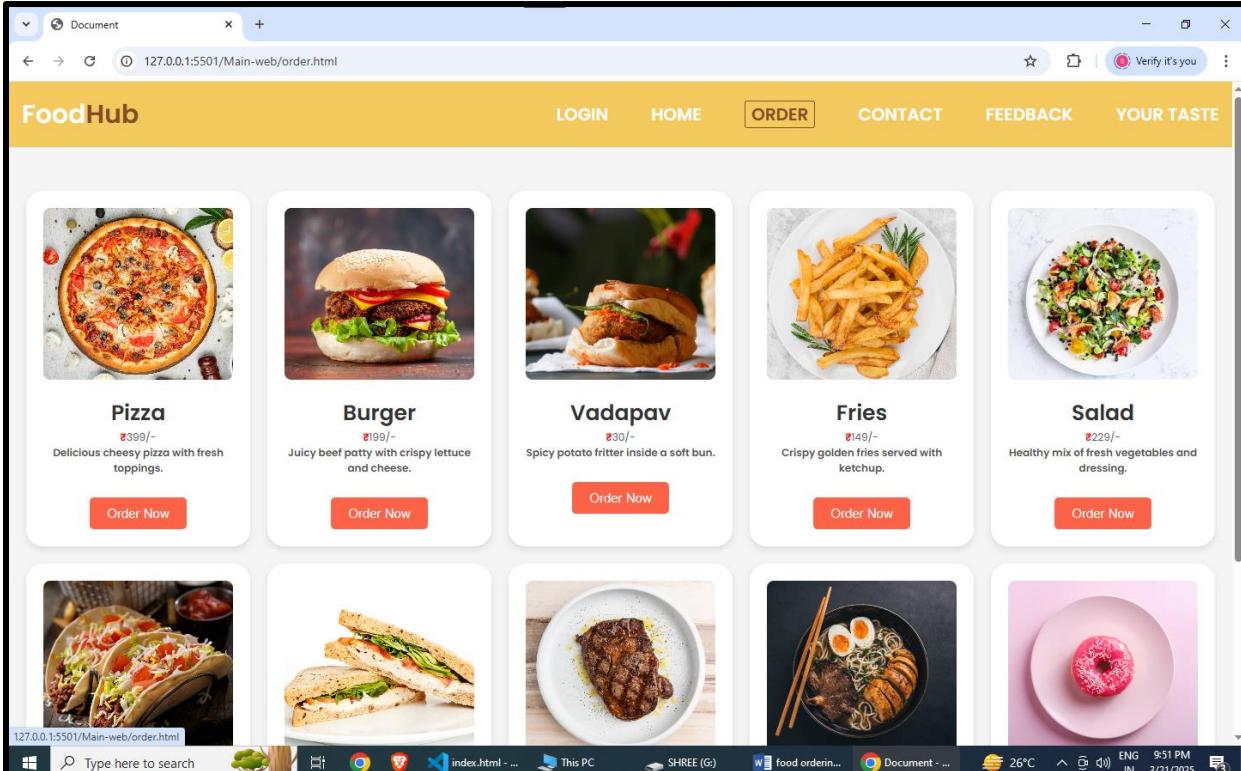
Vadapav ₹30/- Spicy potato fritter inside a soft bun. Order Now

Fries ₹149/- Crispy golden fries served with ketchup. Order Now

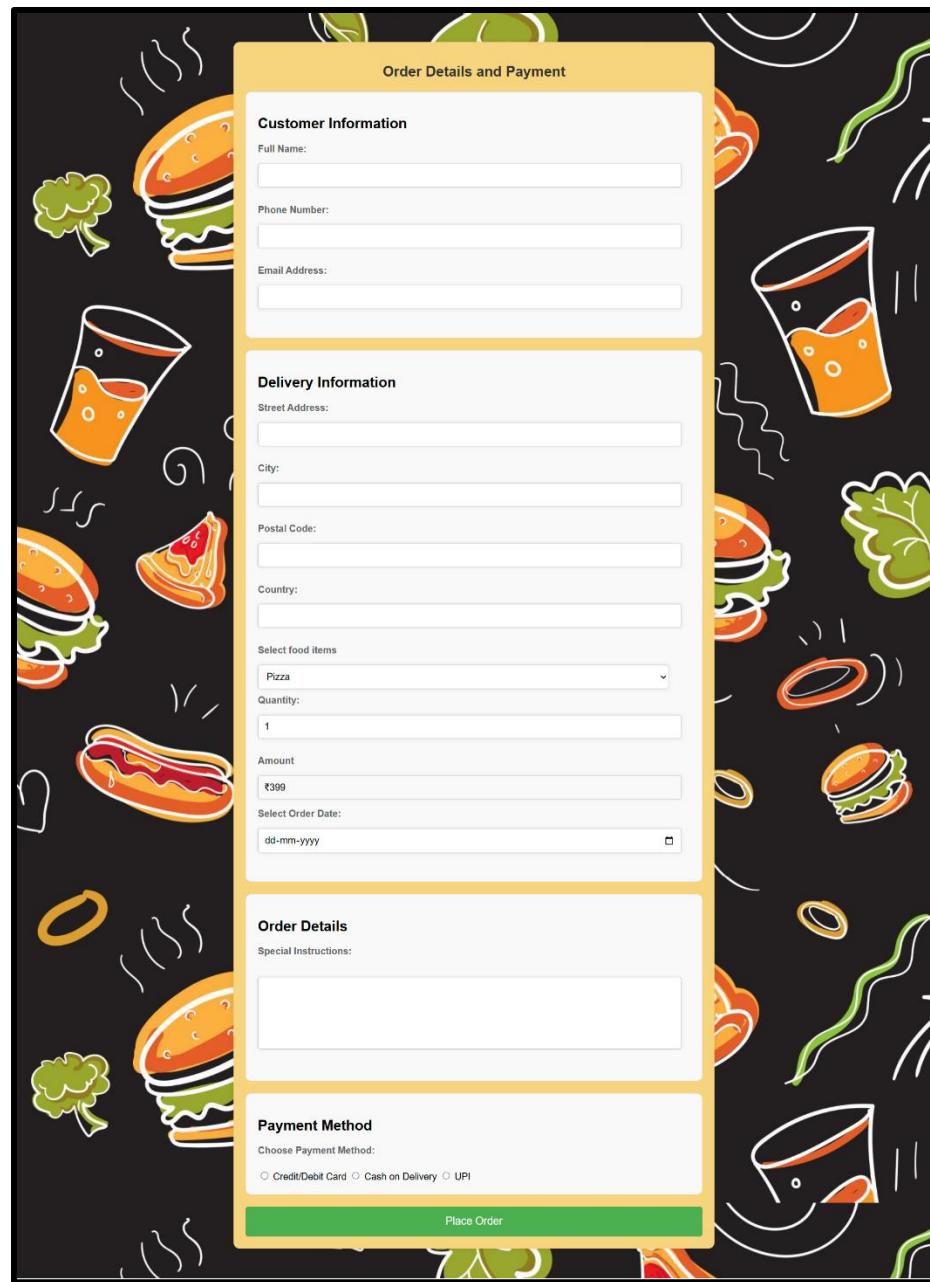
Salad ₹229/- Healthy mix of fresh vegetables and dressing. Order Now

