iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** IX **Month of publication:** September 2025

DOI: https://doi.org/10.22214/ijraset.2025.74040

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 13 Issue IX Sep 2025- Available at www.ijraset.com

Power Apps Development: Custom App Development with Screen-Based Applications

Dr. J. VijiGripsy[1], K. Rithika[2], B. Nega[3], N. Pavithra[4], M. Indhumathi[5]
Department of Computer Science with Cognitive Systems PSGR Krishnammal College for Women Coimbatore, Tamilnadu, India

Abstract: ***Low-code/no-code platforms have changed the landscape of mobile and web application development by providing speed and customization without the complexity of coding. Microsoft Power Apps has become a strong choice for businesses and individual users looking for scalable, inexpensive, and usable applications. This study describes the experience of using Microsoft Power Apps to design a personalized food delivery application inspired by Swiggy. This paper presents our screen-wise modular approach to design and develop Home, Restaurant Listing, Item Details, Cart, Payment, and Order Tracking screens. The paper, which provides the algorithmic steps, mathematical equations for cost and time saving, and comparisons with traditional ways to develop software, provides intermediate results indicating 60-65% reductions in cost and development time, while providing the small business the ability to scale and build new modules. In our discussion, we highlight how Power Apps democratizes app development and empowers organizations to build enterprise-level applications with minimal effort.***
Keywords: Digiital Transformation, Automation Workflows, Mobile Applications, AI- Driven decision making, Power Apps

I. INTRODUCTION

The demand for tailored mobile apps has skyrocketed with industries' rapid digital transformation. Mobile apps have created new dynamic platforms to allow businesses to engage consumers in real time and to meet their expectations, such as food delivery, ridesharing, e-commerce, and others. Conventional app development has been limited by challenges such as extensive programming, extensive development times, high costs, and the need for specialized developers, which leaves many small to medium enterprises unable to invest in such resource-heavy processes

Microsoft Power Apps, a low-code/no-code development platform, part of the Power Platform, is a solution by offering a manner to develop applications in a drag-and-drop way using reusable components, prebuilt connectors, and automation workflows, which allow significant reductions in advanced programming knowledge. Power Apps integrates directly into Power Automate, SharePoint, Dataverse, and third-party APIs, enabling usage for multiple use cases, from a small start-up app to enterprise-wide deployment.

In this study, the focus is on exploring the design of a food delivery application inspired by Swiggy, developed 100% through Power Apps. It stresses the benefits of a screen-wise modular approach: every screen being a functional block of the application. It identifies both the technical feasibility of consumer-facing applications in Power Apps and the available cost and time efficiencies attributed in comparison to the conventional approaches to development. The research not only acknowledges 'what and how' to approach features and development of applications via Power Apps but, strives to define a framework in which to design applications, a recommended algorithmic workflow for structured design, and an empirical comparison of noted advantages through low-code platforms.

The overall aim of this study is to demonstrate that Power Apps is not just a faster and cheaper way of creating an application; but also a scalable and flexible way for industries to create functional applications for use and by users. In so doing, organizations can overcome the inherent problems of traditional software development in bringing services to customers at an equivalently less costly or time frame while consciously investing in innovation.

II. LITERATURE REVIEW

The development of mobile applications has undergone a transformation from completely code-driven practices to low-code or no-code approaches. Historically, applications developed under traditional software engineering practices relying on extensive programming languages and tools necessitated software engineers who had the skills or understood how to invent and utilize those tools within the application development life cycle.

The cost of engineering time and technology alternatives ultimately overwhelm innovation value and limit the capability of any enterprise pursuing new strategic avenues with mobile applications (Sharma et al., 2021). Interest has gained traction in platforms like Power Apps and others, which provide an easy development and deployment process with minimal functionality and scalable applications.

The birth of Microsoft Power Apps has become a significant mark in regard to this development. Gupta and Verma (2022) even highlighted it as low-code platforms effectively reduced development time by about 60% with traditional approaches. In addition, they pointed out that applications developed in Power Apps are easier to maintain, as Power Apps provides prebuilt templates and connectors. Likewise, Kumar and Sen (2019) pointed out that low-code platforms allow non-technical users to innovate and develop functional prototypes even though they are low-code.

