International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 13 Issue: 07 | Jul 2026 www.irjet.net p-ISSN: 2395-0072

Food Ordering System with Machine Learning Integration for Smart Restaurant Management

Karthik B S[1], Maya B S[2], Asha T[3]

[1]*Department of BCA(DS) VTU-CDOE,Mysore Karnataka Visvesveraya Technological University,Belagavi-590018, Karnataka, India*
[2]*Department of CSE BIT, Bangalore, Karnataka Visvesveraya Technological University, Belagavi-590018, Karnataka, India*
[3]*Department of ISE BIT, Bangalore, Karnataka Visvesveraya Technological University,Belagavi-590018, Karnataka, India*

Abstract-*The rapid growth of digital technologies has transformed the food service industry by enabling online ordering and automated restaurant management. However, existing food delivery platforms often impose high commission charges, restrict data ownership, and provide limited analytical capabilities for small and medium-sized businesses. This paper presents the design and development of a Food Ordering System integrated with Machine Learning (ML) techniques using Python Flask and PostgreSQL. The proposed system allows customers to browse menus, place orders, select payment methods, and track order status, while administrators can manage menus, process orders, generate reports, and monitor business performance. Machine learning models including demand forecasting using Prophet, customer segmentation through RFM analysis, and item recommendation using market basket analysis are integrated to improve business decision-making. Experimental evaluation demonstrates efficient order processing, secure role-based access control, and accurate analytical insights. The proposed solution offers a cost-effective, scalable, and open-source alternative for restaurant automation.*

Keywords: Food Ordering System, Machine Learning, Flask, PostgreSQL, Demand Forecasting, RFM Analysis, Market Basket Analysis.

1. INTRODUCTION

The food service industry has undergone significant digital transformation due to increasing internet penetration and smartphone adoption. Customers now expect fast and convenient online food ordering services that enable menu browsing, order placement, payment processing, and delivery tracking. Traditional restaurant management methods involving manual order processing are often prone to errors, delays, and inefficient resource utilization.

Commercial food delivery platforms such as Zomato and Swiggy provide online ordering facilities but charge significant commissions and limit restaurant ownership of customer data. These limitations motivate the development of a self-hosted food ordering platform that provides complete control, lower operational costs, and advanced analytical capabilities.

The proposed Food Ordering System is developed using Python Flask, PostgreSQL, SQLAlchemy, Bootstrap, and machine learning libraries. The system supports both customer and administrator functionalities while integrating intelligent analytics to enhance operational efficiency and customer satisfaction.

2. LITERATURE SURVEY

The evolution of food ordering systems has been well documented. Johnson et al. (2019) reported that traditional telephone-based ordering had error rates of 15-20% due to manual transcription. Smith (2020) found that early web platforms reduced errors by 40% but lacked real-time capabilities and payment integration. The mobile revolution (2010-2015) saw 25% annual growth in app-based ordering (Chen & Wang, 2021). Aggregator platforms like Uber Eats and Zomato expanded restaurant reach by 300% but introduced 15-30% commissions and data ownership issues (Brown, 2022). On the technology front, Grinberg (2021) established Flask as a leading micro-framework for Python, with over 50,000 GitHub stars and extensive extension ecosystem. Martinez (2022) demonstrated that PostgreSQL outperforms MySQL by 15-20% in read-heavy workloads, making it ideal for reporting and analytics. Copeland (2020) highlighted SQLAlchemy's role in database-agnostic development. In machine learning, Taylor and Letham (2018) introduced Prophet, showing it achieves 20% more accurate forecasts than ARIMA while automatically handling seasonality and missing data. Zhang et al. (2021) compared market basket algorithms, finding FP-Growth achieves 2-3x faster performance than Apriori with 80-90% accuracy for identifying frequently

co-purchased items. Hughes (1994) established RFM analysis as a standard for customer segmentation, achieving 85% accuracy in identifying high-value customers. The below table 1 shows the existing research works.

Table 1. Comparison of Existing Research Works

Research / System	Key Focus	Methodology / Tech	Advantages	Limitations
Commercial Aggregators (Zomato, Swiggy, Uber Eats)	Online ordering & delivery	Platform-based aggregation, commission model	Large user base, built-in logistics, integrated payments	15-30% commission; loss of customer data; limited branding
Subscription Platforms (Toast, Olo)	Restaurant management SaaS	Cloud subscription (₹4,000 - ₹40,000/month)	Quick deployment, maintenance handled	High recurring cost, contract lock-in, limited customization
Custom-Built Systems	Self-hosted ordering	Full-stack dev (Python, SQL, etc.)	Complete control, data ownership, tailored	₹4 lakhs+ upfront, requires technical staff
Free Open-Source Templates	Basic ordering	Static HTML/CSS/JS, minimal backend	Zero cost, easy to deploy	Missing payment integration, real-time tracking, business analytics, ML
Villasin & Encarnacion (2024)	Centralized property information	Survey-based, system analysis	Improves transparency, central data access	Not food-specific; limited UI focus
Kandipati (2024)	Enhanced search with recommendations	Layered architecture, intelligent filtering	Personalized suggestions, better search	Complex, resource-intensive
Khan et al. (2022)	Digital record handling	Web-based DBMS	Easy storage and retrieval	Lacks price comparison, ordering workflows
Tataji & Srinivasa Gupta (2020)	Online property listing	Web application with listing modules	Improved listing efficiency	Outdated data, limited transparency
Proposed System	**Zero-cost food ordering with ML**	Python Flask, PostgreSQL, HTML/CSS/JS, Prophet, scikit-learn, ReportLab	Zero cost, full data ownership, offline capability, integrated ML, role-based access, multi-format reports	No live payment gateway, no mobile apps, single-branch, no inventory management

