Societal Transformation: AI and Big Data Journal
Vol. 3(1): 65-78,2025
DOI:10.20547/aibd .253105

Drag & Build: Democratizing App Development through a No-Code Drag-and-Drop Platform

Kashif Laeeq[1]

Abstract: The focus of this study is the design, development, and impact of an example of no-code software application builder using drag and drop. By using programming visually, automating workflows, and integrating autonomously with different backends, this research also mitigates the traditional software development and the non-programmer spends on software development, simplifying and making app building development speedier, easier, and more scalable. This study focuses on architectural structures and the logic building processes coupled with the database, and aims at custom tailoring unmodifiable software that belt on security worries, unreachable deployment, and more, toward problematic customization, unchecked triangulation, unbounded cooperation, and steeped real-time automation. The booming suppliers of easily constructed software applications suggest this no-code solution has an impact on the non-developer portion of society—entrepreneurs, formations of organizations, or institutions. The ability to develop high-level automation on a no-code basis that would overhaul an enterprise, along with autonomous controlling of app development and export to mobile, positions future developments to lift no-code platforms to a new standard with unbounded software development customization, thus controlling the software development world.

Key Words: **No-code development, drag-and-drop interface, workflow automation, scalability**

1. INTRODUCTION

The expansion of digital services, the rise of e-commerce, and the growth of remote work have all contributed to the increased need for mobile and web applications. Traditional app development is time-consuming and requires coding expertise as well as funding. This makes it all but impossible for technical non-users, small businesses, and startups to gain access. The expense of contractor developers is exorbitant and outsourcing is often mired in delays and communication barriers. No-code platforms alleviate all of these issues through visual programming, pre-built components, and drag-and-drop features. Users can design and build applications without any coding. No-code solutions block the technical barriers app development frameworks, no-code solutions are far more cost-effective and accessible. development frameworks, no-code solutions are far more cost-effective and accessible.

[1] Federal Urdu University of Arts, Science & Technology, Karachi, Pakistan
kashiflaeeq@fuuast.edu.pk

Programming without the use of desktop applications encompasses the mental and physical movements of sitting in a chair and stroking the surface of the desk, operating a desktop mouse, and a keyboard. This acts out the performance of black boxing instead of engaging the person in mobile, tactile interactions. This is profoundly appalling to deposit imagination and creativity. Drag-and-drop. The world of no-code solutions facilitates application building without coding barriers. This sounds fantastic, but the conventional platforms come with their own issues utmost of which is lack of advanced features. This often blocks innovation and decisive plans to execute ideas. platforms come with their own issues utmost of which is lack of advanced features. This often blocks innovation and decisive plans to execute ideas.

The drag and drop features of Workflow or Workstreams offer range of platforms that defy conventional boundaries and their holistic design and architecture is a significant departure from the dreary monotony of most application building industry.

The objective of this research is to create, develop, and assess a **Drag & Build App Builder** prototype, which is designed to ease the process of building apps without losing functionality, security, and scalability. The objectives are as follows:

- Enhancing app development accessibility through streamlined UI design, workflow automation, and backend integration.
- Addressing key limitations of existing no-code platforms, including customization constraints, security risks, and scalability challenges.
- Providing an AI-assisted, flexible development environment for rapid app creation.
- Exploring advancements in AI-driven app generation, enterprise automation, and native mobile app exports.