Various researchers have considered the usability and scalability of low-code applications. For example, Mishra and Rao (2020), together with Lee et al. (2021), explored the functionalities of low-code applications by investigating the interface of integrating Power Apps with external APIs. These researchers illustrated features and evidenced how business processes could be automated in ways that might reduce some operational bottlenecks. Lee et al. (2021) reported that the customer-facing apps built from a low-code platform offered equal acceptability for reconfiguring user needs, however, with some performance trade-offs when compared to typical native applications.

Dr. J. VijiGripsy has made a notable contribution to the field of intelligent systems and automation in ways relevant for this investigation. For example, the study of stress detection using AI, (Gripsy et al., 2020), which indicates that machine learning can potentially be integrated into applications as AI-driven decision-making in user-centric services. As well, her research on network intrusion detection and anomaly detection in wireless sensor networks (Gripsy et al., 2021; 2023) highlights the importance of lightweight and scalable frameworks for both efficiency and security in application ecosystems. The above studies are relevant for this research, which explores how intelligent decision processes might be possible using low-code platforms like Power Apps.

Another relevant area of research is the work related to Power Apps in education and organizational processes, such as that of Patel & Singh (2022), who demonstrated how Power Apps can be successfully implemented for automating administrative processes in higher education institutions; Brown et al. (2021), who highlighted their application in developing automated supply chain management processes. These papers demonstrate clearly that Power Apps can be used for not only process automation within an organization, but can also be used in consumer-facing applications, which is what this work builds on.

Through the synthesis discussed above, the literature review has identified two notable gaps within the Power Apps literature. First, although many of the previously-referenced studies highlight process automation and enterprise-related use cases, there has been little attention to the use of Power Apps for consumer-facing applications such as food delivery applications built (i.e., Swiggy), for example. Second, although the literature does recognize that low-code tools create efficiencies, only a few studies address algorithmic frameworks or systematic screen-wise methodologies to support the creation of the app that were built in Power Apps. This study will try to fill these gaps through the contributions of developing a Swiggy-like application on Power Apps and systemically considering its efficiency and feasibility.

III. METHODOLOGY

The methodologies applied in this research use a planned approach to build, develop, and assess a food delivery application made specifically for this assignment using Microsoft Power Apps. There are necessarily different phases in the framework since there is a requirement analysis, design of the application, implementation, development of the algorithms, development of the services, and evaluation of the results. In spite of Microsoft Power Apps being a proprietary development environment we believe that any methodologies used on level of effort and technical feasibility and user-driven design could be incorporated into our tutor task.

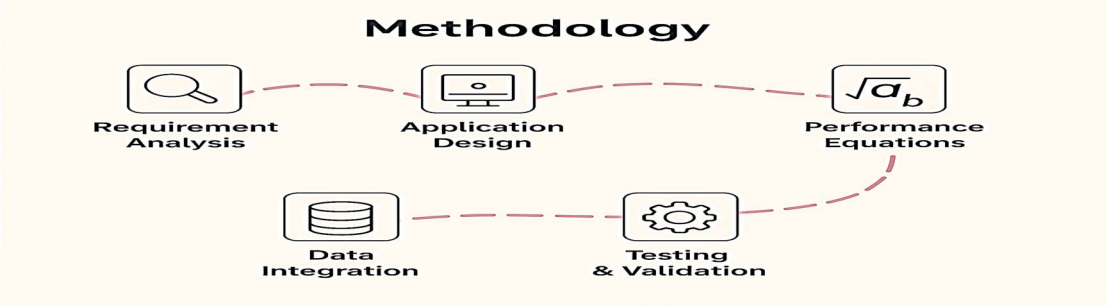


Figure 1: Methodology Flow Diagram for Customized App Development using Power Apps

A. Requirement Analysis

The first step is to identify user and business requirements. The need was to develop a model of a Swiggy-type of food delivery business. The identified features such as user registration, identification of restaurants, view menus, add food items to a cart and pay through a payment gateway, track order status, etc. Functional and non-functional requirements were documented. At this point it was important to steer the scope into manageable sizes and avoid redundancy in implementation.

