1. Comprehensive Website Handbook

1.1. Index

- 1. Comprehensive Website Handbook
    - 1.1. Index
    - 1.2. Introduction
        - 1.2.1. About the Website
        - 1.2.2. Purpose of the Handbook
        - 1.2.3. Target Audience (Developers, Marketers, Testers, etc.)
        - 1.2.4. How to Use This Handbook
    - 1.3. Website Overview
        - 1.3.1. Vision and Mission of the Website
        - 1.3.2. Key Features and Offerings
        - 1.3.3. Target Audience for the Website (end-users)
        - 1.3.4. High-Level Overview of the Website Workflow
        - 1.3.5. Glossary of Terms (for technical and non-technical users)
    - 1.4. Functional Flow
        - 1.4.1. User Flows
        - 1.4.2. Visual Flow Diagrams for Each User Flow
            - delivery order flow
            - takeaway order flow
            - dine-in order flow
        - 1.4.3. Key Use Cases and Scenarios
    - 1.5. Technical Architecture
        - 1.5.1. Technology Stack Overview
        - 1.5.2. High-Level Architecture Diagram
        - 1.5.3. Deployment and Hosting Details
        - 1.5.4. Environment Setup
    - 1.6. Beginner's Guide to Programming
        - 1.6.1. Introduction to Web Development Basics
        - 1.6.2. Overview of Tools and Software to Install
        - 1.6.3. Step-by-Step Guide to Setting Up the Project Locally
        - 1.6.4. Suggested Learning Path
        - 1.6.5. Debugging Basics
    - 1.7. Codebase Structure and Flow
        - 1.7.1. Overview of the Codebase
        - 1.7.2. Code Execution Flow
        - 1.7.3. Understanding Functions and Modules
        - 1.7.4. Step-by-Step Explanation of a Key Feature
        - 1.7.5. Reading the Code
        - 1.7.6. Code Standards and Best Practices
    - 1.8. API Documentation
        - 1.8.1. Overview of API Usage and Purpose
        - 1.8.2. API Endpoint List

        - 1.8.3. Error Codes and Handling
        - 1.8.4. How to Test APIs as a Beginner
    - 1.9. Database Design
        - 1.9.1. Database Schema Overview
        - 1.9.2. Key Tables and Their Purpose
        - 1.9.3. Entity-Relationship Diagrams (ERD)
        - 1.9.4. Sample Queries for Common Use Cases
    - 1.10. User Interface (UI)
        - 1.10.1. Screenshots of All Pages (annotated with descriptions)
        - 1.10.2. Navigation Map
        - 1.10.3. Design Principles Used
    - 1.11. Ad Hoc Process Configuration
        - 1.11.1. Payment Gateway Integration
            - 1.11.1.1. Overview of Payment Gateway Used
            - 1.11.1.2. API Keys, Credentials, and Configuration Steps
            - 1.11.1.3. Step-by-Step Guide for Setting Up Payment Flow
            - 1.11.1.4. Handling Payment Status Updates (webhooks, callbacks)
            - 1.11.1.5. Debugging Common Payment Issues## Ad Hoc Process Configuration
        - 1.11.2. Payment Gateway Integration
            - 1.11.2.1. Overview of Payment Gateway Used
            - 1.11.2.2. API Keys, Credentials, and Configuration Steps
            - 1.11.2.3. Step-by-Step Guide for Setting Up Payment Flow
            - 1.11.2.4. Handling Payment Status Updates (webhooks, callbacks)
            - 1.11.2.5. Debugging Common Payment Issues
        - 1.11.3. Messaging Service Integration (e.g., SMS, WhatsApp)
            - 1.11.3.1. Overview of Messaging Providers (e.g., Twilio, Firebase)
            - 1.11.3.2. Setting Up API Access and Authentication
            - 1.11.3.3. Sending SMS or WhatsApp Messages (sample code snippets)
            - 1.11.3.4. Configuring OTP Logic and Expiry Timers
            - 1.11.3.5. Error Handling and Logging for Message Delivery Failures
        - 1.11.4. WhatsApp OTP Setup
            - 1.11.4.1. Details of Provider (e.g., Meta’s WhatsApp Business API)
            - 1.11.4.2. Required Credentials (Phone Numbers, API Tokens)
            - 1.11.4.3. Configuring WhatsApp Templates for OTP Messages
            - 1.11.4.4. Implementing Message Flow (request, send, verify OTP)
            - 1.11.4.5. Troubleshooting Common WhatsApp API Issues
        - 1.11.5. Government Compliance-Related Processes
            - 1.11.5.1. Overview of Compliance Requirements
            - 1.11.5.2. Step-by-Step Implementation (e.g., GST APIs, Aadhaar Verification)
            - 1.11.5.3. Sample Data Formats for Government APIs
            - 1.11.5.4. Error Handling and Validation for Compliance APIs
            - 1.11.5.5. Documentation Links to Government APIs and Resources

