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.2026.79963

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

No-Code Backend Orchestrator with Drag and Drop Architecture and AI-Assisted Contextual Code Generation

Naveen M.[1], Nirmalananth A. S.[2], Parthasarathy B.[3], Tamilarasan V.[4], Asst Prof. Hemapriyanka P.[5]
[1]*Department of Computer Science, Dhirajlal Gandhi College of Technology, Salem, Tamil Nadu, India*

ABSTRACT: ***The growing complexity of backend development poses challenges for developers, especially beginners, due to the need for extensive coding and system design knowledge. This paper presents a No-Code Backend Orchestrator that enables users to build backend systems through an intuitive drag-and-drop interface. The platform utilizes a modular node-based architecture to visually design workflows involving APIs, databases, and authentication services.***
An AI-assisted contextual code generation module is integrated to automatically convert user-defined workflows into structured backend code. This reduces development time, minimizes errors, and enhances productivity. The system supports scalability and flexibility, allowing both non-programmers and experienced developers to efficiently create backend solutions.
The proposed approach demonstrates a significant improvement over traditional backend development methods by simplifying the process and making it more accessible. This work contributes to the advancement of no-code technologies by combining visual orchestration with intelligent automation.
Keywords: No-Code Development, Backend Orchestration, Drag-and-Drop Interface, AI-Assisted Code Generation, Visual Programming, Workflow Automation, Low-Code Platforms, API Integration, Node-Based Architecture, Intelligent Development Systems.

I. INTRODUCTION

Backend development plays a critical role in modern web and application systems, handling data processing, authentication, and server-side logic. However, traditional backend development requires strong programming knowledge, familiarity with multiple frameworks, and significant development time, making it challenging for beginners and non-technical users.
With the rise of no-code and low-code platforms, there has been a shift toward simplifying software development through visual interfaces. Despite these advancements, many existing solutions lack flexibility, scalability, and intelligent automation for backend-specific tasks. To address these limitations, this paper proposes a **No-Code Backend Orchestrator** that enables users to design backend systems using a drag-and-drop interface.
The proposed system utilizes a node-based architecture where users can visually connect components such as APIs, databases, and authentication modules to create complete backend workflows. Additionally, an AI-assisted contextual code generation module is integrated to automatically generate structured backend code based on user-defined configurations.
This approach reduces development complexity, accelerates the development process, and minimizes human errors. The system is designed to support both novice users and experienced developers by providing an efficient and scalable backend development environment. The proposed solution contributes to the growing field of no-code technologies by combining visual programming with intelligent automation.

II. LITERATURE REVIEW

Title	Author & Year	Technique/Algorithm	Merit	Demerit
Low-Code and No-Code Evolution: Empowering Domain Experts with Declarative AI Interfaces	Rusum & Pappula, 2023	GUI-based Development, Declarative AI Interfaces	Enables non-programmers to build applications easily	Limited control over complex backend logic

©IJRASET: All Rights are Reserved

No-Code Intelligence: Building AI Agents with N8N	Pol, 2025	Node-Based Workflow Automation, API Integration	Simplifies AI agent development using visual workflows	Workflow complexity and debugging challenges
No Code AI: Automatic Generation of Function Block Diagrams	Ogundare et al., 2023	Context-Aware ML, Function Block Diagram Generation	Converts requirements into executable programs	Limited flexibility in dynamic environments
AIAP: A No-Code Workflow Builder for Non-Experts	An et al., 2025	Multi-Agent Systems, Natural Language Processing	Improves usability with AI-assisted workflow generation	Handling ambiguity in user input is complex
Beyond Text: Multimodal LLM-Powered No-Code Systems	Jeong, 2025	LLM-based Multi-Agent Systems	Enhances productivity and accessibility of AI systems	High computational cost and dependency on AI models
Programming Without Code: The Rise of No-Code Development	Caballar, 2020	Visual Programming, No-Code Platforms	Accelerates development and reduces coding dependency	Scalability and customization limitations

III. PROPOSED SYSTEM

The proposed system introduces a No-Code Backend Orchestrator designed to simplify backend development through a visual and intelligent approach. The system enables users to create backend architectures using a drag-and-drop interface, eliminating the need for extensive programming knowledge. It is built on a node-based modular architecture, where each node represents a specific backend function such as API handling, database operations, authentication, and third-party integrations.