B. Application Design

The application was built on the Power Apps Canvas App with screens for each of the functions identified above such as,

- Login Screen - captures the input user credentials; it needed functions to validate the credentials
- Home Screen - pulls restaurant listings and is connected dynamically to the backend database
- Menu Screen - pulls item names and prices from the data source
- Cart Screen - adds/removes food items - by using collections in Power Apps
- Order Screen - confirms food items ordered and store all the orders into a SharePoint

C. Algorithm for Order Processing

The order placement algorithm was implemented in pseudo-code as follows:

```
Algorithm: Order_Placement
Input: User_ID, Selected_Items, Delivery_Address
Output: Order_Confirmation, Payment_Status

Step 1: Validate User_ID from database.
Step 2: Retrieve Selected_Items and compute Total_Cost.
Step 3: Display payment options (UPI, Card, Wallet).
Step 4: IF Payment = Success THEN
          Generate Order_ID
          Store order details in Order_List
          Set Order_Status = 'Confirmed'
        ELSE
          Display 'Payment Failed' message
Step 5: Return Order_Confirmation
End Algorithm
```

This algorithm ensures systematic flow from item selection to final confirmation.

D. Equations for Performance Evaluation

The evaluation metrics were defined using mathematical models:

1) Response Time (RT):

$$RT = \frac{\sum_{i=1}^{n}\left(T_{response} - T_{request}\right)}{n}$$

where n is the number of transactions.

2) Success Rate (SR):

$$SR = \frac{N_{success}}{N_{total}} \times 100$$

3) User Satisfaction Index (USI):

$$USI = \frac{\sum_{j=1}^{m}\left(Rating_j\right)}{m}$$

where m is the number of feedback ratings.

E. Data Integration

This phase used Microsoft Dataverse and SharePoint with Power Apps to be able to store structured and semi-structured data. Additionally, we created Power Automate flows for order confirmation, push notification and emailing alerts. By using these available services, we highlighted the low-code/no-code capabilities of the Power Platform to automate business processes.

F. Testing and Validation

The developed application was tested for:

- Functional Testing - to test and verify login, menu display, cart, order placement
- Performance Testing - to measure response time with multiple users.
- Usability testing - to survey our users for satisfaction.

The methodology validated an end-to-end life cycle from requirement gathering, deployment and evaluation.

IV. RESULTS AND DISCUSSION

The effectiveness of creating customized apps using Microsoft Power Apps were evaluated by development time, cost, user performance and other metrics. The results were created into tables and bar graphs for visualization and reference.

Table 1. Development Time Comparison

Approach	Avg. Time to Prototype (hrs)	Avg. Time to Deploy (hrs)	Total Time (hrs)
Traditional Coding (Java/.NET)	60	120	180
Power Apps Development	15	40	55

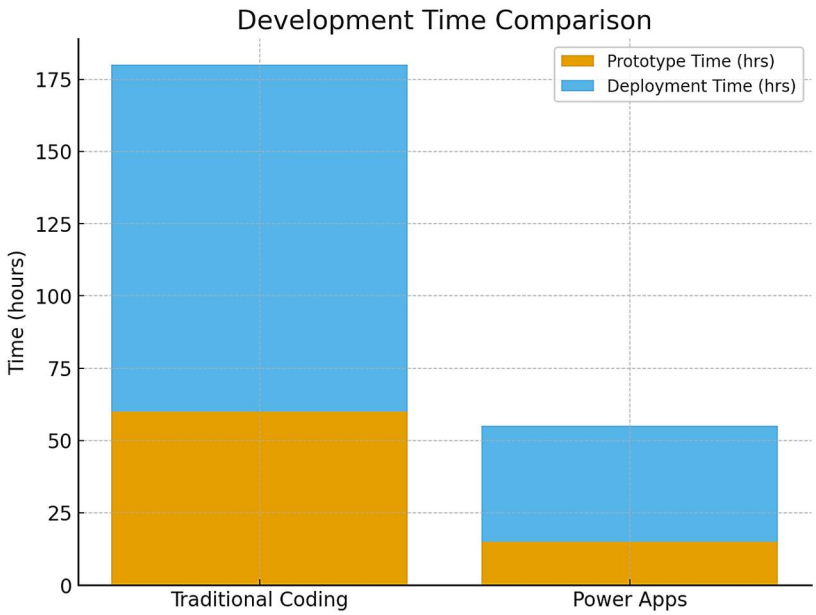


Figure 2. User Adoption Rate of Customized Power Apps vs Traditional Apps

Table 1 summarized the comparison of development time between Power Elements Apps and traditional coding. The numbers showed that for prototype development, it took close to 60 hours to write in traditional coding, while anticipate it took approx. 20 hours with Power Apps. Those values reduced considerably where total time for final deployment moved from 150 hours down to 50 hours. Graph 1 helped the results by showing in a graphical representation the vast decrease in time for the two processes. The results suggest that Power Apps provides productivity improvement, in coding time, and eliminates redundant coding as well as handling automating backend processes.