- 1.12. Testing Guidelines
    - 1.12.1. Overview of Testing Strategy
    - 1.12.2. Functional Testing Scenarios
    - 1.12.3. Technical Testing
    - 1.12.4. Bug Reporting Guidelines
- 1.13. Deployment and Maintenance
    - 1.13.1. Deployment Process (step-by-step)
    - 1.13.2. Version Control Guidelines
    - 1.13.3. Backup and Recovery Plan
- 1.14. Troubleshooting Guide
    - 1.14.1. Common Issues and Fixes
    - 1.14.2. Debugging Tips for Developers
- 1.15. Security Considerations
    - 1.15.1. Security Practices Implemented
    - 1.15.2. Guidelines for Handling Sensitive Data
- 1.16. FAQ
    - 1.16.1. Common Questions by Non-Technical Staff
    - 1.16.2. Questions Related to API Usage
    - 1.16.3. Testing and Debugging FAQs
- 1.17. Appendix
    - 1.17.1. Resources and References
    - 1.17.2. Links to Tools, Libraries, and Frameworks Used
    - 1.17.3. Glossary of Technical Terms

1.2. Introduction

1.2.1. About the Website

The website isa platformfor the userto order food online. The website provides a list of restaurants, their menus, and allows users to place orders for delivery or pickup or dine in . Users can create accounts, save their favorite orders, and track the status of their orders in real-time.

1.2.2. Purpose of the Handbook

The purpose of this handbook is to provide a comprehensive guide to the website's architecture, codebase, and functionality. It is intended for developers, testers, and other stakeholders who need to understand how the website works, how to set it up locally, and how to maintain and troubleshoot it.

1.2.3. Target Audience (Developers, Marketers, Testers, etc.)

The target audience for this handbook includes:

- Developers who need to understand the codebase, APIs, and database design.
- Testers who need to know how to test the website and report bugs.
- Marketers who need to understand the website's features and target audience.

- Project managers who need to oversee the development and deployment of the website.
- Non-technical staff who need a high-level overview of the website's functionality.
- New team members who need to onboard quickly and understand the project.
- Anyone interested in learning about web development and programming.

1.2.4. How to Use This Handbook

This handbook is organized into sections that cover different aspects of the website, from the high-level overview to the technical details of the codebase and database design. You can use the table of contents to navigate to specific sections or read through the entire handbook to get a comprehensive understanding of the website.

1.3. Website Overview

1.3.1. Vision and Mission of the Website

The vision of the website is to provide a seamless and convenient online ordering experience for users, connecting them with their favorite restaurants and enabling them to order food with ease. The mission of the website is to offer a wide variety of food options, ensure timely delivery, and provide a user-friendly interface that makes ordering food a pleasant experience.

1.3.2. Key Features and Offerings

The website offers the following key features and offerings:

- User registration and account creation
- Restaurant listings with menus and reviews
- Order placement for delivery, pickup, or dine-in
- Real-time order tracking
- Favorite orders and reordering
- Payment gateway integration
- Messaging service integration for order updates

1.3.3. Target Audience for the Website (end-users)

The target audience for the website includes:

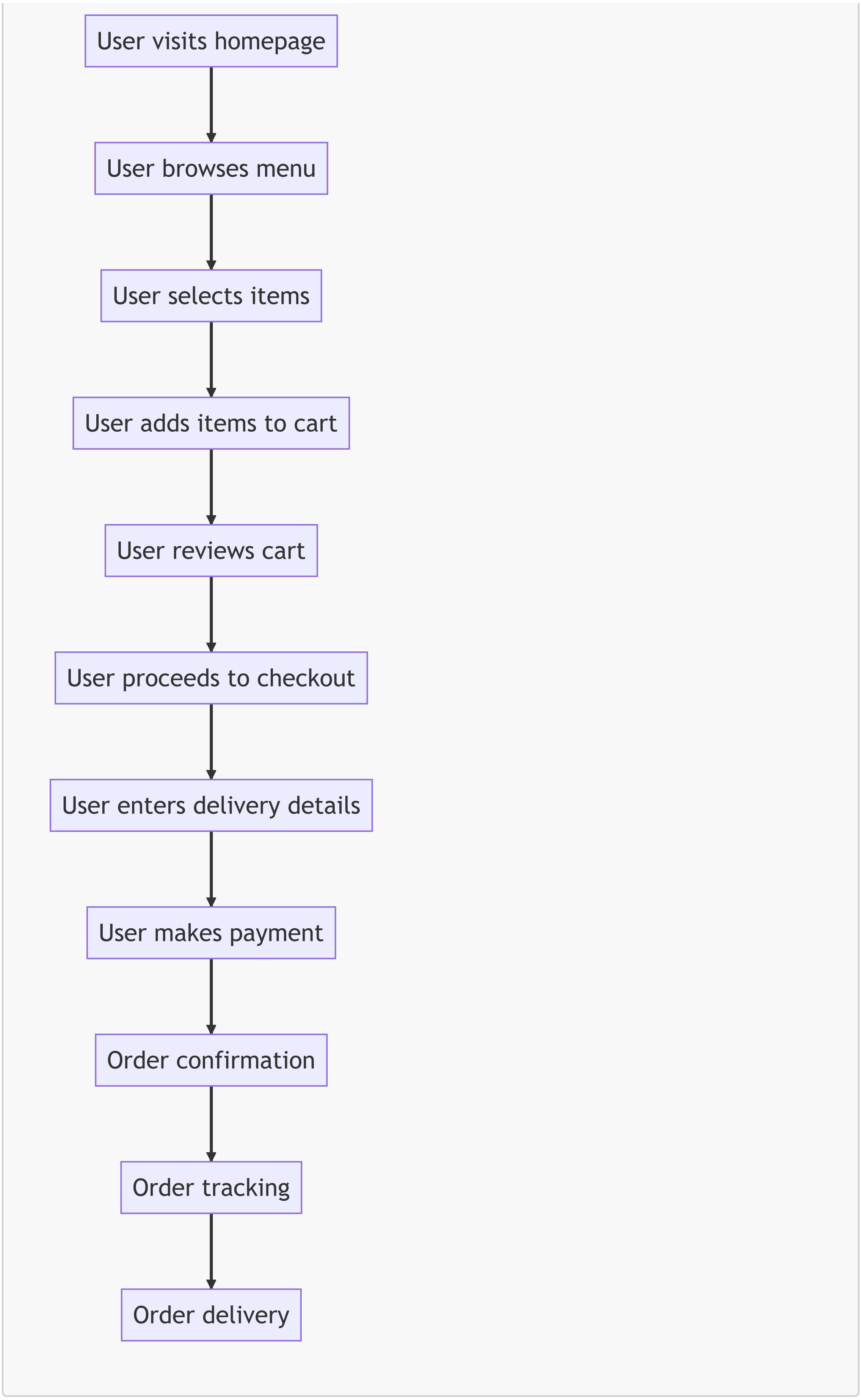
- Working professionals who want to order food for lunch or dinner
- Families looking to order meals for home delivery
- Students who want to order food for study sessions
- Tourists and travelers looking for local cuisine
- Food enthusiasts who want to explore new restaurants
- Event organizers who need catering services
- Anyone who prefers the convenience of online food ordering
- Anyone who wants to avoid the hassle of cooking

