A Project Report on

ONLINE FOOD ORDERING AND DELIVERY PLATFORM

submitted in partial fulfilment of the Degree of Master of Computer Applications (MCA)
By

Mr. Keerthiraj Hemanth Moger
P03ME21S0035

Under the guidance of
Mr. Chandan Hegde
Assistant Professor, Department of MCA

SURANA COLLEGE
AUTONOMOUS
Affiliated to Bangalore University | Re-accredited by NAAC with A+
Recognized by AICTE - New Delhi

Department of MCA

September 2023

SURANA COLLEGE (AUTONOMOUS)

Re-Accredited by NAAC with 'A+' Grade
Affiliated to Bangalore University & Approved by AICTE
C-17, Kengeri Satellite Town, Bangalore - 60

DEPARTMENT OF MCA

SURANA COLLEGE
AUTONOMOUS
Affiliated to Bangalore University | Re-accredited by NAAC with A+
Recognized by AICTE - New Delhi

CERTIFICATE

This is to certify that **Mr. Keerthiraj Hemanth Moger (Reg.no:P03ME21S0035)** is a bonafide student of MCA fourth semester, Surana College (Autonomous), has successfully completed the project work entitled **"Online Food Ordering and Delivery Platform"** for the partial fulfilment of the requirements for the award of the degree of MCA fourth semester. The matter embodied in this project work has not been submitted to any other university for the award of any other degree.

Internal Guide **Director, MCA**

External Examiners:

ACKNOWLEDGEMENT

I take this opportunity to sincerely thank to all those who have encouraged me either directly or indirectly in completing the project.

I am thankful to **Dr. A Srinivas**, Director of MCA, Surana College and **Dr. K Balaji**, HOD MCA, Surana College, for giving this opportunity to undergo the project.

I am extremely grateful to my trainer at **IBM SkillBuild**, for his/her constant support, insights and timely guidance.

I thankful to my guide **Mr. Chandan Hegde**, Assistant Professor, MCA for his valuable guidance, suggestions and constant support.

Mr. Keerthiraj Hemanth Moger
P03ME21S0035

DECLARATION

I hereby declare that the project on **Online Food Ordering and Delivery Platform** embodies project affirmation carried out for the completion of fourth semester MCA curriculum at the department of MCA, Surana College (Autonomous), Bangalore, under the supervision of **Mr. Chandan Hegde**.

Place: Bangalore
Date:

Mr. Keerthiraj Hemanth Moger
P03ME21S0035

ABSTRACT

An "Online Food Ordering and Delivery Platform" is a web-based application that revolutionizes the way users order food from various restaurants and have it delivered to their doorstep. The platform provides customers with an easy-to-use and efficient solution for exploring restaurant options, selecting dishes, ordering, and tracking deliveries in real time. Our goal is to improve the overall ordering experience, provide users with convenience and choice, and help our restaurant partners reach a wider customer base. This website contains a comprehensive list of restaurants in various cities and regions. Users can search for restaurants based on location, cuisine, ratings, and other filters. An efficient search function with various filters allows users to quickly find specific restaurants and cuisines.

Contents

1 Introduction **1**

1.1 Problem Statement 2
1.2 Features: 2
1.3 Software Requirement Specification 3
1.3.1 Cost estimation of the project 3
1.3.2 Proposed Solution 4
1.3.3 Concrete and verifiable goals 4
1.3.4 Aim of the Project 4
1.3.5 Software Specification 4
1.3.6 Hardware Specification 5
1.3.7 Objective 5
1.3.8 Scope of the product 6
1.3.9 User Experience 7

2 Requirement Analysis **9**

2.1 Identification of need: 9
2.2 Feasibility Study: 10
2.2.1 Economical Feasibility 10
2.2.2 Technical Feasibility 11
2.2.3 Operational Feasibility 11
2.3 Preliminary Product Description: 11
2.3.1 Benefit to Organization 12
2.3.2 The Initial Cost 12
2.3.3 Running Cost 12
2.3.4 Need for Training 12
2.4 System Analysis: 13
2.5 Modules 14
2.5.1 Administrator module 14
2.5.2 User/Customer Module 14

3 Introduction to Technologies **17**
3.1 ReactJS . . . 17
3.1.1 Advantages of React.js: . . . 19
3.1.2 Disadvantages of React.js: . . . 19
3.2 Visual Studio . . . 20
3.3 HTML . . . 20
3.3.1 Features of HTML: . . . 20
3.3.2 Advantages of HTML: . . . 21
3.3.3 Disadvantages of HTML: . . . 21
3.4 CSS . . . 22
3.4.1 Features of CSS: . . . 22
3.4.2 Advantages of CSS: . . . 23
3.4.3 Disadvantages of CSS: . . . 23
3.5 JavaScript . . . 24
3.5.1 Features of JavaScript: . . . 24
3.5.2 Advantages of JavaScript: . . . 25
3.5.3 Disadvantages of JavaScript: . . . 25

4 Proposed System **27**
4.1 Proposed System Completion . . . 28
4.2 End Users: . . . 28
4.2.1 Restaurant Discovery: . . . 29
4.2.2 Online Food Ordering: . . . 29
4.2.3 Table Reservations: . . . 29
4.2.4 Ratings and Reviews: . . . 29
4.2.5 Collections and Recommendations: . . . 29
4.2.6 Online Payment Options: . . . 30
4.2.7 Membership Programs: . . . 30
4.2.8 Customer Support: . . . 30

5 System Design **31**
5.1 Usability Goals . . . 31
5.1.1 Client -side web (front-end) . . . 32
5.1.2 Graphic User Interface . . . 32
5.2 LML Diagrams: . . . 32
5.2.1 Actor . . . 32
5.2.2 Use case . . . 33
5.3 Usecase Diagrams: . . . 33
5.4 Admin Use Case Diagram . . . 35

5.5 User Use Case Diagram 36
5.6 User Flow Chart 37
5.7 Application Contents 40
5.8 Dataflow Diagram: 42

6 Implementation **45**
6.1 Implementation Issues & Challenges 45
6.1.1 User without background 45
6.1.2 The screen size of different device 46
6.1.3 Server performance 46
6.2 Components in Reactjs 46
6.2.1 Introduction to Components 46
6.3 Reactjs DOM manipulation 49
6.4 Development Tools 52
6.4.1 Web Technology 52
6.4.2 System Platform 53
6.4.3 Project Management Tool 53
6.4.4 Visual Paradigm Community Edition 53
6.4.5 Test Plan 53

7 System Testing **54**

8 Screenshots **56**

9 Challenges and Solutions **60**
9.1 Solutions and values: 62

10 Results **65**
10.1 Plan and Define Requirements 65
10.2 Design and Development 65
10.3 Database and Infrastructure 65
10.4 Payment Gateway Integration 65
10.5 Location-based Services 66
10.6 Ratings and Reviews System 66
10.7 Order Management and Delivery Tracking 66
10.8 Quality Assurance and Testing 66
10.9 Deployment and Launch 66
10.10Marketing and User Acquisition 67

11 System Strength and Limitation **68**
11.1 System Strength 68
11.2 System Limitation 68

12 Conclusion and Future work **69**
12.1 Conclusion: 69
12.2 Future Enhancement 70

Chapter 1

Introduction

The "Online Food Ordering and Delivery Platform" is a front-end web development project that focuses on creating a dynamic and user-friendly online food ordering platform. ReactJS, a popular JavaScript library known for its efficiency in building interactive user interfaces, is utilized to develop the front end of this application.

In today's fast-paced world, online food delivery systems have become increasingly popular, providing convenience and accessibility to customers seeking a variety of dining options.This project aims to cater to the growing demand for seamless food ordering experiences, where customers can explore multiple restaurants, browse diverse menus, customize their orders, and track deliveries in real-time. The front-end development of the application involves implementing key features such as restaurant listing and search functionalities, displaying restaurant menus, managing the user's shopping cart, providing real-time order tracking, and ensuring a responsive design that accommodates various devices. ReactJS allows to create highly interactive and responsive user interfaces that improve the overall user experience.

By focusing solely on front-end web development, this project lays the groundwork for a comprehensive "Online Food Ordering and Delivery Platform" that can be integrated with backends and databases in future iterations. The front-end interface aims to provide a seamless and pleasant user experience, allowing customers to easily order their favorite dishes from various restaurants at any time.

1.1 Problem Statement

The goal of the project is to create an application using interaction design while at the same time making the JavaScript library more responsive in programming. The purpose of this project is to develop an interactive and user-friendly application that allows customers to easily place orders and have their orders delivered to their designated location.

As web application standards continue to improve and technology requirements become more complex, frameworks have become an integral part of web development. Even assuming that everything can be reinvented, reinventing the wheel to achieve technology so advanced is utterly absurd. For this reason, using a framework recommended by thousands of developers around the world is a very sensible approach to building rich, interactive web applications.[1] Some frameworks include Vuejs, Angular, jQuery, and many more.[2] The most used framework today is React. React.js is an open-source JavaScript library used to build user interfaces. Used for view layer management for web and mobile apps[3]. React is used when developing single-page web or mobile applications because it is great for capturing rapidly changing data that needs to be recorded. React Virtual Dom (Document Object Model) greatly speeds up Dom operations and is very easy to learn, mainly thanks to his JSX syntax. React can be used server-side or client-side[4].

The goal of this project is to create an application using JavaScript libraries. This app is a food ordering site. Once an app is created, it is evaluated with a focus on usability, efficiency, and effectiveness goals.

1.2 Features:

The food ordering system will include the following features:

- User Registration and Authentication.
- Search and Filter functionality.
- Menu Display with categories and subcategories.
- Cart Management.
- Order Placement with delivery or pickup options.

- Payment Integration for online transactions.
- Real-time Order Tracking.
- Discounts and Offers.
- Order History and Reviews.
- Admin Panel for managing menu items, orders, and customer support.

1.3 Software Requirement Specification

The software requirements specification is generated at the end of the scan job. The functionality and performance assigned to the software within a system engineering framework are refined by establishing a complete informational description, a detailed functional and behavioral description, and an indication of the requirements. efficiency. performance and design constraints, compliance claims, and other claim-related data.

1.3.1 Cost estimation of the project

The cost of the software accounts for a small percentage of the total cost of the computer system. Several factors are taken into account and can affect the final cost of the software, such as the human, technical, and physical availability of the software, etc.

The main point to consider when estimating the cost of a project is its size. Although the software sizing feature is comprehensive, approximate dots and lines of code are also used to "size" each software and its price.

The cost estimates I make for the project also depend on baseline measurements collected in previous projects, and these measurements are used in conjunction with estimators to develop project allocation and allocation reports.

We have basically estimated this project mainly on two bases

- Estimated effort. This is the total number of man-hours required to develop the project. This even includes the time it takes to make documentation and manuals.
- Estimated Hardware Required: This includes the cost of the PC and the hardware required for the development of this project.

1.3.2 Proposed Solution

Create components for menu display, shopping cart management, order summary, and checkout. Global state management using API or Redux context. Collects menu items and simulates user commands. Implement routing for navigation. Use CSS or style elements for design. Payment management integration with Stripe is possible. Create functional tests. Deploy your application to a hosted platform. Backend customization and integration can be extended based on your specific needs in the future.

1.3.3 Concrete and verifiable goals

Therefore, Reacts should be designed with simplicity in mind, with all the necessary features while removing the unnecessary ones to provide a better user experience.

- Describe how to use React for web applications such as ordering food.
- Develop a food ordering application using React taking into account the principles of interaction design.
- Evaluate the developed application for usability.

1.3.4 Aim of the Project

The purpose of the development of the "Online Food Ordering and Delivery Platform" project is to replace the traditional ordering method with a computer system. Another main reason for developing this project was to produce job overview reports quickly and in a good format whenever needed. Online food ordering systems are very extensive. This Reactjs project can be used by restaurants and fast food customers to track their orders. This project is easy, fast and accurate. Less storage space required.

1.3.5 Software Specification

1. Front End: HTML, CSS, JavaScript, ReactJS.
2. Operating System: Windows, macOS, Linux
3. Web Browser: Google Chrome, Mozilla Firefox, Safari, or Microsoft Edge

1.3.6 Hardware Specification

1. Processor: Dual-core processor or higher
2. RAM: 4GB or higher
3. Storage: At least 50GB of free disk space

1.3.7 Objective

- To develop a responsive and intuitive user interface, ensuring easy navigation and seamless interaction for users exploring the platform.
- Implement a comprehensive restaurant listing feature that allows users to search for restaurants based on their preferences like cuisine type, rating, and location.
- View each restaurant's menu with different dishes, prices, and descriptions.
- Create an easy-to-use shopping cart system that allows users to add dishes from different restaurants, and review orders.
- Implementing real-time order tracking will allow users to monitor the status of their shipments from order confirmation to shipment completion.
- Make sure your application is designed with a responsive layout to provide the best user experience on different devices such as desktops, tablets, and mobile phones.
- Optionally implement user registration and authentication functionality using ReactJS to personalize the user experience and ensure privacy.
- Lays the groundwork for future integrations with backends and databases, enabling seamless communication with server-side functionality.
- Optimize front-end application speed and performance to ensure fast load times and smooth operation.
- Ensure application compatibility with major web browsers, improving accessibility for users on various platforms.

- Continuously improve the user interface and interactions based on user feedback to improve overall user satisfaction.

1.3.8 Scope of the product

This project focuses on Reactjs as a user interface framework related to human-computer design. Next, consider how to evaluate responsive applications against usability goals and design principles. Future work will also be included in this project.