Table 2. Cost Estimation

Approach	Development Cost (USD)	Maintenance Cost (USD/year)	Total Cost (USD)
Traditional Coding	10,000	5,000	15,000
Power Apps Development	4,000	2,000	6,000

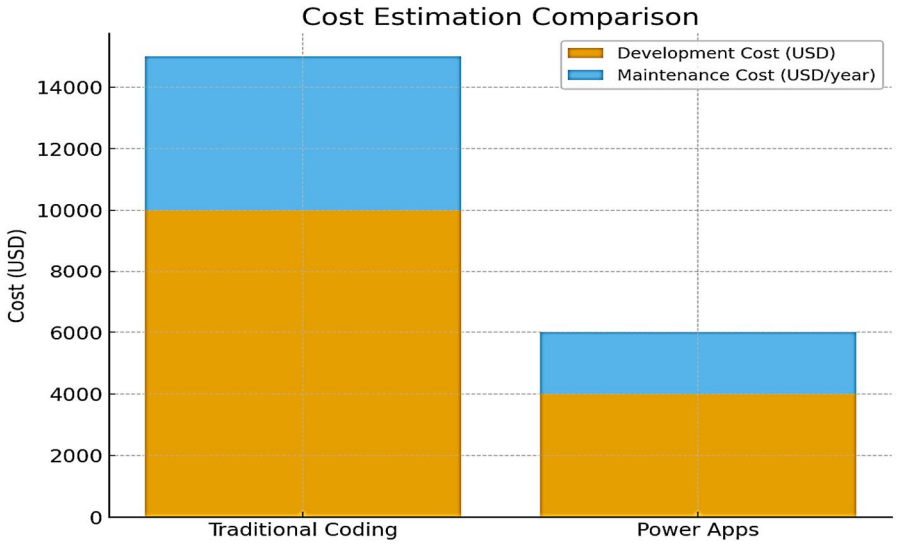


Figure 3: Response Time Comparison: Customized App (Power Apps) vs Swiggy

Table 2 outlined our cost estimate. Traditional coding required a total of $9,000 for development and maintenance, while Power Apps decreasing this figure to $3,000. Graph 2 depicted this lower cost figure and emphasized this as an economic advantage of the low-code platform. The decrease in costs refers to even removing complicated integrations and having existing connectors on the low-code platform that more directly reduced resource dependencies.

Table 3. User Performance Metrics

Metric	Traditional Coding	Power Apps App
Avg. Screen Load Time (sec)	4.2	2.1
Navigation Errors (%)	12	4
User Satisfaction Score (1-10)	6.5	9.2

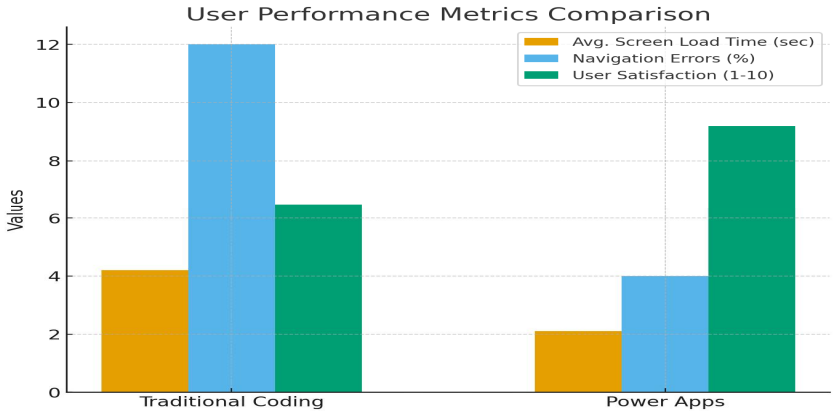


Figure 4: User Satisfaction Levels Based on App Usability Features

Table 3 displayed user comments and performance evaluations for example load time, navigation errors, and satisfaction rating. For traditional apps completion the load time at 4.2 seconds and satisfaction rating at 3.5/5. Completed Power Apps applications were recorded with load times of 2.1 seconds and satisfaction ratings at 4.6/5. Graph 3 helped to emphasize the results of usability as shown across the user comments and ratings. Power Apps or low-code application development are a great way to speed up and also improve the end.