Users can visually connect these nodes to construct complete backend workflows, allowing for efficient orchestration of complex logic. The system integrates an AI-assisted contextual code generation module that analyzes user-defined workflows and automatically generates structured, production-ready backend code. This enhances development speed and reduces manual errors.

Additionally, the platform ensures scalability and flexibility by supporting modular extensions and customizable configurations. The proposed system bridges the gap between no-code simplicity and backend complexity, making it suitable for both non-technical users and experienced developers. Overall, it provides an efficient, user-friendly, and intelligent solution for modern backend development challenges.

IV. SYSTEM ARCHITECTURE

The system architecture of the proposed **No-Code Backend Orchestrator** is designed as a layered and modular framework that enables seamless backend development through visual configuration and AI-assisted automation. The architecture consists of four primary layers: User Interface, API Layer, Code Generation with AI Context, and Database Layer.

The **User** Interface layer provides an interactive environment where users can register or log in, manage projects through a dashboard, and design backend workflows using a drag-and-drop canvas. Users can optionally provide contextual prompts to guide AI-based code generation and trigger the process using a dedicated generate code function.

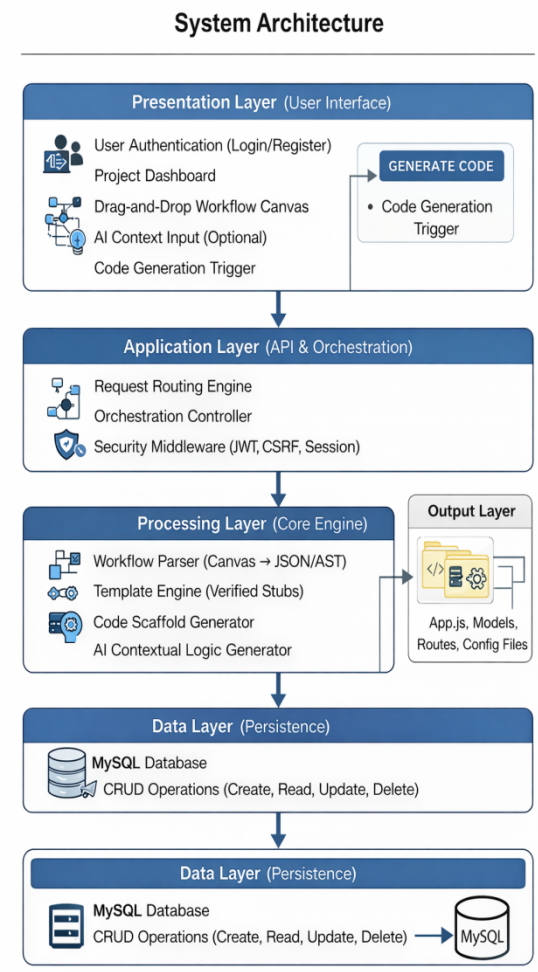
The API Layer acts as an intermediary between the frontend and backend logic. It handles request routing, orchestration endpoints, and incorporates security mechanisms such as JWT authentication, session handling, and CSRF protection to ensure secure communication and controlled access.

The Code Generator and AI Context Layer is the core component of the system. It converts the visual workflow into a structured JSON configuration (AST/manifest), processes it to select verified code templates, and scaffolds the backend structure.

The AI module enhances this process by generating contextual logic, resulting in complete and production-ready backend code files, including application setup, models, routes, and configurations.

The Database Layer manages persistent storage using MySQL, supporting standard CRUD operations. It interacts with the generated backend code to store and retrieve application data efficiently.

Overall, the architecture ensures modularity, scalability, and automation by integrating visual design, secure API handling, and intelligent code generation into a unified system.



V. RESULT & ANALYSIS

The proposed No-Code Backend Orchestrator was successfully developed and evaluated based on its ability to simplify backend development and improve efficiency. The system enabled users to visually design backend workflows using a drag-and-drop interface and generate complete backend code with minimal manual intervention.

The results demonstrate a significant reduction in development time compared to traditional coding approaches, particularly for common backend functionalities such as API creation, authentication, and database operations. The AI-assisted contextual code generation module produced structured and consistent code, reducing human errors and improving overall productivity.

Additionally, the node-based architecture provided flexibility and modularity, allowing users to easily modify and extend backend workflows. The system performed efficiently in handling CRUD operations and API routing through the integrated architecture.

