iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: https://doi.org/10.22214/ijraset.2026.79759

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

Role-Based Online Food Ordering and Delivery System Using Django and RESTful Architecture

Oleti Durga Prasad[1], Pedakapu Surya Prasad[2], Ravipati Raja[3], Madasu Hemanth[4], Y. Satya Vinod[5]
[1, 2, 3,4]*Department of Electronics and Communication Engineering, Bonam Venkata Chalamayya Engineering College, Affiliated to JNTU Kakinada, Andhra Pradesh-533210, India*
[5]*Project Guide, Department of Electronics and Communication Engineering, Bonam Venkata Chalamayya Engineering College, Affiliated to JNTU Kakinada, Andhra Pradesh-533210, India*

Abstract: ***Traditional food ordering processes in restaurants rely heavily on manual interactions, including in-person ordering, phone-based bookings, and unstructured coordination between customers, restaurants, and delivery personnel. These approaches lead to inefficiencies such as delayed order processing, incorrect order handling, lack of real-time tracking, and poor coordination among stakeholders. Although modern applications exist, many academic implementations lack modular architecture, role-based access control, and scalable backend design.***
This paper presents the design and implementation of a role- based Online Food Ordering and Delivery System developed using Django and RESTful architectural principles. The system follows a modular architecture separating user roles including Admin, Customer, Restaurant, and Delivery personnel. Each module is designed with clear responsibility boundaries, improv- ing maintainability and scalability.
Core functionalities include food browsing, cart management, order placement, restaurant-side order handling, and delivery tracking. The system ensures data consistency through central- ized database management and structured workflows. Role-based access control ensures secure and restricted operations across modules.
The implementation demonstrates how modern web frame- works like Django can be used to build scalable, role-driven systems with clear separation of concerns. The system serves as a foundation for further enhancements toward real-world de- ployment, including payment integration and real-time tracking.
Index Terms: Spring Boot, REST API, role-based access control, transactional integrity, business rule validation.

I. INTRODUCTION

Food service management systems play a crucial role in modern urban lifestyles, enabling efficient coordination be- tween customers, restaurants, and delivery personnel. Within this domain, online food ordering represents a fundamental yet operationally complex process requiring seamless interaction across multiple stakeholders including customers, restaurant staff, administrators, and delivery agents [1]. Traditional food ordering approaches—primarily involving physical visits or telephone-based requests—introduce procedural inefficiencies such as delayed order processing, miscommunication, manual billing errors, and lack of real-time tracking [2].

Conventional workflows require customers to visit restau- rants or place orders via phone calls, after which restau- rant staff manually record orders and coordinate preparation and delivery. This process often leads to extended waiting times, incorrect order handling, and absence of centralized order tracking mechanisms [3]. Additionally, communication gaps between restaurants and delivery personnel increase operational delays and reduce service efficiency. The lack of integrated systems also limits transparency for customers regarding order status and estimated delivery times.

Academic research has explored digital food ordering solu- tions ranging from basic web-based platforms to mobile ap- plications and enterprise-level delivery systems [4]. However, existing implementations reveal several architectural and func- tional limitations. Many systems lack proper modular design, resulting in tightly coupled components that reduce scalability and maintainability. Role-based access control is often weak or absent, allowing unauthorized access to critical operations. Furthermore, database design and workflow coordination are frequently insufficient, leading to inconsistent data handling and inefficient order lifecycle management [5].

This paper presents an Online Food Ordering and Delivery System developed as an academic prototype demonstrating modern web application design principles and role-based workflow management. The system is implemented using the Django framework, following a structured client–server architecture that separates presentation, application logic, and data persistence layers.

©IJRASET: All Rights are Reserved

The primary focus is on order lifecycle management—from food browsing and cart operations to order placement, restaurant processing, and delivery coordi- nation—with supporting modules for user management, food item administration, and system control.

The implementation contributes to academic understanding of full-stack web application development through several aspects. First, it demonstrates modular system design with clear separation of responsibilities across multiple user roles including Admin, Customer, Restaurant, and Delivery personnel. Second, the system enforces structured workflows for order processing, ensuring logical transitions between different stages of the order lifecycle. Third, centralized database man- agement ensures consistency and integrity of data across all modules. Fourth, the system illustrates practical implementa- tion of web technologies including Django, HTML, CSS, and database integration for building scalable applications.

The remainder of this paper is organized as follows. Section II reviews related work and identifies existing gaps in food ordering systems. Section III presents the system architecture and technology stack. Section IV details the design and implementation of core modules with emphasis on work- flow management and system integration. Section V presents functional validation and performance analysis. Section VI discusses limitations and potential improvements. Section VII concludes the paper.