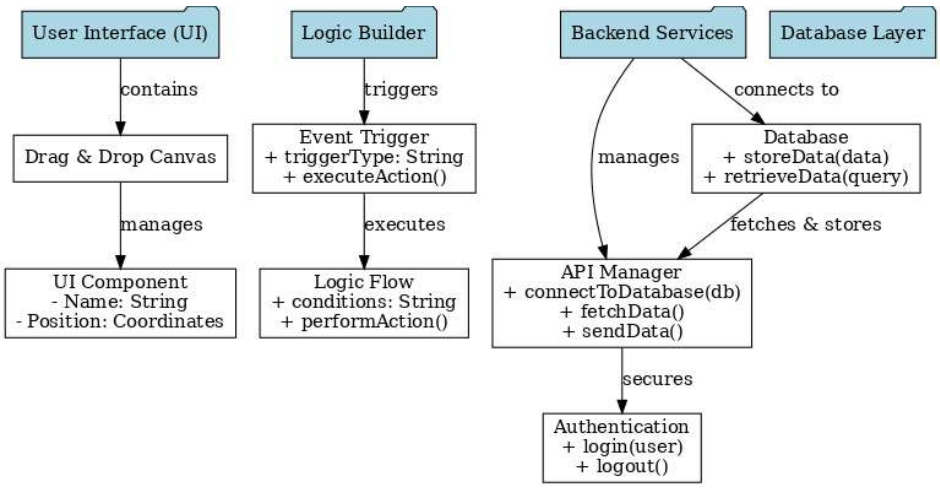

Figure 1: System Architecture of Drag & Build

The outcomes of this research fosters the advancement of no-code development which permits the creation of applications in a faster, easier, and scalable manner, which addresses the ever-growing importance of app development in today's digital age. This paper focuses on the architectural design, logic building techniques and integration strategies of a no-code app

builder. The distinctive feature of this paper is the modular approach which provides an end to end solution. The paper goes beyond the conceptual paradigm and incorporates elementary design validation via system architecture and workflow system design. Hence the study is adequately positioned as an architectural/conceptual level foundational work with a focus on applied research and validation.

2. LITERATURE REVIEW

2.1 Evolution of No-Code Platforms

No-code development now includes fully-fledged application development platforms instead of only simple website builders such as Wordpress and Wix. Work flow automation tools, along with visual programming environments such as Bubble and Adalo, were milestones. Recent developments, such as automation powered by AI in OutSystems and AppGyver, have improved scalability and backend integration [15]. Now, enterprise solutions, AI-driven automation, and seamless API integrations for improved versatility are the primary focus.

2.2 Current No-Code Solutions & Their Shortcomings

The principal no-code platforms are Bubble, versatile web app builder without native mobile support; Adalo, with backend control mobile development simplified; and Thunkable, limited cross-platform with no user interface customization. Common drawbacks are limited customization, gated scalability, and high monthly rates [6]. Numerous platforms have unsophisticated solutions to complex workflows, high-performance processing, and enterprise security. Drag & Build seeks to address the lack of flexibility, real-time database connections, AI automation, and robust security.

2.3 No-Code Development Technology Stack

Development of no-code software employs no-code technologies for frontend, backend, and database boundaries.

- Frontend: Implements React.js, Vue.js, or Flutter for interactive drag and drop UIs with components modularized and rendered in real time.
- Backend & Databases: Incorporates and integrates Firebase (real-time database), Node.js (server-side logic), and PostgreSQL/MySQL (structured data storage) to enhance scale and security.
- APIs & Integrations: Stripe and PayPal (payment gateways), Google Analytics and Mixpanel (analytics), and Zapier and Integromat (automation services) are all seamlessly connected via RESTful APIs, GraphQL, and webhooks.

Using modular, API-based technologies, Drag & Build provides effortless enterprise system integration, database integration, and scalable enterprise app deployment, increasing the efficiency and accessibility of no-code development platforms. Table 1 shows a comparison of NoCode App builders.

Table 1: Feature Comparison of No-Code App Builders

Feature	Drag & Build	Adalo	Thunkable
Drag & Drop UI	✅ Yes	✅ Yes	✅ Yes
Customization	✅ High	✅ Moderate	❌ Limited
AI Assistance	✅ Yes	❌ No	❌ No
API Integration	✅ Yes	✅ Yes	❌ Limited
Database Support	✅ Yes (SQL & NoSQL)	✅ Yes (Internal DB)	✅ Yes (Limited)
Pricing Model	Free & Paid	Paid	Free & Paid

3. METHODOLOGY

3.1 System Architecture of Drag & Build

The architecture of a Drag-and-Drop App Builder is designed to provide a user-friendly, scalable, and performance-driven development environment for non-technical developers [9]. Figure 2 shows the workflow. It consists of various components that work together in unison to provide a seamless app-building experience, such as a visual interface, automation of logic, backend integration, and database connectivity.