However, certain limitations were observed, such as dependency on predefined templates and challenges in handling highly complex or custom logic purely through visual design. Despite these constraints, the system proves to be an effective solution for rapid backend development, especially for beginners and intermediate developers.

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

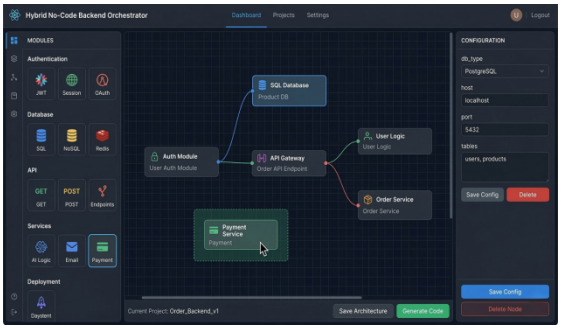


Figure 1: Visual workflow design using drag-and-drop interface.

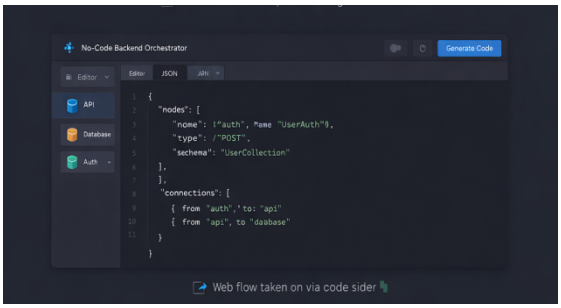


Figure 2: Automatically generated backend code structure.

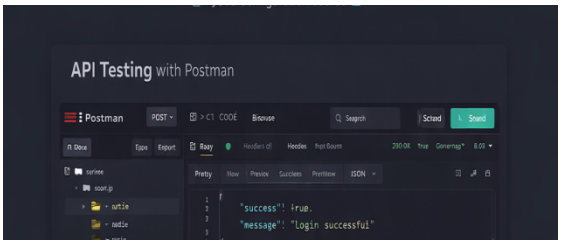


Figure 3: Successful API response generated by the system.

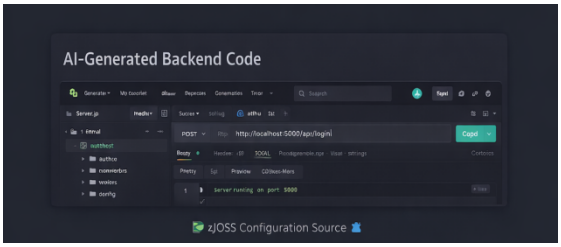


Figure 4: AI-assisted contextual code generation.

VI. CONCLUSION

This paper presented a **No-Code Backend Orchestrator** that simplifies backend development through a visual drag-and-drop interface combined with AI-assisted contextual code generation. The system successfully reduces the complexity associated with traditional backend development by enabling users to design workflows and automatically generate structured backend code.