The project scope for front-end web development of an online food delivery system using ReactJS includes designing and implementing an interactive and user-friendly user interface that allows customers to order food from different restaurants and track their delivery. It is included. The focus of this project was the user interface aspect of the application, responsible for displaying information and facilitating user interaction.

User Interface Design:

- Designing a responsive and visually appealing user interface with an intuitive layout for easy navigation and seamless user experience.

Restaurant Listing and Search:

- Implementing a comprehensive restaurant listing feature, showcasing essential restaurant details, such as cuisine type, ratings, and location. - Providing users with the ability to search for restaurants based on preferences like cuisine or proximity.

Menu Display and Customization:

- Displaying each restaurant's menu with a variety of dishes, prices, and descriptions. - Allowing users to customize their orders by selecting options like toppings, sides, and quantities.

Cart Management and Checkout:

- Creating a user-friendly shopping cart system that enables users to add dishes from different restaurants and review their orders. - Facilitating a streamlined and secure checkout process.

Real-time order tracking (optional)

-Optionally, implement real-time order tracking to notify users of their delivery status, from order confirmation to delivery.

User Authentication and Security (Optional)

- Optionally, integrating a secure user registration and login system to personalize user experiences and manage user data.

Responsive Design:

- Ensuring the application's responsiveness across various devices, including desktops, tablets, and mobile phones.

Cross-Browser Compatibility Testing:

- Testing the application on major web browsers to ensure compatibility and consistent user experience.

Performance Optimization:

- Optimizing the frontend code to improve loading times and overall performance.

The frontend web development project scope does not include backend functionalities, database management, or server-side operations. The focus is solely on building an efficient and engaging frontend interface using ReactJS to provide users with a seamless food ordering experience.

The successful completion of this frontend web development project will result in a functional and feature-rich frontend application, enabling users to explore restaurants, view menus, customize orders, and place food delivery requests with ease. The project's scope lays the foundation for further integration with backend systems to create a complete online food delivery system.

1.3.9 User Experience

User experience refers to how a product works and is used by people in the real world. It's about what people think about the product, how satisfied they are, and how satisfied they are with using the product. Many aspects of user experience can be considered and many ways to

take them into account when designing interactive products. It's usability, functionality, aesthetics, content, look and feel. [11] The ease of use of a product or service determines whether you can complete your task or achieve your goal with it. If you want people to use it often, it should be simple to use. If consumers have to spend a lot of time just understanding the system, they are likely to drop your program. Functionality refers to whether a design works and helps users achieve their goals and needs. When a design is highly functional, it gets the job done and does it successfully. Human-computer interaction, information architecture, efficiency, and effectiveness are important elements of interaction design, and design principles are used to decide how the application should be developed. The completed application will be evaluated for the overall website design, notable features that have been responded to and used in the developed application.

Chapter 2

Requirement Analysis

An "Online Food Ordering and Delivery Platform" in ReactJS requires comprehensive demand analysis to ensure its effectiveness. It will support multiple user roles, such as client and administrator. User authentication and authorization are essential to protect system access. The system needed an intuitive and responsive user interface for customer navigation and restaurant management.

2.1 Identification of need:

The old manual system had many drawbacks. The process of storing, maintaining, and retrieving information is cumbersome and time-consuming as the entire system must be maintained manually. Records are never used in any systematic order, which raises many issues related to specific transactions in specific contexts. If information is found that has to go through different registers, the document never survives, as is the case with reports. Importing and retrieving records is always an unnecessary waste of time. Another problem is that it is very difficult to find errors when importing records. Difficulty updating records after importing them.

The reason is that there is a lot of information that must be retained and remembered when running a business. For this reason, we have partially automated (computerized) the functions of the current system. The current system is very labor intensive and requires him to enter the same information in three different places.

The following points should be well considered:

- The new system provides documentation and reports and may also include some reports to aid administration. Decision-making and cost management, but these reports don't get the attention

they deserve, so reports and information of this kind are well defined and deserve the attention. Details of the information required for each document or report.

- Frequency and distribution required for each document.
- Possible sources of information for each document and report.
- The introduction of computerized systems solves the task of organizational record keeping. Most importantly, information retrieval is done with the click of a mouse. Therefore, the proposed system saves time in business operations and facilitates information flow by providing valuable information reports.

2.2 Feasibility Study:

After conducting a study on the project "Online food ordering and delivery platform" and analyzing all the existing or required functions of the system, the next task is to conduct a feasibility study of the project. All projects are possible, with unlimited resources and unlimited time.

A feasibility study involves examining all possible ways to come up with a solution to a given problem. The proposed solution must meet all user requirements and must be flexible enough that future changes can be easily made based on future requirements.

2.2.1 Economical Feasibility

- This is a very important aspect to consider when developing a project. We chose minimalism-based technology possible cost factor.
- All hardware and software costs must be borne by the organization.
- Overall, we feel that the benefits of organizing will receive from the recommendation system will definitely pass initial and subsequent operating costs of the system.

2.2.2 Technical Feasibility

This includes studying the features, performance, and constraints that can affect the ability to achieve an acceptable system. For this feasibility study, we studied the complete functionalities to be provided in the system as described in the System Requirements Specification (SRS) and checked if everything is possible. Use different types of user interface platforms.

2.2.3 Operational Feasibility

Undoubtedly, the proposed system is completely GUI based, and very user-friendly and all inputs should be considered even for a novice. In addition, proper training has been conducted to familiarize users with the nature of the system so that they feel comfortable with the new system. According to our research, customers feel comfortable and satisfied because the system has reduced their load and activities.

2.3 Preliminary Product Description:

The first phase of the system development life cycle is a preliminary investigation to determine the feasibility of the system. The purpose of the preliminary investigation is to assess the applicability of the project. It is not a design study or a set of details to describe the business system in all its aspects. Instead, the collection of information helps committee members assess the merits of the project application and make an informed assessment of the feasibility of the proposed project.

Analysts working on the preliminary investigation should accomplish the following objectives:

- Clarity and understanding of project requirements
- Determine the size of the project.
- Evaluate the costs and benefits of alternative methods.
- Determine the technical and operational feasibility of the alternatives approach.
- Report results to management, with recommendations justification for accepting or rejecting the proposal.

2.3.1 Benefit to Organization

The organization will obviously be able to gain benefits such as savings in operating costs, reduced paperwork, better utilization of human resources, and a better image that increases goodwill.

2.3.2 The Initial Cost

The initial cost to configure the system will include the cost of hardware-software (operating system software, utilities) & labor (configuration & maintenance). The same is subject to the organization. The "online food ordering and delivery platform" was developed to overcome common problems in the convenient manual system. This software is supported to eliminate and in some cases reduce the difficulties encountered by this legacy system. Moreover, this system is designed for the specific needs of the business to carry out operations smoothly and efficiently. The "online food ordering system" was developed to overcome common problems in the hands-on guidance system. This software is supported to eliminate and in some cases reduce the difficulties encountered by this legacy system. Moreover, this system is designed for the specific needs of the business to carry out operations smoothly and efficiently.

2.3.3 Running Cost

In addition, the initial cost of long-term costs will include system operating costs, including AMC, fixed costs, personnel costs, and related software update/refresh costs.

2.3.4 Need for Training

To keep the system running smoothly, users and administrators need to be trained when the system is installed. Make training locations available to clients. We spoke with the manager who manages the center's financial operations, the staff who keep records in many of the register books, and the reporting manager about the existing system, their requirements, and expectations for the new proposed system. We then performed a system survey of the entire system based on the customer's requirements and the additional features they wanted to incorporate into their system. Even without this proposed system, reliable, accurate, and secure data would be a complex task, as there

was no record to track all activities performed by Kaybus' online food ordering and delivery platform. It was thought that there was. The new system that I have proposed and developed will make the organization's work easier. Help staff create the necessary reports to help them track progress and services.

All important activities to be performed are computerized through this system, greatly simplifying administrative tasks.

2.4 System Analysis:

System analysis gathers and interprets facts, diagnoses problems, provides information about your online grocery ordering system, and recommends system improvements. This is a problem-solving activity that requires focused communication between system users and system developers. System analysis or research is an important stage of any system development process. The system is inspected and analyzed down to the last detail. Systems analysts act as interrogators, digging deep into how current systems work. The system is considered as a whole and the inputs to the system are determined. An organization's services are linked to various processes. Systems analysis involves identifying problems, identifying relevant variables and decisions, analyzing and synthesizing various factors, and determining optimal or suboptimal solutions or programs of action. Most satisfying. A detailed investigation of the process should be performed using various techniques such as interviews, questionnaires, etc. The data collected from these sources should be carefully considered before reaching any conclusions. The bottom line is understanding how the system works. This system is called the existing system. Existing systems are now being thoroughly investigated to identify problem areas. Designers act as problem solvers and try to solve the problems facing the company. Solutions are provided in the form of suggestions. The proposals are then analyzed analytically based on existing systems to select the best one. The proposal is sent to the user for their approval. Suggestions are reviewed based on user requirements and changes are made accordingly. This is a loop that ends when the user is satisfied with the suggestions. Preliminary research is the process of gathering and interpreting facts using information for further study of the system. A preliminary investigation is a problem-solving activity that requires focused communication between system users and system de-

velopers. He carries out various feasibility studies. These studies can provide an approximation of system behavior from which decisions can be made about the strategies to follow for effective system study and analysis.

2.5 Modules

- Administrator module
- User/Customer Module

2.5.1 Administrator module

Admin can view all the information about the user edit the all details about the customer.

- Create food category
- Manage food categories
- Add food item
- Manage Food item
- Manage user order

2.5.2 User/Customer Module

This functionality provided:

- View product's list
- Register
- Place orders

Admin Module

- Dashboard: In this section, admins can easily see all the details like the Total number of orders, unconfirmed orders, confirmed orders, and the total number of dishes prepared. Total food pickup.
- Total food deliver Total Cancelled orders and Total user.
- Reg Users: In this section, the admin can manage registered users (view/update).

- Food Category: In this section, admin can manage the food category (Add and Update),
- Food Menu: This section allows admins to manage (add and update) food menus.
- Onders: In this section, the administrator can view the details of the grocery order and also has the right to change the order status depending on the current status.
- Reports: This section allows administrators to view order details, order counts, and sales reports by date. Admins can also update profiles, change passwords, and recover passwords.

User module

Food menu In this section, users learn what kind of food is available in restaurants

- My account: This section allows users to enter their password, view and update their profile, and log out of their account.
- My order: This section allows users to view their history after logging in.
- Cart: This section allows users to add dishes they would like to order.
- Users can also download the invoice and cancel the order if they wish.

Brief Information about homepage

On this page, the guest user (unregistered user) can view the restaurant menu, and search for dishes based on dish name and guest user can also register at a restaurant and the registered user can login. Users can also track their orders without logging in.

This is the first page a user opens on your website. The home page has a sidebar menu option that allows users to access anything they need in the sidebar. As the user scrolls through the house, it not only shows different products but also videos of him demonstrating certain foods. Users can add desired grocery items to their shopping

cart. At the bottom of the page is a footer with the restaurant's location and contact details. We display all products on our site. The details submitted include the food product name, food image, description, and price. Figure 5 shows what the website looks like and what components are integrated into the home page.

Food

This page contains a list of food products available in the restaurant. The details in the product include The Food Name, Description,and Price.

Blog page

This page talks about foods and their nutritional values.

Order Now

In this page, all the user needs to fill the Name, Email, Telephone number and delivery address.

My orders

This page contains products added to the Cart that a user desires to purchase. The information about the products includes Food Name, Description and Price.

Footer

The footer contains the site's location and contact details, as well as links. Users can clearly understand how the application is built even without knowing how to program. It provides simplicity of available information and shows how structures fit together and interact with each other. In an architectural design example above, i.e. pages (Feed, Blog and many others) are called components in reacts, but any user can see them as a connected structure for smooth operation.

Chapter 3

Introduction to Technologies

3.1 ReactJS

React is a JavaScript library for rendering user interfaces (UI). A user interface is made up of small units such as buttons, text, and images. React lets you combine them into reusable and nestable components. Anything from a website to a phone app can be split into on-screen components. In this chapter, you will learn how to create, customize and conditionally display React components.

Specific functionality in React that is so suitable to access for this web application.

- React makes it easier to create dynamic web applications by requiring less coding and providing more functionality, unlike JavaScript, which often quickly becomes complex to code.
- Build web applications faster because React uses a virtual DOM. Instead of updating all components again like traditional web applications, the virtual DOM compares the previous state of the components and only updates the elements in the real DOM that have changed.
- React is easy to learn because it combines basic HTML and JavaScript concepts with some useful extras.
- Components are the building blocks of React applications, and an app typically consists of multiple components. You can use these components again and again in your Reactjs application[6]
- React Developer Tools are extensions for Chrome which is created by Facebook. This can be used to debug react applications faster and easier. Some notable features for Reactjs are:

- Declarative – Follows the declarative programming paradigm. This makes the program easier to read and omits low-level details. Written code is easier to think about because it is more abstract and describes the solution rather than the method.[7]

- Components - Independent reusable pieces of code. They serve the same purpose as JavaScript functions, but they work alone and return HTML.

  Components come in two types, Class components, and Function components

- Functional Components - Functional components are also returned to their HTML and behave like a class component. But functional components can be written with much less code easier to understand

- • Class-based component - A class component must contain: Extend the React.Component statement. This statement allows Extends React. Component and provides access to components process of reaction. Component functionality. components too require requires a Render() method. This method returns HTML.

- Virtual DOM – instead of manipulating the browser's DOM view. In fact, React creates a virtual DOM in memory. Do everything you need to do before making changes to browser DOM.

- Lifecycle Methods - Every component in React has a lifecycle. It can be monitored and manipulated during three main phases. These phases are Mounting, Updating, and Unmounting.