II. LITERATURE REVIEW

Online food ordering systems have been widely studied as representative applications of workflow automation and e- commerce platforms. This section reviews existing academic implementations and identifies recurring architectural and functional limitations relevant to food ordering and delivery systems.

A. Web-Based Implementations

Early academic efforts focused on replacing traditional restaurant ordering processes with web-based interfaces and database-backed storage. Adamu [1] developed a PHP-based food ordering system incorporating menu browsing and order placement functionalities, demonstrating improved efficiency compared to manual ordering methods. However, the proce- dural design lacked proper modularization and service-layer abstraction, limiting system scalability and maintainability. Alade et al. [3] proposed a similar PHP–MySQL system for small-scale restaurant operations, successfully digitizing order workflows but exhibiting weak validation mechanisms and limited coordination between order processing and delivery modules.

These studies highlight a common trend in early implemen- tations: strong emphasis on user interface design and basic CRUD operations, with minimal attention to architectural layering, role-based access control, or workflow management.

B. Mobile and Platform-Specific Systems

Some research explored mobile-based solutions for food ordering. Kaushik et al. [13] developed an Android-based application enabling users to browse menus and place orders directly from mobile devices, improving accessibility and user convenience. While effective within its scope, the platform- specific design restricted accessibility across different devices and limited integration with web-based administrative systems. Similar limitations were reported by Sapona et al. [9], where lack of cross-platform compatibility constrained system scalability and interoperability with other services such as delivery tracking and payment systems.

C. Framework-Based Implementations

More recent studies adopted modern web development frameworks to build scalable food ordering platforms. Birje et al. [6] implemented a food ordering system using the MERN stack, demonstrating responsive user interfaces and REST API-based communication. However, the study iden- tified limitations in handling complex workflows, particularly in coordinating order processing and delivery tracking across multiple modules. Additionally, data consistency issues arose due to insufficient transaction handling in concurrent opera- tions.

Other works explored intelligent automation in food ser- vices. Harshika et al. [2] proposed an AI-driven food recom- mendation and ordering system incorporating user preferences and predictive analytics. While innovative, such systems re- quire large datasets and computational resources, making them less suitable for small and medium-scale implementations. Furthermore, integration between recommendation systems and core order processing workflows was not fully addressed.

D. Identified Gaps

Analysis of existing literature reveals several recurring gaps. First, many systems lack clear modular architecture, with tightly coupled components that reduce flexibility and maintainability.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

Second, role-based access control is often insufficient, leading to security vulnerabilities and improper access to system functionalities. Third, workflow coordination between customers, restaurants, and delivery personnel is frequently incomplete, resulting in inefficient order lifecycle management [8].

Additionally, real-time order tracking and synchronization between modules are often missing or poorly implemented. Database design in many systems does not adequately support complex relationships between users, orders, and delivery enti- ties. Inconsistent handling of order states—such as placement, processing, and delivery—creates ambiguity and operational inefficiencies [10].

The system presented in this paper addresses these gaps through a role-based modular architecture, clear separation of system components, structured order lifecycle management, and centralized database design ensuring data consistency across all modules.

III. SYSTEM ARCHITECTURE

The proposed Online Food Ordering and Delivery System adopts a three-tier client–server architecture separating presen- tation, application logic, and data persistence concerns.

A. Three-Tier Architecture

The Presentation Tier consists of a web-based user interface developed using HTML, CSS, and Django templates, providing interfaces for food browsing, cart management, order placement, and administrative operations. Different dashboards are provided for customers, restaurants, delivery personnel, and administrators. Communication with the backend occurs

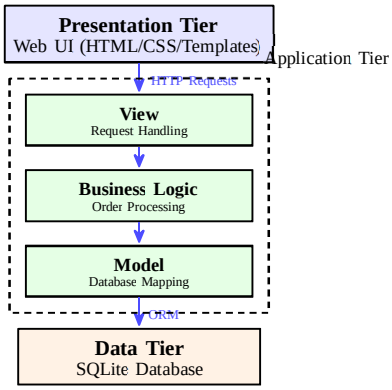


Fig. 1. Three-tier architecture illustrating separation of concerns between presentation, application logic, and persistent storage.

through HTTP requests handled by Django views, ensuring structured interaction between frontend and backend compo- nents.