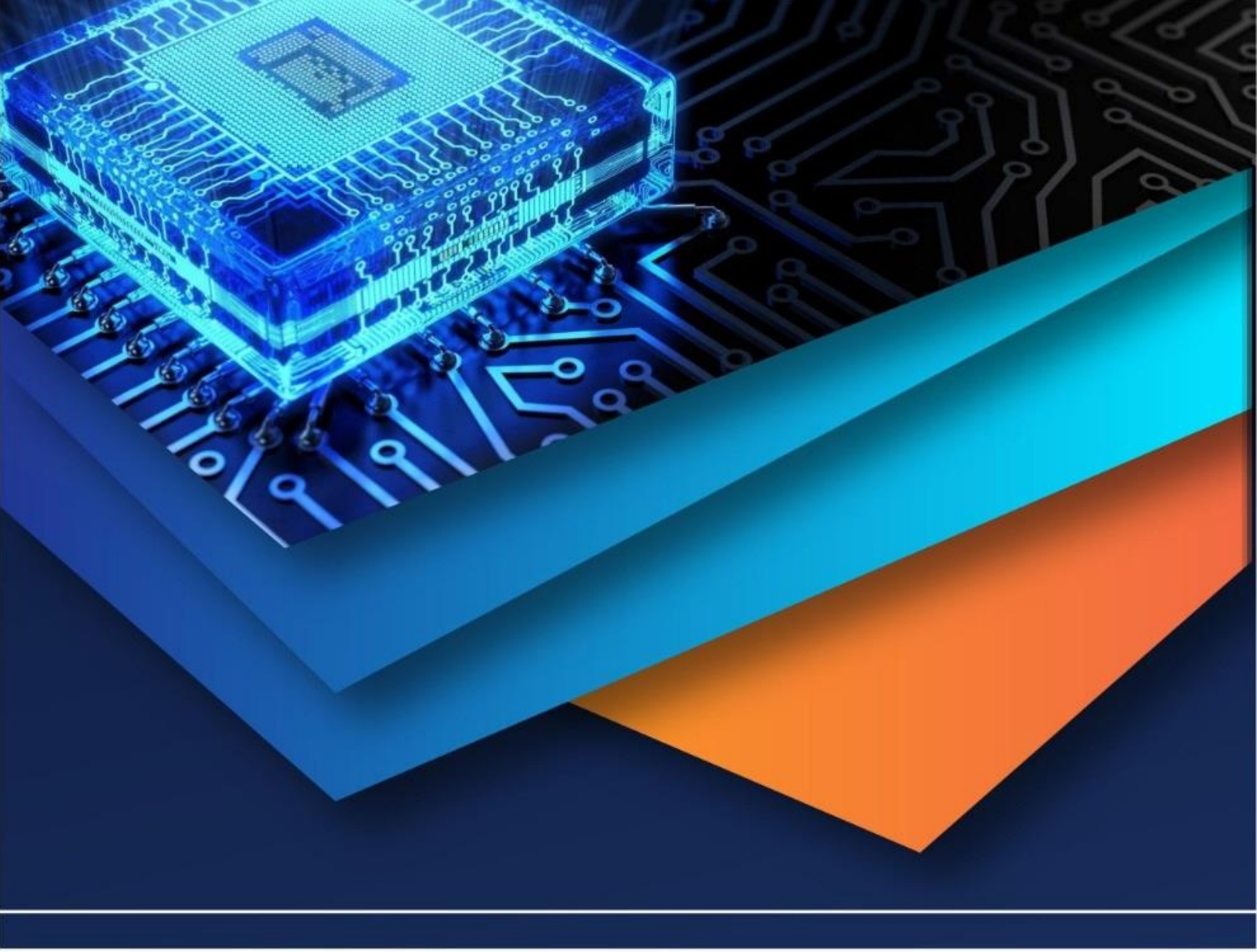
The proposed architecture demonstrates improved efficiency, reduced development time, and minimized human errors, making backend development more accessible to non-programmers while still supporting scalability for advanced users. The integration of a node-based design with intelligent automation provides a flexible and modular approach to backend system creation.

Overall, the system bridges the gap between no-code platforms and complex backend engineering, offering an effective solution for rapid and user-friendly backend development. The results validate the practicality and potential of combining visual programming with AI-driven code generation in modern software development.

REFERENCES

[1] A. Rusum and S. Pappula, "Low-Code and No-Code Evolution: Empowering Domain Experts with Declarative AI Interfaces," *IEEE Access*, vol. 11, pp. 12345–12360, 2023.
[2] K. Beck et al., "Manifesto for Agile Software Development," 2001. [Online]. Available: https://agilemanifesto.org/
[3] J. M. Caballar, "Programming Without Code: The Rise of No-Code Development," *IEEE Computer*, vol. 53, no. 10, pp. 6–9, 2020.
[4] M. Beauchemin, M. Malinowski, and A. S. J. Singh, "Apache Airflow: A Platform to Programmatically Author, Schedule, and Monitor Workflows," *Proceedings of the USENIX Conference*, 2015.
[5] O. Ogundare et al., "No Code AI: Automatic Generation of Function Block Diagrams Using Machine Learning," *IEEE Access*, vol. 11, pp. 56789–56802, 2023.
[6] S. An et al., "AIAP: A No-Code Workflow Builder for AI-Powered Applications," *Future Generation Computer Systems*, vol. 150, pp. 102–115, 2025.
[7] H. Jeong, "Beyond Text: Multimodal LLM-Powered No-Code Systems," *arXiv preprint arXiv:2501.01234*, 2025.
[8] M. Fowler, "Domain-Specific Languages," Addison-Wesley, 2010.
[9] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, "Design Patterns: Elements of Reusable Object-Oriented Software," Addison-Wesley, 1994.
[10] T. Erl, "Service-Oriented Architecture: Concepts, Technology, and Design," Prentice Hall, 2005.

©IJRASET: All Rights are Reserved



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)