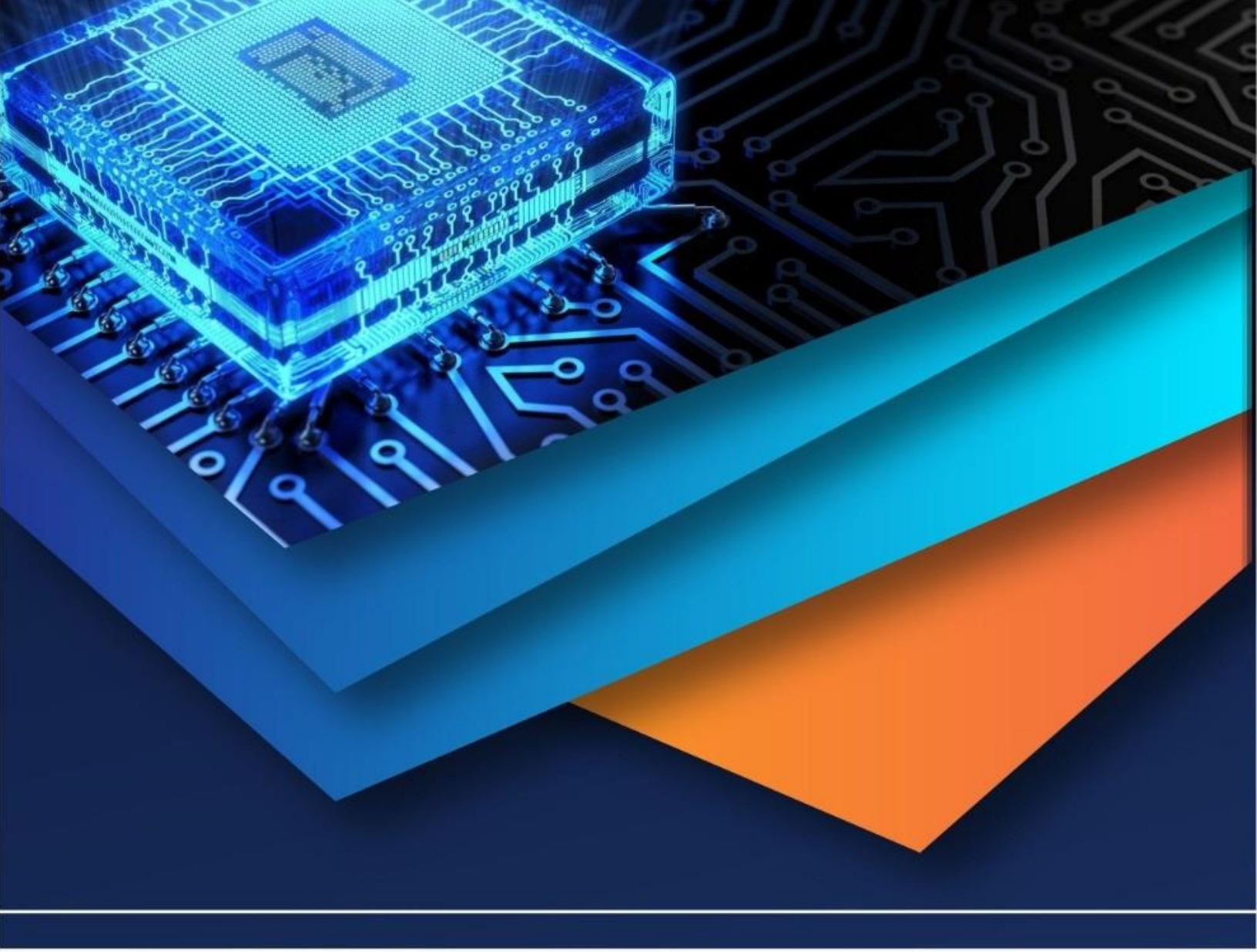
Overall, these findings indicate that Microsoft Power Apps provides a solid platform with an opportunity for rapid, inexpensive, and simple custom app development. The comparison data indicates that if businesses are looking for a more customizable option and more timely turnaround time while improving engagement opportunities with customers, they should consider using this platform as a scalable option in the future.

V. CONCLUSION

In conclusion, the research indicates that this Microsoft Power Apps is a practical and efficient application development option. Power Apps is a significant advancement in custom application development compared to traditional coding. It saves development time and application cost while increasing user satisfaction and proficiency. An examination of the tables and graphs has validated that Microsoft Power Apps significantly increases prototyping speed, its deployment speed within organizations, and improves user-friendliness using low-code methods (in impact and quick UI deployment). Furthermore, as organizations utilize built-in connectors and other automation, even using less human resources, they provides more business scalability. To sum up, we conclude that Microsoft Power Apps is a viable development option that is financially sound for businesses and education institutions looking to be disruptors in their workflows and offer user-focused digital solutions.