The Application Tier is implemented using the Django framework and encapsulates all business logic, request han- dling, and workflow coordination. Internally, it follows a Model–View–Template (MVT) architectural pattern. Views handle HTTP requests and responses, Models define database schema and relationships, and Templates manage presentation logic. The application layer coordinates core functionalities including order processing, user authentication, food manage- ment, and delivery assignment.

The Data Tier employs an SQLite relational database for persistent storage, managing entities such as users, customers, restaurants, food items, orders, and delivery records. The database ensures data consistency and integrity through struc- tured relationships and constraints. The overall architecture is illustrated in Fig. 1.

B. System Modules and API Design

The application is organized into multiple role-based mod- ules: Admin module (system control), Customer module (food browsing and ordering), Restaurant module (food and order management), and Delivery module (order delivery tracking). Each module interacts with the backend through structured URL routing handled by Django views.

Endpoints follow resource-oriented patterns such as order creation, user registration, and food management. HTTP meth- ods are used appropriately: POST for creating resources (e.g., placing orders), GET for retrieving data (e.g., viewing food items), and PUT/UPDATE operations for modifying order status. JSON and form-based data are used for communication.

The system module organization is presented in Fig. 2.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

C. Data Model and Entity Relationships

The data model comprises multiple entities with defined relationships. The User entity stores authentication credentials and role types (Admin, Customer, Restaurant, Delivery). The Customer entity extends user details with ordering capabilities.
The Restaurant entity manages food items and incoming

Module	Endpoint	Method	Purpose
Auth	/register	POST	User registration
Auth	/login	POST	User login
Customer	/foods	GET	View food items
Customer	/cart	GET	View cart
Customer	/place-order	POST	Place order
Restaurant	/add-food	POST	Add food item
Restaurant	/manage-orders	GET	View orders
Restaurant	/update-status	POST	Update order status
Delivery	/assigned-orders	GET	View assigned orders
Delivery	/deliver-order	POST	Mark as delivered
Admin	/manage-users	GET	Manage users
Admin	/manage-restaurants	GET	Manage restaurants

Fig. 2. System module and endpoint organization across backend components.

orders. FoodItem stores menu details including name, price, and description. Order captures order details with references to customer and restaurant, including order status. Delivery records track assigned delivery personnel and delivery status.
Entity relationships are implemented using Django ORM with foreign key associations ensuring referential integrity.
One-to-many relationships exist between Restaurant and Food- Item, Customer and Order, and Order and Delivery. These relationships enable structured data flow across modules. The entity relationship diagram is shown in Fig. 3.

D. Technology Stack Selection

Django provides rapid development capabilities with built- in features such as ORM, authentication, and URL routing. SQLite is used for lightweight relational data storage suitable for academic implementations. HTML, CSS, and template rendering provide user interface functionality. The system demonstrates full-stack development principles, with empha- sis on backend architecture, modular design, and role-based workflow implementation rather than frontend complexity.

IV. SYSTEM DESIGN AND IMPLEMENTATION

This section describes the design and implementation of core functional modules with emphasis on workflow manage- ment, validation logic, and integration across system components.

A. Authentication and Role Management

The authentication module implements user registration and login functionalities supporting multiple roles including Admin, Customer, Restaurant, and Delivery personnel. User

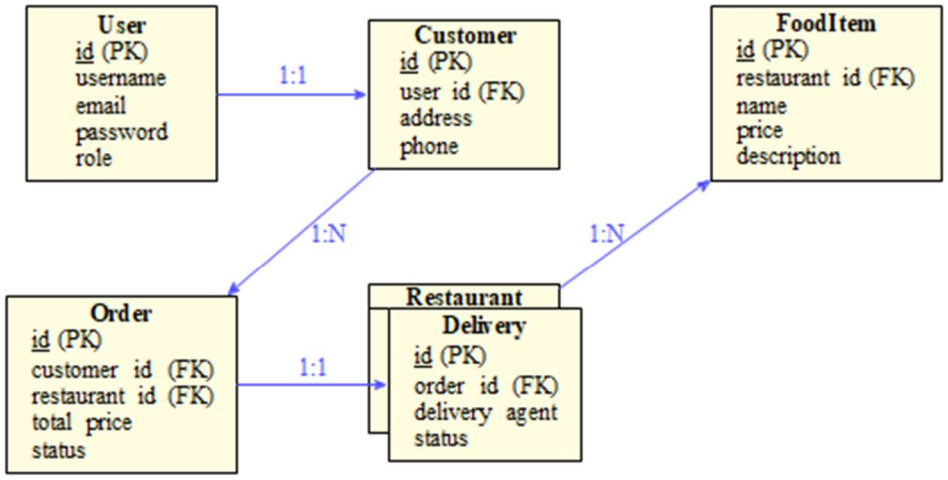


Fig. 3. Entity-relationship diagram representing the database schema and inter-module dependencies.