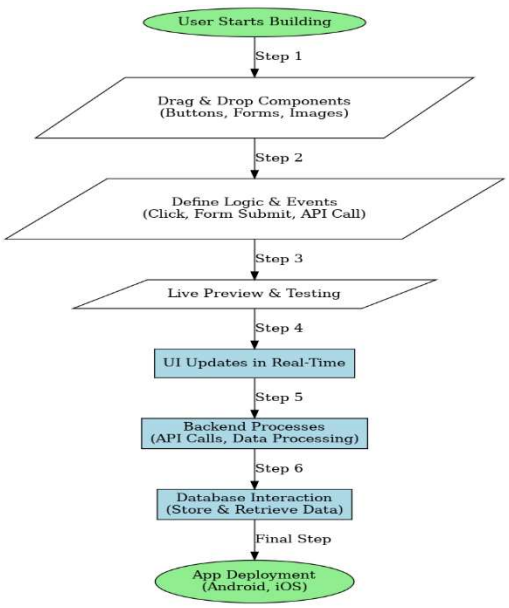

Figure 2: Workflow of Drag & Build

3.1.1 Main Components of the Drag & Drop Architecture

In every component foundational to the architecture, each layer serves a unique purpose based on a a component based subsystem ready for Modular Engineering. For example, in the User Interface(UI) Layer, users are free to arrange various UI objects such as buttons, images, and input fields on a drag-and-drop canvas as they desire. Users can also view in real-time the app changes and use the real-time preview engine which allows app changes to be viewed vigorously. Workflows are described in the Logic Builder section and are defined using trigger events and conditional logic, as well as through flexible, no-code frameworks [10]. Within the app, backend functionalities are well taken care of through cloud storage and API integrations enabling smooth interactions with databases and third party services.

3.1.2 Technical Operation of the Drag & Drop Interface

The smooth drag-and-drop feature can be attributed to dynamic visualization, custom event listeners, and state management on the frontend. When a user drags a UI component from the component library and drops it onto a canvas, event listeners on an object such as onDragStart, onDragOver, and onDrop, which become part of the user interface, track and process the drag-drop action. Each event modifies an organized JSON-based representation of the app which stipulates every components and its properties. Synchronization is driven within the user interface and backend through state management libraries like. For component rendering, the platform employs driven state management systems such as the HTML5 Drag and Drop API, React DnD, and the Konva.js for a smooth, responsive experience.

3.1.3 Defining App Workflows with the Logic Builder

Logic Builder leans on Driven Designs principle starting with an evnt programming methodology, users do define action to be taken based on a Trigger; a trigger may constitute a click, a form submission, or a data alteration. The resolution of a Trigger through Conditionals (if-else) Automabatan workflows permits dynamic interactions. Workflows are saved as declarative JSON, which the logic engine of the platform processes workflows from. The system upholds intricate automation processes with ease, providing multi-step workflows, API integrations, and Database interactions to an app.

3.1.4 Database Connectivity Options

For Data Storage and Data Retrieval, the platform supports different types of databases, which include NoSQL, Relational, and Cloud-based databases. Drag & Build depicts the Pairwise Bond represented under comparison of the Database. Google Firebase Firestore and MongoDB Atlas demonstrates powerful NoSQL databases with dynamic data update functionality needed for Real Time applications [11]. The platform incorporates Relational Database Management systems such as PostgreSQL, MySQL, and Microsoft SQL Server for storing organized data. External databases can be connected using RESTful APIs or GraphQL endpoints, allowing easy integration of third-party data. Local storage systems such as IndexedDB and SQLite allow applications to function offline and synchronize data with cloud storage when connectivity is restored.

