**Mora Food – Online Food Ordering Web Application**

**DECLARATION**

I hereby declare that this project entitled **“Mora Food – Online Food Ordering Web Application”** has been carried out by me under the guidance of **Mr. \_\_\_\_\_\_\_\_\_\_**, Department of Computer Applications, **\_\_\_\_\_\_\_\_\_\_ College**.

I further declare that this project report is the result of my own effort and has not been copied from any source. It has not been submitted to any other university or institution for the award of any degree or diploma.

**Place:** \_\_\_\_\_\_\_\_\_\_

**Date:** \_\_\_\_\_\_\_\_\_\_

**ACKNOWLEDGEMENT**

I express my sincere gratitude to our Principal **Dr. \_\_\_\_\_\_\_\_\_\_** for providing the facilities and encouragement to complete this project successfully.

I am grateful to the Head of the Department, **Mrs. \_\_\_\_\_\_\_\_\_\_**, for her constant support and motivation throughout the course of this project.

I express my deep sense of gratitude and indebtedness to my project guide **Mr. \_\_\_\_\_\_\_\_\_\_** for his valuable guidance, continuous encouragement, and constructive suggestions, which enabled me to complete this project successfully.

I also thank my teachers, friends, and well-wishers for their support and cooperation during the completion of this project.

.

**CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S. NO.** | **DESCRIPTION** | **PAGENO.** |
|  | Abstract | 1 |
| 1 | System Analysis   * 1. 1.1 Existing System   2. 1.2 Proposed System   3. 1.3 Feasibility Study | 2  3  4 |
| 2 | System Requirements Specification   * 1. 2.1 Software Requirements   2. 2.2 Hardware Requirements   3. 2.3 About Software | 6  6  7 |
| 3 | System Design   * 1. 3.1 Modules Description   2. 3.2 Functional Requirements      1. 3.2.1 Data Flow Diagrams      2. 3.2.2 E-R Diagram      3. 3.2.3 UML Diagrams      4. 3.3 User Interface Design | 10  11  11  11  13  14  15 |
| 4 | System Implementation | 19 |
| 5 | System Testing | 20 |
| 6 | System Maintenance | 24 |
| 7 | Conclusion | 25 |
| 8 | Future Enhancement | 26 |
| 9 | Bibliography | 27 |
| 10 | Appendix  10.1 Sample Screens  10.2 Source Code | 28  33 |

# ABSTRACT

## This paper presents Mora Food, a comprehensive, responsive, and user-centric web based food ordering and food discovery application designed to bridge the gap between food lovers, restaurants, and home chefs. In today’s fast-paced digital era, consumers increasingly prefer platforms that are quick, reliable, visually appealing, and easy to use for discovering meals, comparing food options, and placing orders with minimal effort. Mora Food is developed to meet these expectations by offering a seamless and intuitive user experience.

## The application provides a clean and modern interface that allows users to browse restaurants, explore home-cooked meals, view detailed food menus, manage favourite items, access special menus and promotional offers, track order history, and communicate directly through an integrated messaging feature. By combining both restaurant-based and home-chef food services, Mora Food creates a unique ecosystem that supports local food providers while offering customers a wider variety of choices. Mora Food is developed using HTML5 for semantic structure, CSS3 for styling, layout design, and responsiveness, JavaScript for dynamic behaviour and interactivity, and Bootstrap for rapid UI development and an efficient grid system. These technologies ensure that the application delivers consistent performance and responsiveness across multiple devices, including desktops, tablets, and mobile phones. The design follows modern UI/UX principles, focusing on simplicity, accessibility, and smooth navigation.

## To enhance user experience and data persistence, the application utilizes client-side storage (localStorage) to securely store user-related data such as cart items, favourite foods, order history, and user preferences. This project demonstrates the practical implementation of front-end web technologies while adhering to standard software engineering principles, including system analysis, requirement gathering, system design, implementation, testing, and maintenance. Mora Food not only serves as an academic project that strengthens understanding of real-world web development practices but also acts as a scalable foundation for a full-fledged commercial food delivery platform with potential for future enhancements such as backend integration, real-time order tracking, payment gate ways, and user authentication.

## CHAPTER-I

## SYSTEM ANALYSIS

## 1.1 EXISTING SYSTEM

## In the existing food ordering scenario, customers often depend on physical visits, telephone calls, or limited food applications that focus mainly on restaurants. Home chefs and small food businesses receive minimal exposure and marketing support.

## DRAWBACKS

* Manual order placement through calls
* No centralized platform for home chefs
* Lack of order tracking
* Limited menu visibility
* Time-consuming process
* Poor customer feedback handling

These limitations lead to inconvenience for customers and reduced business opportunities for small food providers.

## 1.2 PROPOSED SYSTEM

The proposed Mora Food system provides a unified digital platform for food discovery and ordering. The system offers:

* Online browsing of restaurants and home chefs
* Category-based food exploration
* Special menus and offers
* Favorite food and chef management
* Order history tracking
* User messaging and feedback
* Fully responsive user interface

**ADVANTAGES OF THE PROPOSED SYSTEM:**

* Faster and easier ordering process
* Better exposure for home chefs
* Improved customer satisfaction
* Centralized food discovery platform

## 1.3 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. A feasibility study allows project managers to investigate the possible negative and positive outcomes of a project investing too much time and money.

Three key considerations involved in the feasibility analysis are

* Technical Feasibility
* Economic Feasibility
* Social Feasibility
* Legal Feasibility
* Scheduling Feasibility

## TECHNICAL FEASIBILITY

## The system is developed using well-established web technologies such as HTML5, CSS3, JavaScript, and Bootstrap. These technologies are widely supported and require no special hardware or proprietary software.

## ECONOMICAL FEASIBILITY

## The project is cost-effective as it relies on open-source tools and does not require licensed software.

**OPERATIONAL FEASIBILITY:**

The system is easy to use and does not require special training.

**LEGAL FEASIBILITY:**

The system does not violate any legal or data protection regulations.

## SCHEDULING FEASIBILITY

In this feasibility study, we estimate how much time the system will take to complete, and with our technical skills we need to estimate the period to complete the project using various methods of estimation.

The user-friendly interface ensures easy operation for both technical and non-technical users. No special training is required to use the system.

## CHAPTER-II

## SOFTWARE REQUIREMENT SPECIFICATION

## 2.1 SOFTWARE REQUIREMENTS

* + - Operating system : Windows 10/11
    - Front End : HTML, CSS JavaScript
    - Framework : Bootstrap
    - Browser : Google Chrome, Mozilla Firefox,

Microsoft Edge

* + - Code Editor : VS Code or any modern editor

**2.2 HARDWARE REQUIREMENTS**

* Processor : Intel i3 or Higher
* Clock Speed: 2.7GHZ, Max Turbo Frequency at

4.1GHZ

* RAM: 8 GB
* SSD Capacity: 512 GB
* System Type: 64-bit operating system X64 Based

Processor

**2.3 ABOUT SOFTWARE**

**HTML5 (Hypertext Markup Language – Version 5)**

HTML5 is the latest and most advanced version of HyperText Markup Language, which is used to create and structure web pages on the Internet. It forms the backbone of any web application by defining the structure and layout of content such as text, images, videos, forms, and links. HTML5 introduces semantic elements that clearly describe the meaning of the content, making web pages more readable and accessible to both users and search engines.

Semantic tags such as <header>, <nav>, <section>, <article>, <aside>, and <footer> help organize the web page into meaningful sections. This improves code readability, maintainability, and SEO (Search Engine Optimization). HTML5 also supports multimedia elements like <audio> and <video> without the need for external plugins, enabling richer user experiences.

In the Mora Food application, HTML5 is used to design the overall structure of the website, including the navigation bar, restaurant listings, food cards, order history sections, forms for user messages, and settings pages. HTML5 ensures that the application is lightweight, fast-loading, and compatible with all modern browsers.

Key features of HTML5 include:

* Semantic elements for better structure
* Built-in multimedia support
* Improved form controls and validations
* Cross-platform and cross-browser compatibility

Thus, HTML5 plays a crucial role in creating a well-structured, accessible, and user-friendly web application.

**CSS3 (Cascading Style Sheets – Version 3)**

CSS3 is a powerful styling language used to control the presentation and appearance of HTML elements on web pages. It is responsible for defining colors, fonts, layouts, spacing, animations, and overall visual aesthetics of a web application. CSS3 enhances user experience by making web pages visually appealing and responsive across different screen sizes.

One of the major advantages of CSS3 is its support for responsive web design. Using media queries, CSS3 allows the application layout to adapt dynamically to various devices such as desktops, tablets, and smartphones. CSS3 also introduces advanced features like transitions, animations, gradients, shadows, and flexible box layouts (Flexbox) and grid systems.

In Mora Food, CSS3 is used extensively to design attractive food cards, navigation menus, buttons, icons, banners, and animations such as hover effects and smooth transitions. The responsive layout ensures that users can comfortably browse menus and place orders on any device.

**Key features of CSS3 include:**

* Responsive design using media queries
* Animations and transitions
* Flexbox and Grid layout systems
* Improved font and colour control
* Enhanced user interface design

CSS3 significantly improves the usability, appearance, and responsiveness of the Mora Food application.

**JavaScript**

JavaScript is a widely used scripting language that enables dynamic behavior and interactivity in web applications. Unlike HTML and CSS, which are used for structure and styling, JavaScript handles logic, user interactions, and real-time content updates. It allows web pages to respond instantly to user actions without reloading the page.

JavaScript is essential for implementing features such as adding items to the cart, managing favorites, updating order history, validating user inputs, and displaying messages dynamically. It also enables client-side data storage using localStorage, which helps in saving user preferences, cart data, and order history even after refreshing or closing the browser.

**In the Mora Food application, JavaScript manages:**

* Dynamic loading of food items and restaurants
* Add-to-cart and favorite functionality
* Order history management
* User messages and feedback handling
* Interactive UI components

**Key features of JavaScript include:**

* Event-driven programming
* Real-time updates without page reloads
* Client-side data storage
* Improved performance and user experience

JavaScript makes the Mora Food application interactive, efficient, and user-friendly.

## CHAPTER-III

## SYSTEM DESIGN

**3.1 MODULES DESCRIPTION**

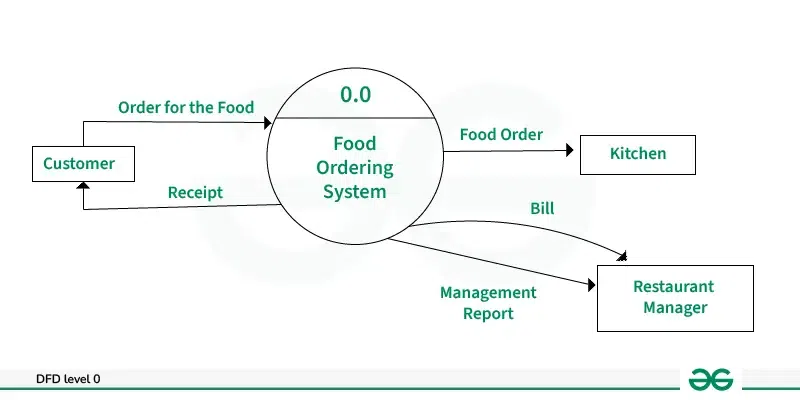
* Main Page – Displays categories, featured items, and navigation
* Restaurants Page – Shows restaurant listings with ratings and locations
* Home Chefs Page – Displays home chefs and their dishes
* Special Menu – Weekly and festival special items
* Favourite’s – Saved meals and chefs
* Offers – Discounts and promotional banners
* Order History – Previous orders stored using localStorage
* Messages – Thank-you messages and complaints
* About – Application information, mission, and vision
* Settings – User preferences and application settings

**3.2 FUNCTIONAL REQUIREMENTS**

**3.2.1 DATA FLOW DIAGRAM**

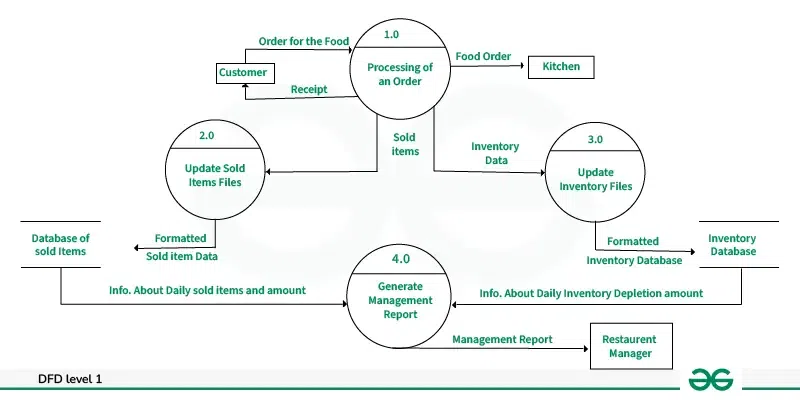
**Level 0 DFD:**

* **Food Delivery App:** This represents the main system, which acts as a central hub for all processes and interactions within the application.
* **User:** Users interact with the application to browse restaurants, place orders, track deliveries, and provide feedback.
* **Restaurant:** Restaurants interact with the application to manage menus, receive orders, update order statuses, and manage inventory.
* **Delivery:** Delivery drivers interact with the application to accept delivery assignments, deliver orders, and update delivery statuses.
* **Inventory:** This process manages inventory levels and the availability of menu items, ensuring that restaurants can fulfill orders accurately.