127.0.0.1:5501/Main-web/order.html

Type here to search SHREE (G) food orderin... Document - ... 26°C ENG IN 9:51 PM 3/21/2025



Order Details



Order Details and Payment

Customer Information

Full Name:

Phone Number:

Email Address:

Delivery Information

Street Address:

City:

Postal Code:

Country:

Select food items:

Quantity:

Amount:

Select Order Date:

Order Details

Special Instructions:

Payment Method

Choose Payment Method:

Credit/Debit Card Cash on Delivery UPI

Place Order

Contact

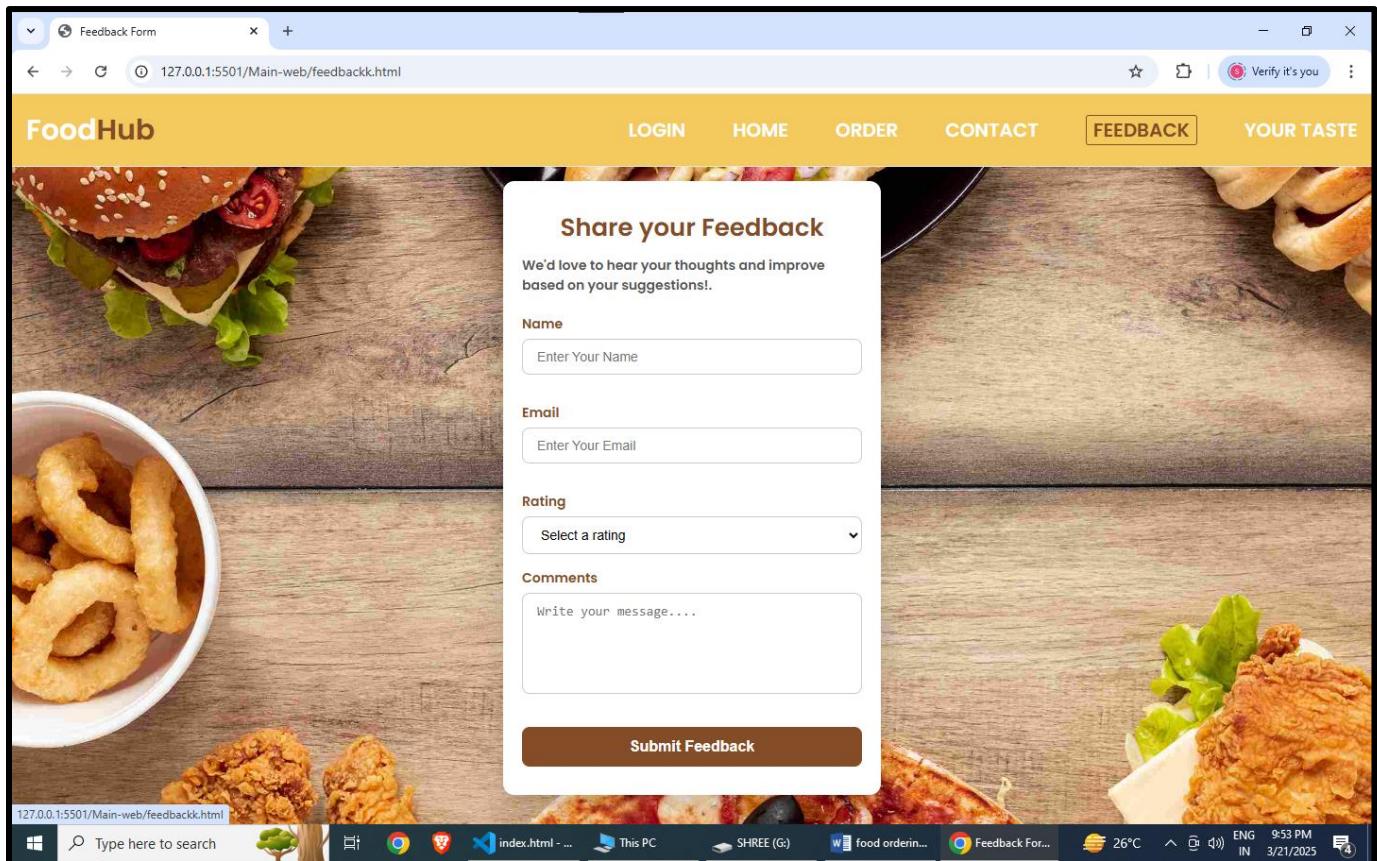
Screenshot of a web browser displaying the "Contact Us" page of a food delivery application named "FoodHub".

The page features a yellow header bar with the "FoodHub" logo on the left and navigation links for LOGIN, HOME, ORDER, CONTACT, FEEDBACK, and YOUR TASTE on the right.

The main content area has a dark background with a large image of a burger and fries on a wooden board on the right side. The title "Contact Us" is centered above three input fields: "Name", "Email", and "Message". A large yellow "SUBMIT" button is positioned below these fields.

The browser's address bar shows the URL `127.0.0.1:5501/Main-web/contact.html`. The taskbar at the bottom of the screen displays various open applications and system status indicators.

Feedback



A screenshot of a web browser displaying a feedback form for a food delivery service named "FoodHub". The page has a yellow header with the logo and navigation links: LOGIN, HOME, ORDER, CONTACT, FEEDBACK (which is highlighted), and YOUR TASTE. The main content area features a background image of various food items like a burger, onion rings, and fried chicken. A central white box contains the feedback form with the following fields:

- Share your Feedback**
- We'd love to hear your thoughts and improve based on your suggestions!.
- Name**:
- Email**:
- Rating**:
- Comments**:
- Submit Feedback** (button)

The browser's address bar shows the URL: 127.0.0.1:5501/Main-web/feedbackk.html. The taskbar at the bottom includes icons for search, file operations, and system status.

Your Taste

The screenshot shows a web browser window with the following details:

- Title Bar:** Your Taste | FOODHUB
- Address Bar:** 127.0.0.1:5501/Main-web/YourTaste.html
- Header:** FoodHub (left), LOGIN, HOME, ORDER, CONTACT, FEEDBACK, YOUR TASTE (right)
- Content Area:** A yellow box containing a form titled "Tell Us About Your Favorite Dish!" with the sub-instruction "We'd love to hear what dish you'd like to see on FoodHub." It includes four input fields: "Your Name", "Your Email", "Dish Name", and "Why do you love it?", followed by a "Submit Suggestion" button.
- Taskbar:** Shows various pinned icons including index.html, This PC, SHREE (G), food orderin..., Your Taste | F..., USD..., ENG IN, and 9:56 PM 3/21/2025.

User Guidelines

We have three main actors in our **FoodHub** system:

- **Admin:** Responsible for managing orders, viewing contact submissions, user suggestions, feedback, and registered users.
- **User (Customer):** Can register, log in, browse dishes, place food orders, submit feedback, and suggest new dishes.
- **System:** Automatically stores user interactions, displays them in the admin dashboard, and manages user access.

Top Use Cases of the FoodHub :

- **Add/Remove/Manage Orders:** Admin can view and manage customer food orders.
- **Browse Dishes:** Users can explore available dishes before placing an order.
- **Submit Feedback:** Users can provide ratings and feedback for their experience.
- **Suggest a Dish ("Your Taste"):** Users can suggest new dishes via a dedicated form.
- **Register / Login / Logout:** Users can create accounts, securely log in.
- **Contact Submission:** Users can send queries or feedback to the admin.