1.3.4. High-Level Overview of the Website Workflow

1.3.5. Glossary of Terms (for technical and non-technical users)

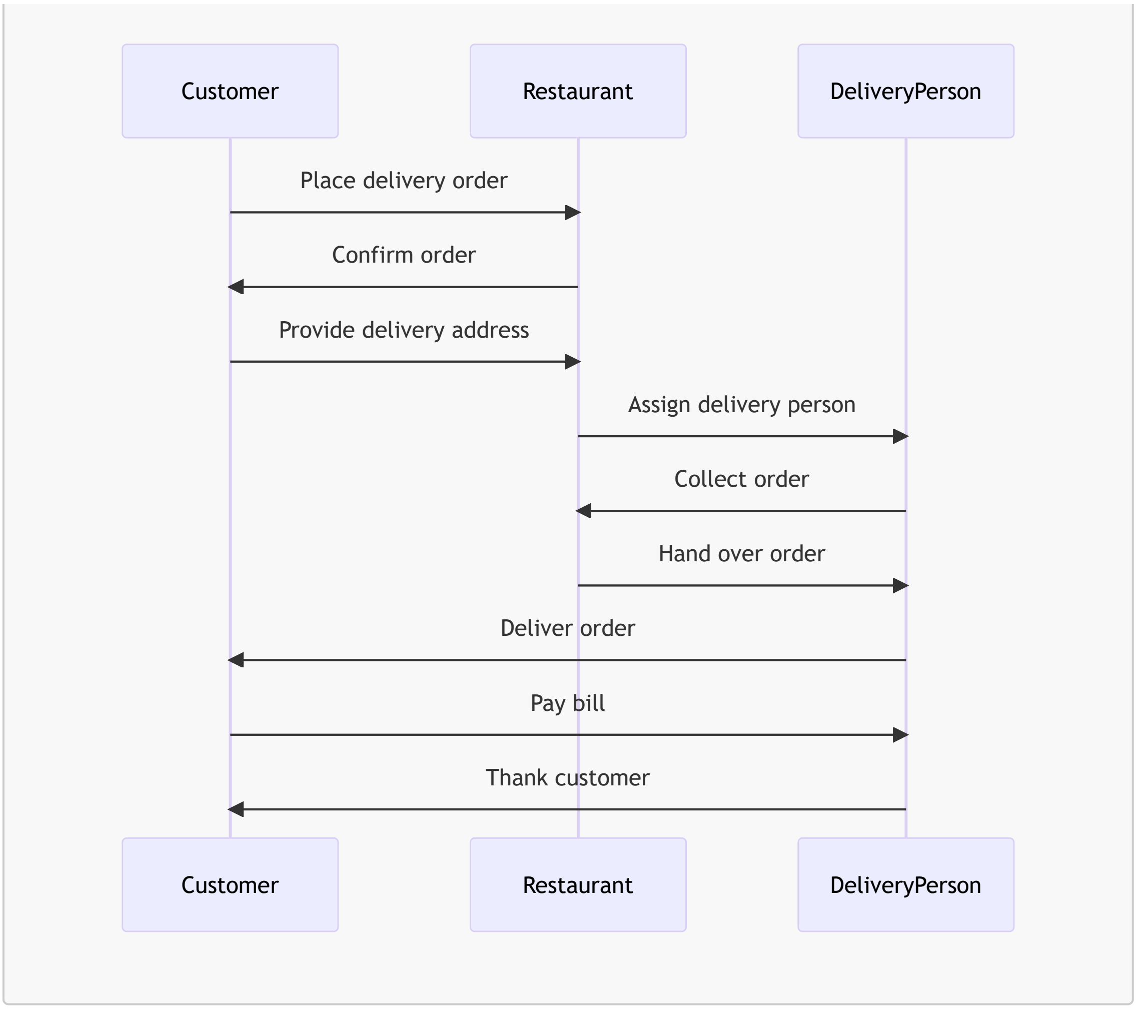
1.4. Functional Flow

1.4.1. User Flows

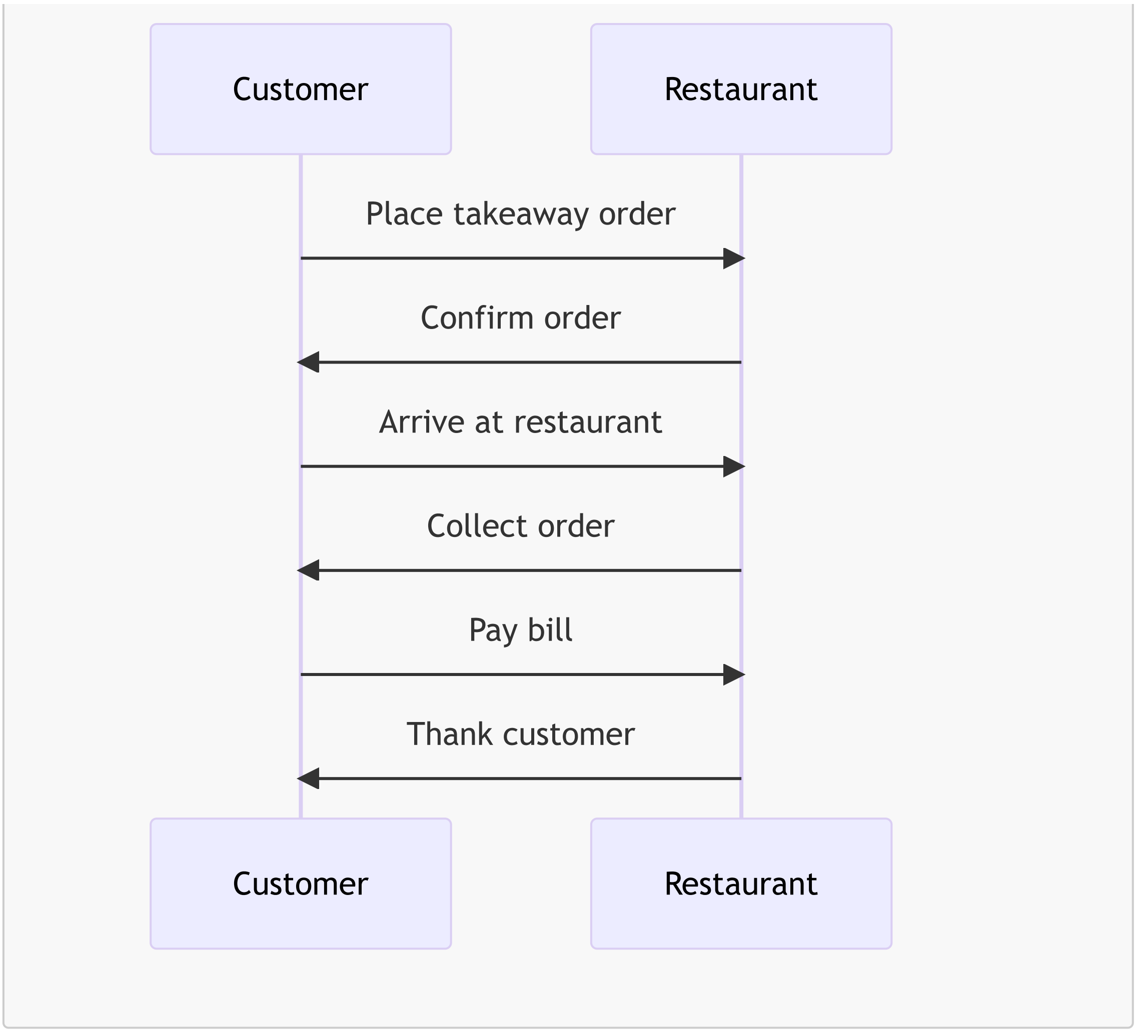


1.4.2. Visual Flow Diagrams for Each User Flow

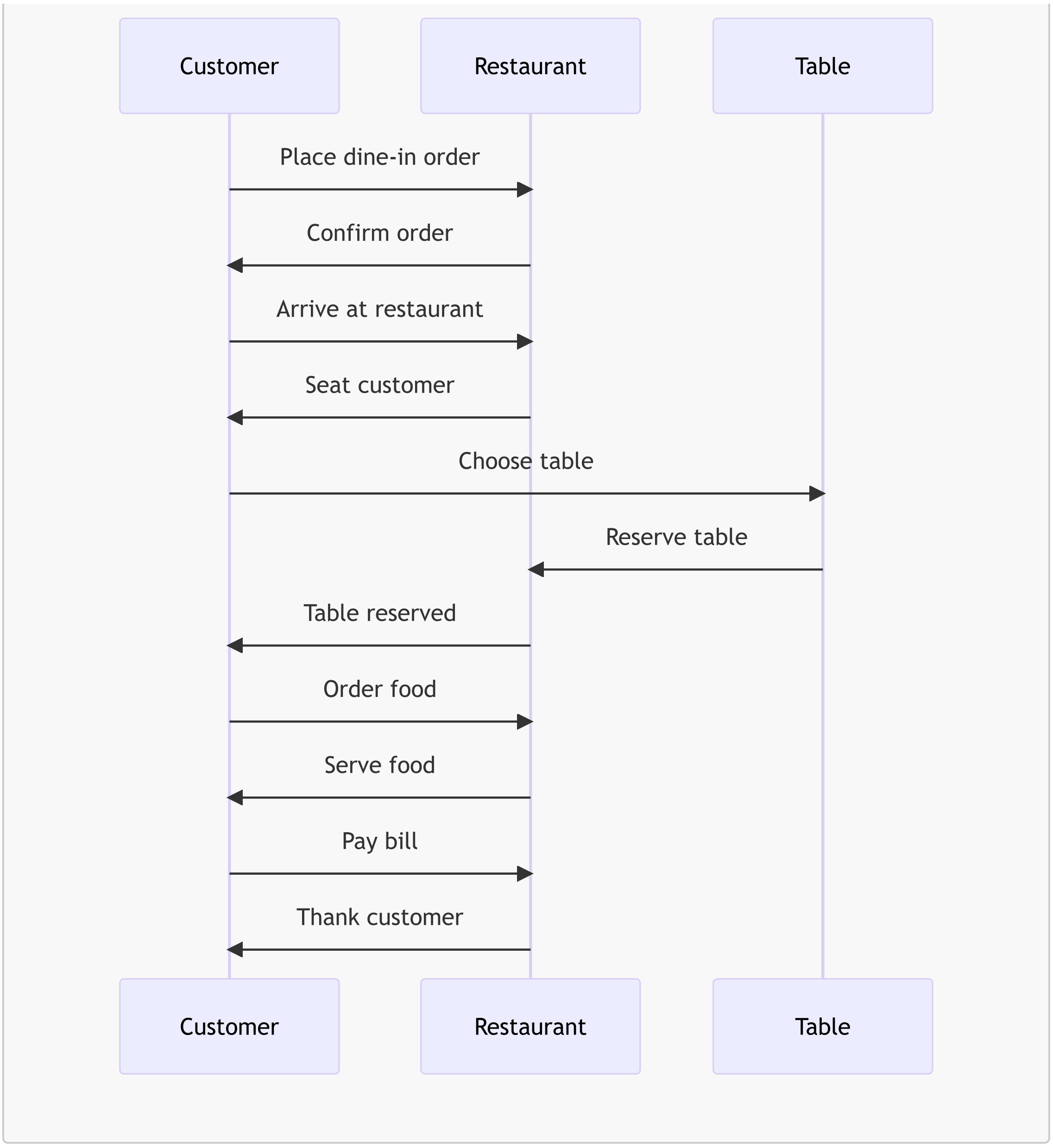
delivery order flow



takeaway order flow



dine-in order flow



1.4.3. Key Use Cases and Scenarios

1.5. Technical Architecture

1.5.1. Technology Stack Overview

The website is built using the following technologies:

- Frontend: Angular - angular is a platform and framework for building single-page client applications using HTML and TypeScript. Angular is written in TypeScript. It implements core and optional functionality as a set of TypeScript libraries that you import into your apps.
- Backend: Node.js - Node.js is an open-source, cross-platform, back-end JavaScript runtime environment that runs on the V8 engine and executes JavaScript code outside a web browser.
- Database: MongoDB - MongoDB is a general-purpose, document-based, distributed database built for modern application developers and for the cloud era.
- Hosting: firebase.com - Firebase is a platform developed by Google for creating mobile and web applications. It was originally an independent company founded in

2011. In 2014, Google acquired the platform and it is now their flagship offering for app development.

- Payment Gateway: razorpay - Razorpay is a payment gateway that allows businesses to accept, process, and disburse payments with its product suite.
- Messaging Service: whatsapp - WhatsApp is a messaging service that allows users to send text messages, voice messages, images, and videos over the internet.
- Other Tools: Git, Postman, VS Code - Git is a distributed version control system for tracking changes in source code during software development. Postman is a collaboration platform for API development that allows users to design, mock, document, monitor, and test APIs. VS Code is a source-code editor developed by Microsoft for Windows, Linux, and macOS.

1.5.2. High-Level Architecture Diagram

1.5.3. Deployment and Hosting Details

The website is deployed on firebase.com and hosted on Google Cloud Platform. The deployment process involves building the Angular frontend and deploying it to firebase hosting. The backend is deployed as a Node.js application on firebase functions. The database is hosted on MongoDB Atlas.

1.5.4. Environment Setup

1. create a firebase project
2. enable firestore and storage
3. create a web app
4. copy the firebase config and paste it in the environment.ts file
5. run `npm run dev` to run the project in development mode
6. run `firebase use staging` to use the staging environment
7. run `firebase deploy` to deploy the project

1.6. Beginner's Guide to Programming

1.6.1. Introduction to Web Development Basics

Web development is the process of building websites and web applications using a combination of HTML, CSS, and JavaScript. HTML is used to create the structure of a web page, CSS is used to style the page, and JavaScript is used to add interactivity and dynamic behavior to the page. Web development also involves working with backend technologies like Node.js and databases like MongoDB to create full-stack applications.

1.6.2. Overview of Tools and Software to Install

1. Node.js - Node.js is a JavaScript runtime built on Chrome's V8 JavaScript engine. You can download Node.js from the official website and install it on your machine.
2. Angular CLI - The Angular CLI is a command-line interface tool that you use to initialize, develop, scaffold, and maintain Angular applications directly from a command shell.

3. MongoDB - MongoDB is a general-purpose, document-based, distributed database built for modern application developers and for the cloud era. You can download MongoDB from the official website and install it on your machine.
4. Git - Git is a distributed version control system for tracking changes in source code during software development. You can download Git from the official website and install it on your machine.
5. Postman - Postman is a collaboration platform for API development that allows users to design, mock, document, monitor, and test APIs. You can download Postman from the official website and install it on your machine.
6. VS Code - Visual Studio Code is a source-code editor developed by Microsoft for Windows, Linux, and macOS. You can download VS Code from the official website and install it on your machine.
7. Firebase CLI - The Firebase Command Line Interface (CLI) provides a variety of tools for managing, viewing, and deploying to Firebase projects. You can install the Firebase CLI using npm.
8. Angular Material - Angular Material is a UI component library for Angular that provides a set of high-quality UI components built with Angular and TypeScript. You can install Angular Material using npm.
9. Razorpay - Razorpay is a payment gateway that allows businesses to accept, process, and disburse payments with its product suite. You can sign up for a Razorpay account and get API keys to integrate with your application.
10. WhatsApp Business API - The WhatsApp Business API allows businesses to communicate with customers over WhatsApp. You can sign up for a WhatsApp Business API account and get API credentials to send messages.

1.6.3. Step-by-Step Guide to Setting Up the Project Locally

1. Clone the repository from GitHub using the `git clone https://github.com/Harsh98992/digitalMenu.git` command.
2. Install Node.js from the official website.
3. Install Angular CLI using the `npm install -g @angular/cli` command.
4. Install MongoDB from the official website.
5. Install Git from the official website.
6. Install Postman from the official website.

1.6.4. Suggested Learning Path

If you are new to web development, here is a suggested learning path to get started:

1. Learn HTML, CSS, and JavaScript basics.
2. Learn Angular basics and build a simple application.
3. Learn Node.js basics and build a simple backend application.
4. Learn MongoDB basics and integrate it with your backend application.
5. Learn how to deploy your application to firebase hosting.
6. Learn how to integrate a payment gateway like Razorpay.
7. Learn how to integrate a messaging service like WhatsApp.
8. Learn how to test and debug your application.

1.6.5. Debugging Basics

Debugging is the process of finding and fixing errors in your code. Here are some basic debugging techniques:

1. Use console.log() statements to print values and debug information.
2. Use the browser developer tools to inspect elements, view console logs, and debug JavaScript code.
3. Use breakpoints in your code to pause execution and inspect variables.
4. Use the Angular CLI to run the project in development mode and view error messages in the console.
5. Use the Postman tool to test APIs and view response data.
6. Use the VS Code debugger to step through your code and inspect variables.
7. Use the Firebase CLI to view logs and debug cloud functions.

1.7. Codebase Structure and Flow

1.7.1. Overview of the Codebase

The codebase is organized into the following directories:

- `src` - Contains the source code for the Angular frontend application.
- `src/app` - Contains the Angular components, services, and modules.
- `src/assets` - Contains static assets like images, fonts, and stylesheets.
- `src/api` - Contains API endpoints and services for interacting with the backend.

1.7.2. Code Execution Flow

The code execution flow follows these steps:

1. The user interacts with the frontend application by browsing menus, selecting items, and placing orders.
2. The frontend application makes API calls to the backend to fetch data, submit orders, and track order status.
3. The backend application processes API requests, interacts with the database, and sends responses back to the frontend.
4. The database stores user data, restaurant data, order data, and other information needed by the application.
5. The payment gateway processes payment transactions and sends payment status updates to the backend.
6. The messaging service sends order updates and notifications to users via WhatsApp.
7. The application logic handles user authentication, authorization, and business logic for order processing.
8. The frontend application displays order status, menus, and other information to the user.
9. The backend application handles API requests, database queries, and external service integrations.
10. The database stores and retrieves data needed by the application.

11. The payment gateway processes payment transactions and sends payment status updates.
12. The messaging service sends order updates and notifications to users.
13. The application logic handles user authentication, authorization, and business logic.
14. The frontend application displays order status, menus, and other information.

1.7.3. Understanding Functions and Modules

1.7.4. Step-by-Step Explanation of a Key Feature

1.7.5. Reading the Code

1.7.6. Code Standards and Best Practices

1.8. API Documentation

1.8.1. Overview of API Usage and Purpose

1.8.2. API Endpoint List

1.8.3. Error Codes and Handling

1.8.4. How to Test APIs as a Beginner

1.9. Database Design

1.9.1. Database Schema Overview

1.9.2. Key Tables and Their Purpose

1.9.3. Entity-Relationship Diagrams (ERD)

1.9.4. Sample Queries for Common Use Cases

1.10. User Interface (UI)

1.10.1. Screenshots of All Pages (annotated with descriptions)

1.10.2. Navigation Map

1.10.3. Design Principles Used

1.11. Ad Hoc Process Configuration

1.11.1. Payment Gateway Integration

1.11.1.1. Overview of Payment Gateway Used

1.11.1.2. API Keys, Credentials, and Configuration Steps

1.11.1.3. Step-by-Step Guide for Setting Up Payment Flow

1.11.1.4. Handling Payment Status Updates (webhooks, callbacks)

1.11.1.5. Debugging Common Payment Issues## Ad Hoc Process Configuration

1.11.2. Payment Gateway Integration

1.11.2.1. Overview of Payment Gateway Used

1.11.2.2. API Keys, Credentials, and Configuration Steps

1.11.2.3. Step-by-Step Guide for Setting Up Payment Flow

1.11.2.4. Handling Payment Status Updates (webhooks, callbacks)

1.11.2.5. Debugging Common Payment Issues

1.11.3. Messaging Service Integration (e.g., SMS, WhatsApp)

1.11.3.1. Overview of Messaging Providers (e.g., Twilio, Firebase)

1.11.3.2. Setting Up API Access and Authentication

1.11.3.3. Sending SMS or WhatsApp Messages (sample code snippets)

1.11.3.4. Configuring OTP Logic and Expiry Timers

1.11.3.5. Error Handling and Logging for Message Delivery Failures

1.11.4. WhatsApp OTP Setup

1.11.4.1. Details of Provider (e.g., Meta’s WhatsApp Business API)

1.11.4.2. Required Credentials (Phone Numbers, API Tokens)

1.11.4.3. Configuring WhatsApp Templates for OTP Messages

1.11.4.4. Implementing Message Flow (request, send, verify OTP)

1.11.4.5. Troubleshooting Common WhatsApp API Issues

1.11.5. Government Compliance-Related Processes

1.11.5.1. Overview of Compliance Requirements

1.11.5.2. Step-by-Step Implementation (e.g., GST APIs, Aadhaar Verification)

1.11.5.3. Sample Data Formats for Government APIs

1.11.5.4. Error Handling and Validation for Compliance APIs

1.11.5.5. Documentation Links to Government APIs and Resources

1.12. Testing Guidelines

1.12.1. Overview of Testing Strategy

1.12.2. Functional Testing Scenarios

1.12.3. Technical Testing

1.12.4. Bug Reporting Guidelines

1.13. Deployment and Maintenance

1.13.1. Deployment Process (step-by-step)

1.13.2. Version Control Guidelines

1.13.3. Backup and Recovery Plan

1.14. Troubleshooting Guide

1.14.1. Common Issues and Fixes

1.14.2. Debugging Tips for Developers

1.15. Security Considerations

1.15.1. Security Practices Implemented

1.15.2. Guidelines for Handling Sensitive Data

1.16. FAQ

1.16.1. Common Questions by Non-Technical Staff

1.16.2. Questions Related to API Usage

1.16.3. Testing and Debugging FAQs

1.17. Appendix

1.17.1. Resources and References

1.17.2. Links to Tools, Libraries, and Frameworks Used

1.17.3. Glossary of Technical Terms