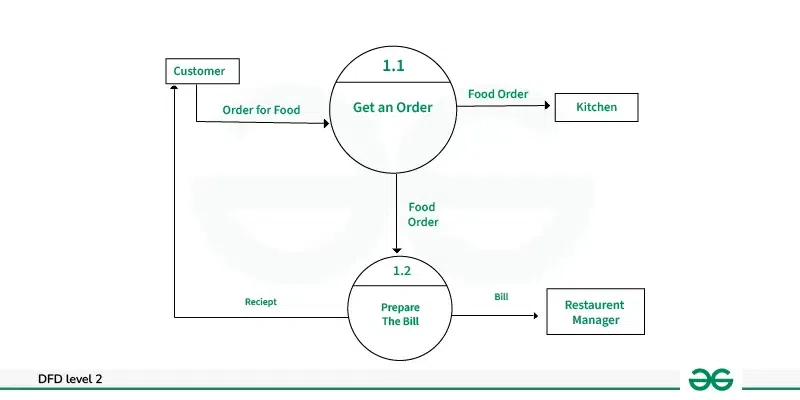
**Level 1 DFD (User Process):**

* **Browse Restaurants:** Users can browse through available restaurants and their menus to make informed choices about where to order from.
* **Order Placement:** Users select dishes, customize their orders, and place them through the application.
* **Track Order:** Users can track the status of their placed orders, including preparation, delivery, and estimated arrival time.
* **Provide Feedback:** Users can provide feedback and ratings for restaurants and delivery drivers based on their experience with the service.

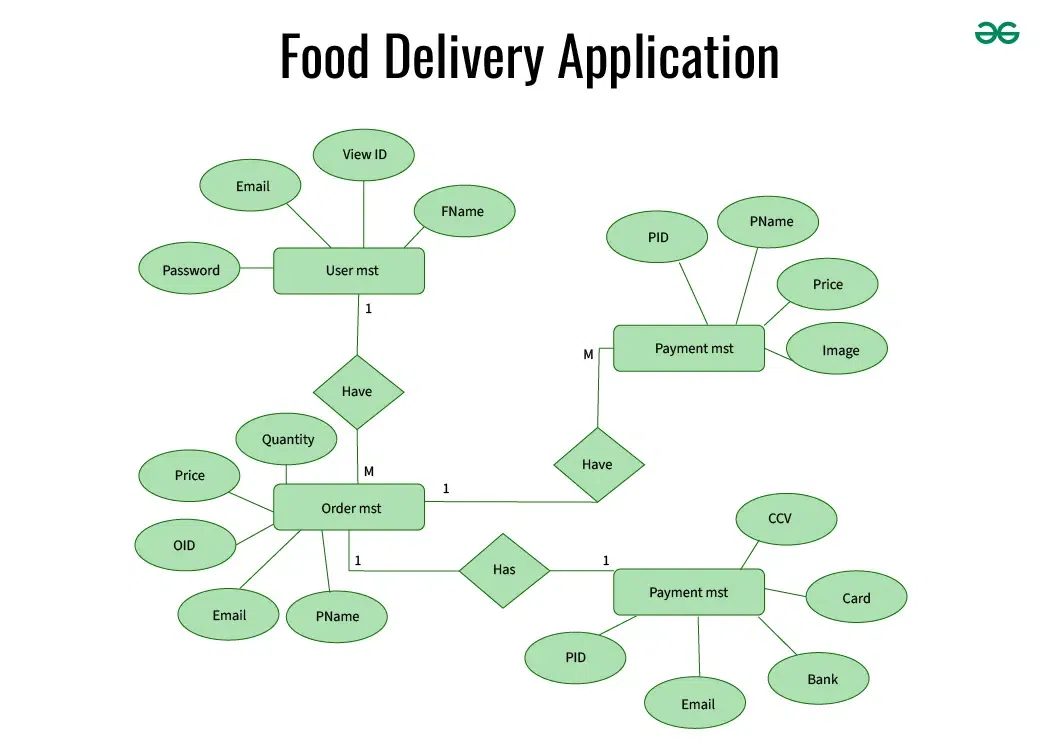


**Level 2 DFD (Restaurant Process):**

* **Manage Menu:** Restaurants manage their menus, including adding, updating, or removing menu items as needed.
* **Receive Order:** Restaurants receive incoming orders from users, acknowledge them, and prepare them for delivery.
* **Update Order Status:** Restaurants update the status of received orders as they progress through the preparation and delivery process.
* **Update Inventory:** Restaurants update their inventory levels based on the fulfillment of orders to ensure accurate tracking of available menu items.