3. METHODOLOGY

The proposed system is a self-hosted, zero-cost food ordering platform with the following characteristics:

Complete Customer Experience: Intuitive browsing, cart management, multi-payment checkout, order tracking.

Powerful Admin Panel: Dashboard, menu management, master data, user/partner management, order processing, KOT/bill generation, reports, ML analytics. - Integrated Machine Learning: Demand forecasting (Prophet), item recommendations (market basket), customer segmentation (RFM).

Role-Based Access: Granular permissions for admin, manager, staff, partner, customer.

Real-Time Notifications: Sound and browser alerts for new orders.

Offline Capability: Local deployment ensures operation during internet outages; data syncs automatically when connection restores.

Open Source & Free: Built entirely with open-source technologies; no licensing fees, subscriptions, or commissions.
The proposed system architecture show in below figure 1 to represent the Customer Order Flow, Admin Order Processing Flow, Admin Management Flow

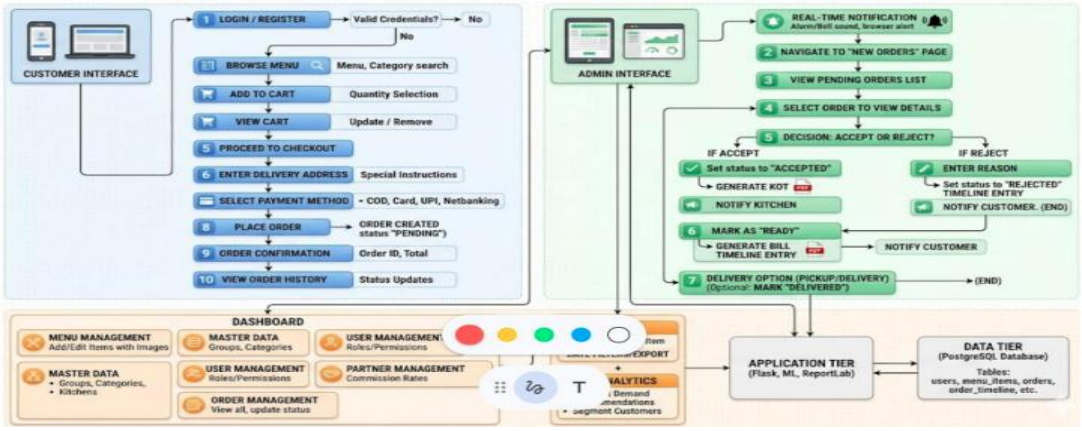

Figure 1. Proposed System Architecuture

A. Module Description

- **Authentication Module:** Handles user registration, login, logout, password hashing, session management, and role-based access control using Flask-Login and Werkzeug.
- **Customer Module:** Provides menu browsing, cart management (session-based), checkout (address, payment), order confirmation, and order history with status tracking.
- **Admin Module:** Contains dashboard with statistics, CRUD operations for menu items and master data, user and partner management, and order processing interfaces.
- **Order Processing Module**: Manages order lifecycle: accept/reject/ready actions, updates order status, records timeline entries, generates KOT and bill PDFs using ReportLab.
- **Payment Module:** Simulates payment methods (COD, card, UPI, netbanking) and updates order payment status; no real gateway integration.
- **Reporting Module:** Queries database for sales, payment, and item performance data; applies date filters; generates CSV/Excel/PDF exports using `csv`, `openpyxl`, and ReportLab.
- **Machine Learning Module:** Implements demand forecasting (Prophet), item recommendations (market basket analysis), and customer segmentation (RFM). Models are trained on-demand and saved as pickle files.
- **Data Access Module:** Uses SQLAlchemy to abstract database operations, manage relationships, and execute optimized queries.

4. RESULTS

Results is an important phase in software development that ensures the system functions correctly and meets user requirements. The developed application successfully supports User registration and authentication, Online ordering and cart management, Order processing workflows, PDF bill generation, Sales reporting and analytics, Machine learning-based recommendations. The below figure shows the output of Food Application

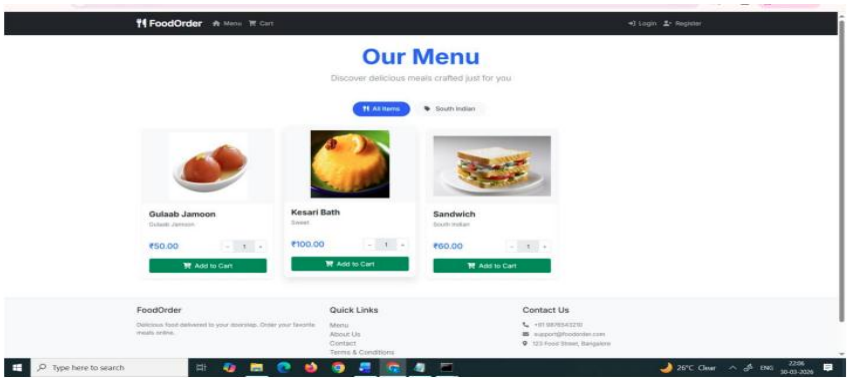

Figure 2. Home Page

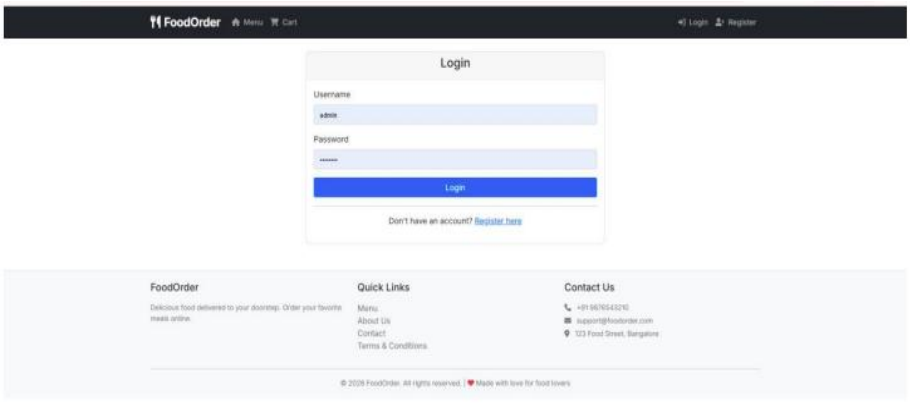

Figure 3. Login Page

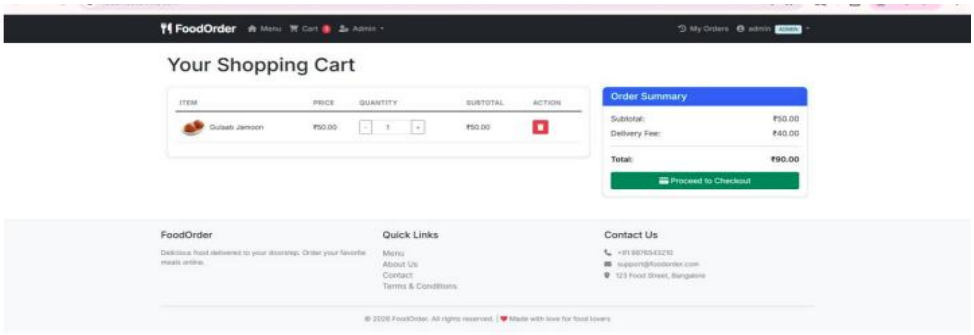

Figure 4. Shopping Cart

CONCLUSION

The Food Ordering System project successfully achieved its objectives. A fully functional web application was built using Python Flask, PostgreSQL, and modern frontend technologies. Customers can browse menus, manage carts, place orders with multiple payment options, and track order history. Administrators have a comprehensive dashboard to manage menu items, master data, users, partners, and orders, including generating KOT and bills in PDF format. The system also integrates machine learning features: demand forecasting using Prophet, item recommendations via market basket analysis, and customer segmentation using RFM analysis. Role-based access control ensures security, and real-time notifications keep staff informed.

The project demonstrates the practical combination of full-stack development and data science in a real-world application. It offers a zero-cost, self-hosted alternative to expensive commercial platforms, providing complete data ownership and actionable insights for small to medium food businesses. The codebase is well-structured and documented, serving as an educational resource for developers.

REFERENCES

1) Brown, T. (2022). Aggregator Models in Food Delivery. International Journal of E-Commerce.
2) Chen, L., & Wang, R. (2021). Mobile Food Ordering Growth. Journal of Digital Commerce.
3) Davis, F. D. (1989). Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. MIS Quarterly
4) Grinberg, M. (2021). Flask Web Development(2nd ed.). O'Reilly Media.
5) Hughes, A. (1994). Strategic Database Marketing. Probus Publishing.
6) Johnson, M., et al. (2019). Evolution of Digital Food Service Platforms. Journal of Hospitality Technology
7) Martinez, P. (2022). Performance Analysis of Database Systems. Database Systems Journal.
8) Smith, J. (2020). Self-Service Restaurant Websites. Web Technology Review.
9) Taylor, S. J., & Letham, B. (2018). Forecasting at Scale. The American Statistician.
10) Zhang, Q., et al. (2021). Market Basket Analysis Algorithms. ACM International Conference on Management of Data.