Installation Procedure:

Setup:

- Install the software of visual Studio
- Install Xampp Database Software
- Install Node.js

For User :

- Install a web browser.
- Ensure internet connectivity.
- Access URL of FoodHub.

Coding

Home Page

Index.html

Code:

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>FOODHUB</title>

    <link rel="stylesheet" href="styleeee.css">

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">

    <link
        href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css">

</head>

<body>

    <header class="header">
```

```
<nav class="navbar">

    <a href="#" class="nav-logo">Food<span
style="color:#854d27">Hub</span></a>

    <ul class="nav-menu">

        <li class="nav-item">

            <a href="../Login and Register/login.html" class="nav-
link">LogIn</a>

        </li>

        <li class="nav-item">

            <a href="index.html" class="nav-link">Home</a>

        </li>

        <li class="nav-item">

            <a href="order.html" class="nav-link">Order</a>

        </li>

        <li class="nav-item">

            <a href="contact.html" class="nav-link">contact</a>

        </li>

        <li class="nav-item">

            <a href="feedbackk.html" class="nav-link">feedback</a>

        </li>

        <li class="nav-item">

            <a href="YourTaste.html" class="nav-link">Your Taste</a>

        </li>

    </ul>
</nav>
```

```
</li>

</ul>

<div class="hamburger">

    <span class="bar"></span>

    <span class="bar"></span>

    <span class="bar"></span>

</div>

</nav>

</header>

<!-- Hero Banner Section --&gt;

&lt;section class="hero-banner"&gt;

    &lt;div class="hero-content"&gt;

        &lt;h1&gt;Craving Something Tasty? &lt;br&gt;Get Delicious Meals Delivered with Ease!&lt;/h1&gt;

        &lt;p&gt;Satisfy Your Hunger with Just a Few Clicks&lt;/p&gt;

        &lt;div class="ord"&gt;&lt;a href="order.html" class="cta-button"&gt;Order now&lt;/a&gt;&lt;/div&gt;

    &lt;/div&gt;

&lt;/section&gt;

&lt;footer&gt;</pre>
```

```
<div class="footer-container">

    <div class="footer-left">

        <h3>FOODHUB</h3>

        <p>&copy; 2023 FOODHUB. All rights reserved.</p>

    </div>

    <div class="footer-center">

        <h3>Quick Links</h3>

        <ul>

            <li><a href="index.html">Home</a></li>

            <li><a href="order.html">Order</a></li>

            <li><a href="contact.html">Contact Us</a></li>

            <li><a href="feedbackk.html">Feedback</a></li>

        </ul>

    </div>

    <div class="footer-right">

        <h3>Follow Us</h3>

        <ul>

            <li><a href="#" target="_blank"><i class="fa fa-facebook"></i></a></li>

            <li><a href="#" target="_blank"><i class="fa fa-twitter"></i></a></li>

            <li><a href="#" target="_blank"><i class="fa fa-instagram"></i></a></li>

        </ul>

    </div>

</div>
```

```
</ul>

</div>

</div>

</footer>

<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"
integrity="sha384-YvpcrYf0tY3IHB60NNkmXc5s9fDVZLESaAA55NDzOxhy9GkcldsIK1eN7N6jleHz
" crossorigin="anonymous"></script>

</body>

</html>
```

Login.html

Code:

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>FOODHUB</title>

    <link rel="stylesheet" href="index.css" />

</head>

<body>

    <div class="form-container" id="loginFormContainer">

        <form id="loginForm">

            <h2>Login</h2>

            <label for="role">Role:</label>

            <select id="role" name="role" required>

                <option value="users">User</option>

                <option value="admin">Admin</option>

            </select>

            <label for="username">Username:</label>
```

```
<input type="text" id="username" name="username" placeholder="Enter  
your username" required />  
  
<label for="password">Password:</label>  
  
<input type="password" id="password" name="password"  
placeholder="Enter your password" required />  
  
<button type="submit">Login</button>  
  
<p>  
    Forgot your password?  
  
    <a href="forgot_password.html">Reset here</a>  
  
</p>  
  
<p>  
    Don't have an account?  
  
    <a href="register.html">Register now</a>  
  
</p>  
  
</form>  
  
</div>  
  
<script>  
  
document  
  
.getElementById("loginForm")  
  
.addEventListener("submit", function(e) {  
  
    e.preventDefault();  
  
    // Pre-populate the username and password fields
```

```
const username = document.getElementById("username").value;

const password = document.getElementById("password").value;

const role = document.getElementById("role").value;

const loginData = {

    username: document.getElementById("username").value,
    password: document.getElementById("password").value,
    role: document.getElementById("role").value,
};

console.log("Login Data:", loginData);

if (!loginData.username || !loginData.password) {

    alert("All fields are required");

    return;
}

fetch("http://localhost:3000/login", {

    method: "POST",
    headers: {
        "Content-Type": "application/json"
    },
    body: JSON.stringify(loginData),
})

.then((response) => response.json())
```

```
.then((data) => {

    if (data.success) {

        alert(data.message);

        if (role === "admin") {

            // Redirect to the admin dashboard page

            window.location.href = "../Admin/admin_dashboard.html"; // Adjust the path as needed

        } else if (role === "users") {

            localStorage.setItem("userId", data.testerId);

            // Redirect to the tester page (or the main page for testers)

            window.location.href = "../Main-web/index.html"; // Adjust the path as needed

        }

    } else {

        alert(data.message);

    }

})

.catch((error) => console.error("Error:", error));

});

</script>

</body>

</html>
```

register.html

Code:

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Register</title>

    <link rel="stylesheet" href="index.css">

</head>

<body>

    <div class="form-container" id="registerFormContainer">

        <form id="registerForm">

            <h2>Register</h2>

            <label for="username">Username:</label>

            <input type="text" id="username" name="username" placeholder="Enter your username" required>

            <label for="email">Email:</label>

            <input type="email" id="email" name="email" placeholder="Enter your email" required>

        </form>
    </div>
</body>
```

```
<label for="password">Password:</label>

<input type="password" id="password" name="password"
placeholder="Enter your password" required>

<label for="confirmPassword">Confirm Password:</label>

<input type="password" id="confirmPassword" name="confirmPassword"
placeholder="Confirm your password" required>

<button type="submit">Register</button>

<p>Already have an account? <a href="login.html">Login here</a></p>

</form>

</div>

<script>

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

    document.getElementById('registerForm').addEventListener('submit',
function(e) {

    e.preventDefault();

    const formData = {

        email: document.getElementById('email').value,
        username: document.getElementById('username').value,
```



```
        headers: {

            'Content-Type': 'application/json'

        },


        body: JSON.stringify(formData)

    })

    .then(response => response.json())

    .then(data => {

        console.log('Data received:', data);

        alert(data.message);

        if (data.success) {

            window.location.href = 'login.html';

        }

    })

    .catch(error => {

        console.error('Error:', error);

    });

});

});