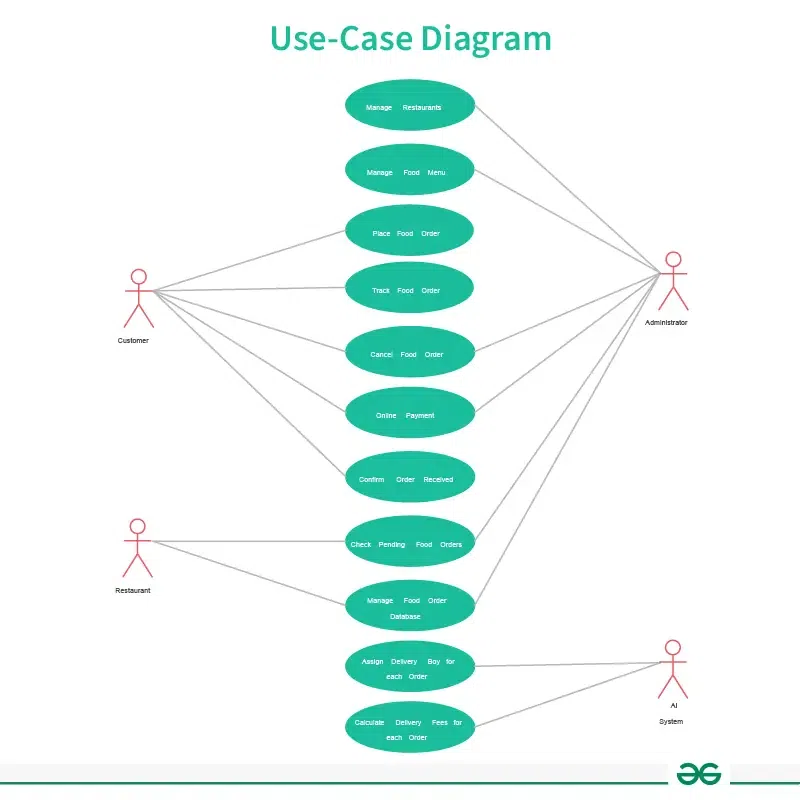


**3.2.2 E-R DIAGRAM**



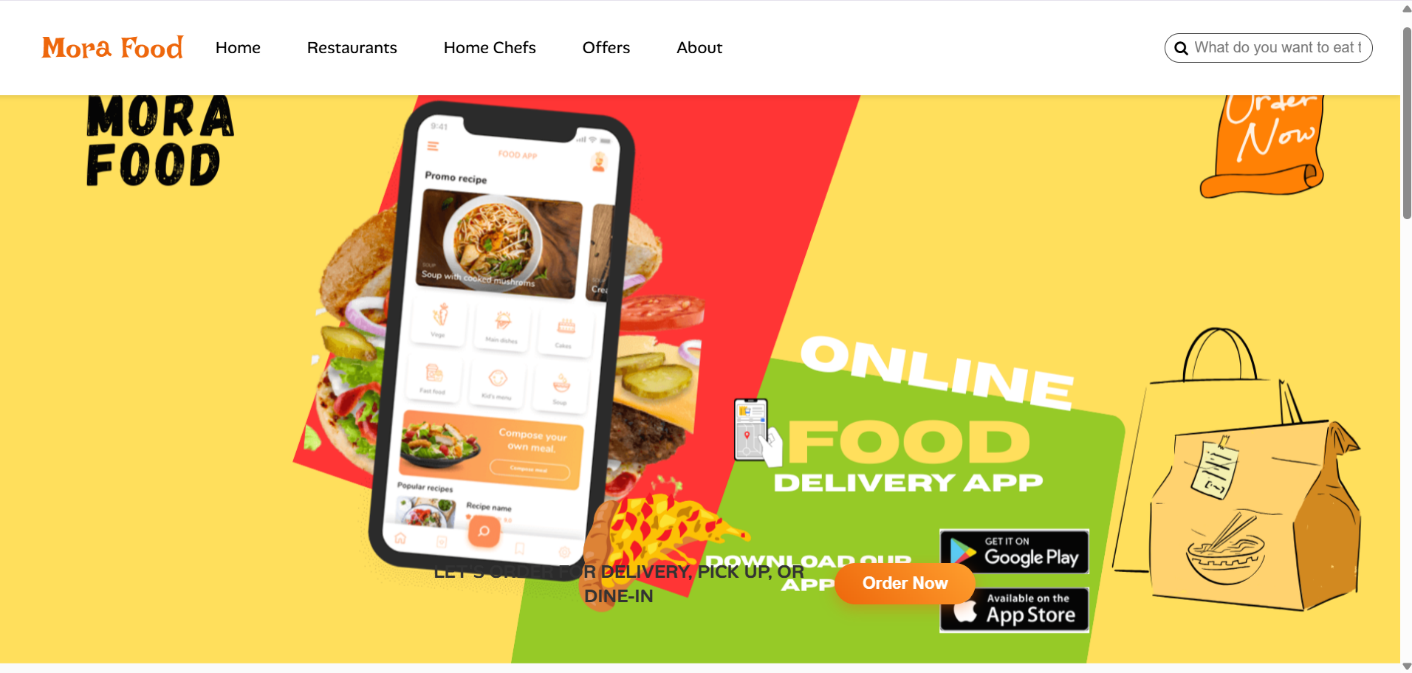
**3.2.3 UML DIAGRAM**

**USECASE DIAGRAM**

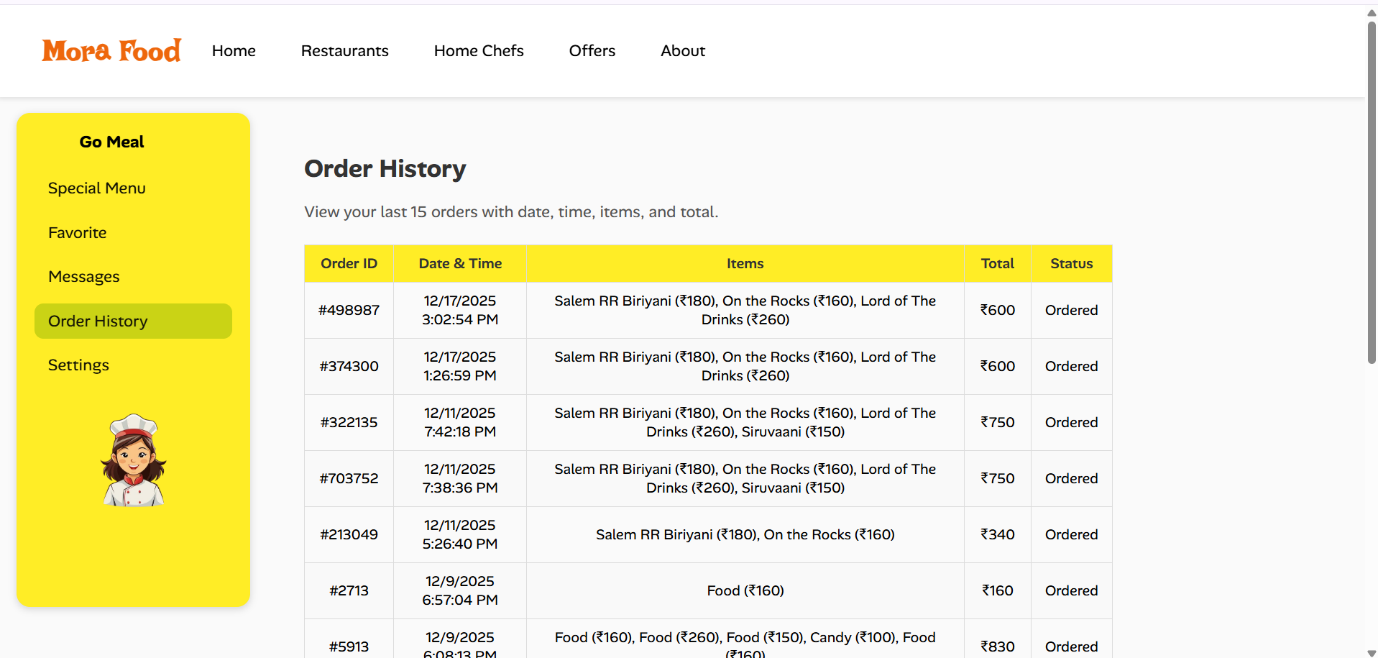
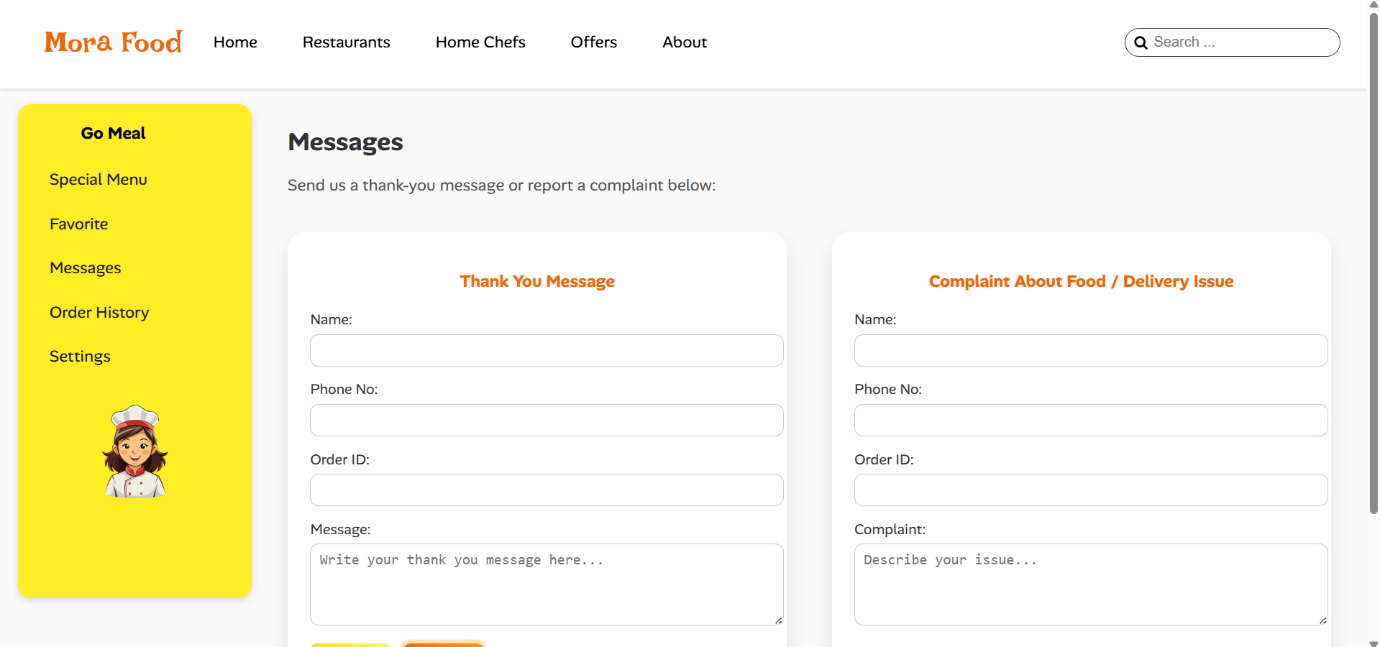
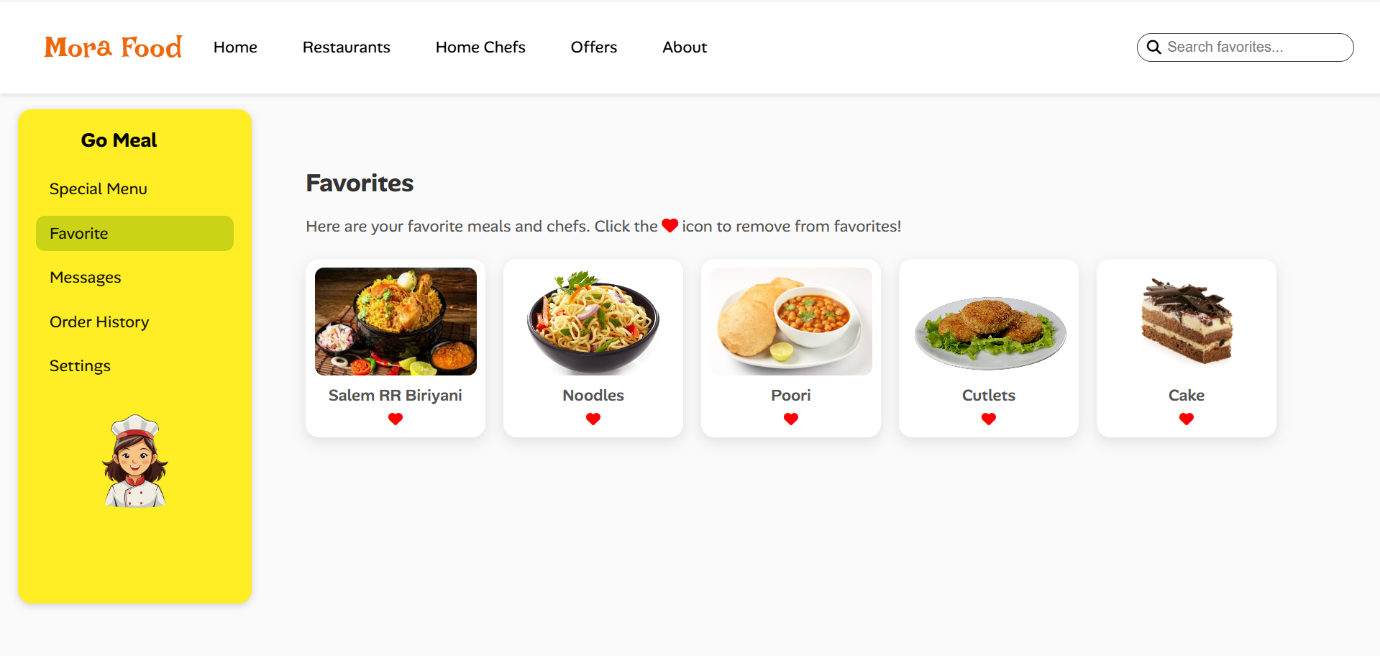
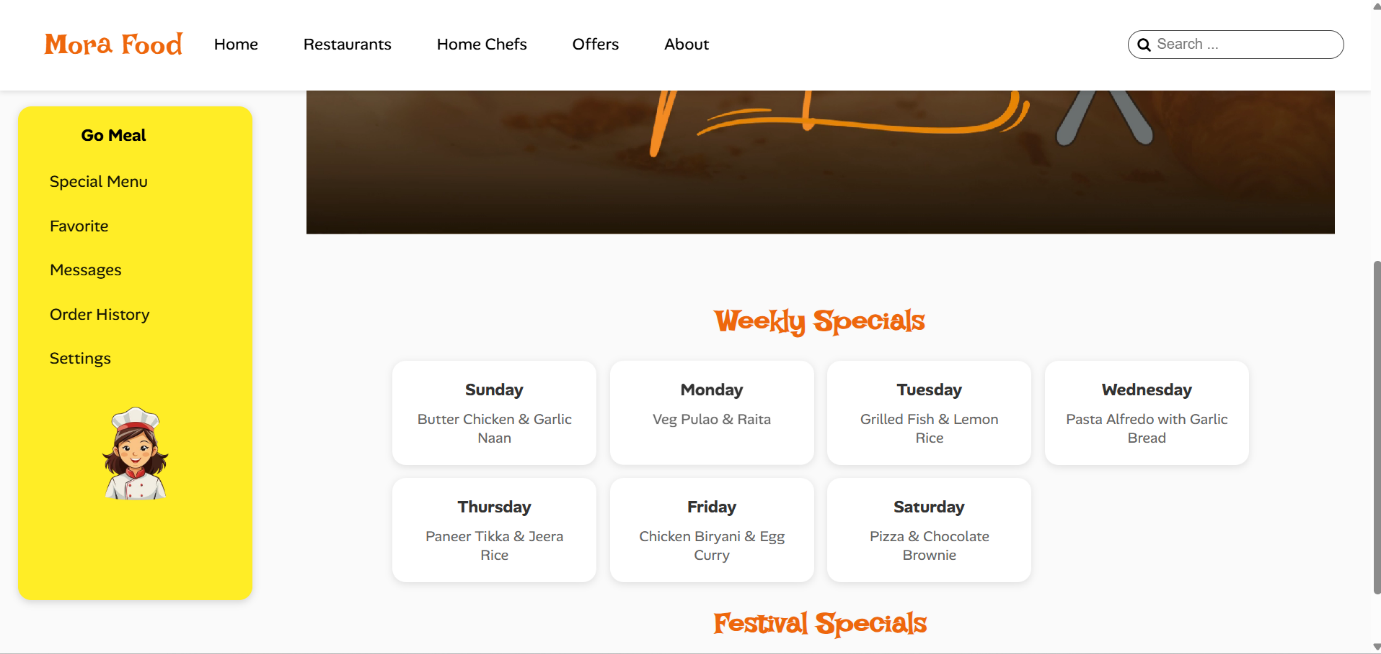
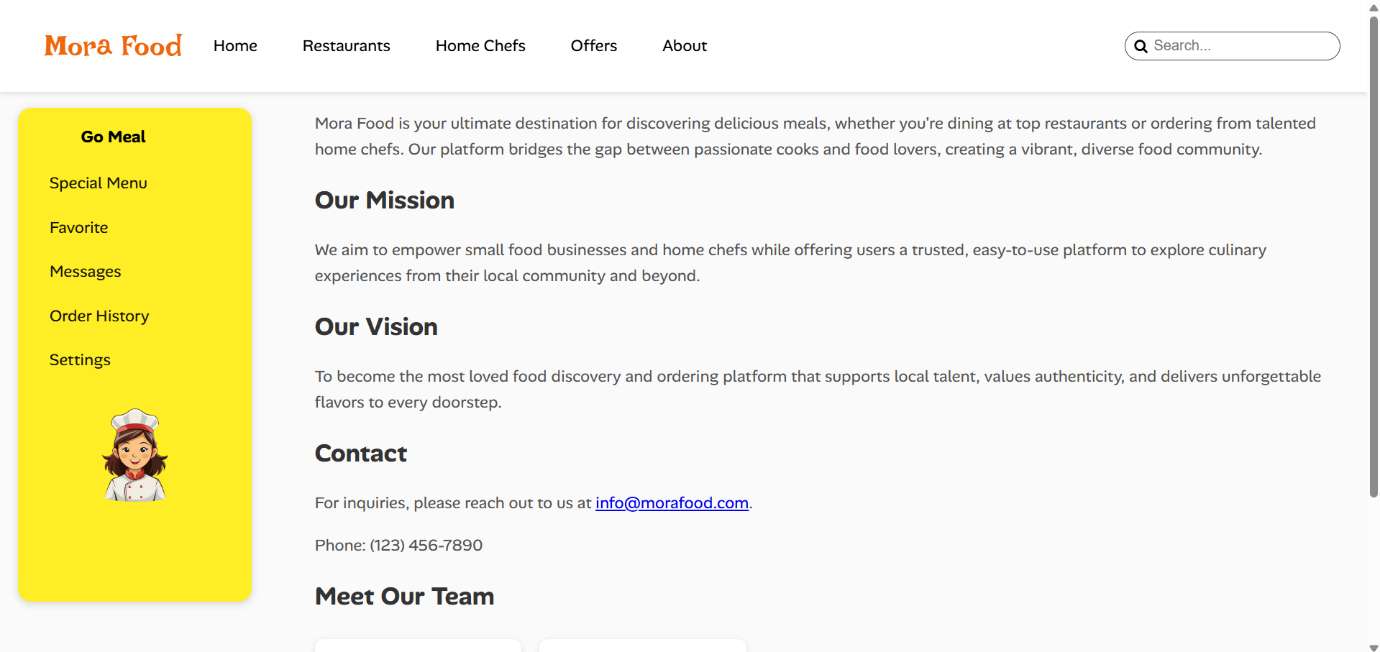
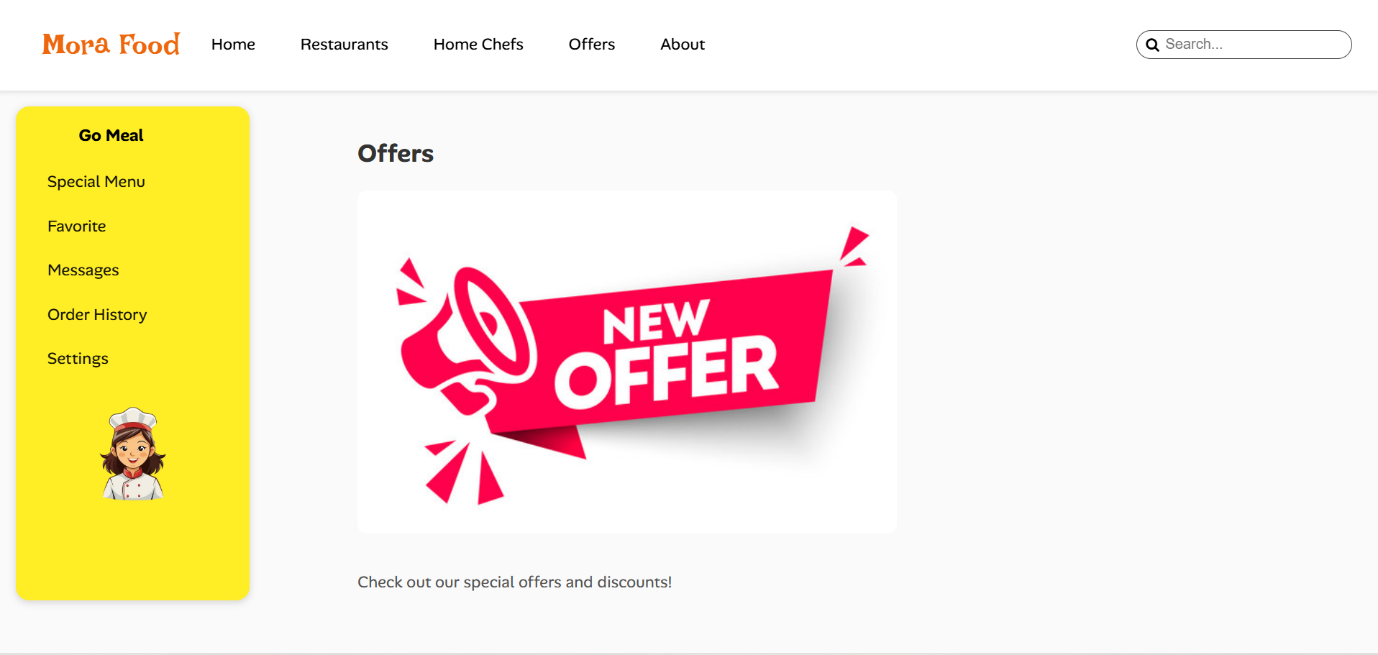
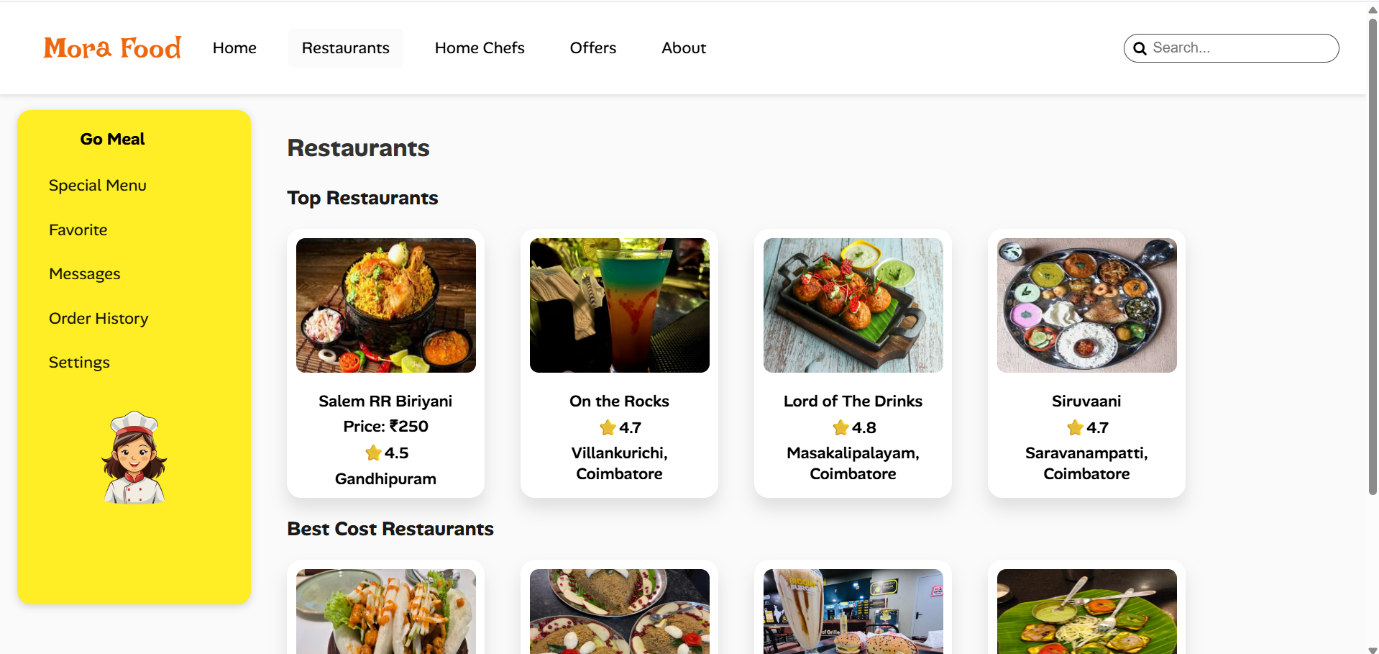
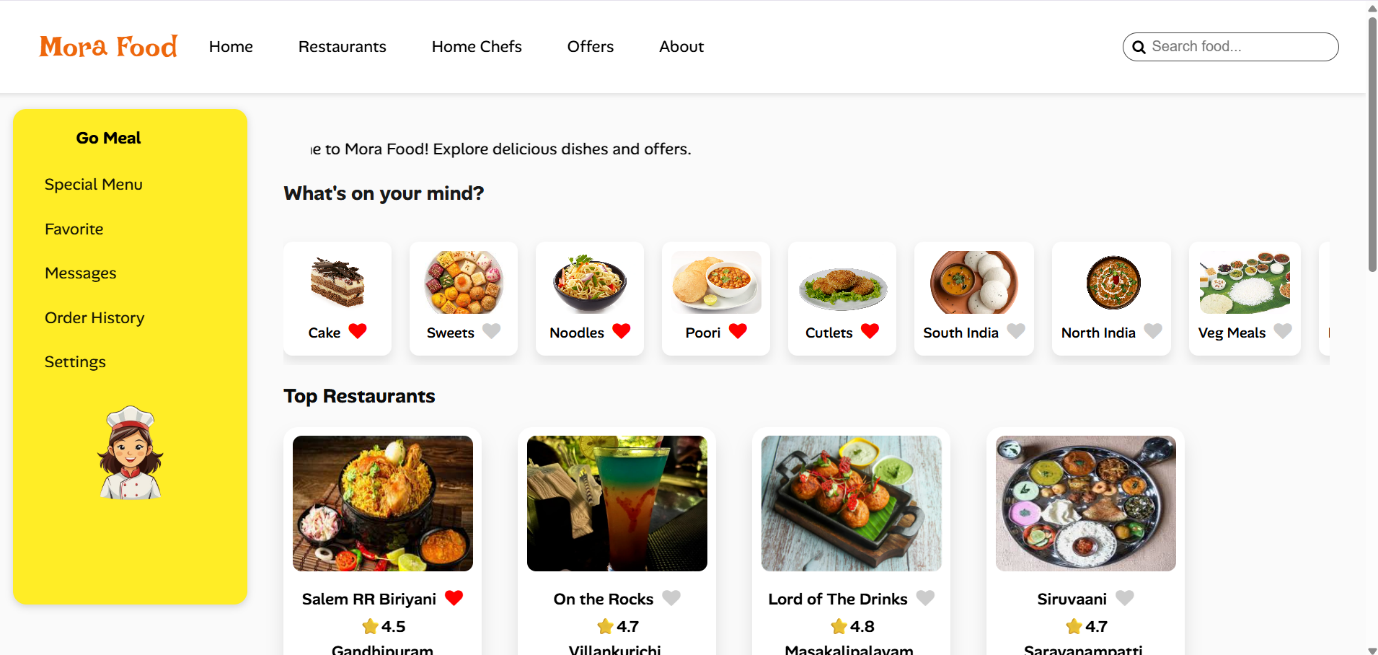
****

**3.3 USER INTERFACE DESIGN**

Home Page



User Output



## CHAPTER-IV

## SYSTEM IMPLEMENTATION

Implementation is the stage in the project where the theoretical design is turned into a working system. The most critical stage is achieving a successful system and in giving confidence on the new system for the users, what it will work efficient and effectively. It involves careful planning, investing of the current system, and its constraints on implementation, design of methods to achieve the change over methods.

The implementation process begins with preparing a plan for the implementation of the system. According to this plan, the activities are to be carried out in these plans; discussion has been made regarding the equipment, resources and how to test activities.

The coding step translates a detail design representation into a programming language realization. Programming languages are vehicles for communication between human and computers programming language characteristics and coding style can profoundly affect software quality and maintainability. The coding is done with the following characteristics in mind.

* Ease of design to code translation.
* Code efficiency.
* Memory efficiency.
* Maintainability.

The user should be very careful while implementing a project to ensure what they have planned is properly implemented. The user should not change the purpose of project while implementing. The user should not go in a roundabout way to achieve a solution; it should be direct, crisp and clear and up to the point.

## CHAPTER-V

## SYSTEM TESTING

System Testing is the testing of a complete and fully integrated software product. Usually software is the only one element of a larger computer based system. Ultimately, software is interfaced with other software/hardware systems. System Testing is actually a series of different tests whose sole purpose is to exercise the full computer based system. Software Testing is an important review of specification, design and coding. The increasing visibility of software as a system element and costs associated with the software failure are motivating forces for well-planned through testing.

Though the test phase is often thought of separate and distinct from the development effort first developers and then testing is a concurrent process that provides valuable information for the development team. There are at least three options for integrating project builder into the test phase.

Testers do not install project builder, use project builder functionality to compile and source-control the modules to be tested and hand them off to the tester, whose process remains unchanged.

The testers import the same project or the project that the developer uses. Create the project based on the development project but customized for the testers (For example- It does not include support documents, source) who imports it.

## TESTING OBJECTIVES

There are several rules that can serve as testing objectives. They are,

* Testing is executing a program with the intent of finding an error.
* A good test case is one that has a high probability of finding an undiscovered error.
* A successful test is one of that uncovers the undiscovered error.

If testing is conducted successfully according to the objective stated above, It will uncover the error in the software.

## TYPES OF TESTING

Testing is the process of executing the program with the intent of finding errors. Testing cannot show the absence of defects, It can only show that software errors are present. The testing principles used are

* Tests are traceable to customer requirements.
* 80% of errors will likely be traceable to 20% of program modules.
* Testing should begin ‘in-small’ and progress towards testing ‘in-large’.

The types of testing are

* Unit testing
* Integration testing
* White box testing
* Black box testing

In our project, we used unit testing and validation testing for checking the system.

## UNIT TESTING

Unit testing is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output. In procedural programming, a unit may be an individual program, function, procedure, etc. In object-oriented programming, the smallest unit is a method, which may belong to a base/ super class, abstract class or derived/ child class. Unit testing frameworks, drivers, stubs, and mock/ fake objects are used to assist in unit testing.

In our system, we tested each module as a separate unit. Each program from the modules are tested and corrected.

## INTEGRATION TESTING

Integration Testing ensures that the product actually meets the client's needs. It can also be defined as to demonstrate that the product fulfils its intended use when deployed on appropriate environment. integration Testing is carried out as a whole system to ensure the system/product works well and that meets the requirements of the customer. We validated our project by doing this integration testing. It gives the desired result and satisfies the customer need.

**VALIDATION TESTING**

Validation testing can be defined in many ways, but a simple definition is that can be reasonably expected by the customer. After validation test has been conducted, one of two possible conditions exists.

* The functions or performance characteristics confirm to specification and are accepted.
* A deviation from specification is uncovered and a deficiency list is created.

Proposed system under consideration has been tested by using validation testing and found to be working satisfactorily.

**WHITE BOX TESTING**

White box testing, sometimes called glass-box testing is a test case design method that uses the control structure of the procedural design to derive test cases. Using white box testing methods, the software engineer can derive test cases that

* Guarantee that all independent paths with in a module have been exercised at least once.
* Exercise all logical decisions on their true and false sides.
* Execute all loops at their boundaries and with in their operational bounds and
* Exercise internal data structure to assure their validity.

For example in this project white box testing is performed against patient module. Without entering text if we apply it displays the message “First add record then save it” else it should be saved.

**BLACK BOX TESTING**

This method treats the coded module as a black box. The module runs with inputs that are likely to cause errors. Then the output is checked to see if any error occurred. This method cannot be used to test all errors, because some errors may depend on the code or algorithm used to implement the module.

## CHAPTER-VI

**SYSTEM MAINTENANCE**

System maintenance is an essential phase of the software development life cycle that begins after the system has been successfully implemented and delivered to the users. It involves the continuous process of monitoring, updating, modifying, and improving the software to ensure that it remains efficient, reliable, secure, and relevant over time. In real-world usage, software systems often encounter unexpected errors, changing user requirements, and evolving technological environments such as new browsers, devices, and operating systems. Therefore, system maintenance is required to correct defects (corrective maintenance), adapt the application to new technical environments (adaptive maintenance), enhance system performance and usability based on user feedback (perfective maintenance), and prevent future issues through code optimization and documentation updates (preventive maintenance). In the Mora Food – Online Food Ordering Web Application, system maintenance plays a crucial role in ensuring smooth operation of features such as dynamic cart management, favorites, order history, responsive layouts, and client-side data storage using localStorage. Regular maintenance activities include fixing JavaScript errors, improving UI responsiveness, updating Bootstrap components, optimizing CSS and scripts for better performance, and ensuring compatibility across multiple browsers and devices. Proper system maintenance increases the reliability and stability of the application, enhances user satisfaction, reduces long-term operational costs, and extends the lifespan of the software. It also provides a strong foundation for future enhancements such as backend integration, online payment systems, user authentication, and mobile application development, making the system scalable and sustainable in the long run.

## CHAPTER-VII

## CONCLUSION

In conclusion, the Mora Food online food ordering web application has been successfully designed and developed using modern front-end web technologies such as HTML5, CSS3, JavaScript, and Bootstrap to provide a responsive, interactive, and user-friendly platform. The system effectively addresses the limitations of traditional food ordering methods by offering digital access to restaurants, special menus, home chefs, offers, favorites, and order history through a single integrated interface. By implementing dynamic cart management, real-time updates, and persistent data storage using localStorage, the application ensures smooth user interaction and continuity across browser sessions. The modular design approach improves maintainability and scalability, while responsive layouts ensure consistent performance across desktops, tablets, and mobile devices. Thorough system analysis, structured design, careful implementation, and systematic testing have contributed to the reliability and usability of the application. Overall, this project demonstrates the practical application of web development concepts and highlights how modern front-end technologies can be effectively utilized to build scalable, efficient, and user-centric web applications suitable for real-world food ordering scenarios.

## CHAPTER-VIII

## FUTURE ENHANCEMENT

1. Integration of a backend server using technologies such as PHP, Node.js, or Java to handle user authentication, order processing, and secure data storage.
2. Implementation of a centralized database (MySQL or MongoDB) to manage users, orders, restaurants, menus, and payment records efficiently.
3. Addition of online payment gateway support (UPI, debit/credit cards, net banking, wallets) for secure and cashless transactions.
4. Real-time order tracking system with live status updates and notifications for order confirmation, preparation, and delivery.
5. Development of a dedicated mobile application using frameworks like React Native or Flutter for better performance and accessibility on smartphones.
6. Introduction of an admin and restaurant dashboard for managing menus, prices, offers, and monitoring orders and sales reports.
7. Implementation of user review and rating features to improve trust, transparency, and service quality.
8. Personalized food recommendations based on user preferences and order history using data analytics.
9. Location-based restaurant filtering and map integration to show nearby restaurants and delivery availability.
10. Enhanced security features such as encrypted data storage, secure login sessions, and protection against common web vulnerabilities.

## CHAPTER-IX

## BIBLIOGRAPHY

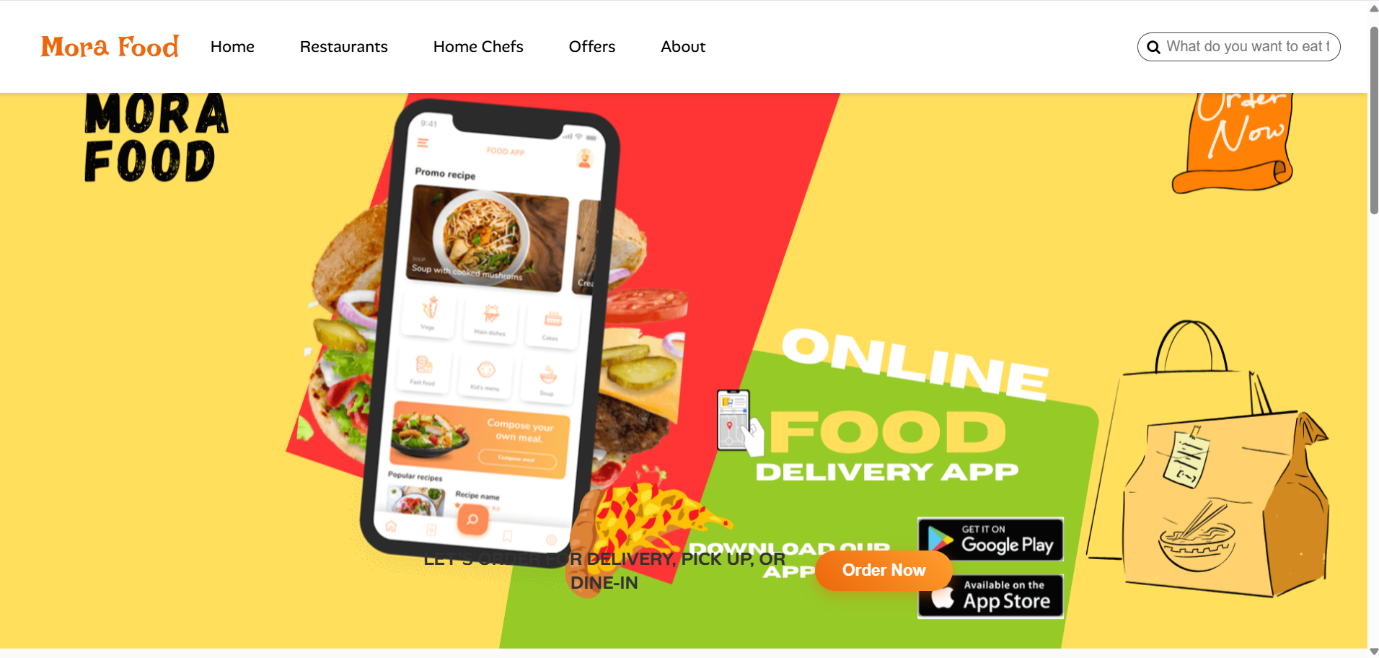
* W3Schools, “HTML5 Tutorial,” for understanding semantic structure and page layout in web applications, Available at: <https://www.w3schools.com/html/>
* W3Schools, “CSS3 Tutorial,” for styling, animations, media queries, and responsive design, Available at: <https://www.w3schools.com/css/>
* W3Schools, “JavaScript Tutorial,” for implementing interactivity, dynamic content updates, and client-side data handling, Available at: <https://www.w3schools.com/js/>
* Bootstrap Official Documentation, “Bootstrap Framework,” for responsive grid system and UI components in food ordering applications, Available at: https://getbootstrap.com/docs/
* Mozilla Developer Network (MDN), “Web APIs and Front-End Development,” for in-depth reference on HTML, CSS, JavaScript, and browser compatibility, Available at: <https://developer.mozilla.org/>
* Pressman, R. S., Software Engineering: A Practitioner’s Approach, McGraw-Hill Education, for system analysis, design, testing, and maintenance concepts.
* Sommerville, I., Software Engineering, Pearson Education, for understanding SDLC phases and project documentation standards.
* Duckett, J., HTML and CSS: Design and Build Websites, Wiley Publishing, for modern UI design principles used in food applications.
* Flanagan, D., JavaScript: The Definitive Guide, O’Reilly Media, for advanced JavaScript concepts applied in cart and order management.
* GeeksforGeeks, “Online Food Ordering System Concepts,” for functional understanding of food delivery platforms, Available at: <https://www.geeksforgeeks.org/>