  - Mounting means putting elements into the DOM.
  - A component is updated whenever there is a change in the component‘s state or props.
  - Unmounting is when a component is removed from the Dom.

- JSX – JSX stands for JavaScript XML. JSX allows us to write HTML in React. JSX makes it easier to write and add HTML in React.

- React Hooks - Hooks allow function components to have access to state and other React features. Because of this, class components are generally no longer needed.[8]

- Architecture beyond HTML – The basic architecture of React applies beyond rendering HTML in the browser. For example, Facebook has dynamic charts that render to tags,[12] and Netflix and PayPal use universal loading to render identical HTML on both the server and client.[9]

Material-UI, the React component library based on Google Material Design, allows for faster and easier stylized web development. With basic React framework familiarity, you can build a deliciously material app with MaterialUI.[10]

3.1.1 Advantages of React.js:

- Reusability and Maintainability: React's component-based architecture encourages code reusability, resulting in easier maintenance and more organized codebases.
- Performance: React's Virtual DOM efficiently updates the actual DOM, reducing unnecessary re-renders and resulting in better overall performance.
- Developer Efficiency: JSX allows developers to write UI code in a more intuitive and familiar manner, resulting in increased developer productivity.
- Community and Documentation: React has a large and active community, providing extensive documentation, tutorials, and open-source libraries, which facilitates learning and problem-solving.
- Testability: React's component-based nature makes it easier to write unit tests for individual components, ensuring code quality and application stability.
- React Native: React can be used with React Native to build mobile applications for both iOS and Android, leveraging a single codebase for cross-platform development.

3.1.2 Disadvantages of React.js:

- Learning Curve: For developers new to React or component-based architectures, there can be a learning curve to understand the concepts and best practices.

- Tooling Complexity: Setting up and configuring the development environment, build tools, and bundlers may require additional effort and understanding.
- Verbosity: JSX code can sometimes become verbose, especially for more complex components, which may lead to increased file sizes.
- Frequent Updates: React.js and its ecosystem undergo regular updates and changes, requiring developers to keep up-to-date with the latest practices and migration strategies.
- Steep Initial Setup: For small projects, setting up a React.js environment might be overkill, and simpler frameworks or libraries may be more suitable.

Overall, React.js is a powerful and efficient library that offers several advantages, especially when building complex and interactive user interfaces. While it may have a few downsides, its large community and extensive ecosystem provide solutions to many of its challenges, making it a top choice for web developers.

3.2 Visual Studio

Visual Studio Code is a lightweight and powerful source code editor that runs on your desktop and is available for Windows, macOS, and Linux. Provides built-in support for JavaScript, TypeScript, Node.js, and a rich extensibility ecosystem for other languages (C, C#, Java, Python, PHP, Go, etc.) and runtimes (.NET, Unity, etc.)[15]

3.3 HTML

HTML, which stands for HyperText Markup Language, is the standard markup language used to create and structure web pages. It forms the backbone of web page content by defining the structure and layout of elements on a page.

3.3.1 Features of HTML:

- HyperText: HTML supports hyperlinks, allowing users to navigate from one web page to another easily.

- Markup Language: HTML uses tags to define the structure of content, such as headings, paragraphs, lists, images, etc.
- Platform Independence: HTML is platform-independent, meaning it can be viewed and accessed from various devices and operating systems.
- Versatility: HTML can be combined with other technologies, like CSS (Cascading Style Sheets) and JavaScript, to create dynamic and interactive web pages.
- Compatibility: HTML is compatible with different web browsers, ensuring that web pages look consistent across various platforms.
- Scalability: HTML is designed to be scalable, allowing developers to create simple static pages or complex, dynamic web applications.

3.3.2 Advantages of HTML:

- Ease of Use: HTML is relatively easy to learn and use, making it accessible to beginners and experienced developers alike.
- Widely Supported: HTML is supported by all major web browsers, making it a universal language for web development.
- SEO-Friendly: Properly structured HTML can help improve a website's search engine ranking, leading to increased visibility and traffic.
- Accessibility: HTML provides semantic elements that can be used to improve accessibility for users with disabilities, enhancing the overall user experience.
- Free and Open Standard: HTML is an open standard, which means it is freely available for everyone to use and modify.

3.3.3 Disadvantages of HTML:

- Limited Styling: HTML alone doesn't provide advanced styling capabilities. For layout and presentation, CSS is required, which may increase the complexity of web development.

- Lack of Interactivity: HTML is static by nature, lacking advanced interactivity and functionality. To add dynamic features, developers often need to use JavaScript or other technologies.
- Versioning and Compatibility: Different versions of HTML have been released over the years, leading to compatibility issues between older and newer browsers.
- Security Concerns: HTML can be vulnerable to various web attacks like cross-site scripting (XSS) if not handled carefully.
- Learning Curve for Advanced Features: While HTML basics are easy to grasp, mastering advanced features and best practices might require more effort and time.

3.4 CSS

CSS, which stands for Cascading Style Sheets, is a style sheet language used to control the presentation and layout of HTML documents. It is an essential part of modern web development and works in conjunction with HTML to enhance the visual appeal and interactivity of web pages.

3.4.1 Features of CSS:

- Separation of Content and Presentation: CSS allows developers to separate the content (HTML) from the presentation (styling), making it easier to maintain and update web pages.
- Cascading Style Rules: CSS follows a cascading model, where styles are applied in a hierarchical manner. This allows for easy and efficient style inheritance and overrides.
- Selectors: CSS provides various selectors to target specific HTML elements and apply styles selectively, giving developers fine-grained control over the appearance of elements.
- Media Queries: CSS supports media queries, enabling the creation of responsive designs that adapt to different screen sizes and devices.

- Flexibility: CSS offers a wide range of styling options, including colors, fonts, spacing, borders, animations, and more, providing the flexibility to customize the look and feel of web pages.
- External Style Sheets: CSS allows styles to be defined in external files, promoting code reusability and consistency across multiple web pages.

3.4.2 Advantages of CSS:

- Consistent Styling: By using CSS, developers can ensure consistent styling throughout a website, making it easier to maintain a cohesive brand identity.
- Page Load Speed: Separating style from content reduces the size of HTML files, leading to faster page load times and improved user experience.
- Responsive Design: With media queries, CSS enables the creation of responsive and mobile-friendly websites, catering to users on various devices.
- Code Reusability: External style sheets facilitate code reusability, as the same styles can be applied across multiple pages, reducing redundancy and enhancing code maintainability.
- Browser Compatibility: CSS ensures consistent rendering of styles across different browsers, streamlining the development process and ensuring a more reliable user experience.

3.4.3 Disadvantages of CSS:

- Complexity: CSS can become complex, especially in larger projects or when dealing with browser-specific quirks, leading to potential difficulties in debugging and maintenance.
- Performance Impact: Extensive use of CSS, especially for animations and transitions, can impact website performance, causing slower rendering times.
- Learning Curve: Mastering CSS and understanding its advanced features may require significant effort and time, particularly for beginners.

- Browser Support: Older browsers may not fully support modern CSS features, leading to inconsistencies in the rendering of styles and requiring fallback solutions.
- Specificity Issues: The cascading nature of CSS can sometimes lead to specificity conflicts, where styles compete for precedence, potentially causing unexpected results.

3.5 JavaScript

JavaScript is a versatile and widely used programming language that enables dynamic and interactive web development. It is mainly used for front-end development, but with Node.js it can also be used for back-end development.

3.5.1 Features of JavaScript:

- High-level Language: JavaScript is a high-level language, meaning it is designed to be easy to read and write for developers.
- Interactivity: JavaScript allows developers to create dynamic and interactive web pages, enabling features like form validation, animations, and user input processing.
- Event-Driven: JavaScript is event-driven, responding to user actions and interactions, such as mouse clicks, keystrokes, and touch events.
- Asynchronous Execution: JavaScript supports asynchronous programming, enabling non-blocking operations like fetching data from servers without freezing the entire application.
- Cross-platform: JavaScript can be executed on various platforms, including web browsers and servers (using Node.js), making it a versatile language for full-stack development.
- Rich Ecosystem: JavaScript has a vast ecosystem of libraries and frameworks (e.g., React, Angular, Vue.js) that enhance development efficiency and extend its capabilities.

3.5.2 Advantages of JavaScript:

- Client-Side Interactivity: JavaScript significantly enhances the user experience by adding interactive elements to web pages, reducing the need for frequent server requests and page reloads.
- Speed: As a client-side language, JavaScript executes instantly in the user's browser, reducing server load and improving the overall speed and responsiveness of web applications.
- Versatility: JavaScript can be used for various purposes, such as building web applications, mobile apps (using frameworks like React Native), and server-side applications (with Node.js).
- Community and Resources: JavaScript has a vast and active developer community, providing extensive documentation, tutorials, and support for developers.
- Easy Integration: JavaScript can be easily integrated into HTML and CSS, making it simple to add interactivity and dynamic behavior to existing web pages.
- Cross-browser Compatibility: Modern JavaScript frameworks and libraries help ensure cross-browser compatibility, making it easier to build consistent applications.

3.5.3 Disadvantages of JavaScript:

- Security Risks: JavaScript can be susceptible to security vulnerabilities like cross-site scripting (XSS) attacks if not handled properly.
- Browser Support: While modern browsers support JavaScript, some older versions or less popular browsers may have limited or inconsistent support for certain features.
- Performance Overhead: JavaScript execution can consume significant resources, leading to performance issues, especially in complex applications or on low-powered devices.
- Learning Curve: For beginners, JavaScript's asynchronous and event-driven nature can be challenging to grasp, making it relatively harder to learn compared to simpler languages.

- Code Maintainability: As JavaScript codebases grow, they may become challenging to maintain due to its flexibility and potential for poorly structured code.
- Dependency Management: Working with extensive JavaScript libraries and frameworks can introduce dependency management challenges, especially with versioning and updates.

Chapter 4

Proposed System

This online application allows end-users to skip check-in lines, select food from e-menus, read e-menus, and order groceries online. The user simply selects the desired food item. Select a dish from the menu card and see the results directly in your restaurant management screen. Using this application reduces the load on the server and may even abort the job. The advantage of this is that users can use the app to order food directly from the chef online, as the waiters may not be available if the restaurant is full. To register, users receive a username and password.

"Online Food Ordering and Delivery Platform" System Proposal: The purpose of the proposed system is to develop an improved equipment system. The proposed system can overcome all limitations of existing systems. This system provides sufficient security and reduces manual effort.

- Security of data
- Ensure data accuracies.
- Proper control of the higher officials.
- Minimize manual data entry.
- Minimum time needed for the various processing.
- Greater efficiency.
- Better service.
- User-friendliness and interactive minimum time required.

4.1 Proposed System Completion

The proposed system is designed and developed to address all the issues outlined in the first chapter of this report. First, the developed system solves the difficult problem of order tracking and provides the functionality to achieve the project's goal of preventing fraudulent listings. Instructions. By operating the system, staff can place orders on the system. The system automatically queues food orders on a first-come, first-served basis, and kitchen staff monitor the food queues and serve customers accordingly. can provide service. Plus, it eliminates all the manual processes associated with traditional methods of delivering grocery orders. In addition, a system has been developed that allows administrators to update all information about the court as required. This feature helps restaurants remove duplicate menu tags that contain incorrect information, and allows staff and customers to view up-to-date menu information using this feature.

In this way, restaurants can help overcome the difficulty of updating mind map information, providing culinary updates, and managing the potential increase in system operating costs. Fixed updates will be automatically updated by the system. by the manager. In addition, the project also achieved the goal of ease of use for employees and consumers by reducing the system workload by allowing consumers to view all updates through their mobile phone client. Handmade by restaurant staff. Ultimately, the project's goal of helping restaurants with pre-planning was achieved as the system enabled managers to create different types of reports to help with restaurant pre-planning. Managers can analyze the generated reports to plan the restaurant's next operations and improve the restaurant's operational efficiency. In summary, this system achieved all important results in accordance with all mentioned issues and project goals.

4.2 End Users:

"Online Food Ordering and Delivery Platform" is a popular food delivery and restaurant discovery platform that connects users with local restaurants and provides various features to enhance the dining experience. As an end user of this app, user can enjoy several benefits and functionalities:

4.2.1 Restaurant Discovery:

The "Online Food Ordering and Delivery Platform" website allows you to explore a wide range of restaurants in your vicinity. You can search for restaurants based on cuisine, location, popularity, and user reviews. The app provides detailed information about each restaurant, including menus, photos, ratings, and reviews, helping you make informed choices.

4.2.2 Online Food Ordering:

One of the primary features of this app is its food delivery service. You can browse through menus, select your desired dishes, customize orders, and place them for home delivery. The app provides real-time order tracking, ensuring you know the status of your delivery.

4.2.3 Table Reservations:

This app enables users to book tables at their preferred restaurants directly through the app. You can check the availability, select a suitable time slot, and reserve a table for dining in. This feature is particularly useful during peak hours or for special occasions.

4.2.4 Ratings and Reviews:

This app encourages users to share their dining experiences by allowing them to rate and review restaurants. As an end user, you can read reviews from other customers, giving you insights into the quality of food, service, ambiance, and overall dining experience at various establishments.

4.2.5 Collections and Recommendations:

"Online Food Ordering and Delivery Platform" curates collections of restaurants based on themes, such as "Best Romantic Restaurants" or "Budget-Friendly Eateries," making it easier for users to discover new dining options. The app also offers personalized recommendations based on your past preferences and browsing history.

4.2.6 Online Payment Options:

This app provides secure and convenient online payment options, including credit/debit cards, net banking, and digital wallets. This feature allows you to complete transactions seamlessly within the app.