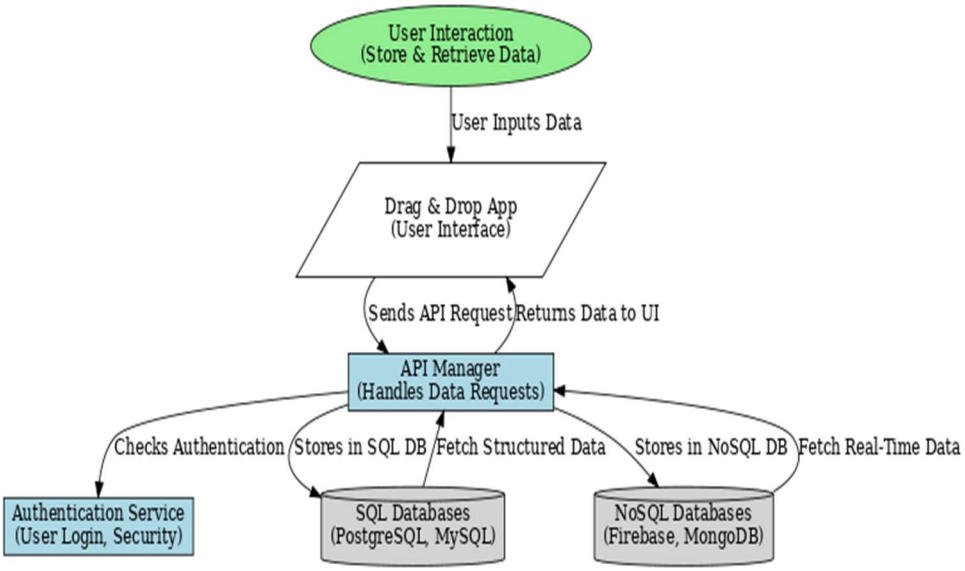

Figure 3: Database Connectivity in Drag & Build

By integrating modular components, real-time rendering, automated workflow, and diverse database integration, the Drag-and-Drop App Builder ensures a successful but simple development process, enabling anyone, irrespective of industry, to develop apps with ease.

3.1.5 Deployment & Maintenance – Management of Updates and Improvements

After extensive testing, the platform will be hosted on the cloud platforms AWS, Firebase Hosting, or Vercel, depending on the needs for availability and scalability. Figure 4 illustrates the testing and deployment pipeline. The deployment phase contains CI/CD (Continuous Integration and Deployment) pipelines for automated versioning and deployment, bug fixing, and new feature releasing. For monitoring user participation and feedback, improvement areas are gated and analytics are incorporated to assist in tracking engagement and improvement areas [12]. Maintenance in this context deals with patching security holes, improving performance, and adding new features to retain the platforms market position and operational efficacy.

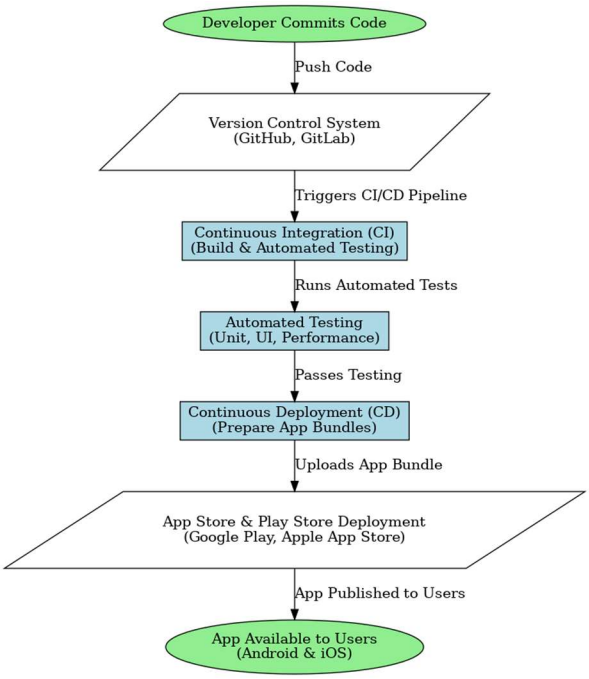

Figure 4: Testing & Deployment Pipeline for Drag & Build

With this structured development cycle, the Drag-and-Drop App Builder can achieve its mission to provide an effortless, no-code app development experience while ensuring scalability, usability, and security at the same time.