## CHAPTER-X

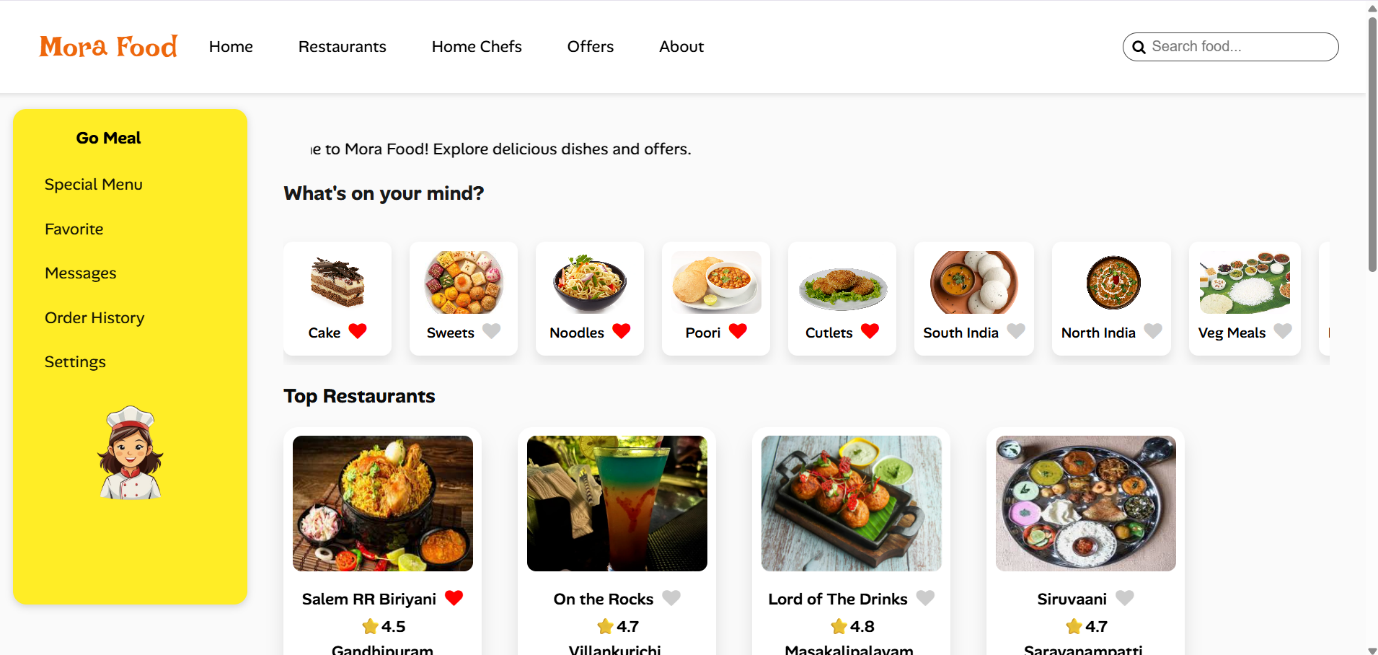
## APPENDIX

**10.1 SAMPLE SCREENS**

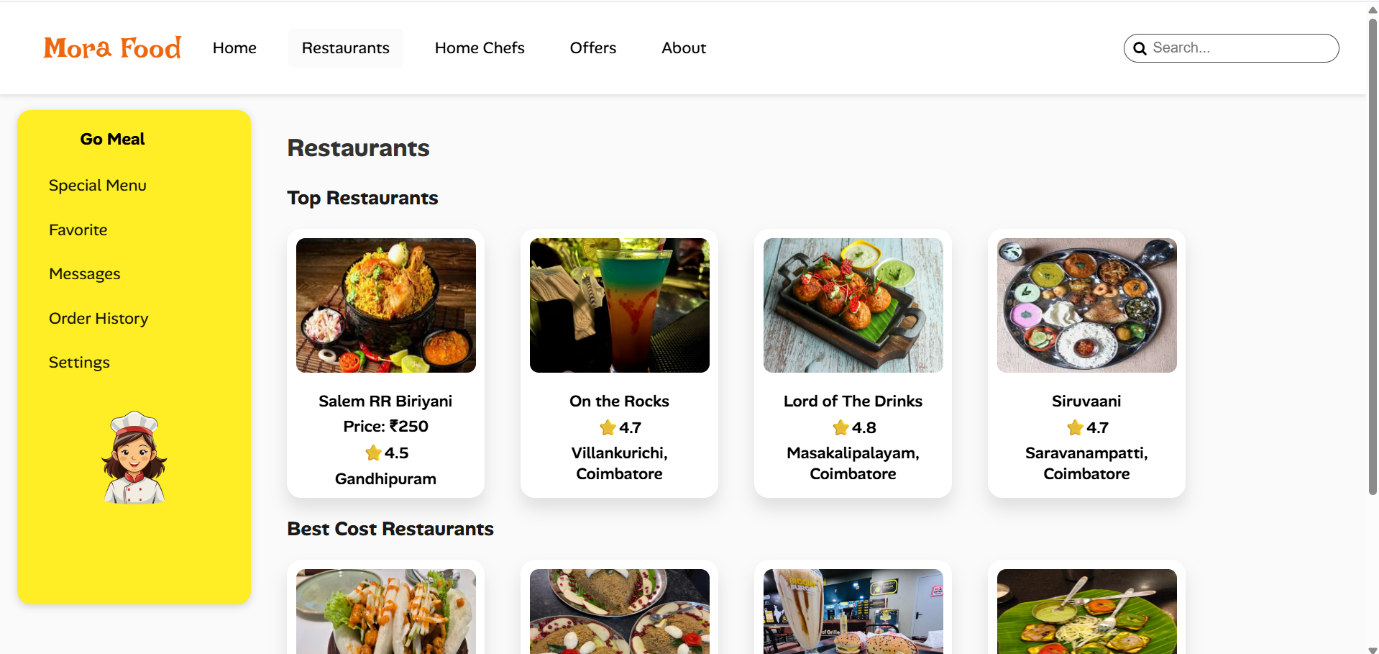
Home Page

****

Main Page

****

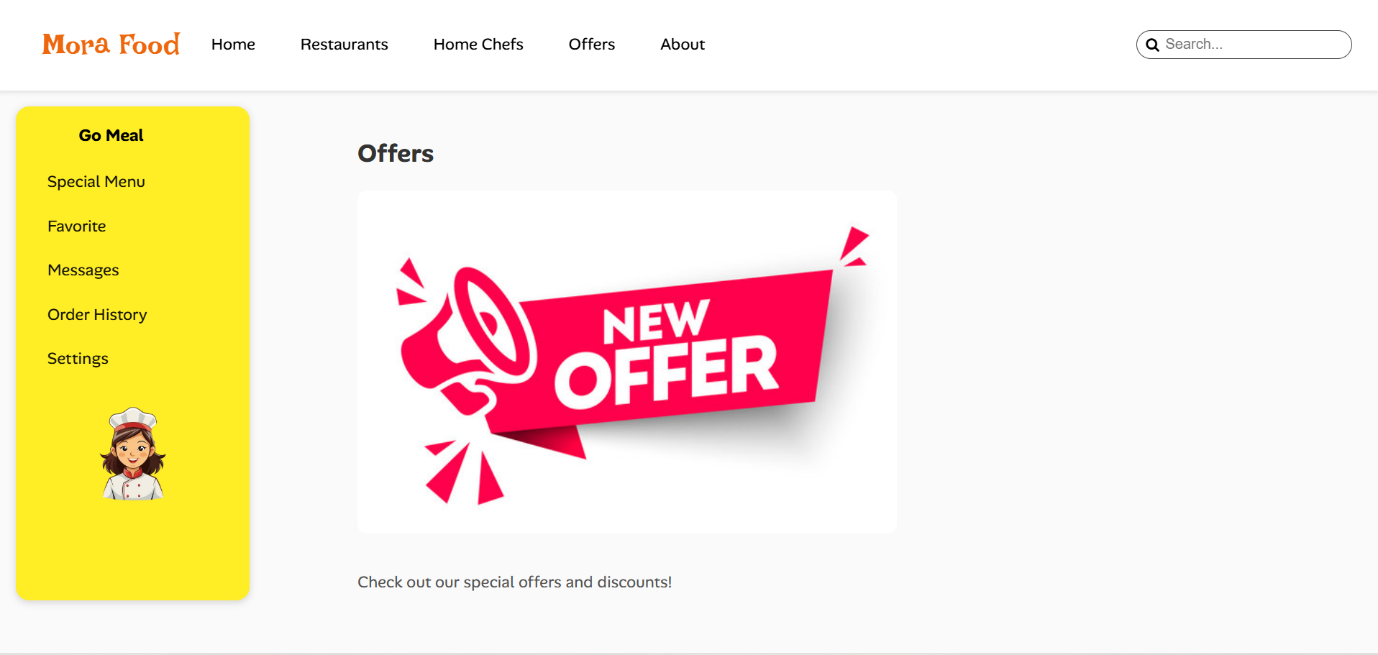
Restaurants Page

****

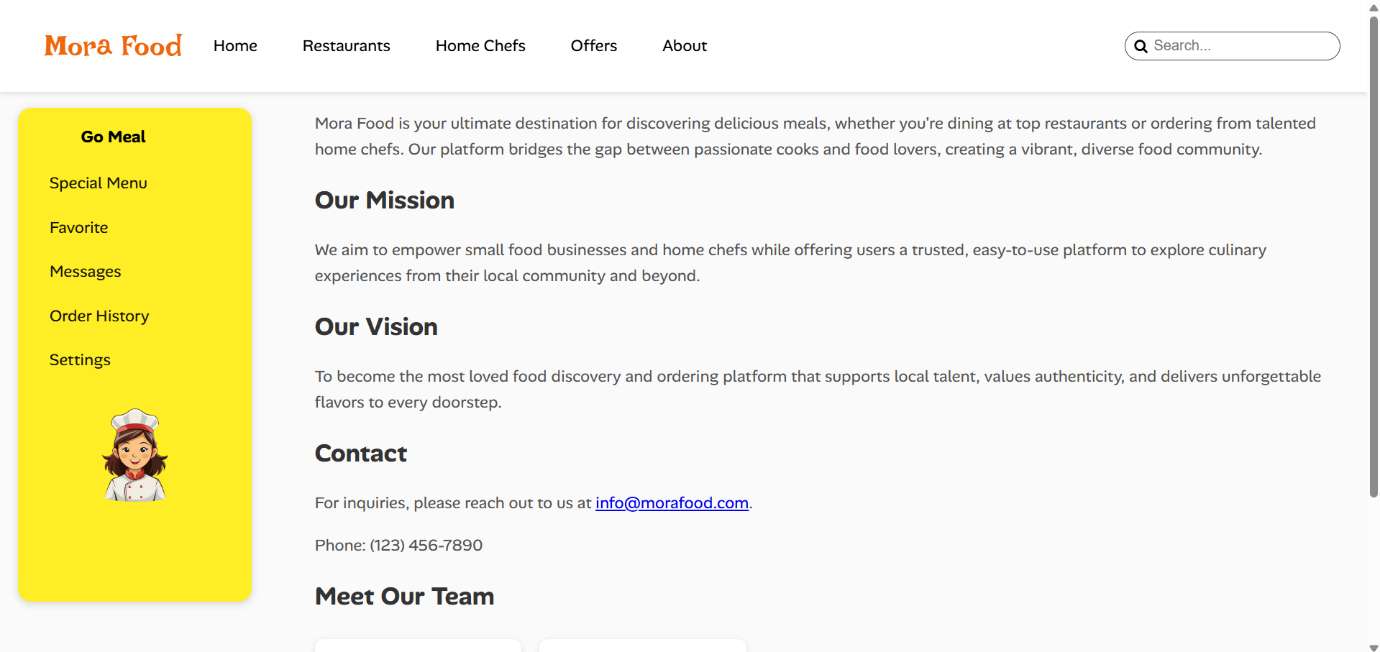
HomeChefs Page

****

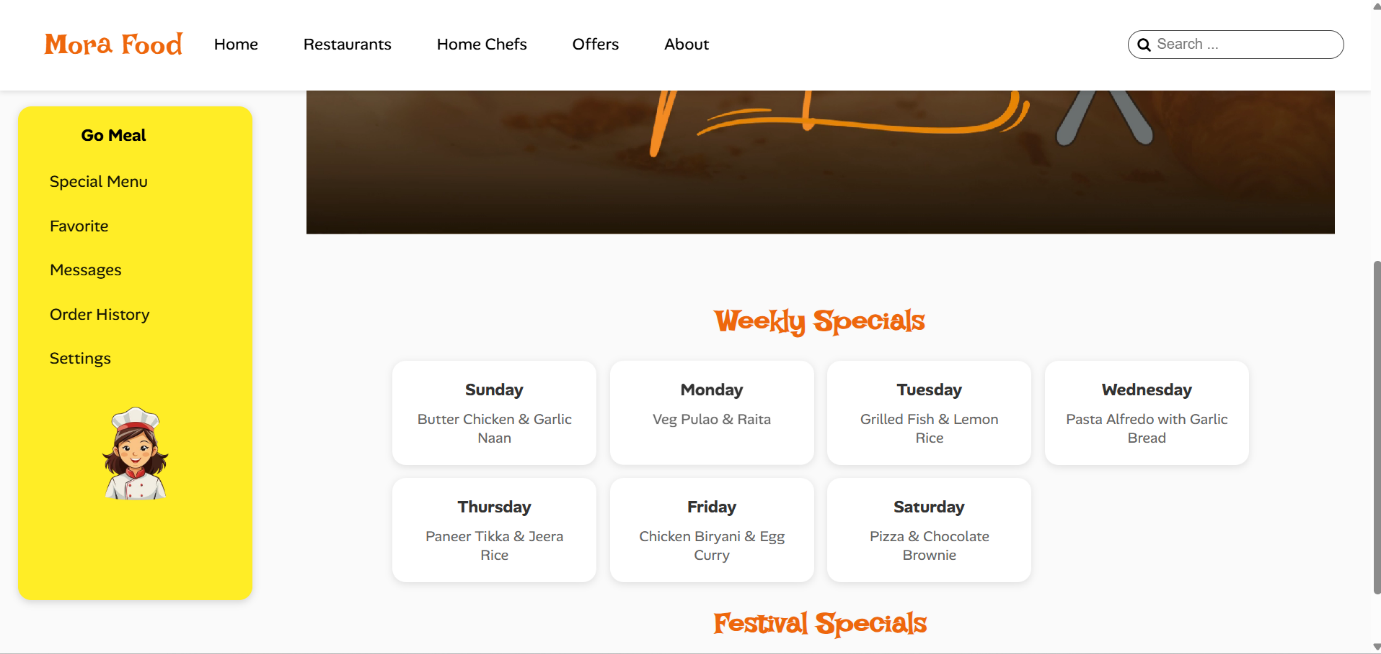
Offers Page

****

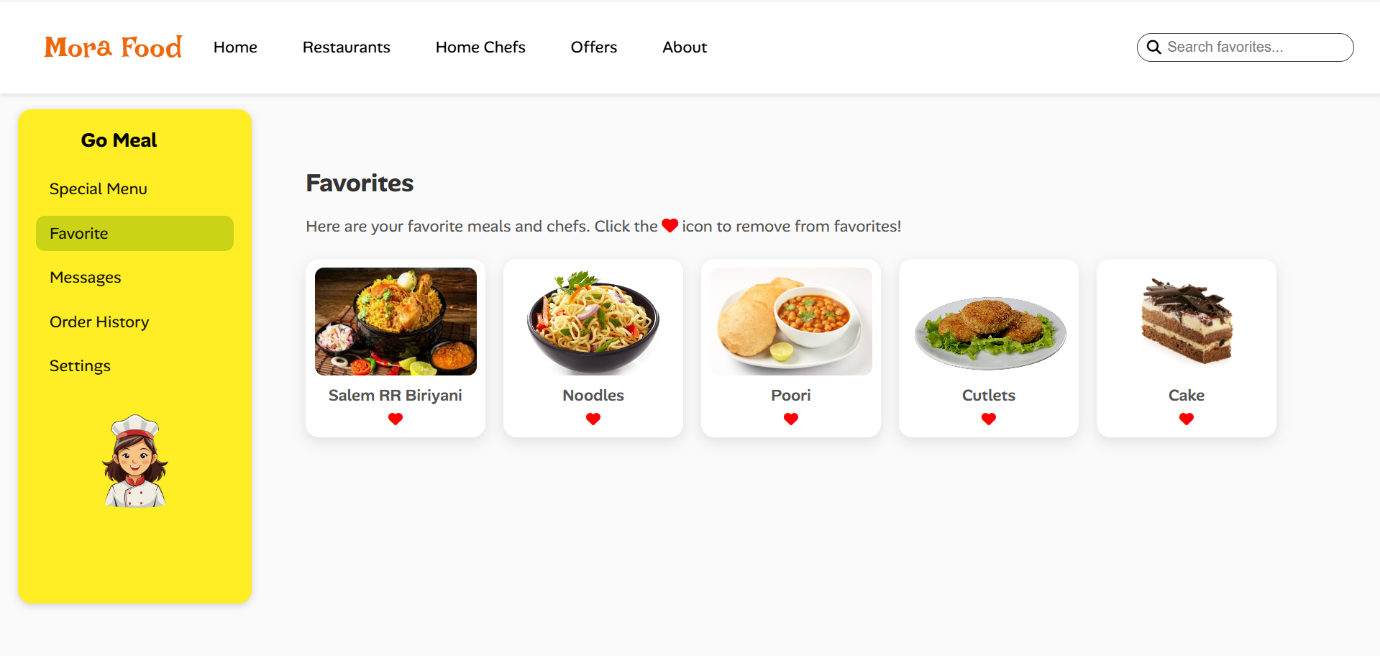
About Page

****

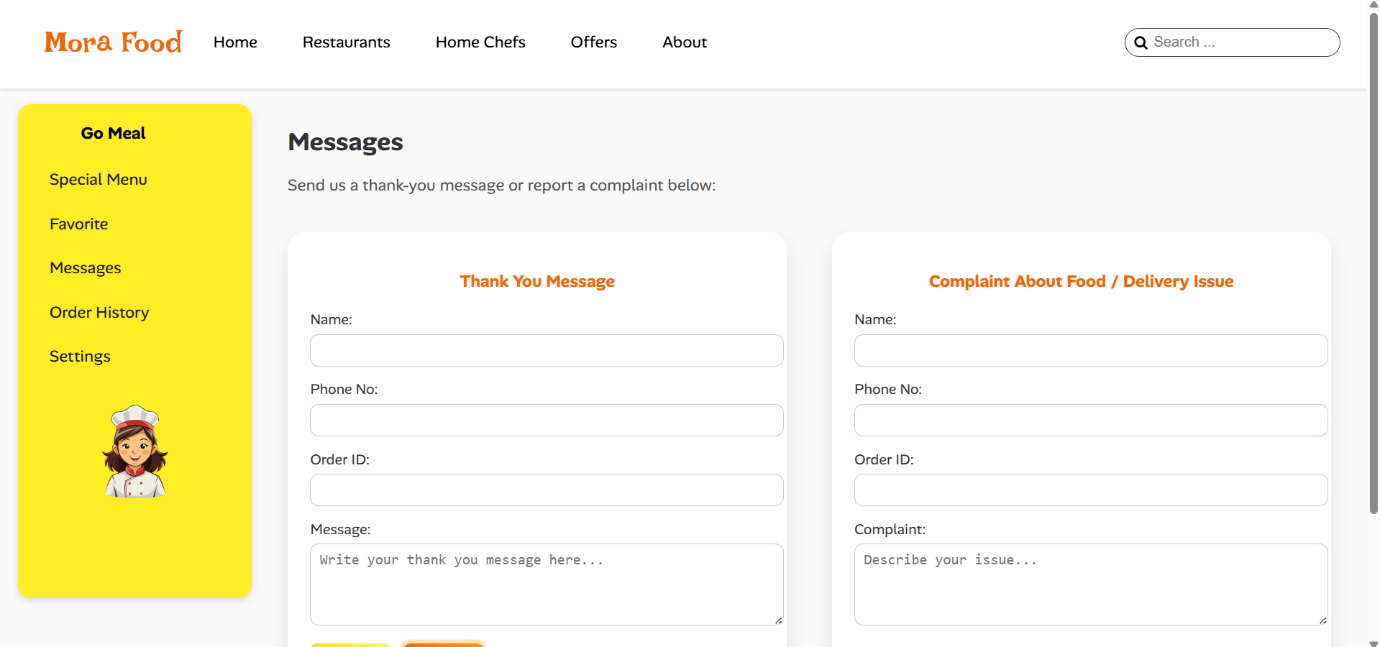
Special Menu Page

****

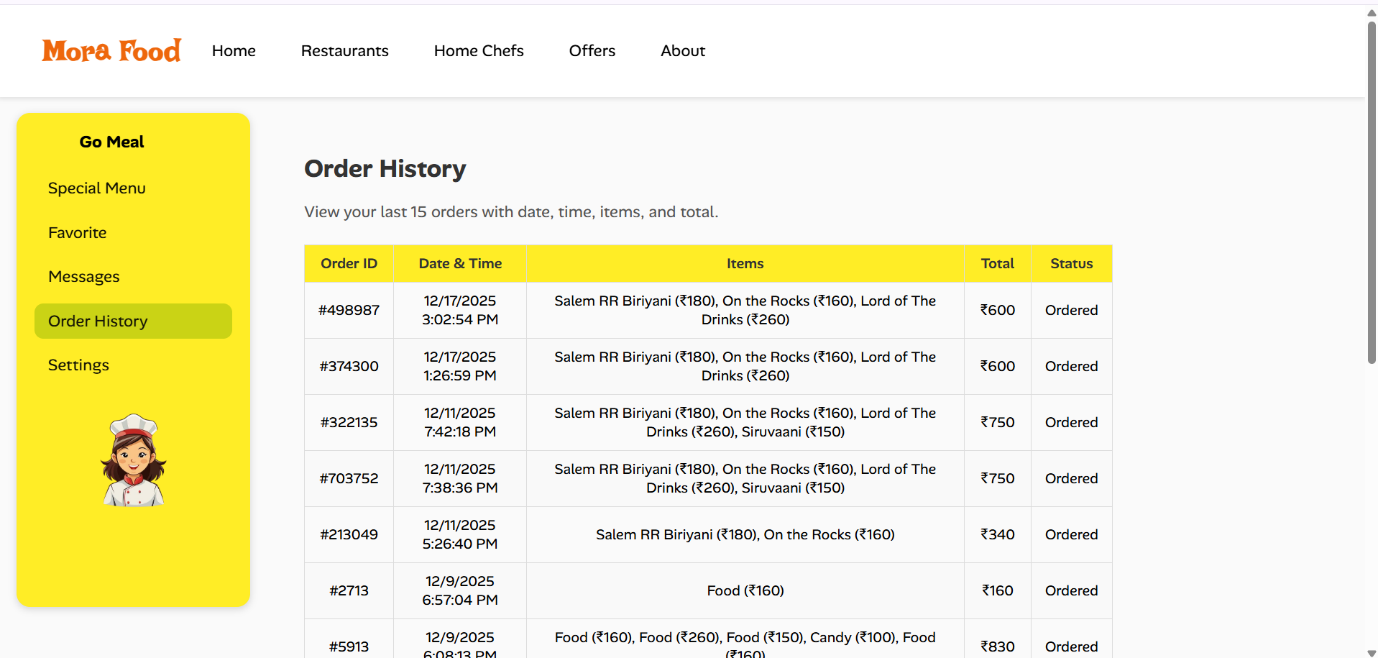
Favorite Page

****

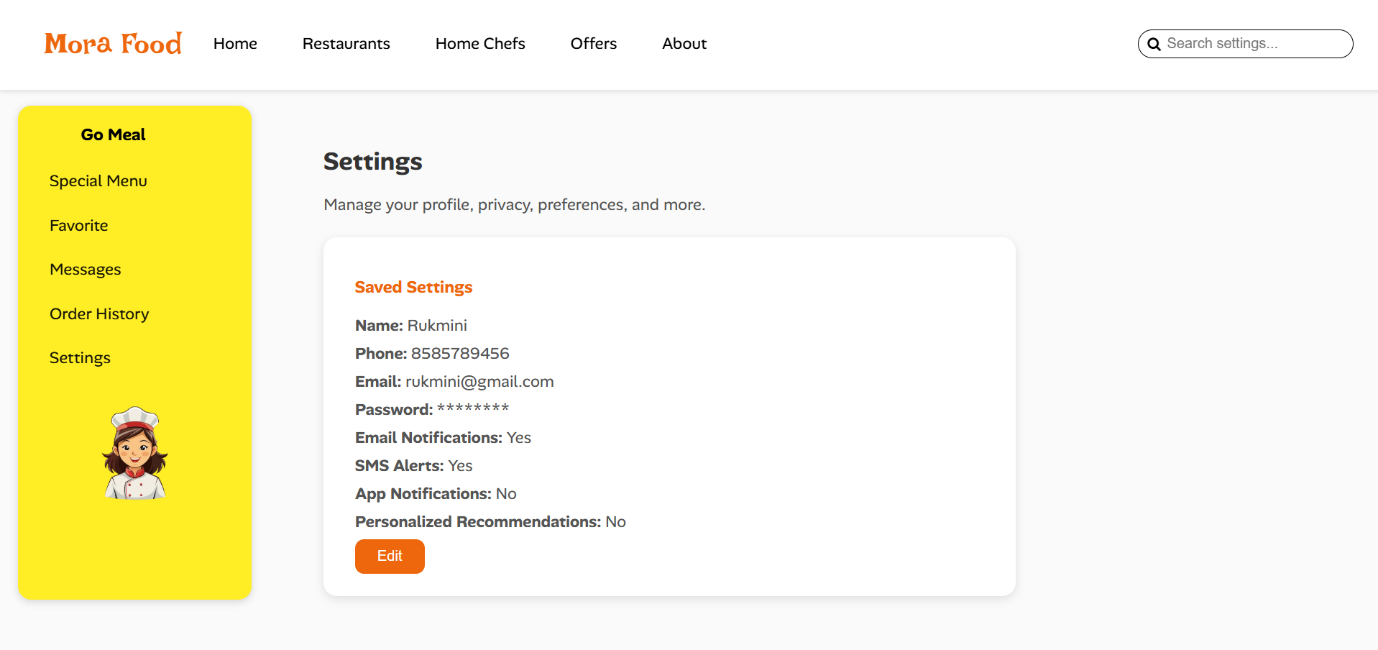
Message Page

****

Order Page

****

Setting Page



**10.2 SOURCE CODE**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

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

<title>Mora Food</title>

<!-- Font Awesome -->

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

<!-- Google Fonts -->

<link href="https://fonts.googleapis.com/css2?family=Alan+Sans:wght@300..900&family=Irish+Grover&display=swap" rel="stylesheet" />

<style>

  body { background-color: hsl(0, 0%, 98%); font-family: "Alan Sans", sans-serif; overflow-x: hidden; margin: 0; }

  h1 { font-family: "Irish Grover", cursive; font-weight: 400; color: #ee670c; margin-left: 20px; cursor: pointer; }

  .nav { display: flex; align-items: center; gap: 20px; background-color: white; padding: 10px 30px; position: fixed; top: 0; left: 0; right: 0; box-shadow: 0 2px 5px rgba(0,0,0,0.1); z-index: 10; }

  .nav a { text-decoration: none; color: black; font-size: 18px; padding: 10px 15px; border-radius: 5px; transition: background-color 0.3s; }

  .nav a:hover { background-color: #f0f0f0; }

  .search { margin-left: auto; display: flex; align-items: center; border: 1px solid #444; border-radius: 20px; padding: 5px 10px; }

  .search input { border: none; outline: none; background: none; font-size: 16px; margin-left: 5px; }

  .title-p { margin: 15px 30px; font-size: large; white-space: nowrap; overflow: hidden; }

  .title-p p { display: inline-block; animation: moveText 15s linear infinite; }

  @keyframes moveText { from { transform: translateX(-100%); } to { transform: translateX(100%); } }