4.2.7 Membership Programs:

This website offers membership programs like Food Gold, which provides exclusive deals, discounts, and complimentary dishes at partner restaurants. These programs provide additional benefits and savings for regular users.

4.2.8 Customer Support:

In case of any queries, issues, or concerns, It provides customer support through various channels, including in-app chat support, email, and helpline numbers. You can reach out to them for assistance regarding orders, payments, or general app-related inquiries.

Overall, as an end user of this website, you have access to a comprehensive range of features and services that make it convenient to explore, order from, and engage with local restaurants.

Chapter 5

System Design

Design is the first step in the development stage of any technology or principle intended to define a device, process or system in sufficient detail to enable it to be realized. Once software requirements have been analyzed and determined, software design involves designing, coding, implementing, and testing the three technical activities necessary to build and verify software. Design activities are very important in this phase because it is during this phase that decisions are made that ultimately affect the success of the software implementation and the discussion of software maintenance. These decisions ultimately affect system reliability and maintainability. Only through design can a customer's requirements be accurately translated into a complete software or system. Quality is driven by design. Software design is the process of transforming requirements into a software representation. Software design he does in two phases. In the preliminary design, requirements are converted into data.

5.1 Usability Goals

It refers to the usefulness and ease of use of a product or system. When we design a product, we consider usability goals to ensure that the product provides the expected solution to the identified problem [13]. These usability goals are:

- Effectiveness
- Efficiency
- Easy to learn
- Safety

- Good utility
- Memorability

5.1.1 Client -side web (front-end)

In web development, the term "client-side" refers to everything you see in your web application. This happens on the client his machine. This includes what the user sees, such as text, images, videos, and other user interface, and actions your app takes in the user's browser. HTML, CSS, and JavaScript work as a client-side browser. Additionally, many developers are integrating client-side processes into their application architecture, moving from server-side execution to regular client-side execution in their modern web applications. Client-side processes are most often written in JavaScript. The client side is also called the user interface, but the two terms are not synonymous. "Client-side" only refers to where the process runs, whereas "UI" refers to the kind of process that runs on the client-side.[14]

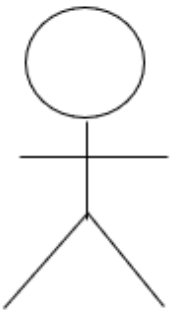
5.1.2 Graphic User Interface

A GUI (graphical user interface) is a system of interactive visual components for computer software. The graphical interface displays objects that convey information and represents actions that the user can perform. Objects change color, size, or visibility as the user interacts with them. GUI objects include icons, sliders, and buttons. These graphic elements are sometimes embellished with audio or visual effects such as transparency and shading. GUIs are considered more user-friendly than text-based command line interfaces, such as the shell of Unix-like operating systems. The GUI was first developed at Xerox PARC by Alan Kay, Douglas Engelbart and a group of other researchers in 1981. Apple then introduced the Lisa computer with a GUI on January 19, 1983.[16]

5.2 LML Diagrams:

5.2.1 Actor

A coherent set of roles that users of use cases play when interacting with the use cases.



5.2.2 Use case

A description of the sequence of actions, including variants, that a system performs that yields an observable result of the value of an actor.

UMI stands for Unified Modeling Language UML is a language for specifying, visualizing, and documenting systems. This is the step in the development of any product after analysis. The goal is to create a model of the entities involved in the project that will then have to be built. Representation of entities that must be used in the product under development must be designed.

There are various kinds of methods in software design: They are as follows

- Use case Diagram
- Sequence Diagram
- Collaboration Diagram
- Activity Diagram
- State chat Diagram

5.3 Usecase Diagrams:

A use case is a description of a set of action sequences. Graphically, it appears as an ellipse with a solid line consisting only of its name. A

use case diagram is a behavior diagram that shows a set of use cases and their actors and relationships. It's a mix of use cases and actors. An actor represents a real-world object. Featured sender. Supporting Actor Benefits. The purpose of a use case is to define consistent behavior without revealing the internal structure of the system. A use case typically represents a sequence of interactions between the user and the system. These interactions consist of a sequence of main lines representing common interactions between the user and the system. The use case model is an important analysis and design artifact (task). Use cases can be represented by drawing a use case diagram and writing extended accompanying text on the figure.

In the use case diagram, each use case is represented by an ellipse with the name of the use case written inside the ellipse. All the ellipses of the system are enclosed within a rectangle which represents the system boundary. The name of the system being modeled appears inside the rectangle. The different users of the system are represented by using a stick person icon. The stick person icon is normally referred to as an Actor. The line connecting the actor and the use cases is called the communication relationship. When a stick person icon represents an external system it is annotated by the stereotypes external system.

In the use case diagram, each use case is represented by an ellipse with the use case name written inside the ellipse. All ellipses in the system are surrounded by rectangles that represent the bounds of the system. The name of the modeled system appears inside the rectangle. Different users of the system are represented by stick figure icons. Stick figure icons are usually called actors. The lines connecting actors and use cases are called communication relationships. If the stick figure icon represents an external system, it is tagged with the external system stereotype. Use case diagrams to model the behavior in your system and help developers understand user needs. Stick figures represent so-called actors Use case diagrams to help you get an overview of your system and clarify who can do it and, more importantly, what they can't do. A use case diagram contains use cases and actors and shows the interactions between use cases and actors. The purpose is to show the interaction between use cases and actors. Express system requirements from the user's perspective. An actor can be an end-user of a system or an external system.

5.4 Admin Use Case Diagram

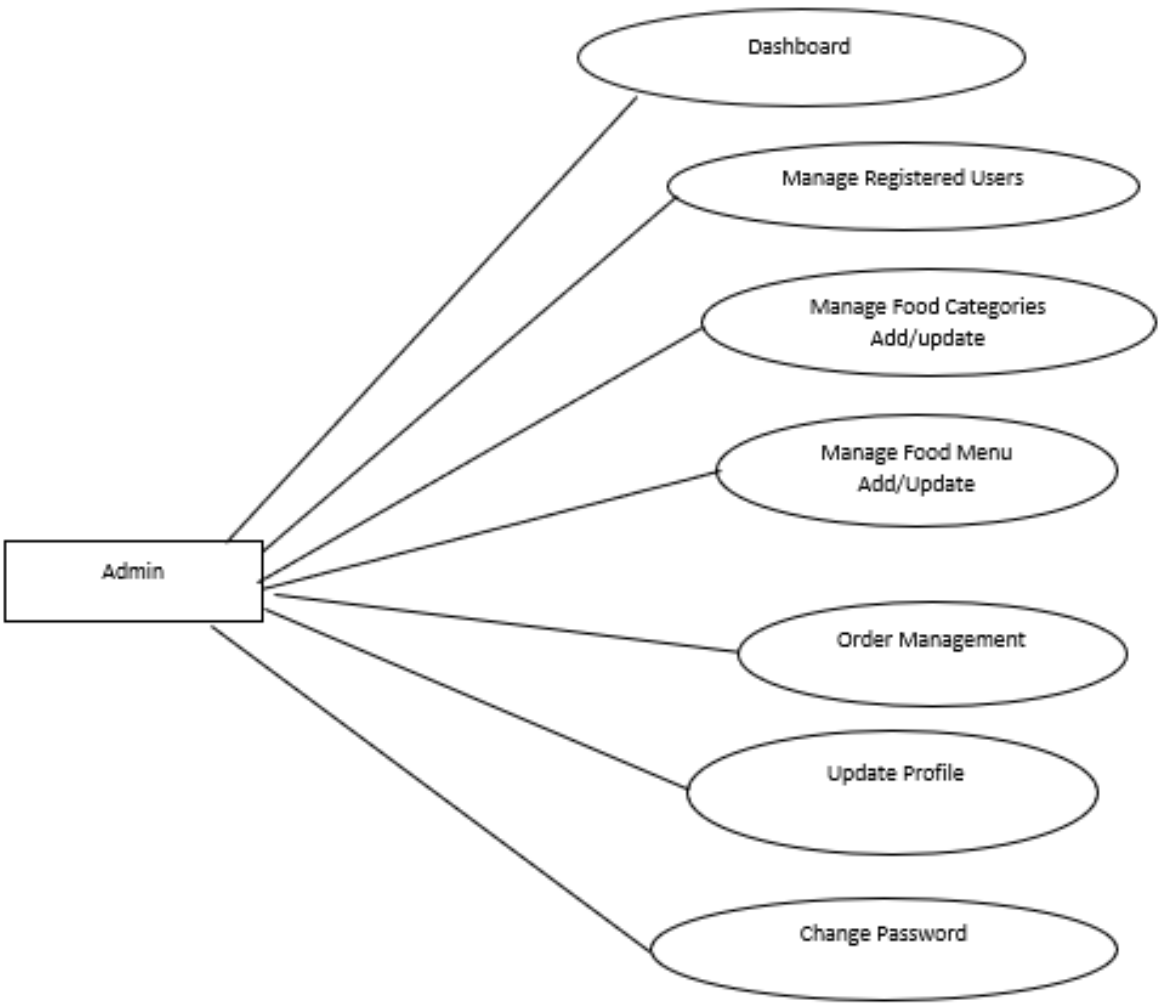


Figure 5.1: Admin Use Case Diagram

5.5 User Use Case Diagram

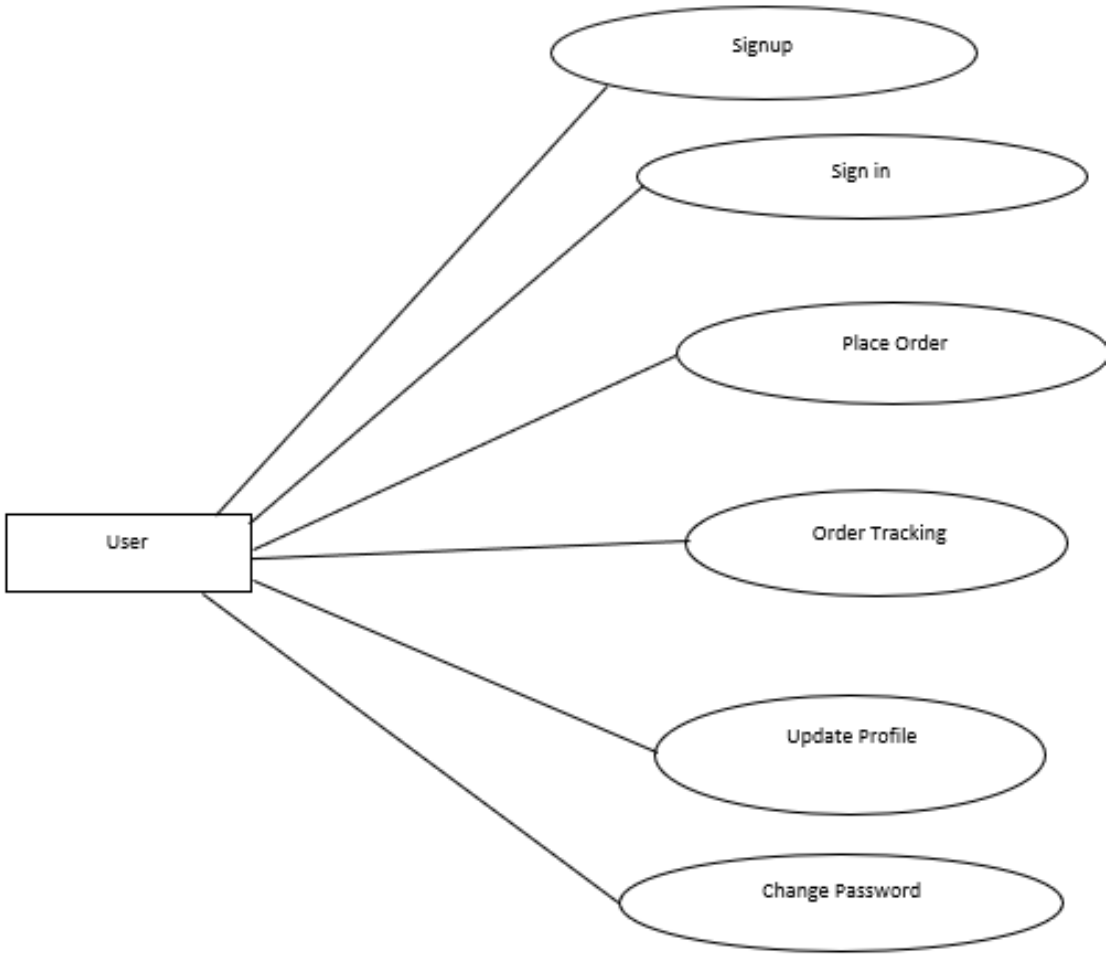


Figure 5.2: User Use Case Diagram

5.6 User Flow Chart

The site can be described by the stages a user takes in the web application. These steps are taken into consideration when building the project and for others to be able to understand the process with ease.

In order to place a food order, the user has the following options

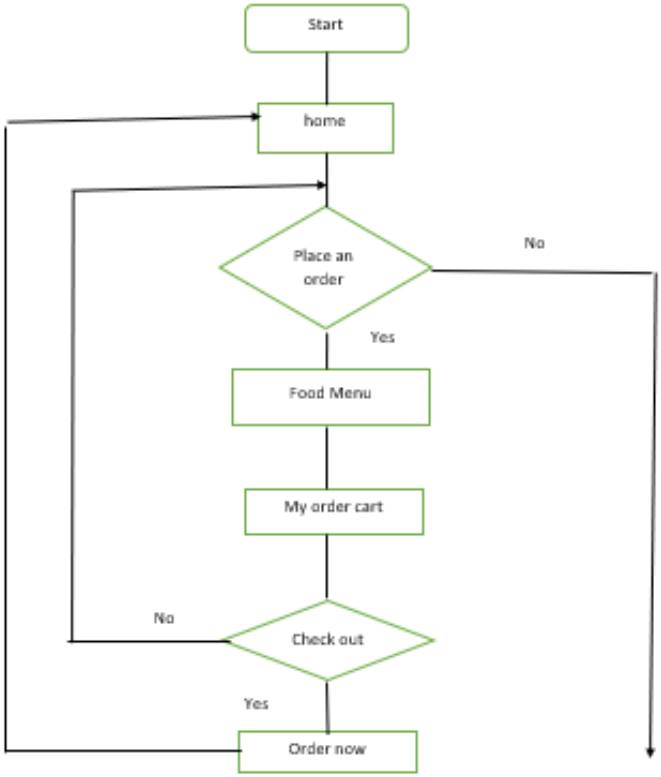


Figure 5.3: The flow chart of the application when ordering food

Home

This is the first page the user lands into when he or she opens the website. The home page has a sidebar Menu option where a user can get access to anything on the sidebar that he pleases. Figure 6 present when the chart will begin and shows the necessary steps to be taken in order to make a purchase or cancel it. As the user scrolls through the home page, there are a variety of products show cased as well as a video demonstration of some food varieties. The user has an option of "Adding to Cart" the food he pleases.

Food Menu

Home, can one navigate to food menu where their varieties of foods to choose from. Figure 8.3 shows the next main steps the user is willing to take.

Cart

Once products are added to the cart, the user can navigate to the "Wishlist" page to check the list of selected items in the cart. The user has an option of removing any undesired product.

Order Now

On satisfaction of the selected products, the user can continue to finally place the order. He will be required to fill in the name, email and delivery address then finally complete the order. In Figure 8.3, the order now is the last step for the user to complete his or her purchase.

When a user wants to read about the weekly websites offers, blog posts and promotional services, the user navigates through the following process.

Home

This is the first page the user lands into when he/she opens the website. The home page has a sidebar Menu option where a user can access to anything on the sidebar that he pleases. Figure 8.4 clearly shows the beginning stages.

Sidebar

Sidebars are options where to go through different pages on the web application. The user can either go back to homepage or continue to the blog page.

Blog

Figure 8.4 shows the user has been able to arrive at the right place, user can then read more about the restaurant in the Blogs page In the Side Bar Menu. To navigate to the food varieties in the website, a user can easily access them through this step.

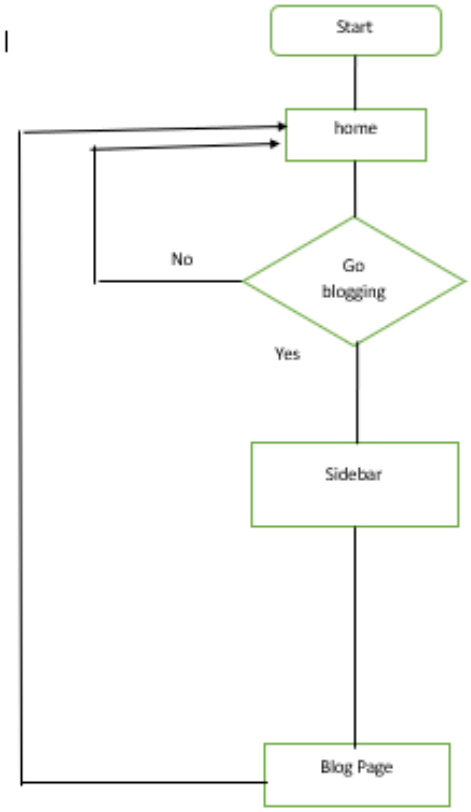


Figure 5.4: Flow chart to access blog page

Home

This is the first page the user lands into when he/she opens the website. The home page has a sidebar Menu option where a user can get access to anything on the sidebar that he or she pleases.

Sidebar

Sidebars are options where to go through different pages on the web application. The user can either go back to homepage or continue to the blog page. Figure 8.5 shows the next step to take in order to reach the users interest page.

Food page

In this page, user can select the food of their choice. In Figure 8.5, the page shows where user can choose varieties of foods.

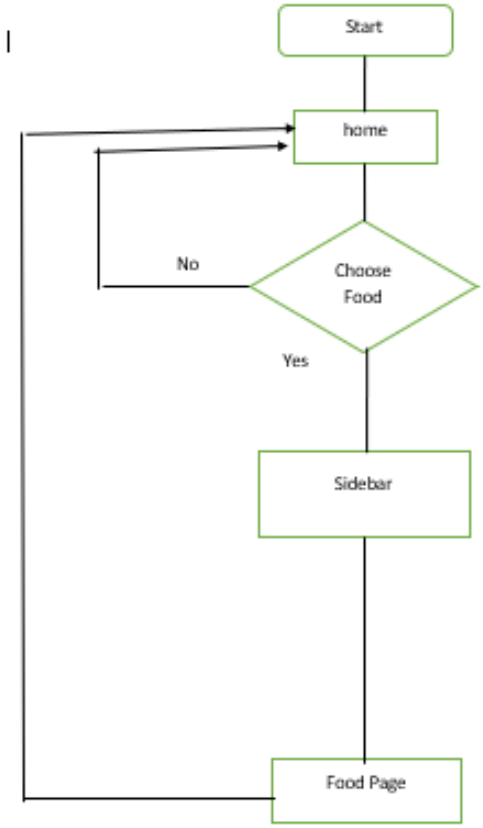


Figure 5.5: This flow chart is to order varieties of food

5.7 Application Contents

The target of the web application is people from all walks of life and must find the information needed on the platform with ease. The knowledge needed to build or have these were from the answers from people who were interviewed. Each users have their opinions but the majority or those with clear point of view were used to build the application. Design principles (effectiveness and efficiency) were also included when building the application. Each content are components on its own and by bring it together makes the web application. These contents are:

Navbar

A navigation system enables visitors to find content by searching and browsing can not only improve the chances of visitors browsing your site longer — it can also improve the chances of them taking action on

your site.On a website, a navigation menu is an organized list of links to other webpages, usually internal pages. Navigation menus appear most commonly in page headers or sidebars across a website, allowing visitors to quickly access the most useful pages [22]. The Navbar has a heading, cart and an icon(sidebar) at the right-hand side. Clicking the icon will send you to a different page and that will be the sidebar. It is simple without submenus and it is for every user can simply go through and find any information they need without any worries.

Sidebar

The sidebar is also called hamburger navigation menu. The sidebar consistent of pages where the user is maybe interested to view or want know more about. Such page on the application is food and that where food varieties are located.

Products

This is components is collections of products. The products are is where each product has details for the user to gain enough information. Each product on the have details like image, name, description, price and many more.

Featured

This is components that includes video, a button and a description. This will also attracts users and want to browse more on the site.

Cart

This component is where foods which are placed by the user waiting to be ordered. It is where products are stored and ready to be processed to next step, that is payment or cancel order.

Footer

This component is where the user gets all the necessary details about the website, this information is opening hours, maps, contact us, quick links, telephone and many more

5.8 Dataflow Diagram:

The data flow diagram is the starting point of the design phase and functionally decomposes the requirements specification. A DFD consists of a series of bubbles connected by lines. Bubbles represent data transformations and lines represent data flow in the system. DFD describes what data is flowing, not how it is processed. Therefore, hardware, software and data structures are not considered. A data flow diagram (DFD) is a graphical representation of the "flow of data" through an information system. DFDs can also be used to visualize data processing (structured design). Data Flow Diagram (DFD) is an important modeling technique for analyzing and structuring information processes. DFD literally means a diagram that describes the flow or movement of information in a process. DFD maps this information flow within the process based on inputs and outputs. A DFD can be called a process model. A data flow diagram is a graphical description of data in a system, and the way data is processed and transformed is called a data flow diagram (DFD).

Unlike detailed flowcharts, DFDs do not provide detailed descriptions of the data in the system and the modules that graphically describe how the data interacts with the system. The number of symbols in the data flow diagram and the following symbols are from DeliMarco.

There are seven rules for construct a data flow diagram.

- Arrows should not cross each other.
- Squares, circles and files must wears names.
- Decomposed data flows must be balanced.
- No two data flows, squares or circles can be the same names.
- Draw all data flows around the outside of the diagram.
- Choose meaningful names for data flows, processes & data stores.
- Control information such as record units, password and validation requirements are not penitent to a data flow diagram

Additionally, DFDs can be used to visualize data processing and structured design. This basic DFD can be decomposed into lower-level diagrams showing smaller steps and details of the system being

modeled. In a DFD, data items are transferred from an external data source or internal data store to an internal or external data store through an internal process. I usually start by drawing a context-level data flow diagram that shows interactions between the system and external actors that act as data sources and data sinks. In Context Diagrams (also known as Level 0 DFDs), the interaction of a system with the outside world is modeled solely in terms of data flows across system boundaries. A context diagram represents the entire system as a single process and does not show its internal organization.

This contextual level of his DFD is then "leveraged" to produce a Level 1 DFD that details the modeled system. This DFD level indicates how the system is divided into subsystems (processes). Each subsystem (process) handles one or more data streams to and from external agents and provides overall system functionality. all systems. The Level DFD is then expanded and split into more detailed and descriptive descriptions of the project, similar to the Level 2 DFD. A Level 2 DFD consists of a set of data streams that ultimately describe a complete project description. software.

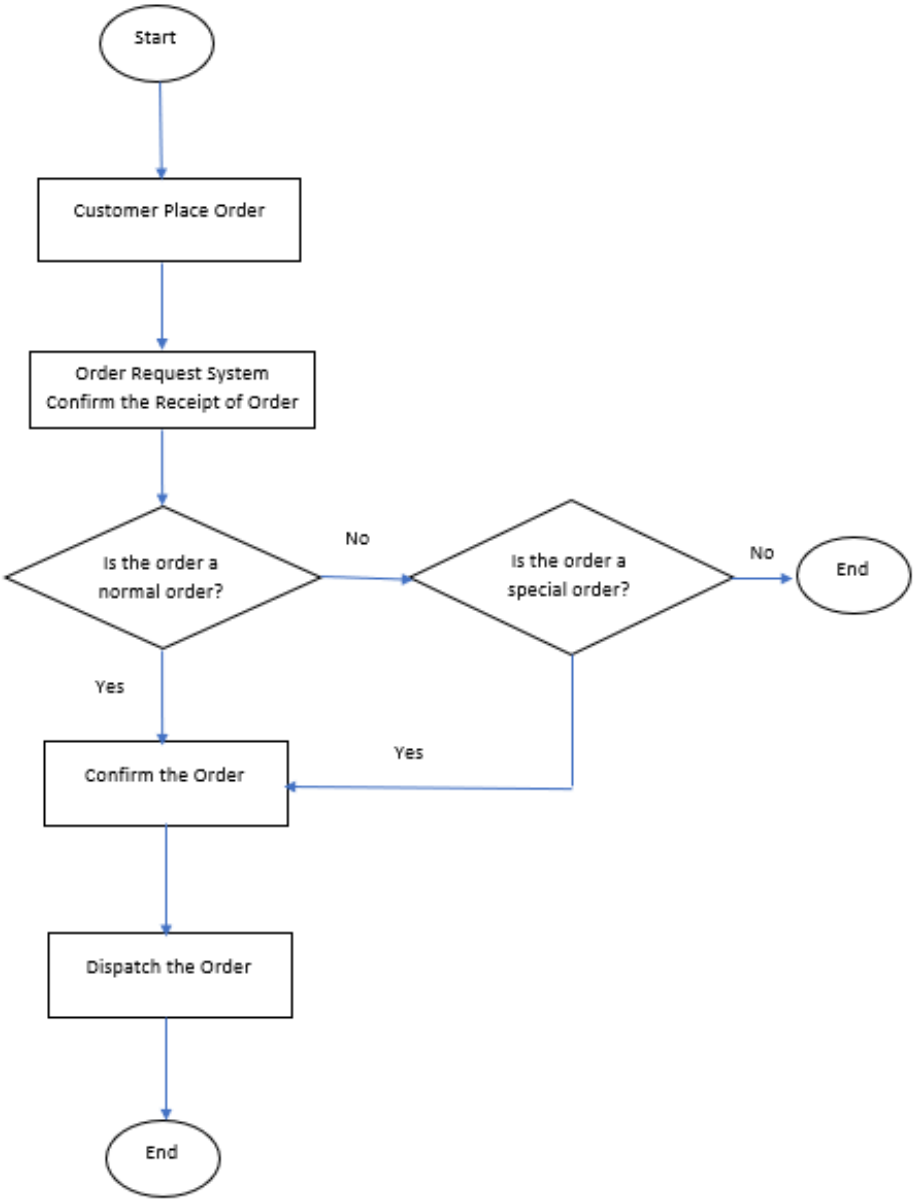


Figure 5.6: Data Flow Diagram

Chapter 6

Implementation

At the end of the system design, we start the actual coding to develop the proposed system. During the development phase, database table structures are first created to provide suitable data types for system backend development, system connectivity, and data transfer. A desktop client program is then developed, followed by a Web service that allows the mobile client program to communicate with the server. Finally, a client program is developed on the mobile side. During the testing phase, several test cases are run to test the system and determine its reliability and accuracy. Based on the test cases, a system test report is produced for further review to identify system weaknesses and make appropriate improvements. Restaurant employees receive special training during the implementation phase. B. Use of the system, processes for managing various events, and policies to be followed. when using the system.