3.2 Development Phases of Drag & Drop App Builder

In the case of the Drag-and-Drop App Builder, the entire platform is built in a scaled, secure, and user-friendly manner, through a meticulously planned lifecycle process, which spans five steps: Requirement Analysis, Design, and Prototyping, Implementation, Testing, and Deployment and Maintenance. Each phase is instrumental in solving a user's problem, increasing the comfort of using the platform, and ensuring the app's usefulness in the future.

3.2.1 Requirement Analysis – Addressing User Pain Points and Needs

The first step is aimed towards problem solving and designing a suitable system. Many entrepreneurs, small businesses and non mannered users consider coding as a challenging, time consuming, and an expensive step, which most of the time is, in the process of app development. Other complex themselves through visual interfaces which unzip the cloud barrier integration through which development is simplified through a Drag and Drop interface including functionalities which are built in. Real time feedback and feedback over the processes of customization and complex within deployment stages are subjected to the flexible process of code-free development.

3.2.2 Design & Prototyping – Laying out the UI for Usability

The user interface (UI) is simple and modular. Users can employ the drag and drop canvas to construct components. Users can access the reusable components from the sidebar. There is also a preview panel that provides feedback. Prototyping is conducted with the help of Figma, Adobe XD, or Sketch, ensuring unused and best practices of UI/UX such as a minimal design, logic sequence, and adaptable layouts, are carefully executed as premade elements of the development phase.

3.2.3 Implementation – Coding Techniques and Frameworks Used

Console development covers frontend and backend to combination of newest frameworks and tech. The frontend is responsive and interactive and could be built using React, Vue, or Flutter. The backend is built using Node with Express, Firebase, or Django and handles server endpoints for operation. Container components like Redux or Vuex are utilized for optimal data and flow across the components.

3.2.4 Testing – Ensuring Functionality and Security

A strategy that defines platforms performance, security, and reliability verification IS A MUST. Testing components in isolation using either Jest or Mocha performs Unit Testing, while Cypress or Selenium conducts E2E testing. E2E testing captures and confirms successful drag-and-drop resolves. Load testing captures systems performance and behavior with multiple active users. Penetration testing assists in auditing security for data breaches, unauthorized data, and API exploitation. Real users provide feedback in usability testing to improve the experience before rollout.

3.2.5 Deployment & Maintenance – Management of Updates and Improvements

After extensive testing, the platform can be scaled and made reliable by deploying to the cloud on AWS, Firebase Hosting, or Vercel, etc. with Vercel AWS connect. Automated pipelines for continuous integration assure Automated Versioning, Bug Fixing, Automated Testing, and Deprecation of Deprecated Modules. Feedback, engagement, and improvement surveys capture the experience and guide feature funnels. Automated Deprecation can be used on surfaced security vulnerabilities from either patching or rest to prevent unauthorized data, optimize Performance, or ensure the platform IS competitive and efficient.

In the circle of feedback, the Drag-and-Drop App Builder structured development cycle achieves its goals of scalability, usability, and security while maintaining no code app development.

3. EXPECTED RESULTS

The development of a Drag-and-Drop App Builder aims to deliver a holistic no-code development platform that enables users to build applications effortlessly. The end prototype will have a user-friendly interface, real-time customization, back-end integration, and single-click deployment features. By eliminating the inconvenience of traditional coding, the platform will be accessible to a broad base of users, from small businesses to enterprises, to create apps rapidly for diverse industries.

4.1 Expected Functionalities of the Final Prototype

The prototype will incorporate a drag-and-drop based visual editor that enables users to design apps with buttons, forms, images, navigation menus, and other ready-made elements. It will offer a real-time feedback preview and users will be able to instantly see the changes they are making. The users will be able to configure and design the workflows with labeled triggers and conditional logic without writing codes through the logic builder. Effortless connectivity will enable real-time data storage and retrieval with Firebase, PostgreSQL, and MongoDB. Users will be able to export their apps as Windows and native Android/iOS apps with one-click deployment.