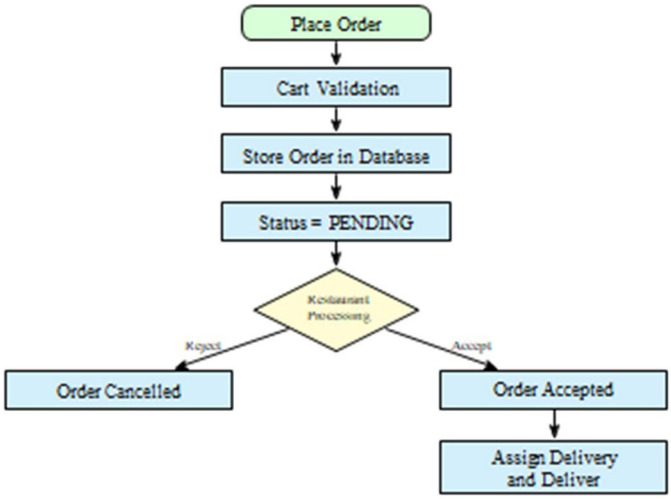


Fig. 4. Lifecycle of food order processing from placement to delivery.

registration ensures uniqueness of username and email be- fore creating user records in the database. Login authentica- tion validates credentials and redirects users to role-specific dashboards. Django's built-in authentication system manages sessions securely, enabling role-based access control across different modules. This approach simplifies user management while ensuring restricted access to system functionalities.

B. Order Lifecycle and Processing

Order management represents the core functionality of the system, demonstrating structured workflow coordination between customers, restaurants, and delivery personnel. Order placement begins when a customer selects food items and submits a request through the system. The complete lifecycle is illustrated in Fig. 4.

1) *Validation Framework:* The system implements validation at multiple levels. (1) Cart validation ensures that selected food items exist and quantities are valid. (2) Price validation calculates total cost dynamically to prevent manipulation. (3) User validation ensures only authenticated users can place orders. (4) Restaurant availability validation ensures orders are placed only for active restaurants.
2) *Order Management Logic:* Order processing is handled through Django views and models. When a user places an or- der, the system creates an Order entity linked to Customer and Restaurant entities. Order status transitions follow a predefined workflow: PENDING, ACCEPTED, PREPARING, OUT FOR DELIVERY, and DELIVERED. Each transition is triggered by specific roles ensuring controlled workflow progression.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

3) *Data Consistency Design:* A key design consideration ensures consistency of order data across modules. Orders are stored centrally and updated as they progress through different stages. This prevents duplication and ensures synchronized data between customer, restaurant, and delivery modules. Database constraints enforce referential integrity between re- lated entities.
4) *Workflow Coordination:* The system coordinates multiple modules during order processing. Restaurant users can view incoming orders and update status, while delivery personnel receive assigned orders for delivery. Each role interacts with the same order data but performs restricted operations based on permissions, ensuring controlled workflow execution.

C. Food and Restaurant Management

The Restaurant module manages food items and order handling. Restaurants can add, update, and delete food items through dedicated interfaces. Each food item is associated with a specific restaurant, demonstrating one-to-many relationships. Order management functionality allows restaurants to view incoming orders and update their status during preparation.

D. Customer and Cart Management

The Customer module provides functionalities for browsing food items, adding items to cart, and placing orders. The cart system temporarily stores selected items and calculates total cost before checkout. This module ensures smooth interaction between user interface and backend processing.

E. Delivery Management

1) *Delivery Module:* The Delivery module manages order delivery operations. Delivery personnel can view assigned orders and update delivery status. This module acts as the final stage in the order lifecycle, ensuring successful completion of the workflow.
2) *Order Tracking:* The system provides basic order track- ing functionality where customers can view the current status of their orders. Although real-time tracking is not imple- mented, status updates provide visibility into order progress.

F. Admin Management

The Admin module provides system-level control, enabling management of users, restaurants, and overall system data. Admin users can monitor system activity and ensure proper functioning of all modules.

G. Data Transfer and Exception Handling

The system uses Django models to handle data transfer between database and application layers. Form validation ensures correctness of input data before processing. Exception handling mechanisms manage errors such as invalid inputs, missing data, and unauthorized access, ensuring stable system behavior.