6.1 Implementation Issues & Challenges

During the system implementation phase, several challenges need to be addressed as it involves end-users testing the production system in different scenarios. Possible challenges you may face include:

6.1.1 User without background

Users must have basic knowledge of Android mobile and computer operating systems to use the system. It will be difficult to train users because knowledge base information can be obtained easily from the Internet.

6.1.2 The screen size of different device

This will be one of the problems you will encounter when deploying the system. Because user can download mobile app from official website and use it as client to place order. Therefore, if the size of the user device screen is too small or too large. Content and application interface may not be consistent.

6.1.3 Server performance

When implementing a real-time system, three will be a large number of clients accessing the server at the same time. As a result, it can slow down connections and system performance, and even cause server downtime if the problem goes out of control.

6.2 Components in Reactjs

Certainly! Components are fundamental building blocks in React.js, a popular JavaScript library for building user interfaces. They play a crucial role in creating reusable and modular UI elements, allowing developers to efficiently manage complex applications.

6.2.1 Introduction to Components

In React, a component is a self-contained unit that encapsulates both the visual elements and the behavior of a specific part of a user interface. It's like a Lego block that you can assemble to create a larger structure. React components are divided into two main types: functional and class components.

Functional Components

Functional components are the simplest type of React components. They are essentially JavaScript functions that return JSX (a syntax extension for JavaScript often used in React to define UI elements). These components are primarily used for rendering UI based on the props they receive. A prop is data passed to a component from its parent component. Functional components do not have their own internal state or lifecycle methods, which makes them lightweight and easy to reason about.

Here's an example of a functional component:

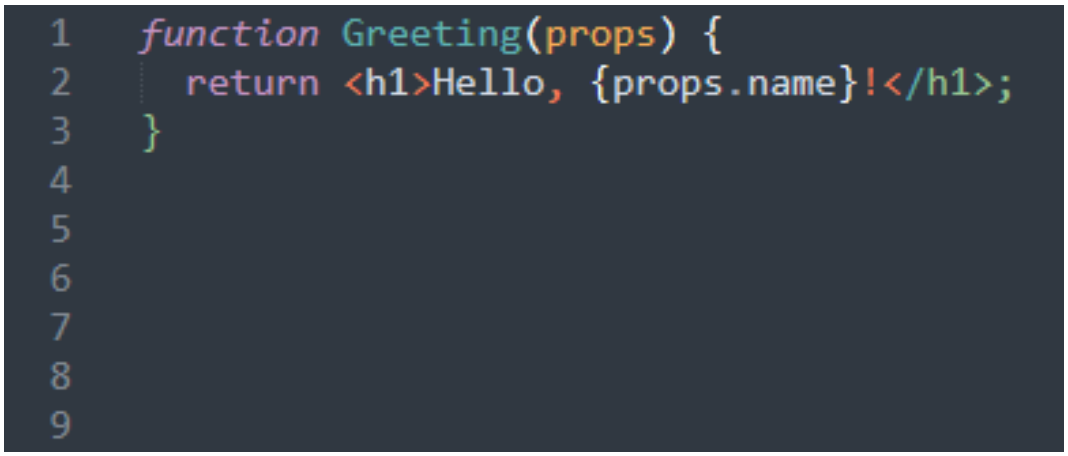


Figure 6.1: Image of Functional Component Example

Class Components

Class components are more powerful and versatile. They are defined as JavaScript classes that extend the 'React.Component' class. Class components have their own state, which is an object that holds data that can change over time and trigger re-renders of the component. They also have access to lifecycle methods, which are special functions that are called during various stages of a component's life, such as when it's mounted to the DOM or updated.

Here's an example of a class component that includes state and a lifecycle method:

```
class Counter extends React.Component {
 constructor(props) {
   super(props);
   this.state = { count: 0 };
 }

 increment() {
   this.setState({ count: this.state.count + 1 });
 }

 render() {
   return (
     <div>
       <p>Count: {this.state.count}</p>
       <button onClick={() => this.increment()}>Increment</button>
     </div>
   );
 }
}
```

Figure 6.2: Image of Class Component Example

Component Composition

One of the key features of React is component composition. This means that you can use components within other components to create more complex user interfaces. This helps break down the UI into smaller, manageable pieces that can be reused across the application.

For instance, you might have a ‘Header‘ component that includes a logo and navigation links, and a ‘Footer‘ component that contains copyright information and social media links. You can then include these components in various parts of your app, ensuring consistency and saving development time.

Props and State

Props (short for properties) are a way to pass data from a parent component to a child component. This enables dynamic content and allows components to be configured based on their usage. Props are immutable, meaning they cannot be modified by the child component.

State, on the other hand, is used to manage data that can change over time within a component. It’s mutable and can be updated using the ‘setState()‘ method. When state changes, React automatically re-renders the component to reflect the updated data.

Lifecycle Methods

Class components have lifecycle methods that provide hooks into various stages of a component’s existence. These methods include ‘componentDidMount‘, ‘componentDidUpdate‘, and ‘componentWillUnmount‘, among others. They allow you to perform actions like fetching data from a server, setting up timers, or cleaning up resources when a component is removed from the DOM.

React Reconciliation

React’s Virtual DOM and reconciliation process are key to its performance optimization. When state or props change in a component, React generates a virtual representation of the updated UI, compares it to the previous representation, and then updates only the necessary parts of the actual DOM. This minimizes the number of expensive DOM operations and makes React applications efficient.

Building Complex Applications

In larger applications, components can be organized into a hierarchy, forming a tree structure. This makes it easier to manage and scale the application. State and props flow downward from parent components to child components, and data can be shared and communicated efficiently through the component tree.

Component Libraries

React's component-based architecture has led to the creation of various component libraries and UI frameworks, such as Material-UI and Ant Design. These libraries provide pre-designed, customizable components that follow best practices for user interface design, helping developers create visually appealing and consistent apps more quickly.

React's component-based architecture is at the core of its simplicity, reusability, and efficiency. Components enable developers to break down complex user interfaces into manageable pieces, create reusable building blocks, and efficiently update the DOM when data changes. Whether you're building a simple web app or a sophisticated enterprise application, mastering the concept of components is essential for creating maintainable and scalable React applications.

6.3 Reactjs DOM manipulation

Certainly! DOM manipulation is a fundamental concept in web development that involves dynamically changing the structure, content, or style of a web page using JavaScript. In the context of React.js, a JavaScript library for building user interfaces, DOM manipulation is handled in a unique and efficient way that differs from traditional approaches.

Understanding the DOM

The Document Object Model (DOM) is a representation of a web page's structure and content as objects. Each HTML element, attribute, and text node is represented as a distinct object in the DOM tree. JavaScript can interact with the DOM to modify the page's appearance and behavior dynamically.

Traditional DOM Manipulation

In traditional web development, manipulating the DOM directly can be challenging and inefficient. Changing elements frequently can lead to performance bottlenecks, as each change triggers reflows and repaints in the browser, causing layout recalculations and slow rendering.

For example, consider updating a simple counter on a page using vanilla JavaScript:

```
<button id="increment">Increment</button>
<p id="count">0</p>

<script>
  const incrementButton = document.getElementById('increment');
  const countParagraph = document.getElementById('count');
  let count = 0;

  incrementButton.addEventListener('click', () => {
    count++;
    countParagraph.textContent = count;
  });
</script>
```

Here, every time the button is clicked, the entire DOM is manipulated directly, which can lead to performance issues with more complex interactions.

React's Virtual DOM

React introduces a concept called the Virtual DOM to address these challenges. The Virtual DOM is an abstraction that sits between the actual DOM and your components. Instead of directly manipulating the DOM, React components update the Virtual DOM first, which is a lightweight representation of the real DOM. This operation is fast because it involves updating JavaScript objects, not the actual browser's rendering engine.

Reconciliation Process

When state or props of a React component change, React generates a new Virtual DOM representation of that component and compares it with the previous one. This process is called reconciliation. React's diffing algorithm efficiently calculates the difference between the old

and new Virtual DOM trees and determines the minimal number of changes needed to update the actual DOM.

For instance, in the previous counter example, a React-based approach might look like this:

```jsx
import React, { useState } from 'react';

function Counter() {
  const [count, setCount] = useState(0);

  return (
    <div>
      <button onClick={() => setCount(count + 1)}>Increment</button>
      <p>{count}</p>
    </div>
  );
}


```

Benefits of Virtual DOM

React's Virtual DOM offers several advantages:

- Performance: By minimizing direct manipulation of the actual DOM, React reduces the number of costly operations like reflows and repaints, leading to better performance and smoother user experiences.
- Efficiency:React's diffing algorithm optimizes the updates, ensuring that only the necessary changes are applied to the DOM.
- Abstraction: Developers can focus on building components and logic without worrying too much about low-level DOM manipulation.
- Reusability: Components are reusable and self-contained, making them easier to maintain and test.
- Cross-Browser Compatibility:React abstracts away browser-specific inconsistencies in DOM APIs, providing a consistent programming model.

Updating the DOM

When React determines the necessary updates based on the Virtual DOM comparison, it batch-processes these updates and then applies them to the actual DOM in a process called reconciliation. This means

that React optimizes and groups changes to minimize the impact on performance.

Reconciliation Strategies

React employs different strategies for reconciling updates efficiently:

- Reusing Elements: If an element remains unchanged between updates, React reuses the existing DOM element.
- Keyed Lists: When rendering lists of items, React uses unique keys to efficiently update, add, or remove list items without affecting unrelated items.
- Component Lifecycle Methods: React components have lifecycle methods that allow developers to control actions during different stages of a component's existence, such as 'componentDidMount', 'componentDidUpdate', and 'componentWillUnmount'.

In React.js, the concept of DOM manipulation is handled through a Virtual DOM and a sophisticated reconciliation process. This approach significantly improves performance, optimizes updates, and abstracts away many of the complexities associated with direct DOM manipulation. By using the Virtual DOM, React provides an efficient and effective way to create dynamic and responsive user interfaces while maintaining performance and scalability.

6.4 Development Tools

6.4.1 Web Technology

The proposed system is a cross-platform system consisting of a Windows desktop client and an Android mobile client. Therefore, I need to implement a web service that supports an Android phone client that retrieves data from a database server and saves hacked data to the database server. This project uses Reactjs for UI development.

An integrated Windows web development environment linked to Reactjs. HTML and CSS. Additionally, JavaScript is a programming language used to create web services hosted on servers, which can be called by Android phone clients to perform specific tasks.

6.4.2 System Platform

This project is a cross-platform system consisting of a Windows desktop client and an Android mobile phone client to operate the proposed system. Therefore, IOS mobile phone client devices are not compatible with the proposed system.

6.4.3 Project Management Tool

As a principle of system development, there should be a complete system plan to guide the development stages. Microsoft Project is project management software used by projects for designing systems planning, project scheduling, resource management, development progress tracking, and more.

6.4.4 Visual Paradigm Community Edition

in this project. Visual Paradigm software is used to document multiple system plan diagrams such as use case diagrams, activity diagrams, and class diagrams. Do you like this? Software developers will be able to visualize and communicate their systems more accurately and clearly.

6.4.5 Test Plan

Once system development is complete, we move on to the system testing phase. During the system testing phase, the developed system should be installed on a suitable device for testing purposes. After system installation is complete, system testing tasks are performed by various roles or users. B. Manager Roles and Employee Roles. The purpose of system testing is to identify and determine the level of system stability. At the same time, developers have the opportunity to discover unreported bugs and encountered bugs in the system. stage of development. Any errors or bugs found during system testing activities will be corrected prior to system release. All tests performed prior to the system testing stage are actually tested by the system designers themselves. As a result, system designers have knowledge of the system software logic, which can lead to inconsistent results and test bias. There are four types of tests for developed systems: unit tests, integration tests, system tests, and acceptance tests.

Chapter 7

System Testing

System testing for an "Online Food Ordering and Delivery Platform" in ReactJS involves testing various aspects of the front-end development to ensure that the application functions correctly and meets user expectations. Here are some system testing cases to consider for front-end development:

User Authentication and Authorization

- Test user registration, login, and logout processes.
- Verify that only authenticated users can access certain parts of the application (e.g., order history, profile).

Menu Display and Interaction

- Test that the menu items are displayed correctly.
- Verify that users can view item details and add items to their cart.
- Test any filtering, sorting, or search functionality in the menu.

Cart Functionality

- Test adding and removing items from the cart.
- Verify that the cart displays the correct items, quantities, and total cost.
- Test the checkout process, including applying discounts or promo codes.

Responsive Design

- Test the application on various devices and screen sizes (desktop, tablet, mobile).
- Ensure that the user interface remains usable and visually appealing.

Performance and Load Testing

- Test the application's performance under different load levels.
- Check for slow loading times, especially when accessing the menu or during checkout.

Accessibility Testing

- Test the application for accessibility using tools like screen readers and keyboard navigation.
- Ensure that all users, including those with disabilities, can use the application effectively.

Browser Compatibility

Test the application in different web browsers (e.g., Chrome, Firefox, Safari, Edge) to ensure consistent behavior and appearance.

Cross-Device Testing

Test the application on various devices (Android, iOS) using real devices or emulators/simulators.

Cross-Platform Testing

If the application has a mobile app version, perform testing on both the web and mobile versions to ensure consistent functionality.

By addressing these system testing cases, I can thoroughly test the front-end development of my "Online Food Ordering and Delivery Platform" in React.js and ensure a high-quality user experience.