4.2 Measuring User Experience (UX)

The user experience of the application will be measured qualitatively and quantitatively. Users will be able to finish given tasks to assess the application's ease of use, learning, and overall satisfaction. Usability testing paired with satisfaction assessment alongside goal completion will note where user behavior identification tools, like heatmaps and analytics, place pain points in the interface. Users are able to provide feedback which guides the platform in responsiveness and accessibility, allowing the platform to refine through A/B testing. Goals achieved, total time taken in goal completion, and errors made are metrics for these tests.

4.3 Industries That Benefit from No-Code Development

The no-code platforms will have a very affordable and fast application development approach for all industries. For example, in e-commerce the platform will help with the development of online stores, payment gateway integration, payment processing, and inventory management. On the other hand, the healthcare sector will be able to create applications for booking and managing patient appointments. It will also assist the EdTech sector with the development of internet-based classrooms. In addition, the platform will help start-ups and entrepreneurs build prototypes and test Minimum Viable Products (MVPs) without a development team. For similar reasons, used by other large firms, they will create internal tools, Human Resource Management Systems, and workflow automation applications.

4.4 Expanding Functionality with Third-Party Integrations

To maximize functionality, the platform will have third-party integration support via RESTful APIs, GraphQL, and webhook services. This will enable effortless integration with payment systems (Stripe, PayPal), cloud services (Google Drive, AWS S3), authentication systems (OAuth, Firebase Auth), and analytical platforms (Google Analytics, Mixpanel). Intelligent automation can go further by introducing chatbots, recommendation systems, and voice assistants. Hence, by enabling these integrations, the app builder will present a flexible and scaleable development environment for a broad user base through the drag and drop builder. The complete expected outcomes will certainly deliver a transformative change to the ease of app building which would extend to those users with no technical knowledge whilst still maintaining deep customization options.

4. CHALLENGES & LIMITATIONS

Even though there are many benefits of no-code development, there are quite a few challenges and limitations that need to be overcome in order to provide maximum performance, security, and flexibility. A Drag & Build App Builder needs to resolve all these issues in order to provide a scalable, secure, and customizable development process while still being user-friendly.

5.1 Performance Issues in No-Code Platforms

No-code app development faces optimization issues with performance. An example emulating poor app performance emerging from applications built with no-code platforms loaded with API calls is due to real-time updates being strained alongside caches. No-code/low-code architecture needing 'server-side rendering'' and 'progressive loading' can help manage memory. Integrating 'rendering techniques' with 'caching'' will improve real-time updates alongside triple API calls.

5.2 Balancing Customization with Ease of Use

No-code development has which permits extensive customization is a no-code/low-code architecture with optional scripting frameworks. Modular UI elements can also assist in scrapping excessive scripting and balance user experience whilst needing more toggles. No-code/low-code archiectures can provide more flexibility whilst preserving the drag and drop interface.

5.3 Security Flaws in No-Code Platforms

User authentication, the integration of external entities, the storage of sensitive information, and other interactions all raise access and privacy concerns. No-code applications have severe security vulnerabilities, including the potential for data breaches, leakage of sensitive information exposed through APIs, unmonitored and unmitigated open injection points, and poorly architected access control. To protect the platform, there are some controls in place including OAuth, JWT Tokens, RBAC with end-to-end encryption, and data anonymization, encryption, GDPR, and HIPAA compliance data anonymization. Compliance and security audits, and advanced penetration testing, among others, can automatically identify, evaluate, and remediate vulnerabilities and security compliance gaps.

5.4 Cross-Platform Compatibility

Limitations of the no-code platform. The benchmarking study illustrates the ongoing fight for real time operational interoperability across web, iOS, and Android systems [13]. The registered issues and operational inefficiencies are most often attributed to the variances in user systems, the document processing and rendering engines, and the display screens. In order to optimize user document processing and system performance, responsive frameworks such as Bootstrap and Tailwind CSS, and cross platform development tools such as Flutter and React Native must be combined. The combined use of native mobile export and progressive web app (PWA) further enhances synergetic performance.

Positioned as the platform to level the performance issues with the gaps in customization, the Drag and Drag App Builder has the potential to provide, in an integrated manner, unmatched no code development platform which is responsive to the multilayered needs of the users from the IT world and from other areas of the business, as a result of the containment of the security

and interoperability boundaries as well as the unobstructed usage of the Drag and Build App Builder.