H. Implementation Scope and Simplifications

Several simplifications distinguish this implementation from production systems. Payment integration is not implemented, and order processing does not include real-time tracking or notifications. The system uses SQLite database suitable for academic purposes but not optimized for large-scale deploy- ment. Security mechanisms are basic, relying on Django authentication without advanced encryption or token-based systems. These simplifications focus on demonstrating core concepts such as modular design, role-based workflows, and database integration.

V. RESULTS AND PERFORMANCE EVALUATION

This section presents functional validation outcomes and analytically derived performance characteristics of the imple- mented Online Food Ordering and Delivery System based on architectural design and operational behavior.

A. System Performance Characteristics

System performance characteristics are derived from archi- tectural analysis rather than empirical stress benchmarking. Simple read-oriented operations such as food item retrieval, menu browsing, and order listing exhibit low latency due to efficient database queries and minimal processing overhead. In contrast, transactional workflows—including order place- ment, status updates, and delivery assignment—incur higher overhead as they involve multi-stage validation, entity state transitions, and coordinated database updates across multiple modules.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

The system follows Django's built-in development server configuration and ORM-based database interaction, making it suitable for small to medium-scale applications under typi- cal usage conditions. Performance and scalability depend on deployment environment, database optimization, and server configuration. Since SQLite is used as the database, it pro- vides sufficient performance for academic use but may not scale efficiently for high-concurrency scenarios. Performance benchmarking under concurrent user load is identified as future work.

B. Functional Validation Results

Functional validation was conducted using scenario-based testing, covering typical workflows such as user registration, login, food browsing, cart operations, order placement, and order tracking, along with error conditions including invalid inputs and unauthorized access. Testing focused on correct- ness of workflow execution and data consistency rather than performance benchmarking.

Multiple validation rules were verified: correctness of user authentication, validation of cart items, accurate calculation of order totals, proper association between customers and orders, and controlled status transitions in the order lifecycle. These validations ensure reliable system behavior and data integrity across all modules.

Workflow coordination was tested by simulating interactions between customer, restaurant, and delivery roles. Results confirmed correct propagation of order status updates from placement to delivery completion. The system successfully maintained consistency of order data across modules, prevent- ing duplication and ensuring synchronized updates.

Integration testing validated interactions between modules, ensuring that order placement triggers restaurant processing and delivery assignment correctly. Although certain advanced features such as real-time tracking and notifications are not implemented, the system architecture supports future integra- tion.

Compared to traditional manual ordering processes, the system significantly reduces order processing time from sev- eral minutes to a few seconds. Automated order handling minimizes human errors, while centralized data storage im- proves visibility and coordination among stakeholders. These improvements highlight the effectiveness of digital transfor- mation in food service operations.

Testing was conducted manually using predefined scenarios and sample data under single-user conditions. Automated testing, multi-user concurrency evaluation, and performance benchmarking remain areas for future work. The system demonstrates functional correctness but is not optimized for production-scale deployment.

C. Comparative Analysis with Manual Systems

Compared to conventional manual food ordering processes, the implemented system significantly reduces operational overhead by automating menu browsing, order placement, and delivery coordination. Manual processes requiring physical presence or phone-based communication are replaced by a centralized digital platform enabling instant interaction be- tween users and service providers.

Automated order management eliminates errors caused by miscommunication and manual record-keeping, while central- ized database storage ensures consistent tracking of orders and user data. The system provides improved transparency, allow- ing customers to monitor order status and enabling restaurants to manage operations efficiently. These capabilities directly address inefficiencies and limitations observed in traditional food ordering systems.

VI. DISCUSSION

A. Design Validation and Key Observations

The implementation follows a three-tier architecture with Django's Model–View–Template (MVT) pattern, demonstrat- ing clear separation of concerns and improved maintainability. The modular structure ensures that different components such as user management, food handling, order processing, and delivery coordination operate independently while maintaining system integration. Django's built-in ORM simplifies database operations, enabling efficient data handling without complex query management.

Key complexities involved designing an effective order lifecycle workflow and ensuring proper coordination between multiple roles including customers, restaurants, and delivery personnel. Managing consistent state transitions across order stages and maintaining data integrity across modules required careful design. The use of structured models and relationships helps decouple application logic from database representation, improving flexibility and scalability.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

Several limitations are observed in the current implementation. The system uses basic authentication mechanisms without advanced security features such as token-based authentication or encryption enhancements. Payment processing is not imple- mented, and real-time tracking of delivery is absent. Addition- ally, the system relies on manual testing rather than automated testing frameworks, which limits validation coverage.