Chapter 8

Screenshots

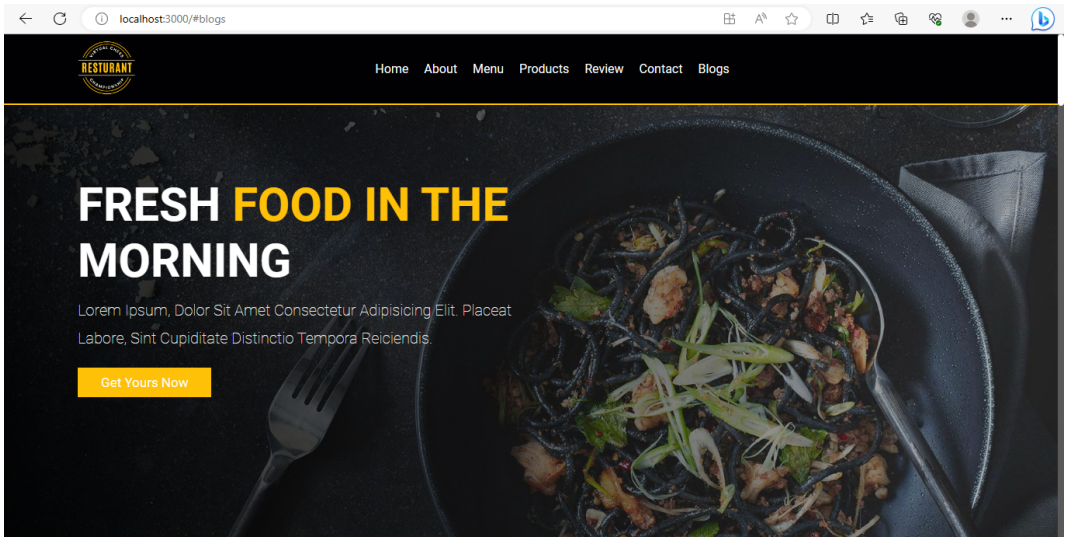


Figure 8.1: Image of Home Page

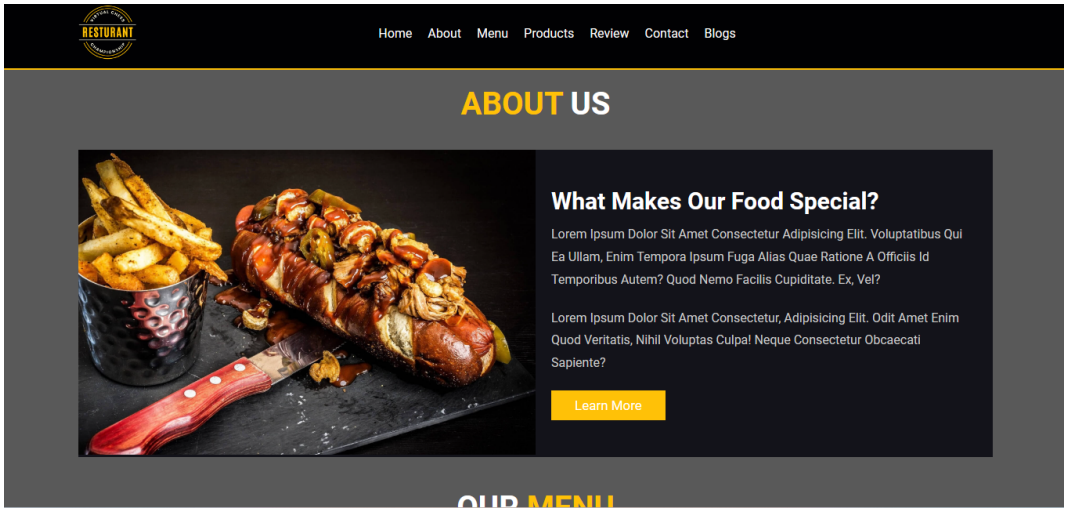


Figure 8.2: Image of About Page

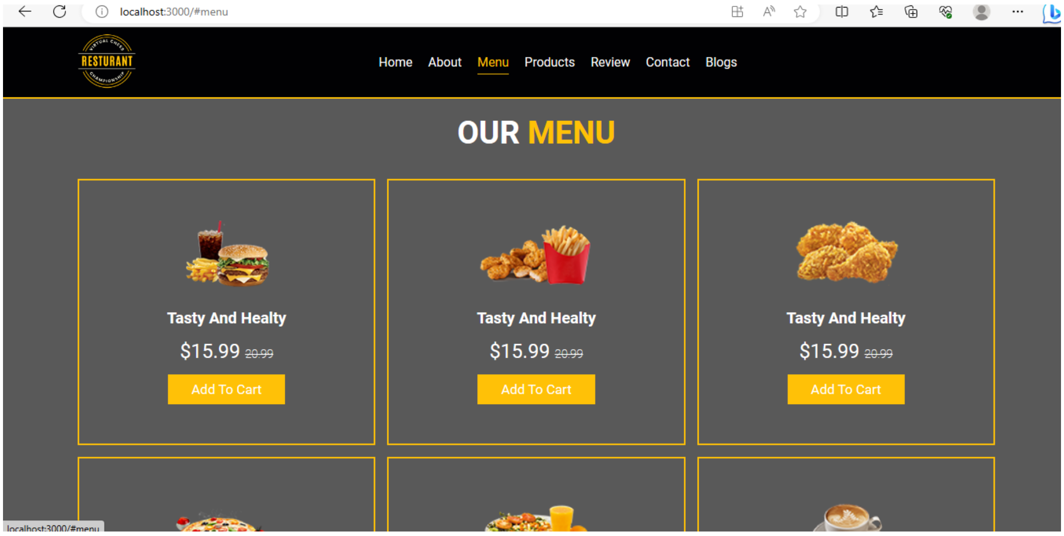


Figure 8.3: Image of Menu Page

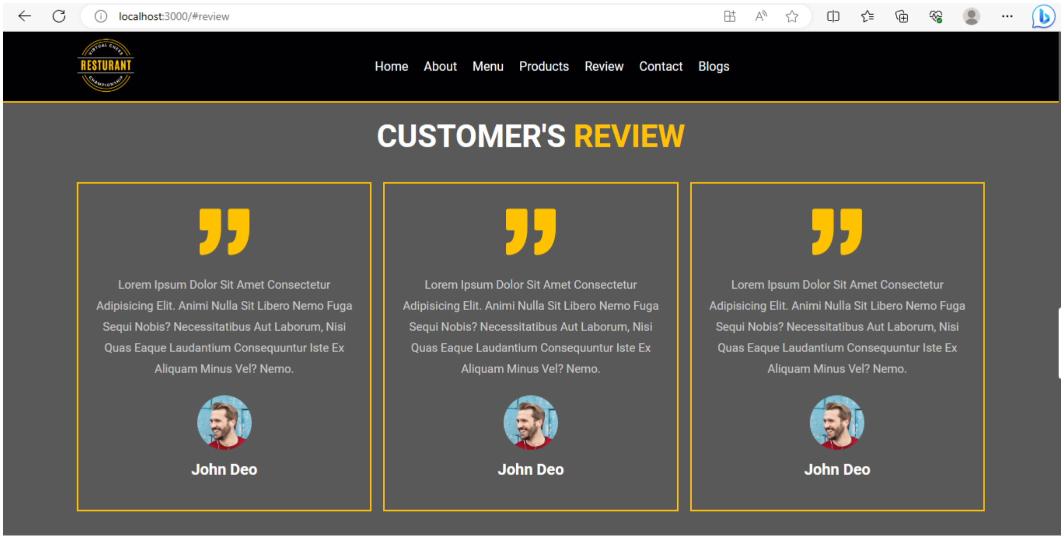


Figure 8.4: Image of Review Page

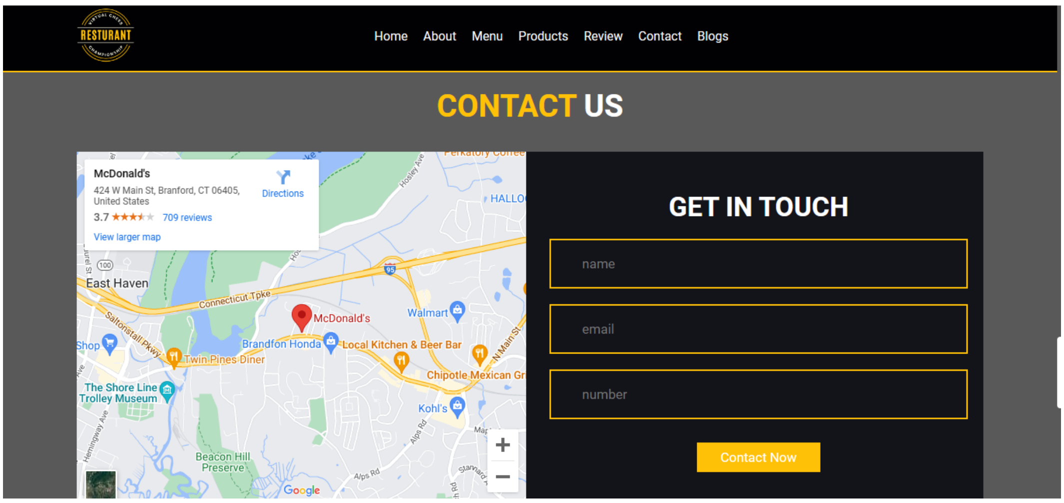


Figure 8.5: Image of Contact Page

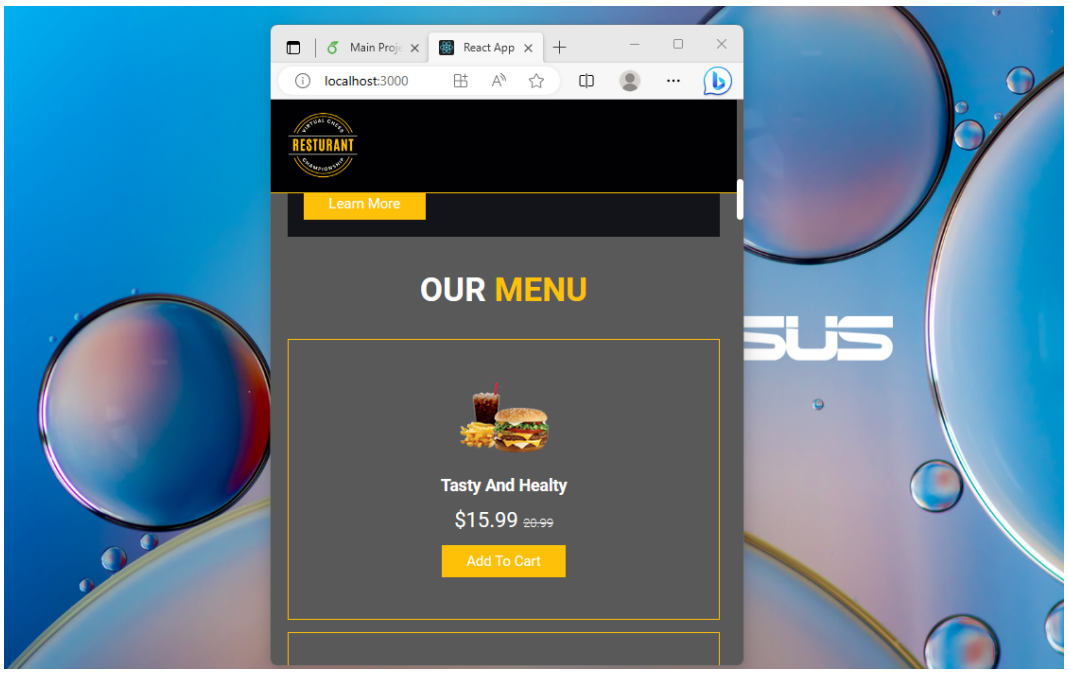


Figure 8.6: Image of Responsive Page

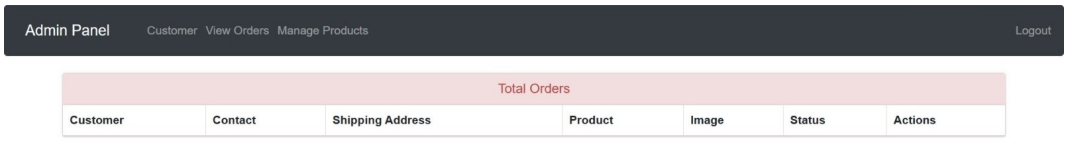


Figure 8.7: Image of Admin Panel

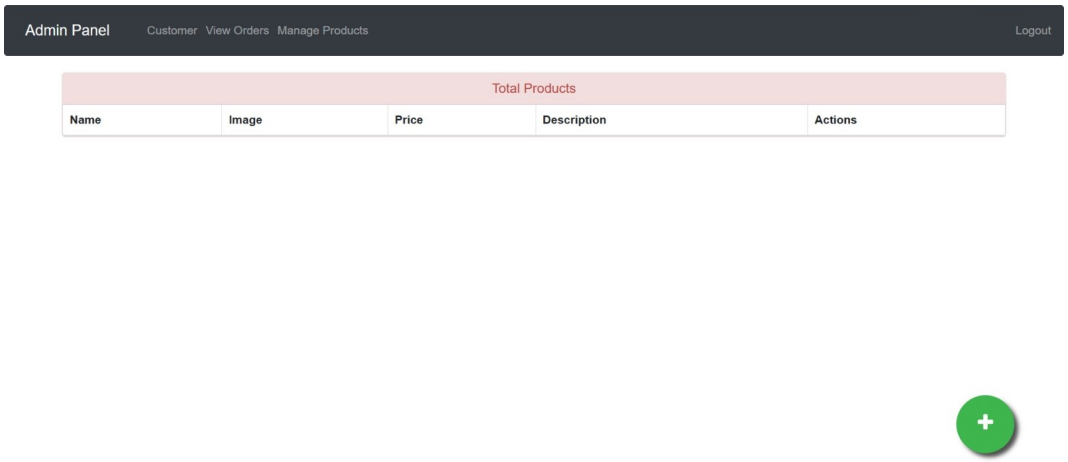


Figure 8.8: Image of Order page

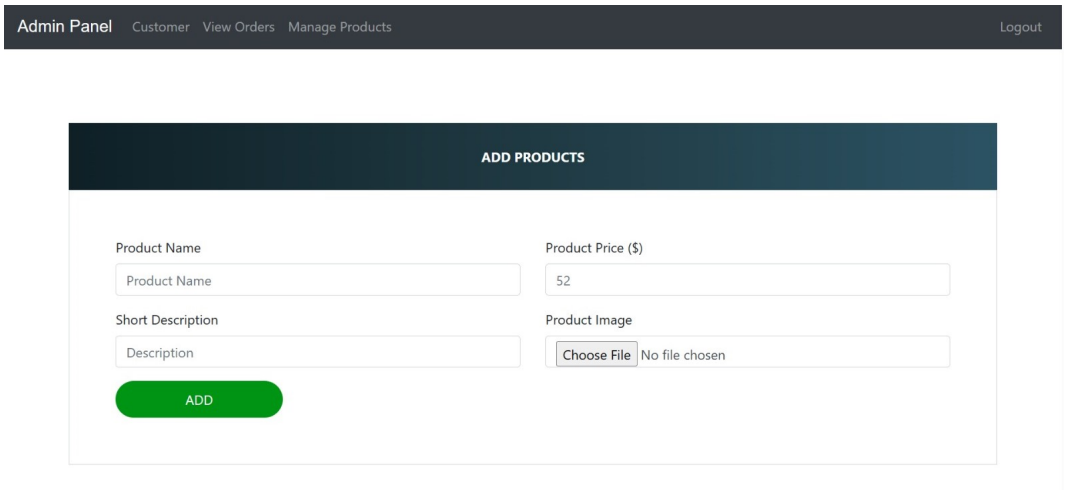


Figure 8.9: Image of Add product Page

Chapter 9

Challenges and Solutions

Responsive Design

Ensuring the application's responsiveness across various devices and screen sizes can be challenging due to differences in screen resolutions and aspect ratios. Solution: Implementing CSS media queries and testing the application on different devices will help achieve a consistent and optimal user experience.

Real-time Order Tracking

Implementing real-time order tracking may require integrating with server-side technologies and handling real-time data updates. Solution: Utilize technologies like WebSockets or server-sent events to enable real-time communication between frontend and backend for order tracking updates.

User Authentication and Account Management

Implementing user authentication and managing user accounts securely requires robust security measures and validation checks. Solution: Use libraries and best practices for user authentication, encrypt sensitive data, and implement proper validation to ensure data privacy and security.

Cross-Browser Compatibility

Different web browsers may interpret CSS and JavaScript differently, leading to inconsistencies in the application's appearance and functionality. Solution: Conduct thorough cross-browser testing to identify and address any compatibility issues.

Performance Optimization

Large JavaScript bundles or unoptimized assets can lead to slower loading times and decreased performance. Solution: Use bundlers like Webpack, optimize images, and employ code splitting to reduce bundle size and improve application performance.

Handling User Feedback

Managing user feedback and addressing feature requests can be overwhelming. Solution: Set up a proper feedback mechanism and prioritize user requests based on their impact and feasibility.

Accessibility

Ensuring the application is accessible to users with disabilities requires adherence to accessibility guidelines. Solution: Follow accessibility standards, use semantic HTML elements, and include proper alternative text for images.

User Experience (UX)

Creating a seamless and delightful user experience requires a thorough understanding of user needs and preferences. Solution: Conduct user testing and gather feedback to iteratively improve the application's usability and UX design.

Maintainability and Code Quality

As the codebase grows, maintaining clean and organized code becomes challenging. Solution: Adopt coding standards, use linters, and refactor code regularly to maintain code quality and readability.

Integration with Backend (Future Consideration)

Integrating the frontend with a backend and managing API calls can be complex. Solution: Plan for a well-defined API contract and establish communication protocols between frontend and backend teams.

Error Handling

Implementing effective error handling to provide meaningful feedback to users requires careful consideration. Solution: Display clear error messages and validations, guiding users to correct their input and actions.

By anticipating and addressing these challenges, the frontend web development project for the online food delivery system using ReactJS can deliver a robust, responsive, and user-friendly application. Continuous testing, feedback gathering, and iterative improvements will contribute to the successful implementation of the system requirements and an enhanced food ordering experience for users.

9.1 Solutions and values:

Here are some solutions and values that an Online Food Ordering and Delivery Platform website can provide:

Restaurant Discovery:

This website would offer a comprehensive database of local restaurants, allowing users to search and discover various dining options based on location, cuisine, ratings, and reviews. This helps users find the right restaurant based on their preferences.

Online Food Ordering:

This website can facilitate online food ordering, enabling users to browse menus, customize orders, and place them for delivery or pickup. Integration with popular payment gateways ensures smooth and secure transactions.

Table Reservations:

This website can include a table reservation feature, allowing users to book tables at their desired restaurants. This feature enhances the dining experience by providing convenience and eliminating waiting times.

Ratings and Reviews:

This website can incorporate a ratings and reviews system, enabling users to share their opinions and experiences about restaurants. User-generated reviews help others make informed decisions while choosing a place to dine.

Menu Display and Updates:

This website can showcase detailed menus for each restaurant, including descriptions, prices, and available options. It should also allow restaurants to update their menus regularly to ensure accurate and up-to-date information.

Location-based Services:

This website can leverage location-based services to display restaurants near the user's current location. This feature improves user convenience by providing relevant options within their vicinity.

Seamless User Experience:

This website should prioritize a user-friendly interface, intuitive navigation, and a visually appealing design. The aim is to create a seamless and engaging experience for users, making it easy to browse, order, and interact with restaurants.

Marketing and Promotions:

To attract both users and restaurants, this website can offer marketing and promotional opportunities. This includes featuring restaurants on the homepage, offering discounts or deals, and implementing loyalty programs.

Customer Support:

Providing robust customer support is crucial. This website should have channels for users and restaurants to reach out for assistance, feedback, or issue resolution. Timely and helpful support enhances user satisfaction and builds trust.

Business Partnerships:

This website can establish partnerships with restaurants and food delivery services to expand its reach and offerings. Collaborations with delivery personnel and logistics providers ensure efficient order fulfillment.

Overall, This website aims to deliver value to both users and restaurants by providing a platform that simplifies the process of discovering, ordering from, and engaging with local restaurants. By incorporating key features and ensuring a positive user experience, this website can create a successful online food delivery and restaurant discovery platform.

Chapter 10

Results

To develop an Online Food Ordering and Delivery Platform website, we will need to:

10.1 Plan and Define Requirements

Determine the key features, functionalities, and goals of your clone website. Identify the scope, target audience, and unique selling points to differentiate your platform from competitors.

10.2 Design and Development

Create wireframes and design mockups to visualize the user interface (UI) and user experience (UX) of your website. Once the design is finalized, proceed with the development phase to build the frontend and backend components, implement features, and integrate necessary APIs.

10.3 Database and Infrastructure

Set up a robust and scalable database architecture to handle user data, restaurant information, menus, orders, and other relevant data. Ensure proper data modeling, indexing, and optimization for efficient retrieval and storage.

