Low-Code and No-Code AI Tools In Modern Web Development

Avinash Amaladoss Sengole Mary
Thakur College of Science and commerce

***Abstract*—Low-code/no-code (LCNC) platforms with integrated AI are revolutionizing modern web development by allowing both professional and citizen developers to build applications**
faster and more efficiently through visual interfaces and pre-built components. These tools use AI to automate tasks like design, content creation, and testing, and can integrate with existing platforms or create new applications, which increases agility and addresses the shortage of skilled developers. These tools democratize access to AI, reduce development time, and open new pathways for innovation across industries.

- **Low-code: Requires some coding knowledge for complex applications, but uses visual tools to speed up development.**
- **No-code: Allows users with no coding experience to build applications using drag- and-drop interfaces and pre-built components.**
- **AI integration: AI assists with design suggestions, generating code snippets, writing content, and automating testing, making the development process more efficient.**

I. SCOPE AND SIGNIFICANCE

Low-code and no-code modular approaches let professional developers quickly build applications by relieving them of the need to write code line by line. They also enable
business analysts, office administrators, small-business owners and others who are not software developers to build and test applications. These people can create applications with little to no knowledge of traditional programming languages, machine code or the development work behind the platform's configurable components.
The scope of LCNC AI in web development:

- Rapid prototyping: Quickly build and test ideas with a minimum viable product (MVP).
- Internal tools: Create custom internal dashboards and applications to streamline business processes.
- Website and app building: Develop websites and mobile applications without writing extensive code.
- Workflow automation: Automate backend processes and integrate different services.

II. OBJECTIVES

The main difference between low-code and no-code development platforms lies in how much coding knowledge the user needs. Low-code development platforms (LCDPs) require some basic coding skills for users to develop and integrate complex applications, while no- code development platforms (NCDPs) do not require programming knowledge at all.
Because most organizations have a wide range of technical skill sets within their workforce, many platforms offer both low-code and no-code tools.

- Increase flexibility: To give businesses the ability to respond to market changes and new opportunities more quickly by rapidly prototyping, testing, and deploying new features and applications.
- Boost developer productivity: To free up professional developers from writing boilerplate code, allowing them to focus on more complex, high-value tasks and innovation.

III. TECHNOLOGIES AND TOOLS

Low-code/no-code platforms stem from earlier rapid application development (RAD) tools such as Excel, Lotus Notes and Microsoft Access that similarly put some development-like capabilities into the hands of business users (i.e., non-IT professionals).
However, those tools required users to thoroughly understand the business apps and their development environments in order to build capabilities. In contrast, with low-code and no-code options' drag-and-drop features, users need either minimal or no knowledge of

the tools or development in general.

Experience with low-code and no-code development indicates that for successful projects it is desirable to work with experienced programmers who can assist with the overall
process, e.g., by identifying potential issues before the app is released.

From a production perspective, it is important to determine who will be using an app
developed using such tools. For example, development with RAD tools generally produces capabilities used by the individual who creates the functionality, or by a limited number of users associated with the creator (e.g., a workgroup or business unit). By contrast, apps produced with low-code or no-code platforms might be robust enough to be used across departments and throughout the entire enterprise, and even by external users such as
customers and business partners. Again, it is important to have access to experienced programmers who can review the app before it is put into production, especially at an enterprise level, and to ensure that security has been addressed.

Below is an unranked, alphabetical list of the most common low-code platform vendors and tools:

- Appian.
- Claris FileMaker.
- Dwkit.
- Google AppSheet.
- Looker 7.
- Mendix.
- Microsoft PowerApps.
- OutSystems.
- Robocoder Rintagi.
- Salesforce Lightning.
- Skyve Foundry.
- SIB Visions VisionX.
- Wix Editor X.
- Yellowfin 9.
- Zoho Creator.

No-code development platforms:

- Airtable.
- AppGyver.
- Google AppSheet.
- Appy Pie.
- Betty Blocks.
- Bubble.
- Carrd.
- Clickfunnels.
- Glide.
- Gumroad.
- Kartra.
- Kissflow.
- Memberstack.
- Notion.
- Outgrow.
- Payhere.
- Quickbase.
- Shopify.
- Stripe.
- Umso.
- Voiceflow.
- Zapier.
- Zudy Vinyl (Zudy acquired by Jitterbit in 2023).

IV. APPLICATIONS AND USE CASES

Low-code and no-code AI tools are increasingly being adopted across diverse sectors, including:

- E-commerce: Product recommendations, personalized marketing, inventory prediction.
- Healthcare: Symptom checkers, predictive analytics for patient outcomes, administrative automation.
- Finance: Fraud detection, credit scoring, customer service chatbots.
- Education: Intelligent tutoring systems, automated grading, content personalization.
- Customer Support: AI-powered chatbots, sentiment analysis for ticket prioritization.
- Marketing and Sales: Content generation, customer segmentation, predictive lead scoring.

V. CHALLENGES AND FUTURE DIRECTIONS

Although many organizations embrace these platforms and development tools to rapidly develop new business apps, they also contend with the problems and challenges generated by these platforms.

Because these tools are low-cost and easy to use, organizational leaders can, and often do, lose track of what their employees are building. This could mean there's no visibility or
oversight to the data being generated, used or even

inappropriately exposed in apps. It also could contribute to more shadow IT.

Customization and scalability are often limited with low-code and no-code tools, and this must be considered when evaluating prospective products. The new app or website might deliver the bare minimum of functions, possibly impacting development time. If more capabilities are required, it might be necessary to increase the dependency on an experienced programmer.

Another potential challenge is how to manage, maintain and test these apps, as well as the potentially escalated infrastructure and storage costs associated with the proliferation of development activity enabled by these platforms.

Additionally, organizations might find that some tasks for which citizen developers or professional development teams have used these tools weren't well-suited to low-code and no-code methods or platforms, and this could represent a significant waste of resources.

Security features of low-code and no-code tools must be carefully evaluated, so that newly developed apps will have sufficient protection from unauthorized access or cyberattacks.

VI. THE FUTURE OF LOW-CODE/NO-CODE APPLICATION DEVELOPMENT

Industry experts predict the future of low code will see continued enterprise adoption, especially for fast development and specific business needs, although low code won't entirely replace traditional application development.

Analysts at Gartner project the low-code market to reach $44.5 billion in revenue by 2026, with a compound annual growth rate (CAGR) of 19.2% from 2021 through 2026. Specifically, the low-code platform segment is projected to expand to more than $18 billion in 2026, with a CAGR of more than 20%.

Low-code and no-code use is likely to increase among LOB workers, while professional developers will continue to use it as well to help them with less complex programming tasks.

According to Forrester, the top areas for low-code use continue to be business process or workflow applications, web and mobile front ends, and customer-facing applications.

Experts predict that eventually low-code will expand into broader areas such as reengineering technology stacks and ecosystems.

Enterprises will continue to practice traditional development for applications that require extensive application functionality, data governance and deployment to specific

architectures or environments. Security considerations are among the top issues to

address, no matter how apps or websites are developed. Use of AI is expected to increase steadily, as many vendors today advertise the inclusion of AI features in their products.