The Rise of the Citizen Developer: An Exploratory Case Study on AI-Agent-Assisted Application Development by Non-Programmers

Category: Case Studies | Published: January 25, 2026

The Rise of the Citizen Developer: An Exploratory Case Study on AI-Agent-Assisted Application Development by Non-Programmers

Abstract

The proliferation of advanced Artificial Intelligence (AI), particularly large language models, has given rise to a new class of development tools: AI coding agents. These agents promise to democratize software development by enabling individuals with little to no formal programming expertise to create functional applications. This blog presents an exploratory case study documenting the journey of a non-programmer using a publicly available, token-limited AI coding agent to develop three distinct software projects of increasing complexity. The projects include a database-driven CRUD (Create, Read, Update, Delete) application, a static website with a content management backend, and a complex, multi-step research agent. The findings demonstrate that current AI agents can successfully scaffold the entire development process, from conceptualization and code generation to debugging and deployment, significantly lowering the barrier to entry for software creation. However, the process also highlights critical challenges related to the reliability of generated code, the need for user oversight, and the emerging questions of responsibility and governance in an era of AI-human collaboration. This study contributes to the understanding of AI's practical impact on software engineering, illustrating the potential of AI teammates while underscoring the pitfalls and the evolving role of human developers.

1. Introduction

The field of Artificial Intelligence (AI) is experiencing a period of explosive growth and widespread excitement, reminiscent of previous technological waves but with a seemingly greater potential to revolutionize diverse domains (Venkatasubramanian, 2018). From legal analytics (Ashley, 2017) to autonomous vehicles (Zhang, 2022), AI is reshaping professional practices and creating new possibilities. A particularly transformative development is the emergence of autonomous coding agents, which function as AI teammates for software developers (Li, 2025). These agents are capable of understanding natural language prompts, generating code, and engaging in a collaborative development process (Wienholt, 2025).

This technological shift has profound implications for individuals without formal programming skills, often referred to as "non-programmers" or "application developers." Historically, creating software required specialized knowledge, but low-code and no-code platforms have begun to lower this barrier (Cabot, 2022). AI coding agents represent a quantum leap beyond these platforms, offering a conversational and dynamic interface for application creation. This has been a long-standing goal in computing, with early research exploring authoring tools to help non-programmers design complex systems like intelligent tutors (Dermeval, 2017) and scientific analysis tools (Kräter, 2021).

This blog presents an exploratory case study of a non-programmer leveraging a freely available AI coding agent to build three software projects of escalating complexity in a single weekend. The study aims to provide an empirical account of the capabilities and limitations of current-generation AI agents in democratizing software development. By documenting the process, challenges, and outcomes, we explore the practical implications of these tools for education, innovation, and the future of software engineering itself.

2. Background and Related Work

The concept of AI agents as active participants in complex systems has a rich history, grounded in foundational work on knowledge representation and logical reasoning to model dynamical systems (Reiter, 2001). Modern AI has evolved from specialized systems, such as those for game playing (Jebari, 2020), to more general, collaborative partners. In software development, this evolution can be seen in the progression from early pattern-based AI scripting tools (McNaughton, 2003) to today's sophisticated coding agents.

Recent studies have begun to quantify the impact of these agents. An analysis of over 456,000 pull requests on GitHub initiated by AI agents found that while they accelerate code submission, their contributions are often structurally simpler and less frequently accepted than human-authored code, indicating a trust and utility gap (Li, 2025). This highlights a critical tension: the promise of productivity versus the need for rigorous oversight.

The educational sphere provides a parallel context for understanding how AI can support novices. Research into conversational agents for introductory programming has shown their potential as supplementary learning tools (Barzanji, 2025). Similarly, studies on AI in simulation-based learning emphasize how virtual agents can guide learners through situated experiences (Dai, 2022). These educational applications share a common goal with coding agents: empowering individuals, whether students or non-programmers, to achieve complex tasks (Lee, 2025; Smith, 2020).

However, this empowerment is not without risks. A significant challenge is the tendency of novices to uncritically accept AI-generated output, raising concerns about academic integrity and the development of critical thinking skills (Chugh, 2025). This mirrors a long-standing issue in game development, where there is a tension between the creative freedom desired by designers (who may not be programmers) and the technical effort required from programmers to debug faulty AI specifications (Llansó, 2009). As AI agents become more autonomous, questions of leadership, responsibility, and even inventorship become paramount (Hedlund, 2022; Smith, 2018; Yanisky-Ravid, 2017).

3. Methodology: A Case Study in AI-Agent-Assisted Development

This study documents an exploratory case study to investigate the process of a non-programmer using an AI coding agent. The subject had no prior professional software development experience. The primary tool was the free-tier version of a commercially available large language model (Claude), accessed via its web interface. This choice introduces a realistic constraint: the model's context window (token limit), which necessitates breaking down complex problems into smaller, manageable conversational turns.

The study was structured around the development of three distinct projects, designed to test the agent's capabilities across a range of tasks and complexities:

1. **Project 1: CRUD Application:** A simple database management tool. The requirements were to create a Python application with a Streamlit user interface to perform Create, Read, Update, and Delete operations on a local MySQL database. This

project tests the agent's ability to generate code for backend logic, database interaction, and frontend UI components.

2. **Project 2: Static Blog Site with JSON Backend:** A personal blog website. The requirements included a responsive front end built with Bootstrap 5 and HTML, which dynamically populates content from local .json files. A separate administrative dashboard was also required to allow the user to input a title, description, and blog content (in HTML) and have it converted into the correct .json format. This project assesses the agent's proficiency in multiple languages (HTML, CSS, Python) and its ability to handle data structuring and file I/O.
3. **Project 3: Autonomous Research Agent:** A sophisticated tool for automated research and content creation. The agent was required to take a topic as input and execute a multi-step plan: (1) devise a research plan for user review, (2) conduct initial research using a locally run LLM via Ollama, (3) perform a gap analysis on the initial findings, (4) conduct targeted web searches to fill identified gaps, and (5) synthesize all information to write a comprehensive blog post in both HTML and Markdown formats. This project evaluates the agent's capacity for higher-order tasks like planning, analysis, and multi-source synthesis, moving beyond simple code generation towards more autonomous, agentic behavior (Sardana, 2024).

Throughout each project, the interaction between the user and the AI agent was documented, including the prompts provided, the code generated by the agent, and the iterative debugging and refinement process.

4. Results

The non-programmer successfully completed all three projects using the AI coding agent as the primary development tool. The process was characterized by a conversational, iterative workflow where the user described desired functionality in natural language and the agent provided code snippets, explanations, and debugging assistance.

Project 1 (CRUD Application): The agent successfully generated the full Python script for the application. It correctly utilized the streamlit library for the user interface and mysql-connector-python for database operations. The user was able to create a functional application that could connect to a local MySQL server, display existing records in a table, and provide forms for adding, updating, and deleting records. The main challenges involved debugging connection errors and handling user input validation, which required several rounds of clarification with the agent.

Project 2 (Blog Site): The agent demonstrated proficiency across multiple technologies. It generated a clean, responsive HTML5 template using Bootstrap 5 CDN links. It also produced a separate Python script for the admin dashboard, which used the tkinter library for a simple GUI and the json library to write user inputs to formatted files. The agent was able to maintain context across different parts of the project, ensuring the JSON structure it created in the backend script matched the structure expected by the JavaScript code it wrote for the frontend.

Project 3 (Research Agent): This project marked a significant increase in complexity, requiring the agent to orchestrate a series of tasks rather than just generate code for a single application. The agent successfully produced a Python script that implemented the specified workflow. It generated a coherent research plan based on a given topic. It correctly used the requests library to interact with the local Ollama API for initial research and employed web scraping libraries like BeautifulSoup for the gap-filling stage. The final output was a well-structured blog post in both HTML and Markdown. This demonstrated a capability aligned with AI planning agents that compose complex services from sub-processes (Shi, 2023). The primary hurdle was managing the complexity and statefulness of the process within the constraints of the conversational interface and token limits, requiring the user to carefully manage the agent's focus.

5. Discussion

The successful completion of these three projects by a non-programmer has significant implications. It provides a practical demonstration that modern AI coding agents can serve as effective force-multipliers, enabling individuals to build non-trivial software without traditional training. This signals a potential paradigm shift in software development, moving towards a future of human-computer collaboration where each party contributes its strengths (Ashley, 2017). The role of the human shifts from a line-by-line coder to a system architect and project manager, providing a specific purpose for the project while guiding the AI's efforts and validating its output.

However, the study also highlights several critical challenges. The agent often produced code with bugs or inefficiencies, and the user, lacking deep technical expertise, had to rely on the agent itself for debugging. This creates a potential feedback loop where the tool that introduced the error is also the primary means of correcting it and requires the user to provide clear instructions so objectives can be consistently met. This underscores the risk of novices

uncritically accepting AI output and the importance of developing new digital literacy skills focused on prompt engineering and critical evaluation of AI-generated artifacts (Chugh, 2025).

Furthermore, as these agents become more capable, the nature of programming itself may change, potentially altering the skills required in emerging technological landscapes like the metaverse (González-Argote, 2022). The case of the research agent, which performed planning and synthesis, pushes the boundaries of simple code generation. It functions as a system that models knowledge and executes actions, echoing foundational AI concepts (Reiter, 2001). This raises complex questions about responsibility. If an autonomous agent developed by a non-programmer causes harm or produces faulty output, who is accountable: the user, the AI developer, or the agent itself? This points to a growing need for frameworks governing AI expert responsibility and the leadership of mixed human-AI teams (Hedlund, 2022; Smith, 2018).

6. Conclusion

This exploratory case study demonstrates that AI coding agents have reached a level of maturity where they can empower non-programmers to create a range of functional software applications. The journey from a simple CRUD app to a complex research agent illustrates a steep reduction in the technical barriers to software creation. This democratization of technology holds immense promise for innovation, enabling domain experts in fields from science (Kräter, 2021) to engineering (Venkatasubramanian, 2018) to build their own tools.

However, the promise of AI is accompanied by significant pitfalls. The reliability of AI-generated code, the potential for over-reliance by novice users, and the profound ethical and legal questions surrounding responsibility and governance must be addressed. The rise of the AI collaborator (Li, 2025) does not eliminate the need for human oversight but rather transforms it. Future research should focus on developing methodologies for verifying AI-generated code, establishing best practices for human-AI collaboration in software development, and creating educational frameworks to equip a new generation of citizen developers with the critical skills needed to navigate this new technological landscape.

References

Ashley, K (2017). Artificial Intelligence and Legal Analytics. *Cambridge University Press eBooks*, . https://doi.org/10.1017/9781316761380

Barzanji, C, & Loitsch, C (2025). Exploring conversational agents for novice programmers: a

scoping review. *Discover Artificial Intelligence*, *5*(1). https://doi.org/10.1007/s44163-025-00521-4

Cabot, J, & Clarisó, R (2022). Low Code for Smart Software Development. *IEEE Software*, *40*(1), 89-93. https://doi.org/10.1109/ms.2022.3211352

Chugh, R, Turnbull, D, Morshed, A, Sabrina, F, Azad, S, Rashid, M, Kaisar, S, & Subramani, S (2025). The Promise and Pitfalls: A Literature Review of Generative Artificial Intelligence as a Learning Assistant in ICT Education. *Computer Applications in Engineering Education*, *33*(2). https://doi.org/10.1002/cae.70002

Dai, C, & Ke, F (2022). Educational applications of artificial intelligence in simulation-based learning: A systematic mapping review. *Computers and Education Artificial Intelligence*, *3*, 100087-100087. https://doi.org/10.1016/j.caeai.2022.100087

Dermeval, D, Paiva, R, Bittencourt, I, Vassileva, J, & Borges, D (2017). Authoring Tools for Designing Intelligent Tutoring Systems: a Systematic Review of the Literature. *International Journal of Artificial Intelligence in Education*, *28*(3), 336-384. https://doi.org/10.1007/s40593-017-0157-9

González-Argote, D (2022). Immersive environments, Metaverse and the key challenges in programming. *Metaverse Basic and Applied Research*, *1*, 6-6. https://doi.org/10.56294/mr20226

Hedlund, M, & Persson, E (2022). Expert responsibility in AI development. *AI & Society*, *39*(2), 453-464. https://doi.org/10.1007/s00146-022-01498-9

Jebari, K, & Lundborg, J (2020). Artificial superintelligence and its limits: why AlphaZero cannot become a general agent. *AI & Society*, *36*(3), 807-815. https://doi.org/10.1007/s00146-020-01070-3

Kräter, M, Abuhattum, S, Soteriou, D, Jacobi, A, Krüger, T, Guck, J, & Herbig, M (2021). AIDeveloper: Deep Learning Image Classification in Life Science and Beyond. *Advanced science (Weinheim, Baden-Wurttemberg, Germany)*, *8*(11), e2003743. https://doi.org/10.1002/advs.202003743

Lee, D, & Ah, S (2025). Case study on designing interactive AI-powered digital literacy lessons for non-programmers in K-12 and analyzing their impact. *The Journal of Korean Association of Computer Education*, *28*(11), 1-10. https://doi.org/10.32431/kace.2025.28.11.001

Li, H, Zhang, H, & Hassan, A (2025). AIDev: Studying AI Coding Agents on GitHub. *arXiv*, . https://doi.org/10.5281/zenodo.16919051

Llansó, D, Gómez-Martín, M, & González-Calero, P (2009). Self-Validated Behaviour Trees through Reflective Components. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, *5*(1), 70-75. https://doi.org/10.1609/aiide.v5i1.12353

McNaughton, M, Redford, J, Schaeffer, J, & Szafron, D (2003). Patttern-Based AI Scripting Using ScriptEase. *Lecture notes in computer science*, , 35-49. https://doi.org/10.1007/3-540-44886-1_6

Reiter, R (2001). Knowledge in Action. *The MIT Press eBooks*, . https://doi.org/10.7551/mitpress/4074.001.0001

Sardana, A, Sethuraman, S, & Kalyanasundaram, P (2024). Compliance-as-Code 2.0: Orchestrating Regulatory Operations with Agentic AI. *Journal of Artificial Intelligence General science ISSN 3006-4023*, *5*(1), 546-563. https://doi.org/10.60087/jaigs.v5i1.366

Shi, D, & Gan, S (2023). YUMA – An AI Planning Agent for Composing IT Services from Infrastructure-as-Code Specifications. *Proceedings of the ... Annual Hawaii International Conference on System Sciences/Proceedings of the Annual Hawaii International Conference on System Sciences*, . https://doi.org/10.24251/hicss.2023.129

Smith, A, & Green, M (2018). Artificial Intelligence and the Role of Leadership. *Journal of Leadership Studies*, *12*(3), 85-87. https://doi.org/10.1002/jls.21605

Smith, J, Truesdell, E, Freeman, J, Magerko, B, Boyer, K, & McKlin, T (2020). Modeling music and code knowledge to support a co-creative AI agent for education. *Zenodo*, , 134-141. https://doi.org/10.5281/zenodo.4245386

Venkatasubramanian, V (2018). The promise of artificial intelligence in chemical engineering:

Is it here, finally?. *AIChE Journal*, *65*(2), 466-478. https://doi.org/10.1002/aic.16489

Wienholt, N (2025). Using an AI Coding Agent. *Apress eBooks*, , 7-27. https://doi.org/10.1007/979-8-8688-1784-7_2

Yanisky-Ravid, S, & Liu, X (2017). When Artificial Intelligence Systems Produce Inventions: The 3A Era and an Alternative Model for Patent Law. *SSRN Electronic Journal*, . https://doi.org/10.2139/ssrn.2931828

Zhang, K, Zhao, L, Dong, C, Wu, L, & Zheng, L (2022). AI-TP: Attention-Based Interaction-Aware Trajectory Prediction for Autonomous Driving. *IEEE Transactions on Intelligent Vehicles*, *8*(1), 73-83. https://doi.org/10.1109/tiv.2022.3155236

© 2026, Dylan A. Bulseco, Ph.D., AI in Education Blog. All rights reserved.

This work is funded in part by NSF DUE 2300796, NSF ITE 2347375 and NSF DRL 2342700.