10.4 Payment Gateway Integration

Integrate secure and reliable payment gateways to facilitate online transactions. Research and select payment partners that support your

target markets and comply with local regulations.

10.5 Location-based Services

Implement geolocation features to enable users to find restaurants near their current location. Integrate mapping APIs to display restaurant locations accurately and provide directions.

10.6 Ratings and Reviews System

Develop a ratings and reviews system that allows users to rate and review restaurants. Implement moderation mechanisms to ensure content quality and manage user feedback effectively.

10.7 Order Management and Delivery Tracking

Build a robust order management system that handles incoming orders, communicates with restaurants, and provides real-time order tracking to users. Coordinate with delivery partners or develop an in-house delivery system if desired.

10.8 Quality Assurance and Testing

Thoroughly test your clone website to identify and resolve any bugs, usability issues, or performance bottlenecks. Conduct both manual and automated testing to ensure the website functions seamlessly across different devices and browsers.

10.9 Deployment and Launch

Prepare your infrastructure for deployment, considering factors such as scalability, security, and performance. Once ready, deploy your clone website to a production environment and make it available to users.

10.10 Marketing and User Acquisition

Develop a marketing strategy to promote your clone website, attract users, and engage with restaurants. Utilize online and offline channels, social media, content marketing, and partnerships to drive user adoption.

Chapter 11

System Strength and Limitation

11.1 System Strength

This system provides customers with a comfortable dining experience by allowing them to view food information and place orders from their mobile phones. The mobile app is easy to use as it has a straightforward graphical user interface and few instructions to follow when ordering through the app. That way, he helps restaurant staff to serve valuable customers quickly. Additionally, the system is inexpensive to deploy and the hardware requirements are not too powerful or too expensive to support the system, making it suitable for most small and medium-sized restaurants. After all, the communication between the restaurant's servers and customers is an intranet, so there's no need for internet access.

11.2 System Limitation

The mobile application is developed in the Android environment, so the system is not compatible with iOS mobile devices. Therefore, 105 mobile phone users may not be able to install mobile phone apps and use the system. Meanwhile, the restaurant has an Android mobile device that aims to solve the above problem. Second, because we have mobile phones, the client devices need to connect to the wireless Internet to communicate with the server. Determining the right location to install and configure your WiFi access point is very important to ensure WiFi signal coverage throughout your restaurant area.

Chapter 12

Conclusion and Future work

12.1 Conclusion:

Ten years later, technological advances and innovations have made it easier and more efficient for people to do their jobs. In many other industries, regions have long used management systems to support business growth. Therefore, there is also a trend for the F&B industry to introduce management systems into their businesses. their business activities. At the end of this project, the system will be able to reduce and replace human labor, spend less time on each transaction, and generate reports for later management to get the most out of the system. . Therefore, it directly affects the profitability of the restaurant. In addition, the system will already support most of the business processes that use the system, which will help reduce operating costs in terms of restaurant labor costs. Therefore, it is believed that this system will lead to long-term expansion of the restaurant business. On the other hand, today's technology makes it easy to meet mobility needs.

Portability is therefore one of the factors to consider in system development. This is because mobility brings many advantages to users such as convenience, accessibility, and easy communication when using systems. We recommend a method combined with food ordering to meet all your needs. On a mobile platform with a computer-based restaurant management system. Integrating the two functions creates a system that allows users to have a mobile experience. Users can process food orders from their smartphones and tablets. In addition, computers have other features such as large screens, so restaurants manage their daily operations through computing platforms. Another

compatible system assists restaurant managers and other drivers who need to communicate with this necessary hardware.

12.2 Future Enhancement

The system can implement real-time notification functions from mobile phone apps to service centers. This feature will allow customers to request customer service through her mobile app instead of making verbal calls to restaurant staff. In addition, the mobile application can also implement functionality that allows customers to update the status of the food being served. For example, a gourmet customer in a restaurant can use a mobile app to order a dish to be served, and when they are full after finishing their main course, they can ask that the next dish not be served. Via mobile application. Finally, your mobile app can implement mini-games to entertain your customers while waiting for their food to be served.

References

1. Aman Goel “10 Best Web Development Frameworks”, published 07 Aug, 2020 https://hackr.io/blog/top-10-web-development-frameworks-in-2020

2. Duomly, “The best front-end framework to learn in 2019”, published Oct 16, 2019 https://dev.to/duomly/the-best-front-end-framework-to-learn-in-2019-dn7

3. Nitin Pandit, What and Why React.js, published Mar 05, 2020 https://www.csharpcorner.com/article/what-and-why-reactjs/

4. 10 Best Web Development Frameworks, published 07 Aug, 2020 https://hackr.io/blog/top-10-web-development-frameworks-in-2020

5. Eric Baer “What is React” https://www.oreilly.com/library/view/what-react-is/9781491996744/ch01.html

6. Chinmayee Deshpande “The Best Guide to Know What Is React” published 28 December, 2021 https://www.simplilearn.com/tutorials/reactjs-tutorial/what-is-reactjs

7. Saurabh Sharma “Declarative Programming & React” published Dec 26,2018 https://dev.to/itsjzt/declarative-programming–react-3bh2

8. W3schools “React Components”

   https://www.w3schools.com/react/react_components.asp

9. Wikipedia “Architecture beyond HTML”

   https://en.wikipedia.org/wiki/React_(JavaScript_library)

   #Architecture_beyond_HTML

10. AmberJ, “5 Delightful Things about Material-UI” Published Nov 11, 2019.https://dev.to/amberjones/5-delightful-things-about-material-ui-5402

11. H. Sharp, Y. Rogers, J. Preece. Interaction Design: Beyond Human-Computer Interaction. 5th Edition. Chichester: John Wiley & Sons Ltd. 2019.

12. Pedro Mendonca "Human-Computer Interaction: What Is It, Why Does It Matter & Best Practices" Published 12 May, 2020 https://exaud.com/human-computer-interaction/

13. Achonwa Alvan" What are usability goals in interactive design?"

    https://www.educative.io/answers/what-are-usability-goals-in-interactive-design

14. CloudFlare, "what do client side and server-side mean? client-side vs serverside

    https://www.cloudflare.com/learning/serverless/glossary/client-side-vs-server-side/

15. Visual Studio "Getting Started" Published 13 January, 2021. https://code.visualstudio.com/docs

16. Computer Hope "GUI" https://www.computerhope.com/jargon/g/gui.htm