  .sidebar { position: fixed; top: 120px; left: 20px; bottom: 60px; width: 220px; background-color: #ffed26; border-radius: 15px; box-shadow: 0 2px 8px rgba(0,0,0,0.15); padding: 20px; }

  .sidebar h3 { margin: 0 0 20px 50px; }

  .sidebar ul { list-style: none; padding: 0; margin: 0; }

  .sidebar li { margin: 10px 0; }

  .sidebar a { text-decoration: none; color: #151506; font-size: 18px; display: block; padding: 8px 15px; border-radius: 10px; transition: background-color 0.3s; }

  .sidebar a:hover { background-color: #cad315; }

  .chef img { width: 150px; border-radius: 10px; margin: 25px auto 10px; display: block; }

  .content { margin-left: 280px; padding: 120px 40px; animation: fadeIn 1s ease-in-out; }

  .content h4 { font-size: 22px; margin-top: -10px; color: #0a0a0aef; }

  .category { margin-top: 20px; }

  .category h2 { font-size: 22px; color: #0a0a0a; }

  .category-list { display: flex; flex-wrap: wrap; gap: 40px; margin-top: 20px; }

  .food-scroll { display: flex; overflow-x: auto; gap: 20px; padding: 10px 0; margin: 20px 0; scroll-snap-type: x mandatory; scrollbar-width: none; -ms-overflow-style: none; }

  .food-scroll::-webkit-scrollbar { display: none; }

  .food-item { flex: 0 0 auto; text-align: center; scroll-snap-align: start; background: white; border-radius: 10px; padding: 10px; box-shadow: 0 6px 12px rgba(0,0,0,0.1); }

  .food-item img { width: 100px; height: 70px; object-fit: cover; border-radius: 10px; }

  .food-item p { font-size: 16px; font-weight: 500; margin: 5px 0; }

  .favorite { cursor: pointer; font-size: 20px; color: #ccc; margin-left: 5px; transition: color 0.3s; }

  .favorite.active { color: red; }

  .category-item { background: white; border-radius: 16px; padding: 10px; box-shadow: 0 12px 18px rgba(0,0,0,0.1); width: 200px; text-align: center; opacity: 0; transform: translateY(40px); animation: slideUp 0.8s forwards; }

  .category-item:hover { transform: translateY(-8px) scale(1.03); box-shadow: 0 15px 25px rgba(238,212,12,0.3); transition: all 0.4s ease; }

  .category-item img { width: 100%; height: 150px; border-radius: 10px; margin-bottom: 10px; transition: transform 0.3s ease; }

  .category-item:hover img { transform: scale(1.05); }

  .category-item p { font-weight: 600; font-size: 18px; margin: 5px 0; color: #000; }