5. FUTURE SCOPE

Future research will concentrate on carrying out simulated trials and a pilot evaluation to determine how effective the suggested strategy is. The approach will be improved for wider use, practical consequences will be highlighted, and empirical data will be included in these assessments.

6.1 No-Code App Development Advances

The evolution of no-code platforms will focus on enhanced customization, automation, and scalability. Development of the future will focus on optimizing design flexibility, backend automation, and backend performance on more sophisticated applications. Advanced collaborative tools will enable multiple users to work on the same project in real-time, making the platform appropriate for large-scale software engineering teams. Users would be able to trace modifications and revert or save to earlier states using enhanced component-level versioning. Also, integration with blockchain technology could enable the development of decentralized apps eliminating coding through the deployment of secure smart contracts.

6.2 Incorporating AI for Intelligent App Development

AI could assist in the automation of UI/UX workflows with real-time suggestions, advanced app predictions, and advanced voice feature integration. Figure 5 illustrates the automated app development process. Users could be complemented with AI intelligent assistants to guide them in application structuring by the best practice techniques to minimize design disfunctionalities and the downstream inefficiencies. Users could have the functionality of the app described in a few sentences and, using NLP, the system is capable of building sophisticated workflows. AI can also automate testing processes to expose and fix UI/UX-related issues pre-deployment.

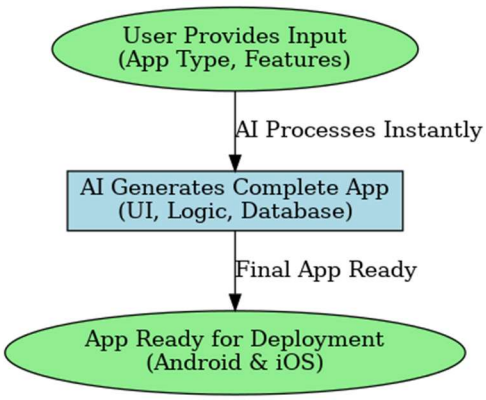

Figure 5: AI-Powered App Creation Workflow

6.3 Enterprise-Level Features for Scalability

To be adopted in businesses, no-code platforms must support large-scale, high-performance applications. Subsequent versions of drag-and-drop app builders can include role-based access control (RBAC), multi-user collaboration, and advanced analytics dashboards for business insights. Enterprise-level database support, such as Amazon RDS, Google BigQuery, and Microsoft Azure SQL, will facilitate the processing of large datasets efficiently [14]. Custom API integrations and microservices architecture will assist organizations in integrating their applications with other enterprise software such as ERP, CRM, and supply chain management systems.

6.4 Supporting Native Android/iOS App Exports

Many people currently expect that the next major feature to be added will be the direct export of fully functional native apps for Android and iOS devices. Most no-code tools still focus on the creation of Progressive Web Apps (PWAs) or hybrid applications. Further developments in Flutter, React Native, and Swift/Kotlin code generation will allow for direct export of native apps with improved performance, offline use, and device-level features like GPS, push notifications, and camera access. Furthermore, AI code generation could enable designers to augment and customize the exported applications with very low effort coding. Adding AI assistance will elevate the drag and drop interface to the next level, providing the power to the users with enterprise-grade functionality, and full stack native mobile support. Thus the future of no-code mobile app builders will bring even greater flexibility and performance, enabling easier app building while eliminating the gap between no-code and code app development.

6. CONCLUSION

No-code development is transforming software creation by making app development faster, more accessible, and cost-effective. Drag & Build provides a scalable, user-friendly platform that enables individuals and businesses to develop applications without coding. By integrating visual programming, workflow automation, and backend connectivity, it overcomes traditional limitations while ensuring customization, security, and cross-platform compatibility.

This research enhances no-code platforms by addressing customization, scalability, and security challenges through a modular, event-driven architecture, AI-powered automation, and seamless API integrations. Future advancements will focus on AI-driven app creation, enterprise solutions, and native mobile exports, ensuring Drag & Build remains at the forefront of no-code innovation.

