

AI SaaS Website Builder: An Intelligent Platform for Instant Website Generation

Dhyan Senghani, Mr. Ishan Thakkar

Department of Computer Science and Engineering, Parul Institute of Technology, Parul University, Gujarat, India

Abstract

Imagine how different the process of building a website was just a few years ago. You would hire a professional web developer, wait for weeks or even months, spend thousands of dollars, or alternatively, struggle with complicated drag-and-drop builders that still required significant technical understanding. These traditional approaches, though functional, created substantial barriers for small businesses, entrepreneurs, and individuals who simply wanted an online presence.

In this paper, we have guided the readers on our ideas towards creating an AI-powered SaaS website builder workflow, how we constructed it and tested it, and how it can be of value to users seeking instant website generation. The platform allows users to describe their desired website in plain language and receive a fully functional, deployed website within minutes. Our features included AI-driven code generation, one-click deployment, credit-based monetization, secure authentication, and smooth user interfaces. The most important contributions here have been the creation of an adaptable architecture to support prompt-to-website conversion, ensuring the interface performs excellently across devices, and actually analyzing the question of whether the platform is working by evaluating user reactions. What we discovered was fairly promising—users saved tremendous development time, non-technical individuals could create professional websites, and the credit-based system proved to be a sustainable monetization model. The AI generation approach proved to be a win-win as users receive instant results when they need them and the platform receives a way of sustaining its operations through subscription and credit purchases. This study contributes to the existing information about AI-powered development tools and provides practical guidance to anyone developing such a type of intelligent website generation platform.

Keywords: AI Website Builder, SaaS Platform, Automated Code Generation, Large Language Models, MERN Stack, One-Click Deployment

1. Introduction

It is quite astonishing how websites are built nowadays. I still recall that my colleagues would spend entire semesters learning HTML, CSS, and JavaScript just to create a basic portfolio page. Nowadays, I simply type a description of what I want, and artificial intelligence generates the entire code for me within seconds. This change from manual coding to AI-assisted generation has utterly revolutionized the business of web development and the way individuals attempt to achieve an online presence.

The new digital landscape has led to the emergence of AI-powered SaaS website builders as the intermediary between ideas and execution. They provide users with a methodical manner to demonstrate what they want and present developers with an efficient method of generating code. Consider what these platforms actually have been: to non-technical users, they represent the ability not to learn programming languages or hire expensive developers—to be able to describe requirements and receive working websites under a single roof. To businesses, particularly the small ones that cannot afford enormous development budgets, they imply literally reaching their customers with professional websites that were previously out of reach.

However, it is not only about individual websites. These platforms have been empowering digital entrepreneurship when they are used to lower the technical barrier to entry. Individuals will launch their ideas faster, the cost of development will reduce among all users, and creative energy will be spent efficiently on business logic rather than technical implementation. In the current fast-paced digital economy, in particular, these AI builders turned out to be essential tools—helping people to launch businesses quickly and enabling entrepreneurs to communicate their vision without technical delays.

Here's what our platform does:

I. AI-Powered Generation allowing individuals to describe their website in natural language and receive complete, functional code including HTML/CSS and content structure.

II. One-Click Deployment so that users can go from generation to live website without any additional technical steps or configuration.

III. Credit-Based Monetization which provides flexible access—users purchase credits to generate websites, with subscription options for frequent users.

IV. Authentication System that is secure everywhere, as after all—user data and generated websites must be protected.

V. MERN Stack Architecture built on MongoDB, Express, React, and Node.js to ensure scalability, performance, and modern development practices.

What follows in this paper is fairly simple: Section 2 examines the efforts of others in AI code generation and SaaS platforms. Section 3 goes down to finer details of the way we constructed everything. Section 4 provides what we discovered after testing the platform. Section 5 discusses what it was all about and where we lacked. Section 6 brings to a close and considers the way we might continue.

2. Literature Review

A. The Evolution of Website Building Tools

The methods of building websites have transformed dramatically over the past three decades. Actually, the first effort to enable non-technical users to create web content was through manual HTML coding in simple text editors. They were not unreasonably innovative in their era. However, they were badly limited in comparison. They required learning complex syntax, understanding browser compatibility issues, and spending hours debugging layout problems. You could not really visualize changes without uploading and refreshing repeatedly.

The first visual website builders emerged in the late 1990s with tools like Microsoft FrontPage and Adobe Dreamweaver. However, in the truth of things, the majority of them simply generated messy code that was difficult to maintain. You would have to work within rigid templates rather than having true creative freedom. They did not capitalize on the interactivity, the dynamism, the ability to really update things in real-time that modern web technologies enabled.

Then came content management systems such as WordPress, Joomla, and Drupal. You could simply install a CMS and choose a theme rather than building everything from scratch. This actually was a game-changer when WordPress began offering thousands of plugins and themes—you could actually extend functionality without writing code, not only read about how to do it.

B. The Rise of Drag-and-Drop Builders

Modern drag-and-drop builders like Wix, Squarespace, and Webflow represent the next evolution. Studies have revealed that those that are really good are merely great at a few major things. Users mainly interact with these platforms using visual interfaces, therefore, the experience should be intuitive. You would like to present users with what they desire and not overwhelm them with hundreds of options. The most interesting features include real-time preview (see changes immediately), responsive design tools (optimize for all screen sizes automatically), and extensive template libraries (start from professional designs).

The fact is that ease of use does matter. When you have an intuitive interface, you will have people creating websites even when they have no technical training. As user experience research claims, drag-and-drop interfaces are excellent for general users, but WYSIWYG (What You See Is What You Get) editing which allows you to see changes exactly as they will appear is much better when the user is seeking immediate visual feedback.

C. The Emergence of AI-Powered Code Generation

The most recent and perhaps most transformative development is AI-powered code generation. With the introduction of large language models like GPT-4 and similar technologies, it has become possible to convert natural language descriptions directly into working code. Research indicates that AI code generation can reduce development time by 40-60% for standard websites.

What makes AI generation credible? The quality of output is important (a model generating 100 lines of correct code is more valuable than one generating 1000 lines with errors). The issue of context understanding (the model must comprehend the user's intent beyond literal words). The general structure is also relevant. However, it is a delicate balance that one has to maintain because you want the AI to be creative and flexible, and you also want to prevent generating insecure or broken code. It is important to have validation and testing systems.

D. Making Money through SaaS Platforms

An AI website builder platform must have a way of making money, and therefore its monetization strategy is significant.

The most common method is the subscription model, where users pay on a recurring basis to access generation features. According to research on SaaS pricing, tiered subscriptions with distinct feature sets are the best—subscriptions enable the user to find the appropriate tier, they are reasonably priced and suitable for users with varying needs and budgets.

The credit-based model has been found to be especially successful for usage-based services. A certain number of free credits allows new users to test the platform—to get those network effects to make the platform worth it. Credit purchases will then generate revenue for users who want to generate more websites. The trick lies in properly explaining what the credits provide.

The transaction-based models charge per generation. This matches revenue with value delivered, but it introduces complication—you are required to have payment processing integration. This has been found to be most effective for occasional users who do not want a recurring subscription.

E. User Experience Matters

The experience that people gain throughout the platform has a strong impact on their future return.

The design of the generation interface influences attitude to quality and usability. Amazingly, the minimalist design

normally beats feature-rich interfaces for initial use. Yet you still would like progressive disclosure—basic options remain easy to use by beginners, but more advanced features are offered when experienced users need them.

The presentation of generated output has an impact on engagement and satisfaction. It has been found that split-view interfaces (code on one side, preview on the other) are more effective for developers, whereas full preview interfaces are superior for non-technical users. A combination of both would allow users to derive the merits of both in a hybrid approach.

The trust-building elements are real. Secure authentication, transparent pricing, clear data policies—all these factors determine user response to platform recommendations. Verification badges, responsive customer support, and clarity on how generation works give greater confidence to the user.

F. Where This Research Fits In

One can find loads of research concerning traditional website builders, and research about the specifics of AI code generation. Still, there are not many articles that have examined in depth AI-powered SaaS website builders that combine natural language processing, code generation, deployment, and monetization altogether. That is the gap, which is being addressed by this research which will involve documentation of the entire design and implementation of such a platform, whether it actually works based on user response, and contribution of information that will be of benefit to other developers who have been developing similar platforms.

3. Methodology

A. How We Structured the System

We constructed the platform using a modular MERN stack architecture—attending to concerns in different layers communicated by clearly defined interfaces. This process enables us to develop, test, and fix parts separately and still have the whole system running fairly well.

The Presentation Layer contains all the bits that users can actually view and interact with: the generation interface where users input prompts, the preview area where generated websites are displayed, and the dashboard where users manage their generated sites and credits. We applied responsive design to ensure consistency of the experiences regardless of whether one is on a desktop, tablet, or phone. With React we have component-based architecture, CSS3 provides layout flexibility and visual presentation, and the technology that runs all interactive elements includes real-time form validation and dynamic preview updates.

The Application Layer contains the business logic itself—namely the AI integration logic, user authentication logic, credit management logic, and deployment handling logic. This is a layer that lies between the presentation and data layers and enforces business rules and ensures consistency of data. Session management is used to track user state through

interactions, and caching policies are used to optimize access to data accessed regularly.

The Data Layer is concerned with permanent storage: user accounts, generated websites, credit balances, and transaction history. MongoDB was used as a NoSQL database to provide flexible schema design, efficient querying, and horizontal scalability. Optimized indexing ensures fast retrieval of user data and website records.

External Integrations add functionality by linking with OpenAI API (for code generation), Stripe API (for payment processing), and deployment services (to publish websites). These were contained by our service interfaces in order to remain independent of the core platform.

B. Core Features

AI-Powered Website Generation

The generation engine is fundamentally the core of the platform. We have placed a large text input area directly in the center which accepts natural language descriptions of the desired website. The AI integration processes this prompt and generates complete website code as you wait, thus it is able to deliver results even before you finish considering the full requirements.

The prompt processing uses LLM technology to understand user intent. We engineered prompts that extract key information: website type (business, portfolio, e-commerce, blog), desired sections (hero, about, services, contact), color preferences, and content requirements.

The code generation produces structured HTML/CSS/JSX output. This includes:

- Complete page structure with semantic HTML5 elements
- Responsive CSS styling that works on all device sizes
- Interactive JavaScript functionality where appropriate
- Mobile-first design principles

The preview functionality allows users to see their generated website immediately without leaving the platform. This real-time feedback loop enables iterative refinement—users can adjust their prompt and regenerate until satisfied.

Authentication System

This segment is entirely about providing users with secure access to their accounts and generated websites. Registration allows new users to create accounts where their credentials are verified with email confirmation.

JWT (JSON Web Token) implementation provides stateless authentication. Tokens are securely stored and validated on each request to protected routes. This ensures that user data remains private and that only authenticated users can access their generated websites and credit balances.

Login and signup systems include:

- Email and password authentication
- Password hashing for security
- Session management for persistent login
- Protected route middleware

Credit-Based Monetization

The credit system balances free access and premium features. New users receive initial free credits so that they can test the platform without making a financial commitment—this prevents the platform from being irrelevant and worthless to potential customers.

Standard credit packages offer additional generations for users who need more than the free tier. Each website generation consumes a certain number of credits based on complexity. Users can see their remaining balance at all times.

The purchase process guides users through the stages: select a credit package, pay via Stripe, and receive credits instantly. Different package sizes are highlighted in clear comparison tables. Payment processing is secure, and confirmation receipts complete the deal.

Credit management displays current balance, tracks consumption history, and allows users to purchase additional credits as needed. Automated deduction ensures credits are only consumed when generation is successful.

Payment Integration

Stripe API integration handles all financial transactions. This includes:

- Secure payment processing
- Multiple payment method support
- Subscription management for premium tiers
- Webhook handling for payment events

The payment flow is designed to be seamless—users select a package, enter payment details, and receive immediate credit addition to their account. All transactions are logged for audit and customer support purposes.

Deployment System

One-click deployment functionality allows users to publish their generated websites instantly. When a user is satisfied with their generated website, they can deploy it with a single click.

The deployment process:

- Generated code is packaged for deployment
- Code is sent to hosting infrastructure
- A unique URL is generated for each deployed site
- Users receive the URL immediately

Render cloud hosting was selected for its simplicity and reliability. The platform handles frontend and backend

hosting, and the one-click deployment feature eliminates technical barriers to going live.

C. User Interface Design

Our main focus was on accessibility, efficiency, and device friendliness.

The generation interface is prompt-based—a large text area and generate button at first sight with no additional clicking. Example prompts are displayed to help users understand what to write. The category selection presents an alternative path for users who prefer guided generation rather than free-form prompts.

The preview interface displays the generated website in an iframe or full view. The most important information is displayed prominently, while code view is available in expandable sections for technical users. Device preview toggles (mobile, tablet, desktop) increase engagement and provide real feedback on how the site will appear on different screens.

The dashboard bundles all management features using tabbed interfaces. Website history, credit balance, account settings, and deployment management are arranged in a logical manner. Contextual tips come in handy to assist the user in completing tasks with ease.

Responsive design implies that all things are displayed correctly depending on screen sizes. Navigation on smaller screens is reduced to mobile-friendly menus, and touch targets are sized to fit fingers correctly.

D. Development Approach

Our approach was iterative in nature—we had to plan it and respond promptly to new requirements.

We began with **requirement gathering** by researching existing AI website builders and asking potential users about their needs and pain points so that the product would become a reality.

System design converted these requirements into architecture, database schemas, and interface mock-ups which informed the build.

Core functions prototype development was achieved to provide working prototypes of AI generation capabilities in order to have initial feedback which allowed us to pivot without spending too much time on each functionality.

Implementation was staged: first AI integration (core value), then authentication and credit system, then payment integration, and finally deployment features. This staged release allowed us to provide value in the early stages and add more value in later stages.

Testing was carried out—unit testing on individual components, integration testing on connections, and acceptance testing by real users. The platform was tested during performance testing in which it was proved to handle expected traffic.

The **production environment** was configured through deployment, and monitoring and analytics enabled continuous optimization according to actual usage patterns.

E. System Architecture Diagram

The following diagram illustrates the complete workflow of the AI SaaS Website Builder platform, from user input to live deployment.



Figure 1: System Architecture and Workflow of AI SaaS Website Builder Platform

The diagram above demonstrates the complete flow:

1. User interacts with the **Presentation Layer** through the React-based interface
2. Requests are processed by the **Application Layer** containing all business logic
3. **External Integrations** handle AI generation, payments, and hosting
4. **MongoDB** persists user data, generated websites, and transaction records
5. The **bottom workflow** shows the sequential process from login to live deployment

4. Results and Analysis

A. Is the Platform in Reality Working?

The platform absolutely functions! Every single aspect which was intended does what it is expected to do.

The **generation feature** responds to every conceivable prompt and delivers working website code which is properly structured—super handy. The AI has become the magic thing—users type in what they want, and the system shows

them the generated website as it is being processed, therefore, they do not have to wait long for results.

The **prompt understanding** accurately interprets various types of requests. I put it to test using different queries—"build a gym website," "create a portfolio for a photographer," "make a restaurant landing page"—it worked well, keeping the structure appropriate and maintaining design consistency.

The **preview functionality** displays the generated content exactly as it would appear live, and the device toggle addition shows the clear appearance of each website on different screen sizes.

The **authentication flow** is solid: it is impossible to access protected routes without login, passwords are properly hashed, and after login, users are redirected where they expect to go.

The **credit purchase checkout** is quite straightforward: You select a package, make payment via Stripe, and credits appear immediately. All transactions are stored such that they can be audited or referenced by customer support whenever there are any requirements.

The **deployment system** is functional. I am able to click deploy, and the live URL appears as soon as the process completes. Deployed sites have proper URLs and are accessible to anyone.

B. What Users Thought

The comments of people who had used the platform informed us about what they liked and what they did not like or what we should do better.

According to **non-technical users**, they are happy as they can create websites without learning to code. They particularly adore the simple prompt interface, which removes all complexity. The free credit option is very popular among students and small business owners as there would be no charge for trying the platform. It is cool that even more advanced users appreciate the speed.

According to **developers**, the platform is very useful for rapid prototyping. They prefer quick generation and the ability to view and edit the generated code. The preview feature attracts plenty of thumbs up since it shows exactly what will be deployed.

The **credit system** receives favorable responses. Users like the fact that they can purchase only what they need. Businesses also like the subscription option for frequent use. Some proposed the introduction of team accounts and shared credits to gain more flexibility.

C. Performance Metrics

Tests were run on performance, and all was fair with normal load. Generation results were returned fast, even when the prompt was complex and detailed. Page loads beat the majority of industry standards—response was brisk and UX was enhanced by smart loading states.

It was tested with concurrent users on the site, and the site was able to cope, but with spikes, additional caching or a larger backend might be needed in the event of future growth. Database connections are pooled to ensure they are not exhausted.

Mobile experience also makes the point—the responsive design works on small screens and even on slower networks. Content loads quickly since we only load essential resources first, which preserves appearance but saves bandwidth.

D. Business Impact

Users and generated websites were good in terms of numbers. Websites with detailed prompts receive significantly higher quality output than vague ones. A definite indication that prompt quality is important to the final result. The paid features certainly increase generation quality—the numbers do confirm that. We do not know that it is always causal, however, the correlation is high.

Users are actively operating on the platform: they make multiple generations and refine their prompts. They do not abandon the platform, and this demonstrates that it is not a novelty. The platform is in fact connecting ideas to reality in a manner that would otherwise be hard to achieve without it. Small businesses receive an opportunity to compete with larger brands.

E. What Didn't Work So Well

Although we are rather strong, we discovered some weak aspects.

Prompt interpretation could be improved. The smarter understanding based on context, examples, and iteration can help make generation relevance better. It is still markedly clumsy to handle ambiguous descriptions, so the prompt engineering should continue to evolve.

The generated code quality is acceptable, but manual refinement options will be fantastic with the provision of code editing, style customization, and component replacement.

Deployment options need to be beefed up as well. It should add custom domain support and SSL certificates to make things look more professional.

Articulating the value of paid features would be more effective in order to enable users to understand what they are purchasing and have them upgrade.

5. Discussion

A. The Implications of This for Development of AI SaaS Platforms

Overall rules in building AI-powered platforms are depicted in this project.

Generation must strike the balance between power and simplicity—a large prompt area with example inputs gave it the impression that it can be used immediately.

Prompt engineering must be continuously tweaked with feedback data. The initial prompt structure is effective yet it can be further refined as per actual generations and user satisfaction.

Real-time preview features are really appreciated among users. An excellent preview interface provides immediate visual feedback so that one can see what is being generated easily.

Credit-based models can be very good—free credits allow users to test the platform, and then monetization works from there. We are able to avoid ruining the user experience yet still make money.

Finally, security is key. We should have straightforward authentication, secure payment processing, and clear data policies that will ensure user trust. Attempting to construct those governance structures is as significant as the cool new features.

B. Challenges to Keep in Mind

I went through the project and understood that some of the largest headaches that we would encounter are already highlighted to anyone interested in building something of the same nature. Plunging into it will present some discouraging bumps which are worth trying to observe.

AI API costs scale with platform growth. As more users generate websites, API costs increase. Infrastructure such as cost optimization and rate limiting has to be scaled with platform expansion.

Generation quality consistency is a challenge. To maintain high quality, we will have to implement prompt refinement, output validation, and fallback mechanisms. The greater the number of users we attract, the more effort we will incur in maintaining that quality.

Fraud prevention grows bigger along with the platforms and their increasing popularity. Compromised accounts, credit abuse, and fake generations are ongoing problems. Security measures should be adapted according to new threats.

Data privacy compliance varies across jurisdictions—data protection regulations, user content ownership, and platform liability. The legal scrutiny and possibly alternative strategies in different market descriptions might become a requirement.

Competitive differentiation needs constant innovation because more and more AI website builders have become eminent. Sustainable competitive advantage can be formed through distinctive features, generation quality, specializations for specific industries, or superior user experience.

C. Where We Could Go from Here

The analysis identified various modifications that will in fact enhance the attractiveness of our platform.

Drag-and-drop editor integration would give users post-generation refinement capabilities including visual editing, component rearrangement, and style customization. Imagine

that users will be able to tweak the generated design without regenerating the entire website. Being able to combine AI generation with visual refinement is a win for user control.

Better UI generation would also produce more sophisticated designs including animations, advanced interactions, and modern design patterns. Marketing users already cherish visual appeal. Providing users with generation options for different design styles (minimal, bold, elegant, playful) would allow them to match their brand identity.

Multi-language support would enable users to generate websites in their native language, expanding the platform's global reach. It would transform the platform from English-only to truly international, making it accessible to billions more users.

Team collaboration features—shared workspaces, role-based access, commenting—would increase platform stickiness and create enterprise value. It would be more of a collaborative platform than a single-user tool.

E-commerce generation would mean users are able to generate fully functional online stores with product listings, cart functionality, and payment integration, making it more valuable to businesses and enabling transaction revenue opportunities.

Machine learning improvements would be able to make generation more relevant, learn from user feedback, personalize recommendations, and continuously improve the platform through data-driven optimization.

6. Conclusion

This study included the design, implementation, and evaluation of a comprehensive AI-powered SaaS website builder platform that connected user ideas to functional websites using natural language processing, automated code generation, one-click deployment, and credit-based monetization. The platform has managed to illustrate the potential of creating value among users who want websites without technical barriers and entrepreneurs who want rapid online presence.

Overall, the project demonstrates that creating an intelligent and easy-to-use AI website builder is achievable and can actually assist users in launching their online presence quickly, which is essential in today's digital economy.

The major impact comes in the form of a modular architecture that enables AI-driven generation, responsive user interfaces that are accessible to all devices, systematic credit management, and practical advice to developers who build similar platforms. Findings have shown that AI-powered website builders provide a valuable opportunity to democratize web development, and the credit-based monetization model is sustainable as a long-term revenue generator, besides providing users with flexible access to generation capabilities.

This modular structure implies that we will be able to test both free and premium options, refine the UI on all devices, and build improvements based on actual user feedback. Moreover, we have demonstrated how AI generation remains profitable and increases accessibility.

The study contributes to the existing knowledge of AI-powered development platforms and offers practical insight into the creation of intelligent website generation applications. The platforms will have a greater role as digital entrepreneurship continues to digitize and as user expectations evolve over time, ensuring that such tools become essential drivers of economic activity and creative expression.

The expansion goals for future implementation of identified limitations will be achieved by using better prompt engineering, drag-and-drop refinement, mobile applications, team collaboration features, and machine learning integration—to continuously elevate the value of the platform for all explained stakeholders.

To the future, we will have to refine generation quality, add visual editing, introduce collaboration features, enhance analytics, and continue to integrate new AI capabilities to make the platform a better experience for everyone.

References

- [1] Anderson, C. (2009). *Free: The Future of a Radical Price*. Hyperion.
- [2] Chen, M., Tworek, J., Jun, H., et al. (2021). Evaluating Large Language Models Trained on Code. OpenAI Technical Report.
- [3] Evans, D. S., & Schmalensee, R. (2016). *The New Economics of Multisided Platforms: Matchmakers*. Harvard Business Review Press.
- [4] Fader, P. (2012). *Customer Centricity: Focus on the Right Customers for Strategic Advantage*. Wharton Digital Press.
- [5] Ghose, A. (2017). *Tap: Unlocking the Mobile Economy*. MIT Press.
- [6] Hagiu, A., & Wright, J. (2015). Multi-sided platforms. *International Journal of Industrial Organization*, 43, 162-174.
- [7] Kaplan, A. M., & Haenlein, M. (2010). Users of the world, unite! The challenges and opportunities of Social Media. *Business Horizons*, 53(1), 59-68.
- [8] Nielsen, J. (2012). *Mobile Usability*. New Riders.
- [9] OpenAI. (2023). *GPT-4 Technical Report*. OpenAI Research.
- [10] Osterwalder, A., & Pigneur, Y. (2010). *Business Model Generation*. John Wiley & Sons.
- [11] Parker, G. G., Van Alstyne, M. W., & Choudary, S. P. (2016). *Platform Revolution: How Networked Markets Are Transforming the Economy*. W. W. Norton & Company.

- [12] Resnick, P., & Zeckhauser, R. (2002). Trust among strangers in internet transactions: Empirical analysis of eBay's reputation system. *Economics of the Internet and E-commerce*, 11(2), 127-157.
- [13] Spool, J. M. (2009). *Web Site Usability: A Designer's Guide*. Morgan Kaufmann.
- [14] Tiwana, A. (2013). *Platform Ecosystems: Aligning Architecture, Governance, and Strategy*. Morgan Kaufmann.
- [15] Tullis, T., & Albert, B. (2013). *Measuring the User Experience: Collecting, Analyzing, and Presenting Usability Metrics*. Morgan Kaufmann.
- [16] Vaithilingam, P., Zhang, T., & Glassman, E. L. (2022). Expectation vs. Experience: Evaluating the Usability of Code Generation Tools Powered by Large Language Models. *CHI Conference on Human Factors in Computing Systems*.
- [17] Zuev, D. (2023). The Economics of AI Code Generation: Productivity, Quality, and Cost Implications. *Journal of Software Engineering Research*.