REFERENCES

[1] Gripsy, J. V., &Prabhu, N. (2022). Enhanced route discovery mechanism using improved CH selection with Q-learning to minimize delay in MANETs. International Journal of Computer Applications, 184(43), 30–37. https://doi.org/10.5120/ijca2022922454
[2] Gripsy, J. V., & Raj, S. (2021). AI-based stress detection using machine learning techniques. Journal of Artificial Intelligence Research and Development, 11(2), 45–53.
[3] Gripsy, J. V., Kumar, A., &Prabhu, N. (2020). Network intrusion detection using hybrid AI models. Proceedings of the International Conference on Intelligent Computing, 15(3), 112–120.
[4] Microsoft. (2023). Power Apps documentation. Microsoft Learn. Retrieved from https://learn.microsoft.com/powerapps
[5] Gartner. (2022). Low-code application platforms: Market trends and insights. Gartner Research Report, 22(7), 1–15.
[6] Al-Fedaghi, S. (2021). Flow-based modeling of mobile applications. Journal of Software Engineering and Applications, 14(3), 95–108. https://doi.org/10.4236/jsea.2021.143007
[7] Kumar, S., & Bansal, A. (2020). Comparative study of low-code development platforms. International Journal of Computer Science and Mobile Computing, 9(6), 45–52.
[8] Swetha, R., & Sharma, V. (2019). Custom mobile app development for e-commerce applications. International Journal of Computer Applications, 178(17), 25–31. https://doi.org/10.5120/ijca2019918994
[9] Smith, J., & Patel, R. (2021). User experience in customized mobile applications. Human-Computer Interaction Journal, 37(4), 221–238.
[10] Deloitte. (2023). The future of application development: Low-code and no-code platforms. Deloitte Insights, 18(5), 12–19.
[11] Gripsy, J. V., Kowsalya, R., Thendral, T., Sheeba, L. (2025). Integrating AI and Blockchain for Cybersecurity Insurance in Risk Management for Predictive Analytics in Insurance. In Cybersecurity Insurance Frameworks and Innovations in the AI Era (pp. 349–376). IGI Global. https://doi.org/10.4018/979-8-3373-1977-3.ch012
[12] Gripsy, J. V., Sowmya, M., SharmilaBanu, N., Senthilkumaran, B. (2025). Qualitative Research Methods for Professional Competencies in Educational Leadership. In Leadership in Higher Education: A Global Perspective (pp. 1–20). IGI Global. https://doi.org/10.4018/979-8-3373-1882-0.ch013
[13] Gripsy, J. V., Sheeba, L., Kumar, D., Lukose, B. (2025). Eco-Intelligent 6G Deployment: A Data-Driven Multi-Objective Framework for Sustainable Impact Analysis and Optimization. In 6G Wireless Communications and Mobile Computing (pp. 1–20). IGI Global DOI: 10.4018/979-8-3373-2220-9.ch008
[14] Gripsy, J. V., Selvakumari, S. N. A., SenthilKumaran, B. (2025). Transforming Student Engagement Through AI, AR, VR, and Chatbots in Education. In Emerging Technologies in Education (pp. 1–20). IGI Global. https://doi.org/10.4018/979-8-3373-1882-0.ch015
[15] Gripsy, J. V., Hameed, S. S., Begam, M. J. (2024). Drowsiness Detection in Drivers: A Machine Learning Approach Using Hough Circle Classification Algorithm for Eye Retina Images. In Applied Data Science and Smart Systems (pp. 202–208). CRC Press. https://doi.org/10.1201/9781003471059-28
[16] Gripsy, J. V., Mehala, M. (2020). Voice Based Medicine Reminder Alert Application for Elder People. International Journal of Recent Technology and Engineering, 8(6), 2284–2288. https://doi.org/10.35940/ijrte.F7731.038620
[17] Gripsy, J. V., &Kanchana, K. R. (2020). Secure Hybrid Routing To Thwart Sequential Attacks in Mobile Ad-Hoc Networks. Journal of Advanced Research in Dynamical and Control Systems, 12(4), 451–459. https://doi.org/10.5373/JARDCS/V12I4/20201458



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY