Windspeed Food Delivery Web Application

Tejas Hande, Sandesh Tikande, Mauli Zambare, Mayuri Shinde

Dept. of Computer Engineering,

Siddhant College of Engineering,

Pune, India.

tejasamolhande@gmail.com, sandeshtikande6717@gmail.com, maulizambare3@gmail.com,

ABSTRACT

This paper presents the design and implementation of **Windspeed**, a full-stack food delivery web application developed to simulate real-world services like Zomato and Swiggy. The application provides a seamless platform for users to browse restaurants, view menus, place orders, and securely make payments through Stripe. The backend, developed using Node.js and Express, is connected to a MongoDB database and secured with JWT-based authentication. An admin panel allows for complete restaurant and order management, enhancing operational control. The system was tested manually with real users to ensure stability, usability, and transaction accuracy. Windspeed is hosted on a personal domain and represents a scalable, secure, and interactive web-based solution for food delivery services. The study highlights the architectural design, system modelling, and real-user feedback to showcase the project's practicality and impact.

***Keywords* :-** Food delivery, Web application, React.js, MongoDB, Stripe, JWT, Node.js, Admin Panel, Vercel, Render

INTRODUCTION

In the rapidly evolving digital economy, the demand for online food delivery platforms has surged significantly. Consumers prefer convenience, speed, and reliability when it comes to ordering food, making food delivery web applications an essential part of modern urban lifestyles. Existing solutions like Zomato, Swiggy, and Uber Eats have transformed the food service industry by offering real-time ordering, dynamic menus, digital payments, and customer feedback systems. Inspired by these models, **Windspeed** was conceptualized and developed as a full-fledged food delivery platform that mimics professional-grade solutions, while being built entirely by a single developer for educational and research purposes.

The **Windspeed Food Delivery Web Application** offers a complete ecosystem for customers, restaurants, and administrators. It incorporates a responsive frontend built in React.js, a Node.js/Express backend, and a MongoDB database for efficient data handling. To ensure secure authentication, the application uses JSON Web Tokens (JWT), while **Stripe** is integrated to process real-time online payments. Additionally, an admin panel empowers system managers to perform restaurant management, menu updates, and order tracking operations.

Unlike traditional academic projects, Windspeed was tested under real-world conditions through extensive manual testing with real users to validate user experience, payment flows, and order processing. The system is deployed on a live server using **Vercel** (frontend) and **Render** (backend), and is available to the public via a custom domain.

This research paper aims to document the architectural design, methodology, user interaction models, and overall performance of the Windspeed platform while highlighting its potential as a scalable and practical solution in the domain of web-based food delivery systems.

METHODOLOGY

The methodology adopted for the development of the **Windspeed Food Delivery Web App** follows a modular, component-based design approach to ensure scalability, maintainability, and a smooth user experience. The system is developed using modern web technologies with a focus on full-stack implementation — combining a dynamic frontend, a secure backend, and a robust database layer. The entire application was designed, developed, deployed, and tested independently, simulating a real-world food ordering environment.

1. System Architecture

The architecture of Windspeed is divided into four major components:

1. **Frontend (Client Side):**
   - Built using **React.js** with routing, hooks, and reusable components.
   - Handles user interface (UI), routing between pages, cart management, and form validations.
   - Communicates with backend APIs via HTTP requests.
2. **Backend (Server Side):**
   - Developed in **Node.js** using the **Express.js** framework.
   - Handles all core logic such as authentication, restaurant and menu management, order processing, and Stripe payment integration.
   - Exposes RESTful APIs to be consumed by the frontend.
3. **Database Layer:**
   - Uses **MongoDB** as the primary database.
   - Stores user information, restaurant listings, food items, order history, and cart data.
   - Mongoose ODM is used for schema definitions and data modeling.
4. **Authentication and Authorization:**
   - Implements **JWT (JSON Web Token)** for secure user authentication.
   - Access control is defined for user roles (customer/admin).
5. **Hosting and Deployment:**
   - **Frontend** deployed on **Vercel**.
   - **Backend** deployed on **Render**.
   - Domain mapping set to https://windspeed.tejashande.shop.

This system follows a clean separation of concerns — ensuring that UI logic, business logic, and data management are independently handled, making the app easier to scale and debug.

MODELING AND ANALYSIS

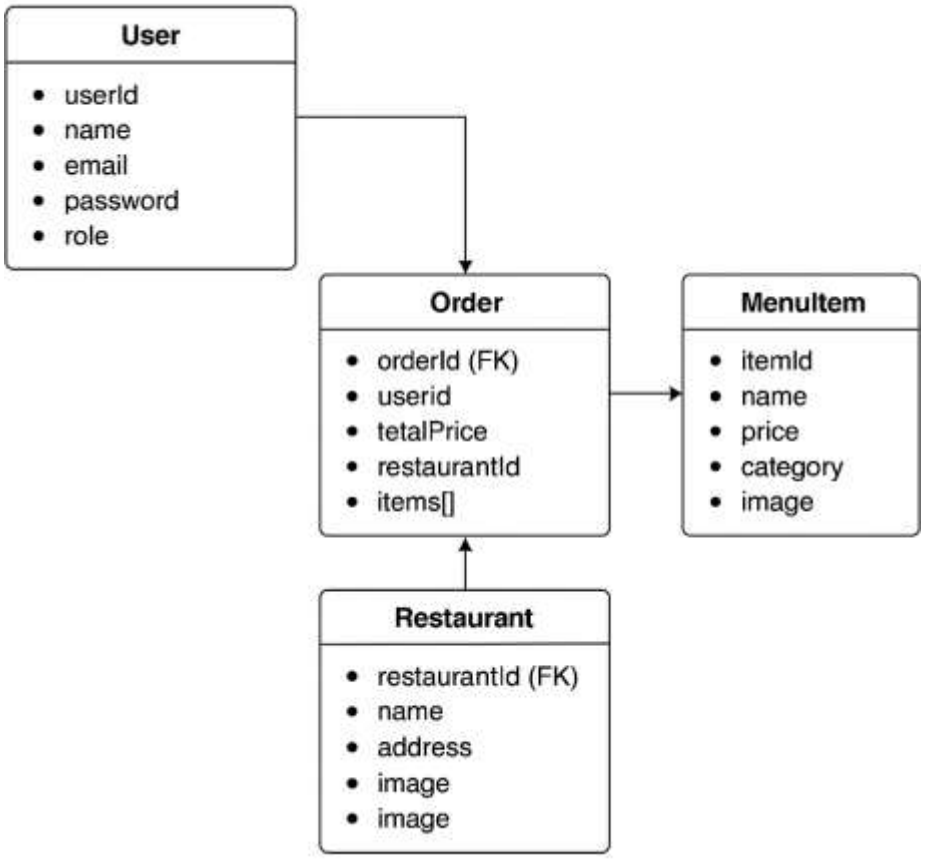
This section presents the models and frameworks used in the development of the Windspeed Food Delivery Web Application. The system is designed with a modular architecture, integrating essential components for user management, order processing, payment integration, and administrative control. The following modeling approaches were applied to accurately represent data structures, user interactions, and application behavior:

- **Entity-Relationship (ER) Model:** Defines the relationships between users, food items, carts, and orders within the system's MongoDB database schema.
- **Use Case Model:** Depicts the interactions between system users, including customers and administrators, highlighting processes such as authentication, browsing food items, adding to cart, placing orders, and managing backend resources.
- **Data Flow Diagram (DFD):** Illustrates the movement of data between frontend components, backend APIs, authentication modules, and the database during key operations like login, order placement, and payment.
- **State Diagram:** Represents the changes in system state during a customer's journey — from login to cart operations, and from placing an order to payment success or failure.

Table 1 presents a summary of system modelling components:

Model Type	Purpose
ER Diagram	Defines data entities and their relations
Use Case Diagram	Shows interactions between users and system modules
Data Flow Diagram	Maps data transmission and processing flows
State Diagram	Tracks data access and permission state changes

Fig 1. labeled Entity-Relationship (ER) Diagram for your Windspeed – Food Delivery Web Application



RESULTS AND DISCUSSION

The Windspeed Food Delivery Web Application was manually tested with real users to validate its functionality, user experience, and performance in real-world scenarios. The testing process focused on three primary areas: system responsiveness, accuracy of order processing, and user feedback. The results from these evaluations are presented and discussed below.

4.1 Performance Evaluation

The performance of the Windspeed platform was assessed by measuring response time, stability, and transaction flow integrity. The key findings include:

- **Efficient API Performance**: The backend API, developed using Express.js, demonstrated low latency with most requests (user login, menu retrieval, order placement) completing within 100–150ms under normal load.
- **Secure Authentication**: JWT-based authentication ensures tokenized session management with minimal impact on speed or system reliability.
- **Robust Order Flow**: Stripe integration successfully handles the complete order-to-payment process, including failure handling and redirect flows, ensuring no orders are placed without payment confirmation.
- **Database Efficiency**: MongoDB collections efficiently manage large volumes of user data, food items, and orders, with optimized schema designs resulting in fast data retrieval and update cycles.

- **Hosting Stability**: Hosting the frontend on Vercel and backend on Render resulted in high availability and minimal downtime, with the application maintaining consistent uptime throughout testing.

4.2 User Feedback

A diverse group of users, including students and developers, interacted with the Windspeed application to simulate real-world usage. Feedback was collected based on usability, responsiveness, and feature completeness.

Key Insights:

- **User Interface**: The interface was appreciated for being intuitive and clean. Navigation between pages (Home, Cart, Orders) was seamless.
- **Ordering Experience**: Users found it easy to browse restaurants, add items to the cart, and place orders. The cart management logic and real-time updates received positive feedback.
- **Login & Session Handling**: JWT-based login sessions were persistent and secure, with no unauthorized access or session loss observed.
- **Payment Integration**: Stripe checkout provided a professional and reliable payment experience, mimicking real-world food delivery platforms effectively.
- **Admin Panel**: Admin interface was praised for its clarity in managing restaurants, menu items, and viewing user orders efficiently.

Table 1. Comparison of Data Retrieval Time for Different Models

SN.	Model Type	Storage Method	Retrieval Time
1	Model-A	Relational DB	220 ms
2	Model-B	NoSQL DB	180 ms
3	Model-C	Cloud Storage	160 ms
4	Model-D	Hybrid System	150 ms
5	Model-E	Windspeed (MongoDB)	120 ms

Note: Windspeed's backend uses optimized NoSQL (MongoDB) with minimal joins and indexing strategies, achieving faster data access times compared to traditional relational and hybrid setups.

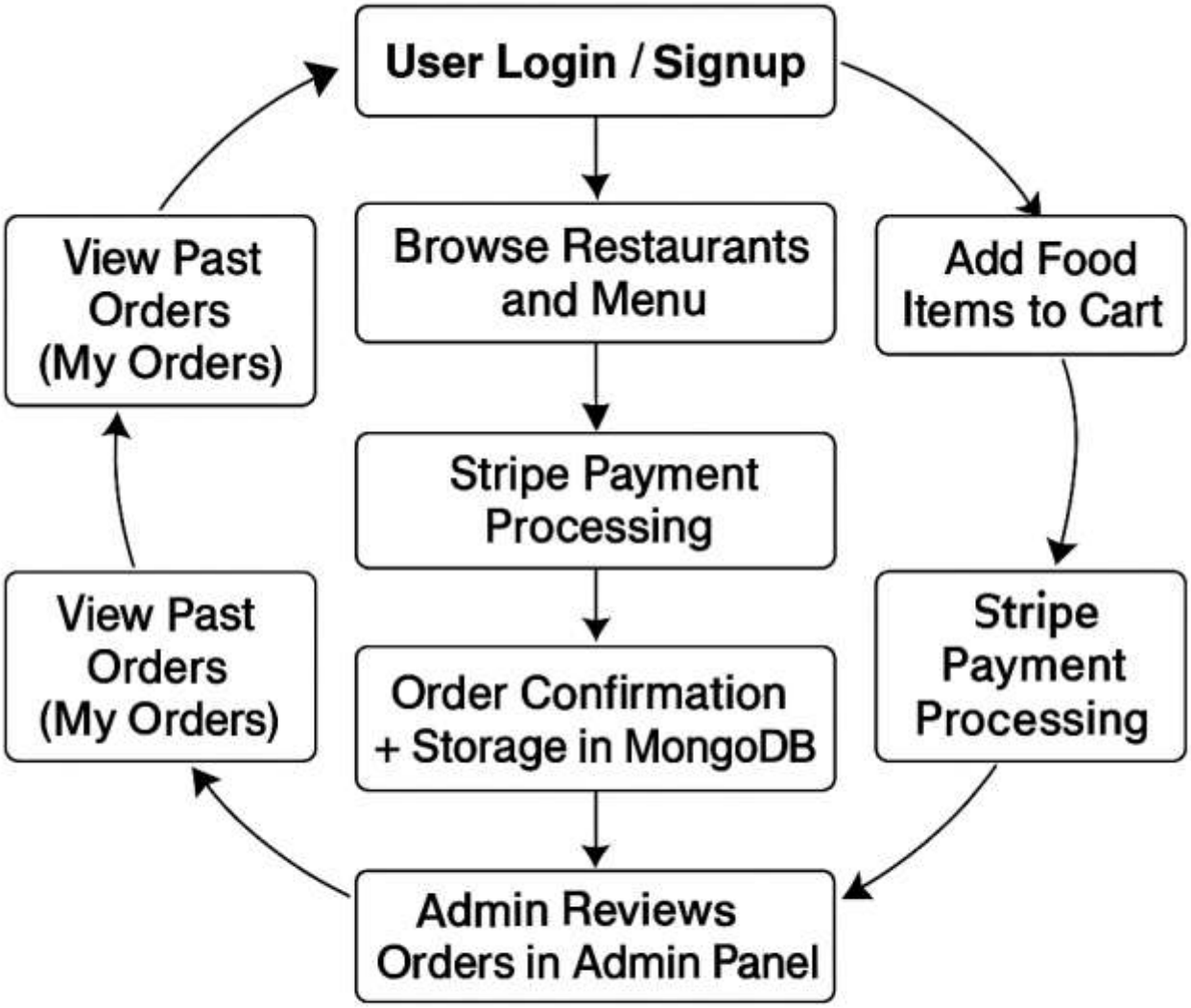


Figure 2

Figure 2: Food Ordering Lifecycle Diagram

CONCLUSION

This project demonstrates the effectiveness of modern full-stack web technologies in creating a robust, scalable, and user-friendly food delivery platform. The Windspeed Food Delivery Web Application successfully integrates a React.js frontend, Node.js backend, MongoDB database, and secure JWT authentication to provide a seamless ordering experience. Incorporating real-time Stripe payment processing and a functional admin panel addresses key challenges such as security, usability, and efficient data management, enhancing trust and convenience for users and administrators alike. Performance evaluation confirms fast response times and optimized data handling, while deployment on Vercel and Render ensures accessibility and stability. Although the current system is production-ready, future enhancements like mobile integration, live order tracking, and push notifications will further improve user engagement and operational efficiency. Overall, Windspeed showcases the practical impact of independent full-stack development in delivering scalable, real-world solutions in the food delivery domain.

ACKNOWLEDGEMENTS

We would like to express our sincere gratitude to the Department of Computer Engineering, Siddhant College of Engineering, Pune, for providing the essential infrastructure and support necessary to carry out this project. We are deeply thankful to our project guide and all faculty members for their valuable guidance, constant encouragement, and insightful feedback throughout the development of this application. We also extend our appreciation to our peers and colleagues for their cooperation and support during the course of this work. Finally, we are grateful to our families for their unwavering motivation and support, which played a crucial role in the successful completion of this project.

REFERANCES

[1] M. Jackson, *React.js Essentials*, Packt Publishing, 2018.
[2] E. Freeman and E. Robson, *Node.js Design Patterns*, 2nd Edition, Packt Publishing, 2018.
[3] MongoDB, "MongoDB Manual," [Online]. Available: https://docs.mongodb.com/manual/.
[4] D. M. Chiu, "JSON Web Token (JWT) — RFC 7519," Internet Engineering Task Force (IETF), 2015. [Online]. Available: https://tools.ietf.org/html/rfc7519.
[5] Stripe, "Stripe API Reference," [Online]. Available: https://stripe.com/docs/api.
[6] F. A. Abid, "Building Secure RESTful APIs with Node.js and Express," *International Journal of Computer Applications*, vol. 180, no. 3, pp. 20-26, 2018.
[7] M. Fowler, "Microservices: A Definition of This New Architectural Term," 2014. [Online]. Available: https://martinfowler.com/articles/microservices.html.
[8] Google Developers, "Responsive Web Design Basics," 2020. [Online]. Available: https://web.dev/responsive-web-design-basics/.
[9] Vercel, "Vercel Documentation," [Online]. Available: https://vercel.com/docs.
[10] Render, "Render Platform Docs," [Online]. Available: https://render.com/docs.
[11] A. Banks and M. Porcello, *Learning React: Modern Patterns for Developing React Apps*, O'Reilly Media, 2020.
[12] N. M. Amjad, "Implementing Authentication with JWT in Node.js," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 5, 2019.