</script>

</body>

</html>
```

order.html

Code:

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <link rel="stylesheet" href="order.css">

    <title>Document</title>

</head>

<body>

    <header class="header">

        <nav class="navbar">

            <a href=".index.html" class="nav-logo">Food<span
style="color:#854d27">Hub</span></a>

            <ul class="nav-menu">

                <li class="nav-item">

                    <a href=".Login and Register/login.html" class="nav-
link">LogIn</a>

                </li>
            
```

```
<li class="nav-item">  
    <a href="index.html" class="nav-link">Home</a>  
</li>  
  
<li class="nav-item">  
    <a href="order.html" class="nav-link">Order</a>  
</li>  
  
<li class="nav-item">  
    <a href="contact.html" class="nav-link">contact</a>  
</li>  
  
<li class="nav-item">  
    <a href="feedbackk.html" class="nav-link">feedback</a>  
</li>  
  
<li class="nav-item"><a href="YourTaste.html" class="nav-link">Your  
Taste</a></li>  
  
</ul>  
  
<div class="hamburger">  
    <span class="bar"></span>  
    <span class="bar"></span>  
    <span class="bar"></span>  
</div>  
</nav>
```

```
</header>

<section class="menu">

    <div class="food-item">

        <h2>Pizza</h2>

        <p><span class="price">₹399/-</span></p>

        <p class="description">Delicious cheesy pizza with fresh toppings.</p>

        <button class="order-btn" onclick="orderNow('Pizza')">Order Now</button>

    </div>

    <div class="food-item">

        <h2>Burger</h2>

        <p><span class="price">₹199/-</span></p>

        <p class="description">Juicy beef patty with crispy lettuce and cheese.</p>

        <button class="order-btn" onclick="orderNow('Burger')">Order Now</button>

    </div>

    <div class="food-item">
```

```


<h2>Vadapav</h2>

<p><span class="price">₹30/-</span></p>

<p class="description">Spicy potato fritter inside a soft bun.</p>

<button class="order-btn" onclick="orderNow('Vadapav')">Order Now</button>

</div>

<div class="food-item">



<h2>Fries</h2>

<p><span class="price">₹149/-</span></p>

<p class="description">Crispy golden fries served with ketchup.</p>

<button class="order-btn" onclick="orderNow('Fries')">Order Now</button>

</div>

<div class="food-item">



<h2>Salad</h2>

<p><span class="price">₹229/-</span></p>

<p class="description">Healthy mix of fresh vegetables and dressing.</p>

<button class="order-btn" onclick="orderNow('Salad')">Order Now</button>
```

```
</div>

<div class="food-item">

    <h2>Taco</h2>

    <p><span class="price">₱175/-</span></p>

    <p class="description">Crispy taco shell filled with seasoned meat.</p>

    <button class="order-btn" onclick="orderNow('Taco')">Order Now</button>

</div>

<div class="food-item">

    <h2>Sandwich</h2>

    <p><span class="price">₱79/-</span></p>

    <p class="description">Freshly made sandwich with layers of flavor.</p>

    <button class="order-btn" onclick="orderNow('Sandwich')">Order Now</button>

</div>

<div class="food-item">

    <h2>Steak</h2>

    <p><span class="price">₱599/-</span></p>

    <p class="description">Tender, juicy steak grilled to perfection.</p>
```

```
<button class="order-btn" onclick="orderNow('Steak')">Order  
Now</button>  
  
</div>  
  
<div class="food-item">  
  
      
  
    <h2>Ramen</h2>  
  
    <p><span class="price">₹299/-</span></p>  
  
    <p class="description">Hot and flavorful Japanese noodle soup.</p>  
  
    <button class="order-btn" onclick="orderNow('Ramen')">Order  
Now</button>  
  
</div>  
  
<div class="food-item">  
  
      
  
    <h2>Donut</h2>  
  
    <p><span class="price">₹49/-</span></p>  
  
    <p class="description">Sweet and fluffy donut with chocolate glaze.</p>  
  
    <button class="order-btn" onclick="orderNow('Donut')">Order  
Now</button>  
  
</div>  
  
</section>  
  
<footer>  
  
<p>© 2025 FOODHUB. All rights reserved.</p>
```

```
</footer>

<script>

    // Get all the "Order Now" buttons

    const orderNowButtons = document.querySelectorAll('.order-btn');

    // Add event listener to each "Order Now" button

    orderNowButtons.forEach(button => {

        button.addEventListener('click', () => {

            // Redirect the user to the ordDetails page

            window.location.href = 'ordDetails.html';

        });

    });

</script>

</body>

</html>
```

orderDetails.html

```
</html>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Order Details and Payment</title>

    <link rel="stylesheet" href="ordDetails.css">

</head>

<body>

    <div class="container">

        <h1>Order Details and Payment</h1>

        <form id="orderForm">

            <!-- Customer Information -->

            <section>

                <h2>Customer Information</h2>

                <label for="name">Full Name:</label>

                <input type="text" id="name" name="name" required><br><br>
```

```
<label for="phone">Phone Number:</label>

<input type="text" id="phone" name="phone" required><br><br>

<label for="email">Email Address:</label>

<input type="email" id="email" name="email" required><br><br>

</section>

<!-- Delivery Information -->

<section>

    <h2>Delivery Information</h2>

    <label for="address">Street Address:</label>

    <input type="text" id="address" name="address" required><br><br>

    <label for="city">City:</label>

    <input type="text" id="city" name="city" required><br><br>

    <label for="postalCode">Postal Code:</label>

    <input type="text" id="postalCode" name="postalCode"
required><br><br>

    <label for="country">Country:</label>

    <input type="text" id="country" name="country" required><br><br>
```

```
<label for="selectfood">Select food items</label>

<select id="selectfood" name="selectfood">

    <option value="pizza">Pizza</option>

    <option value="burger">Burger</option>

    <option value="vadapav">Vadapav</option>

    <option value="fries">Fries</option>

    <option value="salad">Salad</option>

    <option value="taco">Taco</option>

    <option value="sandwich">Sandwich</option>

    <option value="steak">Steak</option>

    <option value="ramen">Ramen</option>

    <option value="donut">Donut</option>

</select>

<div id="quantity-container">

    <label for="quantity">Quantity:</label>

    <input type="number" id="quantity" name="quantity" value="1"
min="1" max="10">

</div>

<br>

<label for="text">Amount</label>

<input type="text" id="amount" name="amount" readonly>
```

```
<label for="orderDate">Select Order Date:</label>

<input type="date" id="orderDate" name="orderDate"
required><br><br>

</section>

<!-- Order Details -->

<section>

    <h2>Order Details</h2>

    <label for="specialInstructions">Special Instructions:</label><br>

        <textarea id="specialInstructions" name="specialInstructions"
style="height: 100px; resize: none;"></textarea><br><br>

</section>

<!-- Payment Method -->

<section>

    <h2>Payment Method</h2>

    <label for="paymentMethod">Choose Payment Method:</label><br>

        <input type="radio" id="creditCard" name="paymentMethod"
value="Credit/Debit Card" required> Credit/Debit Card

        <div id="creditCardDetails" style="display: none;">

            <label for="cardNumber">Credit/Debit Card Number:</label>

            <input type="text" id="cardNumber" name="cardNumber"
placeholder="Enter your card number" required>

            <label for="cardExpiry">Card Expiry:</label>

        </div>
    </section>
</body>
</html>
```

```
<input type="text" id="cardExpiry" name="cardExpiry"
placeholder="MM/YY" required>

    <label for="cardCvv">Card CVV:</label>

    <input type="text" id="cardCvv" name="cardCvv" placeholder="Enter
your card CVV" required>

</div>

<input type="radio" id="cash" name="paymentMethod" value="Cash on
Delivery" required> Cash on Delivery

<input type="radio" id="upi" name="paymentMethod" value="UPI"
required> UPI

<div id="upiDetails" style="display: none;">

    <label for="upiMethod">Select UPI Method:</label><br>

    <div style="display: flex; flex-direction: row;">

        <input type="radio" id="googlePay" name="upiMethod"
value="Google Pay" required> Google Pay

        <input type="radio" id="phonePay" name="upiMethod"
value="Phone Pay" required> Phone Pay

        <input type="radio" id="upild" name="upiMethod" value="UPI ID"
required> UPI ID

    </div>

    <div id="upildInput" style="display: none;">

        <label for="upildValue">Enter UPI ID:</label>

        <input type="text" id="upildValue" name="upildValue"
placeholder="Enter your UPI ID" required>

    </div>

```

```
</div>

</div>

</section>

<button type="submit">Place Order</button>

</form>

</div>

<script>

// Get the credit card radio button

const creditCardRadio = document.getElementById('creditCard');

// Add event listener to the credit card radio button

creditCardRadio.addEventListener('change', () => {

    const creditCardDetails = document.getElementById('creditCardDetails');

    if (creditCardRadio.checked) {

        creditCardDetails.style.display = 'block';

    } else {

        creditCardDetails.style.display = 'none';

        document.getElementById('cardNumber').value = "";

        document.getElementById('cardExpiry').value = "";

        document.getElementById('cardCvv').value = "";

    }

})
```

```
        }

    });

    // Get the UPI radio button

    const upiRadio = document.getElementById('upi');

    // Add event listener to the UPI radio button

    upiRadio.addEventListener('change', () => {

        const upiDetails = document.getElementById('upiDetails');

        if (upiRadio.checked) {

            upiDetails.style.display = 'block';

        } else {

            upiDetails.style.display = 'none';

        }

    });

    // Get the UPI ID radio button

    const upildRadio = document.getElementById('upild');

    // Add event listener to the UPI ID radio button

    upildRadio.addEventListener('change', () => {
```

```
const upildInput = document.getElementById('upildInput');

if (upildRadio.checked) {

    upildInput.style.display = 'block';

} else {

    upildInput.style.display = 'none';

}

});

// Get the order form

const orderForm = document.getElementById('orderForm');

// Add event listener to the form submission

orderForm.addEventListener('submit', (e) => {

    e.preventDefault();

    // Get the selected food item, quantity, and order date

    const selectedFood = document.getElementById('selectfood').value;

    const quantity = document.getElementById('quantity').value;

    const orderDate = document.getElementById('orderDate').value;

    const name = document.getElementById('name').value;

    const phone = document.getElementById('phone').value;
```

```
const email = document.getElementById('email').value;

const address = document.getElementById('address').value;

const city = document.getElementById('city').value;

const postalCode = document.getElementById('postalCode').value;

const country = document.getElementById('country').value;

const specialInstructions =
document.getElementById('specialInstructions').value;

const paymentMethod =
document.querySelector('input[name="paymentMethod"]:checked').value;

// Create a new object to store the order details

const orderDetails = {

    name: name,

    phone: phone,

    email: email,

    address: address,

    city: city,

    postalCode: postalCode,

    country: country,

    foodItem: selectedFood,

    quantity: quantity,

    amount: document.getElementById('amount').value,
```

```
    orderDate: orderDate,  
  
    specialInstructions: specialInstructions,  
  
    paymentMethod: paymentMethod  
  
};  
  
  
  
// Send the order details to the server  
  
fetch('http://localhost:3000/place-order', {  
  
  method: 'POST',  
  
  headers: {  
  
    'Content-Type': 'application/json'  
  
  },  
  
  body: JSON.stringify(orderDetails)  
  
})  
  
.then(response => response.json())  
  
.then(data => {  
  
  console.log('Order placed successfully:', data);  
  
  alert('Order placed successfully!');  
  
  orderForm.reset();  
  
})  
  
.catch(error => {  
  
  console.error('Error placing order:', error);
```

```
        alert('Error placing order. Please try again later.');

    });

});

// Define the prices for each food item

const prices = {

    pizza: 399,

    burger: 199,

    vadapav: 30,

    fries: 149,

    salad: 229,

    taco: 175,

    sandwich: 79,

    steak: 599,

    ramen: 299,

    donut: 49

};

// Get the select food item, quantity, and amount fields

const selectFood = document.getElementById('selectfood');

const quantity = document.getElementById('quantity');

const amount = document.getElementById('amount');
```

```
// Add an event listener to the select food item field

selectFood.addEventListener('change', calculateAmount);

// Add an event listener to the quantity field

quantity.addEventListener('input', calculateAmount);

// Function to calculate the amount

function calculateAmount() {

    const selectedFood = selectFood.value;

    const quantityValue = parseInt(quantity.value);

    const price = prices[selectedFood];

    const calculatedAmount = price * quantityValue;

    amount.value = `₹${calculatedAmount}`;

}

// Call the calculateAmount function initially

calculateAmount();

// Function to toggle credit card fields

function toggleCreditCardFields() {
```

```
const creditCardDetails = document.getElementById('creditCardDetails');

if (document.getElementById('creditCard').checked) {

    creditCardDetails.style.display = 'block';

    document.getElementById('cardNumber').setAttribute('required', 'true');

    document.getElementById('cardExpiry').setAttribute('required', 'true');

    document.getElementById('cardCvv').setAttribute('required', 'true');

} else {

    creditCardDetails.style.display = 'none';

    document.getElementById('cardNumber').removeAttribute('required');

    document.getElementById('cardExpiry').removeAttribute('required');

    document.getElementById('cardCvv').removeAttribute('required');

}

}

// Function to toggle UPI fields

function toggleUPIFields() {

    const upiDetails = document.getElementById('upiDetails');

    if (document.getElementById('upi').checked) {

        upiDetails.style.display = 'block';

    } else {

        upiDetails.style.display = 'none';

    }

}
```

```
        document.querySelectorAll('input[name="upiMethod"]').forEach((input)
=> input.removeAttribute('required'));

        document.getElementById('upildValue').removeAttribute('required');

    }

}

// Function to toggle UPI ID input field

function toggleUPIIdField() {

    const upildInput = document.getElementById('upildInput');

    if (document.getElementById('upild').checked) {

        upildInput.style.display = 'block';

        document.getElementById('upildValue').setAttribute('required', 'true');

    } else {

        upildInput.style.display = 'none';

        document.getElementById('upildValue').removeAttribute('required');

    }

}

// Add event listeners

document.getElementById('creditCard').addEventListener('change',
toggleCreditCardFields);
```

```
document.getElementById('upi').addEventListener('change',
toggleUPIFields);

document.getElementById('upiid').addEventListener('change',
toggleUPIIdField);

// Run functions on page load to ensure proper state

toggleCreditCardFields();

toggleUPIFields();

toggleUPIIdField();

// Modify form submission to prevent errors

document.getElementById('orderForm').addEventListener('submit',
function(e) {

    toggleCreditCardFields();

    toggleUPIFields();

    toggleUPIIdField();

});

document.addEventListener("DOMContentLoaded", () => {

    const orderForm = document.getElementById("orderForm");

    function generateReceipt(orderDetails) {

        const {

            jsPDF
```

```
    } = window.jspdf;

    const doc = new jsPDF();

    doc.setFont("helvetica", "bold");

    doc.setFontSize(18);

    doc.text("FoodHub - Order Receipt", 20, 20);

    doc.setFontSize(12);

    doc.setFont("helvetica", "normal");

    doc.text(`Customer Name: ${orderDetails.name}` , 20, 40);

    doc.text(`Phone: ${orderDetails.phone}` , 20, 50);

    doc.text(`Email: ${orderDetails.email}` , 20, 60);

    doc.text(`Address: ${orderDetails.address}, ${orderDetails.city},
${orderDetails.country}` , 20, 70);

    doc.text(`Postal Code: ${orderDetails.postalCode}` , 20, 80);

    doc.text(`Food Item: ${orderDetails.foodItem}` , 20, 100);

    doc.text(`Quantity: ${orderDetails.quantity}` , 20, 110);

    doc.text(`Amount: ${orderDetails.amount}` , 20, 120);

    doc.text(`Order Date: ${orderDetails.orderDate}` , 20, 130);
```

```
doc.text(` Payment Method: ${orderDetails.paymentMethod}` , 20, 150);

if (orderDetails.specialInstructions) {

    doc.text(` Special Instructions: ${orderDetails.specialInstructions}` ,
20, 170);

}

doc.text("Thank you for your order!", 20, 190);

doc.save(` Order_Receipt_${Date.now()}.pdf`);

}

orderForm.addEventListener("submit", (e) => {

    e.preventDefault();

    // Get order details

    const selectedFood = document.getElementById("selectfood").value;

    const quantity = document.getElementById("quantity").value;

    const orderDate = document.getElementById("orderDate").value;

    const name = document.getElementById("name").value;

    const phone = document.getElementById("phone").value;

    const email = document.getElementById("email").value;
```

```
const address = document.getElementById("address").value;

const city = document.getElementById("city").value;

const postalCode = document.getElementById("postalCode").value;

const country = document.getElementById("country").value;

const specialInstructions =
document.getElementById("specialInstructions").value;

const paymentMethod =
document.querySelector('input[name="paymentMethod"]:checked').value;

const orderDetails = {

    name,

    phone,

    email,

    address,

    city,

    postalCode,

    country,

    foodItem: selectedFood,

    quantity,

    amount: document.getElementById("amount").value,

    orderDate,

    specialInstructions,
```

```
    paymentMethod

};

// Send data to the server (optional)

fetch("http://localhost:3000/place-order", {

  method: "POST",

  headers: {

    "Content-Type": "application/json",

  },

  body: JSON.stringify(orderDetails),

})

.then((response) => response.json())

.then((data) => {

  console.log("Order placed successfully:", data);

  alert("Order placed successfully!");

  orderForm.reset();

  generateReceipt(orderDetails); // Generate and download receipt

})

.catch((error) => {

  console.error("Error placing order:", error);

  alert("Error placing order. Please try again later.");
})
```

```
    });

    });

});

</script>

<script
src="https://cdnjs.cloudflare.com/ajax/libs/jspdf/2.5.1/jspdf.umd.min.js"></script>

</body>

</html>
```

contact.html

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Contact Form</title>

    <link rel="stylesheet" href="styleee.css" />

    <link rel="stylesheet" href="contact.css" />

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">

    <link
        href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css">

</head>

<body>

    <header class="header">

        <nav class="navbar">

            <a href=".index.html" class="nav-logo">Food<span
                style="color:#854d27">Hub</span></a>

            <ul class="nav-menu">

                <li class="nav-item">
```

```
<a href="../Login and Register/login.html" class="nav-link">LogIn</a>

</li>

<li class="nav-item">
    <a href="index.html" class="nav-link">Home</a>
</li>

<li class="nav-item">
    <a href="order.html" class="nav-link">Order</a>
</li>

<li class="nav-item">
    <a href="contact.html" class="nav-link">Contact</a>
</li>

<li class="nav-item">
    <a href="feedbackk.html" class="nav-link">Feedback</a>
</li>

<li class="nav-item"><a href="YourTaste.html" class="nav-link">Your Taste</a></li>

</ul>

<div class="hamburger">
    <span class="bar"></span>
    <span class="bar"></span>
    <span class="bar"></span>
```

```
</div>

</nav>

</header>

<main>

    <div class="contact-container">

        <h1>Contact Us</h1>

        <div id="responseMessage"></div>

        <!-- Display API response message -->

        <form id="contactForm">

            <input name="name" type="text" id="name" class="feedback-input"
placeholder="Name" required />

            <input name="email" type="email" id="email" class="feedback-input"
placeholder="Email" required />

            <textarea name="message" id="message" class="feedback-input"
placeholder="Message" required></textarea>

            <input type="submit" value="SUBMIT" />

        </form>

    </div>

</main>

<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"
integrity="sha384-"
```

```
YvpcrYf0tY3IHB60NNkmXc5s9fDVZLESaAA55NDzOxhy9GkcldsIK1eN7N6jleHz
" crossorigin="anonymous"></script>

<script>

    document.addEventListener("DOMContentLoaded", function() {

        const contactForm = document.getElementById("contactForm");

        const responseMessage =
document.getElementById("responseMessage");

        contactForm.addEventListener("submit", function(event) {

            event.preventDefault();

            const name = document.getElementById('name').value;
            const email = document.getElementById('email').value;
            const message = document.getElementById('message').value;

// const formData = new FormData(contactForm);

            const formData = {

                name: name,
                email: email,
                message: message,
            };
        });
    });
</script>
```

```
const apiEndpoint = "http://localhost:3000/api/contact";

// Fetch contact data from the API
fetch(apiEndpoint, {
  method: "POST",
  headers: {
    'Content-Type': 'application/json; charset=utf-8'
  },
  body: JSON.stringify(formData)
})

.then(response => response.json())

.then(data => {
  // Update the message element with the new data
  responseMessage.innerHTML = `<p>${data.message}</p>`;
  responseMessage.classList.add("success", "show"); // Add 'show' class to make the message visible

  // Set a timeout to reset the form after 2 seconds
  setTimeout(() => {
    contactForm.reset();
    responseMessage.innerHTML = "";
    responseMessage.classList.remove("success", "show");
  }, 2000);
})
```

```
.catch(error => {
    console.error("Error:", error);
    responseMessage.innerHTML = `<p>Error submitting the form.
Please try again later.</p>`;
    responseMessage.classList.add("error", "show"); // Add 'show'
    class to make the message visible

    setTimeout(() => {
        responseMessage.innerHTML = "";
        responseMessage.classList.remove("error", "show");
        }, 3000);
    });
});
});

</script>
</body>
</html>
```

feedbackk.html

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Feedback Form</title>

    <link rel="stylesheet" href="feedbackk.css">

    <link rel="stylesheet" href="styleeee.css">

    <link
        href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;600&display=swap" rel="stylesheet">

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-beta3/css/all.min.css">

</head>

<body>

    <header class="header">

        <nav class="navbar">

            <a href=".//index.html" class="nav-logo">Food<span
                style="color:#854d27">Hub</span></a>

            <ul class="nav-menu">

                <li class="nav-item">
```

```
<a href="../Login and Register/login.html" class="nav-link">LogIn</a>

</li>

<li class="nav-item">
    <a href="index.html" class="nav-link">Home</a>
</li>

<li class="nav-item">
    <a href="order.html" class="nav-link">Order</a>
</li>

<li class="nav-item">
    <a href="contact.html" class="nav-link">contact</a>
</li>

<li class="nav-item">
    <a href="feedbackk.html" class="nav-link">feedback</a>
</li>

<li class="nav-item"><a href="YourTaste.html" class="nav-link">Your Taste</a></li>

</ul>

<div class="hamburger">
    <span class="bar"></span>
    <span class="bar"></span>
    <span class="bar"></span>
```

```
</div>

</nav>

</header>

<!-- fd -->

<div class="feedback-container">

    <h1>Share your Feedback</h1>

    <p>We'd love to hear your thoughts and improve based on your suggestions!.</p>

    <form action="process_feedback.php" method="POST" class="feedback-form">

        <div class="form-group">

            <label for="name">Name</label>

            <input type="text" name="name" id="name" placeholder="Enter Your Name" required>

        </div>

        <div class="form-group">

            <label for="email">Email</label>

            <input type="email" name="email" id="email" placeholder="Enter Your Email" required>

        </div>

        <div class="form-group">

            <label for="rating"><i>Rating</i></label>
```

```
<select name="rating" id="rating" required>

<option value="" disabled selected>Select a rating</option>

<option value="Excellent">Excellent</option>

<option value="Good">Good</option>

<option value="Average">Average</option>

<option value="Poor">Poor</option>

<option value="Terrible">Terrible</option>

</select>

</div>

<div class="form-group">

    <label for="comments">Comments</label>

    <textarea name="comments" id="comments" placeholder="Write your
message...." rows="5" required></textarea>

</div>

<button type="submit" class="btn">Submit Feedback</button>

</form>

</div>

<script>

// Get the feedback form
```

```
const feedbackForm = document.querySelector(".feedback-form");

// Add an event listener for form submission

feedbackForm.addEventListener("submit", (e) => {

  e.preventDefault();

  // Collect form data

  const name = document.getElementById("name").value;
  const email = document.getElementById("email").value;
  const rating = document.getElementById("rating").value;
  const comments = document.getElementById("comments").value;

  // Prepare data to send

  const feedbackData = {
    name,
    email,
    rating,
    comments
  };

  // Send data to the server

  fetch("http://localhost:3000/api/submitFeedback", {
```

```
        method: "POST",

        headers: {

            "Content-Type": "application/json",

        },

        body: JSON.stringify(feedbackData),

    })

    .then((response) => response.json())

    .then((data) => {

        if (data.success) {

            alert(data.message);

            feedbackForm.reset(); // Clear form fields

        } else {

            alert("Error: " + data.message);

        }

    })

    .catch((error) => console.error("Error:", error));

});

</script>

</body>

</html>
```

YoueTaste.html

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Your Taste | FOODHUB</title>

    <link rel="stylesheet" href="yourtaste.css">

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.5.0/css/all.min.css">

</head>

<body>

    <!-- ↴ Navbar Start -->

    <header class="header">

        <nav class="navbar">

            <a href="#" class="nav-logo">Food<span style="color:#854d27">Hub</span></a>

            <ul class="nav-menu">
```

```
        <li class="nav-item"><a href="../Login and Register/login.html" class="nav-link">LogIn</a></li>

        <li class="nav-item"><a href="index.html" class="nav-link">Home</a></li>

        <li class="nav-item"><a href="order.html" class="nav-link">Order</a></li>

        <li class="nav-item"><a href="contact.html" class="nav-link">Contact</a></li>

        <li class="nav-item"><a href="feedbackk.html" class="nav-link">Feedback</a></li>

        <li class="nav-item"><a href="YourTaste.html" class="nav-link">Your Taste</a></li>

    </ul>

    <div class="hamburger">

        <span class="bar"></span>

        <span class="bar"></span>

        <span class="bar"></span>

    </div>

</nav>

</header>

<!-- ✅ Navbar End -->

<!-- ✅ Suggestion Form Start -->
```

```
<section class="taste-form-container">

    <h2>Tell Us About Your Favorite Dish!</h2>

    <p>We'd love to hear what dish you'd like to see on FoodHub.</p>

    <form id="tasteForm">

        <input type="text" id="name" placeholder="Your Name" required>

        <input type="email" id="email" placeholder="Your Email" required>

        <input type="text" id="dishName" placeholder="Dish Name" required>

        <textarea id="dishDescription" rows="4" placeholder="Why do you love it?" required></textarea>

        <button type="submit">Submit Suggestion</button>

    </form>

    <div id="success-message" style="display:none; margin-top: 15px; color: green;">

        ✓ Thank you! Your suggestion has been submitted. ⓘ

    </div>

</section>

<!-- ✓ Suggestion Form End -->

<!-- ✓ Script for Form Handling -->

<script>

    document.getElementById("tasteForm").addEventListener("submit", async
function(event) {
```

```
event.preventDefault();

const name = document.getElementById("name").value;

const email = document.getElementById("email").value;

const dishName = document.getElementById("dishName").value;

const dishDescription =
document.getElementById("dishDescription").value;

try {

  const res = await fetch("http://localhost:3000/suggest-dish", {

    method: "POST",

    headers: {

      "Content-Type": "application/json",

    },

    body: JSON.stringify({

      name,

      email,

      dishName,

      dishDescription,

    }),

  });

  if (res.ok) {

    document.getElementById("tasteForm").reset();
```

```
        const messageDiv = document.getElementById("success-
message");

        messageDiv.classList.add("show"); // Add class to fade in

        messageDiv.style.display = "block";

        // Hide the message after 3 seconds

        setTimeout(() => {

            messageDiv.classList.remove("show"); // Remove fade-in class

            messageDiv.style.display = "none";

        }, 3000);

    } else {

        alert("☒ Failed to submit your suggestion. Please try again.");

    }

} catch (error) {

    console.error("Error:", error);

    alert("⚠ An error occurred. Please check your internet or server.");

}

});