  .price { color: green; font-weight: bold; }

  @keyframes slideUp { from { opacity: 0; transform: translateY(40px); } to { opacity: 1; transform: translateY(0); } }

  @keyframes fadeIn { from { opacity: 0; } to { opacity: 1; } }

  .download-prompt { text-align: center; font-size: large; margin-bottom: 30px; }

  .cart-box { position: fixed; top:140px; right:20px; width:250px; background:white; border-radius:15px; padding:15px; box-shadow:0 6px 18px rgba(0,0,0,0.15); }

  .cart-items{ max-height:200px; overflow-y:auto; font-size:14px; }

  .cart-item{ display:flex; justify-content:space-between; margin-bottom:6px; }

  .cart-total{ margin-top:10px; font-weight:bold; border-top:1px solid #ccc; padding-top:8px; }

  .add-btn{ background:#ee670c; border:none; color:white; padding:6px 10px; border-radius:8px; cursor:pointer; }

  .add-btn:hover{ background:#66ce1c; }

  #cartBox { display: none; }

  #confirmBtn {

  background-color: #28a745; /\* Green color \*/

  color: white;

  border: none;

  padding: 8px 12px;

  border-radius: 8px;

  cursor: pointer;

  font-weight: bold;

  margin-top: 10px;

  width: 100%;

}

#confirmBtn:hover {

  background-color: #218838; /\* Darker green on hover \*/

}

  @media (max-width:865px){ h1{ font-size:20px; margin-top:25px; } .nav{flex-wrap:wrap; gap:5px; padding:10px 5px;} .nav a{font-size:15px;margin-left:10px;} .sidebar{top:125px; left:8px;width:160px;} .content{margin-left:195px;padding:100px 20px;} .category-list{justify-content:center; gap:20px;} }

  @media (max-width:480px){ .sidebar{display:none;} .content{margin:100px 20px; padding:20px;} .category-list{grid-template-columns:repeat(2,1fr); display:grid; gap:15px;} .category-item{width:100%;} .download-prompt{font-size:small;} }

</style>

</head>

<body>

<nav class="nav">

  <h1 onclick="location.href='food.html'">Mora Food</h1>

  <a href="Main.html">Home</a>

  <a href="Restaurants.html">Restaurants</a>

  <a href="Home Chefs.html">Home Chefs</a>

  <a href="Offers.html">Offers</a>

  <a href="About.html">About</a>

  <div class="search">

    <i class="fa fa-search"></i>

    <input type="text" id="searchInput" placeholder="Search food..." onkeyup="liveSearch()">

  </div>

</nav>

<aside class="sidebar">

  <h3>Go Meal</h3>

  <ul>

    <li><a href="Special Menu.html">Special Menu</a></li>

    <li><a href="Favorite.html">Favorite</a></li>

    <li><a href="Message.html">Messages</a></li>

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

    <li><a href="setting.html">Settings</a></li>

  </ul>

  <div class="chef"><img src="Img/chefs.png" alt="chef" /></div>

</aside>

<main class="content">

  <div class="title-p"><p>Welcome to Mora Food! Explore delicious dishes and offers.</p></div>

  <h4>What's on your mind?</h4>

  <!-- Food Items Scroll -->

  <div class="food-scroll">

    <div class="food-item"><img src="Img/cake.png" alt="Cake"><p>Cake <i class="fa fa-heart favorite" onclick="toggleFavorite(this,'Cake','Img/cake.png')"></i></p></div>

    <div class="food-item"><img src="Img/sweets1.jpg" alt="Sweets"><p>Sweets <i class="fa fa-heart favorite" onclick="toggleFavorite(this,'Sweets','Img/sweets1.jpg')"></i></p></div>

    <div class="food-item"><img src="Img/noodles.webp" alt="Noodles"><p>Noodles <i class="fa fa-heart favorite" onclick="toggleFavorite(this,'Noodles','Img/noodles.webp')"></i></p></div>

    <div class="food-item"><img src="Img/Poori.jpg" alt="Poori"><p>Poori <i class="fa fa-heart favorite" onclick="toggleFavorite(this,'Poori','Img/Poori.jpg')"></i></p></div>

    <div class="food-item"><img src="Img/cutlets.png" alt="Cutlets"><p>Cutlets <i class="fa fa-heart favorite" onclick="toggleFavorite(this,'Cutlets','Img/cutlets.png')"></i></p></div>

    <div class="food-item"><img src="Img/south india.webp" alt="South India"><p>South India <i class="fa fa-heart favorite" onclick="toggleFavorite(this,'South India','Img/south india.webp')"></i></p></div>

    <div class="food-item"><img src="Img/North India.png" alt="North India"><p>North India <i class="fa fa-heart favorite" onclick="toggleFavorite(this,'North India','Img/North India.png')"></i></p></div>

    <div class="food-item"><img src="Img/veg Meals.jpg" alt="Veg Meals"><p>Veg Meals <i class="fa fa-heart favorite" onclick="toggleFavorite(this,'Veg Meals','Img/veg Meals.jpg')"></i></p></div>

    <div class="food-item"><img src="Img/Paneer Butter masala.jpg" alt="Paneer Butter Masala"><p>Paneer Butter Masala <i class="fa fa-heart favorite" onclick="toggleFavorite(this,'Paneer Butter Masala','Img/Paneer Butter masala.jpg')"></i></p></div>

    <div class="food-item"><img src="Img/chicken gravy.webp" alt="Chicken Gravy"><p>Chicken Gravy <i class="fa fa-heart favorite" onclick="toggleFavorite(this,'Chicken Gravy','Img/chicken gravy.webp')"></i></p></div>

  </div>

  <!-- Top Restaurants Section -->

  <section class="category">

    <h2>Top Restaurants</h2>

    <div class="category-list">

      <div class="category-item"><a href="Salem RR Biriyani.html"><img src="Img/Salem RR.jpg" alt="Salem RR Biriyani"></a><p>Salem RR Biriyani <i class="fa fa-heart favorite" onclick="toggleFavorite(this,'Salem RR Biriyani','Img/Salem RR.jpg')"></i></p><p>⭐4.5</p><p>Gandhipuram</p><p class="price">₹180</p><button class="add-btn" onclick="addToCart('Salem RR Biriyani',180)">Add to Cart</button></div>

      <div class="category-item"><img src="Img/On rock.jpg" alt="On the Rocks"><p>On the Rocks <i class="fa fa-heart favorite" onclick="toggleFavorite(this,'On the Rocks','Img/On rock.jpg')"></i></p><p>⭐4.7</p><p>Villankurichi, Coimbatore</p><p class="price">₹160</p><button class="add-btn" onclick="addToCart('On the Rocks',160)">Add to Cart</button></div>

      <div class="category-item"><img src="Img/Lord of The Drinks.webp" alt="Lord of The Drinks"><p>Lord of The Drinks <i class="fa fa-heart favorite" onclick="toggleFavorite(this,'Lord of The Drinks','Img/Lord of The Drinks.webp')"></i></p><p>⭐4.8</p><p>Masakalipalayam, Coimbatore</p><p class="price">₹260</p><button class="add-btn" onclick="addToCart('Lord of The Drinks',260)">Add to Cart</button></div>

      <div class="category-item"><img src="Img/ss.jpg" alt="Siruvaani"><p>Siruvaani <i class="fa fa-heart favorite" onclick="toggleFavorite(this,'Siruvaani','Img/ss.jpg')"></i></p><p>⭐4.7</p><p>Saravanampatti, Coimbatore</p><p class="price">₹150</p><button class="add-btn" onclick="addToCart('Siruvaani',150)">Add to Cart</button></div>

    </div>

  </section>

  <!-- Snacks Section -->

  <section class="category">

    <h2>Snacks</h2>

    <div class="category-list">

      <div class="category-item"><img src="Img/cookies.jpg" alt="Cookies"><p>Cookies <i class="fa fa-heart favorite" onclick="toggleFavorite(this,'Cookies','Img/cookies.jpg')"></i></p><p>⭐4.5</p><p>Gandhipuram</p><p class="price">₹110</p><button class="add-btn" onclick="addToCart('Cookies',110)">Add to Cart</button></div>

      <div class="category-item"><img src="Img/candy.jpg" alt="Candy"><p>Candy <i class="fa fa-heart favorite" onclick="toggleFavorite(this,'Candy','Img/candy.jpg')"></i></p><p>⭐4.2</p><p>Gandhipuram</p><p class="price">₹100</p><button class="add-btn" onclick="addToCart('Candy',100)">Add to Cart</button></div>

      <div class="category-item"><img src="Img/dairy-products-milk.jpg" alt="Dairy Products"><p>Dairy Products <i class="fa fa-heart favorite" onclick="toggleFavorite(this,'Dairy Products','Img/dairy-products-milk.jpg')"></i></p><p>⭐4.0</p><p>Gandhipuram</p><p class="price">₹90</p><button class="add-btn" onclick="addToCart('Dairy Products',90)">Add to Cart</button></div>

      <div class="category-item"><img src="Img/sweets.jpg" alt="Sweets"><p>Sweets <i class="fa fa-heart favorite" onclick="toggleFavorite(this,'Sweets','Img/sweets.jpg')"></i></p><p>⭐4.5</p><p>Gandhipuram</p><p class="price">₹120</p><button class="add-btn" onclick="addToCart('Sweets',120)">Add to Cart</button></div>

      <div class="category-item"><img src="Img/chips.webp" alt="Chips"><p>Chips <i class="fa fa-heart favorite" onclick="toggleFavorite(this,'Chips','Img/chips.webp')"></i></p><p>⭐4.3</p><p>Gandhipuram</p><p class="price">₹90</p><button class="add-btn" onclick="addToCart('Chips',90)">Add to Cart</button></div>

      <div class="category-item"><img src="Img/Fruit.jpg" alt="Fruit"><p>Fruit <i class="fa fa-heart favorite" onclick="toggleFavorite(this,'Fruit','Img/Fruit.jpg')"></i></p><p>⭐4.6</p><p>Gandhipuram</p><p class="price">₹150</p><button class="add-btn" onclick="addToCart('Fruit',150)">Add to Cart</button></div>

    </div>

  </section>

</main>

<!-- CART -->

<div class="cart-box" id="cartBox">

  <h3>🛒 Cart</h3>

  <div class="cart-items" id="cartItems"></div>

  <div class="cart-total">Total: ₹<span id="total">0</span></div>

  <button id="confirmBtn">Confirm Order</button>

</div>

<h2 class="download-prompt">For a better experience, download the Mora app now</h2>

<script>

function toggleFavorite(el,name,img){

  let favorites=JSON.parse(localStorage.getItem("favorites"))||[];

  const index=favorites.findIndex(f=>f.img===img);

  if(index===-1){favorites.push({name,img}); el.classList.add("active");}

  else{favorites.splice(index,1); el.classList.remove("active");}

  localStorage.setItem("favorites",JSON.stringify(favorites));

}

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

    let favIcons=document.querySelectorAll(".favorite");

    let favorites=JSON.parse(localStorage.getItem("favorites"))||[];

    favIcons.forEach(el=>{

      const name=el.parentElement.textContent.trim();

      if(favorites.some(f=>name.includes(f.name))){el.classList.add("active");}

    });

  });

  // Cart

  let cart=[]; let total=0;

  function addToCart(name,price){

    cart.push({name,price}); total+=Number(price); renderCart(); showCart();

  }

  function renderCart(){

    const cartItemsEl=document.getElementById("cartItems");

    const totalEl=document.getElementById("total");

    cartItemsEl.innerHTML="";

    cart.forEach((item,idx)=>{

      cartItemsEl.innerHTML+=`<div class="cart-item"><span>${item.name}</span><span>₹${item.price}</span></div>`;

    });

    totalEl.textContent=total;

  }

  function liveSearch() {

  let input = document.getElementById("searchInput").value.toLowerCase();

  let items = document.querySelectorAll(".category-item");

  items.forEach(item => {

    let text = item.textContent.toLowerCase();

    if (text.includes(input)) {

      item.style.display = "block";

    } else {

      item.style.display = "none";

    }

  });

}

  function showCart(){document.getElementById("cartBox").style.display=total>0?"block":"none";}

  document.getElementById("confirmBtn").addEventListener("click",()=>{

    if(cart.length===0){alert("Your cart is empty!"); return;}

    const now=new Date(); const orderId='#'+Math.floor(Math.random()\*1000000);

    const orders=JSON.parse(localStorage.getItem("orders"))||[];

    orders.push({orderId,date:now.toLocaleDateString(),time:now.toLocaleTimeString(),items:cart,total,status:"Ordered"});

    localStorage.setItem("orders",JSON.stringify(orders));

    alert("Order confirmed! Order ID: "+orderId);

    cart=[]; total=0; renderCart(); showCart();

  });

  document.addEventListener("DOMContentLoaded",()=>{ renderCart(); showCart(); });

</script>

</body>

</html>