The focus of this study was a comprehensive architecture development and technical methodology. The methodology has been validated through architectural modeling, workflows and prototype level testing to establish the technical soundness of the approach. However, in future, empirical validation with end user would provide further insights into usability, accessibility and adoption potential. The current methodology is at design and technical development level. However, it still makes a novel contribution by architectural foundation with implementation pathways.

REFERENCES

[1] Shridhar, Shreyas, and Siddharth Bose. "Analysis of low code-no code development platforms in comparison with traditional development methodologies." International Journal for Research in Applied Science and Engineering Technology 9, no. 12 (2021): 508-513.

[2] W. Nurharjadmo, M. A. Khadija, and T. Wahyuning, "Modern No-Code Software Development Android Inventory System for Micro, Small and Medium Enterprises," 2022 IEEE International Conference on Cybernetics and Computational Intelligence (CyberneticsCom), pp. 191-195, 2022. DOI: 10.1109/CyberneticsCom55287.2022.9865265.

[3] S. Heuschkel, "The Impact of No-Code on Digital Product Development," ArXiv, vol. abs/2307.16717, 2023. DOI: 10.48550/arXiv.2307.16717.

[4] L. Korada, "Low Code/No Code Application Development - Opportunity and Challenges for Enterprises," International Journal on Recent and Innovation Trends in Computing and Communication, 2022. DOI: 10.17762/ijritcc.v10i11.11038.

[5] D. I. DeSilva, R. Ranathunga, and R. Shangavie, "Quality Assurance in Low-Code/No-Code Development," 2023 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES), pp. 1-7, 2023. DOI: 10.1109/ICSES60034.2023.10465268.

[6] S. Liu, H. La, A. Willms, and R. Rhodes, "A 'No-Code' App Design Platform for Mobile Health Research: Development and Usability Study," JMIR Formative Research, vol. 6, 2022. DOI: 10.2196/38737.

[7] M. Kulkarni, "Deciphering Low-Code/No-Code Hype – Study of Trends, Overview of Platforms, and Rapid Application Development Suitability," International Journal of Scientific and Research Publications (IJSRP), 2021. DOI: 10.29322/ijsrp.11.07.2021.p11570.

[8] R. Picek, "Low-Code/No-Code Platforms and Modern ERP Systems," 2023 International Conference on Information Management (ICIM), pp. 44-49, 2023. DOI: 10.1109/ICIM58774.2023.00014.

[9] M. Moskal, "No-Code Application Development on the Example of Logotec App Studio Platform," Informatyka, Automatyka, Pomiary w Gospodarce i Ochronie Środowiska, vol. 11, pp. 54-57, 2021. DOI: 10.35784/IAPGOS.2429.

[10] Kesavan, Ram, David Gay, Daniel Thevessen, Jimit Shah, and C. Mohan. "Firestore: The nosql serverless database for the application developer." In 2023 IEEE 39th International Conference on Data Engineering (ICDE), pp. 3376-3388. IEEE, 2023.

[11] Heuschkel, Simon. "The impact of no-code on digital product development." arXiv preprint arXiv:2307.16717 (2023).

[12] Sido, Nourjan, Eksan Ahmed Emon, E. Ahmed, E. Supervisor, and M. Falch. "Low/No Code Development and Generative AI." PhD diss., Aalborg University, 2024.

[13] Wu, Chuhao, José Miguel Pérez-Álvarez, Adrian Mos, and John M. Carroll. "Codeless app development: Evaluating a cloud-native domain-specific functions approach." arXiv preprint arXiv:2210.01647 (2022). .

[14] Desmond, Michael, Evelyn Duesterwald, Vatche Isahagian, and Vinod Muthusamy. "A no-code low-code paradigm for authoring business automations using natural language." arXiv preprint arXiv:2207.10648 (2022).

[15] Faria, Nuno, José Pereira, Ana Nunes Alonso, and Ricardo Vilaça. "Towards generic fine-grained transaction isolation in polystores." In VLDB Workshop on Data Management and Analytics for Medicine and Healthcare, pp. 29-42. Cham: Springer International Publishing, 2021.