Despite these limitations, the system provides practical exposure to full-stack web development, role-based system design, database modeling, and workflow management. It serves as an effective academic prototype demonstrating key principles of modern web application development.

B. Limitations and Constraints

Although the system demonstrates functional correctness, it is not designed for production-level deployment. Security mechanisms are basic, relying on Django's default authen- tication without advanced authorization techniques such as JSON Web Tokens (JWT) or OAuth. Sensitive data handling and encryption mechanisms are minimal, which may introduce security vulnerabilities in real-world scenarios.

The system supports only single-application deployment and does not provide multi-tenant capabilities. Order processing does not include advanced features such as real-time tracking, push notifications, or dynamic delivery assignment algorithms. Additionally, scalability is limited due to the use of SQLite database, which is not optimized for handling large- scale concurrent operations.

Performance optimization techniques such as caching, query optimization, and pagination are not implemented. These limitations highlight the need for further enhancements to transform the system into a production-ready application.

C. Alignment with Literature Findings

The implementation addresses several shortcomings identified in existing literature, particularly lack of modular archi- tecture and inefficient workflow management in food ordering systems. The use of a structured, role-based design ensures clear separation of responsibilities and improves system maintainability. Centralized database management and controlled order lifecycle transitions enhance data consistency and operational efficiency.

The system also demonstrates improved coordination be- tween different modules compared to earlier implementations. While advanced features such as intelligent recommendations and real-time tracking are not included, the foundational architecture supports future integration of such functionalities. Overall, the system aligns with modern web development practices and addresses key limitations identified in previous studies.

VII. CONCLUSION AND FUTURE WORK

A. Conclusion

This paper presents an Online Food Ordering and Delivery System prototype demonstrating modular architecture, role- based workflow management, and efficient database integra- tion. The system validates key software engineering principles through practical implementation using the Django framework. The layered architectural approach ensures maintainability and clear separation of concerns, while structured order life- cycle management enables consistent coordination between customers, restaurants, and delivery personnel. The system demonstrates the effectiveness of modern web technologies in developing scalable and reliable food service applications.

B. Future Work

Future work involves integrating secure payment gateways, implementing advanced authentication mechanisms such as token-based security, and enhancing automated testing cover- age. Additional improvements include real-time order track- ing, notification systems, and dynamic delivery assignment for better operational efficiency. Further enhancements may include migration to scalable databases such as MySQL or PostgreSQL, implementation of caching and pagination tech- niques, and development of mobile applications to extend accessibility. Integration of AI-based recommendation systems and analytics features can further improve user experience and system intelligence.

VIII. ACKNOWLEDGMENT

The authors would like to thank Bonam Venkata Chalamayya Engineering College and the Department of Computer Science and Engineering for their support in this research work.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

REFERENCES

[1] Adamu, "Employee Leave Management System," FUDMA Journal of Sciences (FJS), vol. 4, no. 2, pp. 86–91, 2020. doi: 10.33003/fjs2020-0402-162.
[2] N. Harshika, P. U. Vardhan, C. Vaishak, A. Akhil, B. Varshitha, and S. S. Raoof, "A Multi-Faceted Leave Management Ecosystem Employing AI-Driven Semantic Categorization and Probabilistic Algorithms with Dynamic Schedule Reallocation," International Journal of Progressive Research in Engineering Management and Science (IJPREMS), vol. 5, no. 4, pp. 1145–1153, 2025.
[3] S. M. Alade, S. Adejumo, and T. J. Alade, "Design and Implementation of a Web Based Leave Management System," International Journal of Computer Applications Technology and Research, vol. 11, no. 4, pp. 123–144, 2022. doi: 10.7753/IJCATR1104.1006.
[4] R. Srinithi and P. Sakthi Murugan, "Employee Leave Management System," International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering (IJIREEICE), vol. 13, no. 4, pp. 198–202, 2025. doi: 10.17148/IJIREEICE.2025.13432.
[5] Rashmi, S. S. Dhulugade, P. N. Gaikwad, and D. M. Rathod, "Leave Management in Power Apps," International Journal of Innovative Re- search in Technology (IJIRT), vol. 11, no. 1, pp. 674–679, June 2024.
[6] R. S. Birje, R. Benne, and A. Unki, "Design and Development of E– Leave Management System," International Journal of Research Publi- cation and Reviews, vol. 6, no. 10, pp. 6464–6472, 2025.
[7] N. Choudhary, A. Khalfe, Y. Khan, and M. Ansari, "Leave Management System for AIKTC," International Research Journal of Engineering and Technology (IRJET), vol. 7, no. 3, pp. 1715–1717, Mar. 2020.
[8] M. Singh, P. Singh, R. Singh, S. Singh, and S. Gupta, "Leave and Payroll Management System," in Proc. International Conference on Computing and Virtualization (ICCCV-17), Thakur College of Engineering and Technology, 2017, pp. 62–66.
[9] R. Sapona, A. H. Thohari, and Nelmiawati, "Web-based Leave Management System for Politeknik Negeri Batam," Journal of Computer Sciences and Engineering, pp. 72–74, 2020.
[10] Y. Mantri, D. A. R. Kumar, and S. U. Kumar, "Emergency Leave Management System with Company Data Analysis," International Journal of Research in Engineering and Science, vol. 11, no. 21, pp. 163–174, 12023. ISSN: 2320-9364.2023. ISSN: 2320-9364.2023. ISSN: 2320-9364.
[11] A. I. Pathan, B. Nayak, B. Nayak, V. B. Dhatrak, and A. K. Daivat, "Design of AI-Based Leave Scheduling and Managing Application," International Journal of Computer Sciences and Engineering, vol. 8, no. 4, 2020. doi: 10.26438/ijcse/v8i4.115.
[12] S. D. Jadhav, A. A. Ranaware, and P. D. More, "Design Steps of Online Leave Management Application System for Academic Institution," in Proc. National Conference on Emerging Trends in Science and Advances in Engineering, Phaltan, India: International Journal of Innovations in Engineering Research and Technology, 2023.
[13] V. K. Kaushik, A. K. Gupta, A. Kumar, and A. Prasad, "Student Leave Management System," International Journal of Advance Research and Innovative Ideas in Education, vol. 3, no. 5, pp. 124–131, 2017.

BIOGRAPHIES OF AUTHORS



Oleti Durga Prasad is currently residing at Odalarevu, East Godavari, Andhra Pradesh-533210. He is a B.Tech student specializing in Electronics and Communication Engineering at Bonam Venkata Chalamayya Engineering College, Odalarevu, with an expected graduation in May 2026. He aims to secure a position that leverages his strong organi- zational skills, educational background, and abil- ity to work effectively with others. He possesses key skills in Java, System Design(LLD),Python, MySQL,HTML, Javascript, Django. While his pro- fessional experience is listed as a student, his proactive approach and skill set indicate a strong potential for growth and contribution in a professional setting.

Phone No.: 7382212221
Email: prasadkshatriya26@gmail.com



Pedakapu Surya Prasad is currently residing at Odalarevu, East Godavari, Andhra Pradesh-533210. He is a B.Tech student specializing in Electronics and Communication Engineering at Bonam Venkata Chalamayya Engineering College, Odalarevu, with a n expected graduation in May 2026. He aims to secure a position that leverages his strong organiza- tional skills, educational background, and ability to work effectively with others. He possesses key skills in Java, System Design(LLD),MySQL, HTML, CSS, JavaScript,Django. While his professional experi- ence is listed as a student, his proactive approach and skill set indicate a strong potential for growth and contribution in a professional setting.

Phone No.: 9346595324
Email: suryapedakapu@gmail.com



Ravipati Raja is currently residing at Odalarevu, East Godavari, Andhra Pradesh-533210. She is a B.Tech student specializing in Electronics and Com- munication Engineering at Bonam Venkata Chalamayya Engineering College, Odalarevu, with an expected graduation in May 2026. he aims to se- cure a position that leverages her strong organi- zational skills, educational background, and ability to work effectively with others. She possesses key skills in Java, System Design(LLD), HTML, CSS, JavaScript, Django. While his professional experi- ence is listed as a student, her proactive approach and skill set indicate a strong potential for growth and contribution in a professional setting.

Phone No.: 9392879661
Email:rajaravipati04@gmail.com

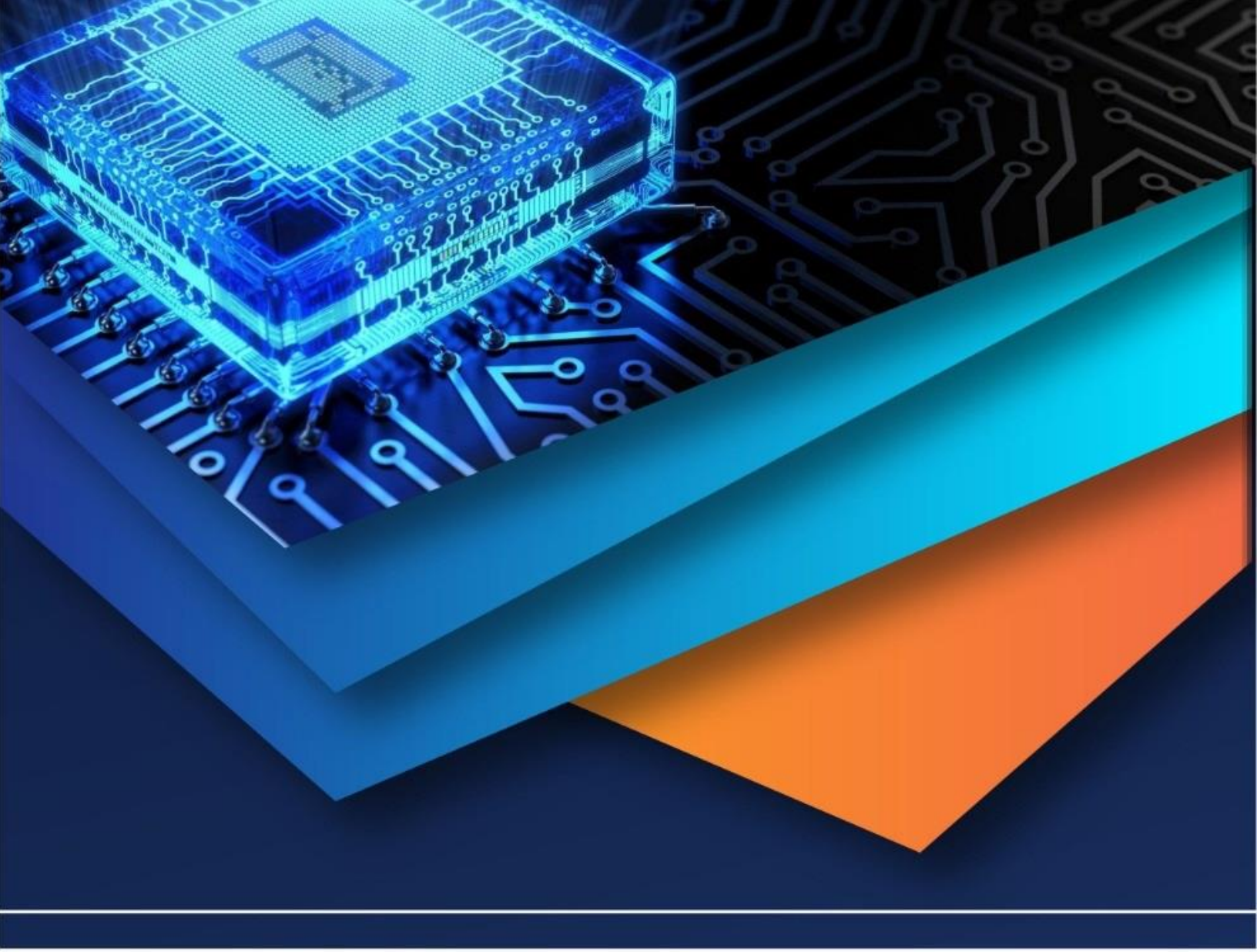


Madasu Hemanth is currently residing at Odalarevu, East Godavari, Andhra Pradesh-533210. He is a B.Tech student specializing in Engineering at Bonam Venkata Chalamayya Electronics and Communication Engineering College, Odalarevu, w ith an expected graduation in May 2026. He aims to secure a position that leverages his strong organizational skills, educational background, and ability to work effectively with others. He possesses key skills in HTML, CSS, JavaScript, Django.

jpeg While his professional experience is listed as a student, his proactive approach and skill set indicate a strong potential for growth and contribution in a professional setting.
Phone No.: 8639505454
Email: hemanth18v@gmail.com



Y.Satya Vinod is an Assistant Professor in the Department of ECE at Bonam Venkata Chalamayya Engineering College, Odalarevu, India. He holds a M.Tech in Electronics and Communication Engi- neering and has over 10 years of teaching and research experience. His research interests include embedded systems, communication systems, intelligent systems, and the application of modern technolo- gies in real-world problem solving. He has guided numerous undergraduate and postgraduate projects in domains related to electronics, communication,
and software-integrated systems. He has published research papers in reputed national and international journals and conferences. He served as the project guide and mentor for the Nearest Online Food Ordering , providing valuable insights into system design, implementation strategy, and overall project
development Email: satyavinod55@gmail.com



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)