</script>

</body>

</html>
```

OUTPUTS

Reports:

order

The screenshot shows a Windows desktop environment with a browser window open to an 'Admin Dashboard - Order Man' page at 'localhost / 127.0.0.1 / food_ord'. The dashboard has a dark blue sidebar on the left with the title 'Admin Dashboard' and a teal header bar. The main content area is titled 'Orders' and displays a table of 5 rows of order data. The table columns are: Order ID, Customer Name, Phone, Email, Address, Food Item, Quantity, Amount, Order Date, Payment Method, and Special Instructions.

Order ID	Customer Name	Phone	Email	Address	Food Item	Quantity	Amount	Order Date	Payment Method	Special Instructions
19	prem shah	9975241812	premshah47@gmail.com	shiroli, kolhapur, 416122, india	steak	1	₹599	2025-03-23T18:30:00.000Z	Cash on Delivery	grilled
17	srish kore	8866523412	shrikore44@gmail.com	kagal, kolhapur, 416202, india	salad	2	₹458	2025-03-22T18:30:00.000Z	Cash on Delivery	extra tomatoes slices
11	joel john	7875679900	joeljohn0@gmail.com	kaneriwadi, kolhapur, 416234, india	fries	4	₹596	2025-03-19T18:30:00.000Z	Cash on Delivery	extra spicy
15	shree patil	7775003756	shripatil9082@gmail.com	shiroli, kolhapur, 416122, india	vadapav	2	₹60	2025-03-13T18:30:00.000Z	Cash on Delivery	extra souce
13	siddhant kamble	9049540141	siddhantk12@gmail.com	kalamba, kolhapur, 416007, india	ramen	3	₹897	2025-03-12T18:30:00.000Z	UPI	make it less spicy

Contact

The screenshot shows a web-based admin dashboard for an order management system. The main content area displays a table titled "Contact Submissions" listing six entries. The table has columns for ID, Name, Email, Message, and Submitted At. The left sidebar contains a navigation menu with links for Orders, Contact Submissions (which is currently selected), Feedback Submissions, Registered Users, YourTaste Submissions, and Logout.

ID	Name	Email	Message	Submitted At
9	Manish Kumar	manish.kuma55r@gmail.com	Can I schedule my order for later?	3/22/2025, 9:45:28 AM
8	Rohit Sharma	rohit.sharma11@gmail.com	Having issues while placing an order	3/22/2025, 9:44:43 AM
7	Simran Kaur	simran.kaur@gmail.com	The delivery was quick, thank you!	3/22/2025, 9:43:48 AM
6	Aarav Mehta	aarav.mehta@example.com	I would love to see more vegetarian options.	3/22/2025, 9:42:59 AM
5	joel john	joeljohn0@gmail.com	need help regarding delivery	3/22/2025, 9:39:50 AM
4	shree	shreepatil414@gmail.com	hi	3/3/2025, 5:27:04 PM

Feedback

The screenshot shows a web-based Admin Dashboard interface. On the left, there's a sidebar with a dark blue background and white text, containing links for Orders, Contact Submissions, Feedback Submissions (which is highlighted in green), Registered Users, YourTaste Submissions, and Logout. The main content area has a light gray background and features a table titled "Feedback Submissions". The table has a dark header row with columns for ID, Name, Email, Rating, Comments, and Submitted At. Below the header, there are six rows of data. The table is set against a background of a city skyline at night.

ID	Name	Email	Rating	Comments	Submitted At
14	Rajat Kapoor	rajatk256@gmail.com	Poor	The order was incorrect. Please improve.	3/22/2025, 9:49:07 AM
13	Meera Shah	meera.shah68@gmail.com	Average	Food was okay, but arrived a bit late.	3/22/2025, 9:48:26 AM
12	Kunal Joshi	kunaljoshi33@gmail.com	Good	Nice packaging and quick delivery.	3/22/2025, 9:47:32 AM
11	Ananya Verma	ananya.verma77@gmail.com	Excellent	The food was delicious and fresh!	3/22/2025, 9:46:52 AM
10	arun	tak@gmail.com	Excellent	Nice service	3/18/2025, 1:20:11 PM
9	shree	shri@gmail.com	Excellent	gh	3/3/2025, 5:48:16 PM

Registered Users

The screenshot shows a web-based admin dashboard for managing registered users. The left sidebar contains navigation links: Orders, Contact Submissions, Feedback Submissions, **Registered Users** (which is the active tab), and YourTaste Submissions. Below these are Logout and Help links. The main content area is titled "Registered Users" and displays a table with five rows of user data. The columns are User ID, Username, Email, and Registered At.

User ID	Username	Email	Registered At
6	shree414	shripatil9082@gmail.com	3/22/2025, 9:55:47 AM
7	joejohn	joejohn0@gmail.com	3/22/2025, 9:56:16 AM
8	siddhant kamble	siddhant@gmail.com	3/22/2025, 9:56:38 AM
9	harshda3322	harshda44@gmail.com	3/22/2025, 9:57:17 AM
10	premshah	premshah47@gmail.com	3/22/2025, 9:57:41 AM

Your Taste

The screenshot shows a web-based Admin Dashboard interface. On the left, a sidebar menu lists several options: Orders, Contact Submissions, Feedback Submissions, Registered Users, and YourTaste Submissions, which is currently selected and highlighted in green. Below the menu is a Logout button. The main content area is titled "Your Taste Submissions" and displays a table of data. The table has columns for ID, Name, Email, Dish Name, Description, and Submitted At. Six rows of data are listed, corresponding to the submissions shown in the screenshot.

ID	Name	Email	Dish Name	Description	Submitted At
25	Vikram Rao	vikram.rao@gmail.com	Masala Ramen Bowl	Ramen noodles cooked in spicy Indian masala broth with veggies and egg.	3/22/2025, 9:54:06 AM
24	Aman Sharma	aman.sharma@gmail.com	Butter Chicken Pizza	Classic pizza with butter chicken topping and a creamy Indian twist.	3/22/2025, 9:52:50 AM
23	Sanya Gupta	sanya.gupt@gmail.com	Mango Sushi Rolls	Refreshing sushi rolls made with mango, avocado, and sticky rice	3/22/2025, 9:51:54 AM
22	Rohit Mehra	rohit.mehra90@gmail.com	Spicy Paneer Tacos	A fusion of Indian spices and Mexican street tacos with paneer filling.	3/22/2025, 9:50:43 AM
21	siddhant	siddhant@gmail.com	momo	idk	3/22/2025, 9:15:28 AM
20	shree	tak@gmail.com	kachori	yess	3/21/2025, 8:40:06 PM

Conclusion

This project successfully fulfills its goal of providing an efficient and user-friendly online food ordering platform. By digitizing the food ordering process, it reduces manual work, improves customer convenience, and streamlines restaurant management. The platform allows users to register, log in, browse menu items, place orders, give feedback, and suggest dishes—all through a simple and responsive web interface.

From the backend perspective, the system is built using Node.js and MySQL, ensuring robust data handling and secure server-side processing. Admins are provided with a dedicated dashboard to monitor orders, manage customer interactions, and view feedback and suggestions. This organized system improves business operations and allows better decision-making based on customer input.

Overall, FoodHub bridges the gap between food providers and customers by offering a centralized, reliable, and scalable solution. It not only enhances the user experience but also equips the management with tools to handle everyday tasks efficiently. With further enhancements like payment integration and order tracking, the platform holds great potential for future growth.

SUGGESTION

- **Order Tracking Feature:** Adding real-time order tracking can boost transparency and trust. Users should be able to view the order status—whether it's being prepared, out for delivery, or delivered.
- **Admin Analytics Dashboard:** Enhance the admin panel with visual charts and analytics to monitor sales trends, popular dishes, user activity, and feedback stats. This will help admins make informed decisions.
- **Personalized Recommendations:** Use customer order history and feedback to suggest dishes tailored to their taste. Implementing this can increase engagement and repeat orders.
- **Email Notifications:** Send automated confirmation emails for orders, account creation, and feedback submissions. This improves communication and adds a professional touch to the platform.
- **User Profile & Order History:** Allow users to view and manage their profiles, including their previous orders, favorite dishes, and saved addresses. This not only enhances user experience but also encourages repeat orders.

Future Enhancements

- **Mobile Application Development:** Create Android and iOS apps for easier access and improved user convenience.
- **Real-Time Order Tracking:** Enable users to track their orders live from preparation to delivery.
- **AI-Based Recommendations:** Suggest dishes based on users' previous orders and preferences for a personalized experience.
- **Multilingual Support:** Add multiple language options to cater to users from diverse backgrounds.
- **Rewards and Loyalty Program:** Introduce incentives like discounts or points to retain and encourage repeat customers.

Bibliography

Websites:

- www.Google.com
- www.YouTube.com
- www.chatgpt.com
- www.blackbox.com

