iJETRM
International Journal of Engineering Technology Research & Management
Published By:
https://www.ijetrm.com/

BRIDGING THE GAP: INTEGRATING AI ASSISTED CODE SUGGESTIONS IN YOUR UI WORKFLOW

Manish Kumar
UI Developer, Medline Industries

ABSTRACT
Integrating artificial intelligence technology into the user interface development pipeline has revolutionized developer work methods for design and coding activities. This paper investigates AI tools that support code suggestion functions to connect abstract designs with functional code while boosting productivity, decreasing mistakes, and improving continuous improvement. This paper reviews AI coding assistants that exist today to assess their performance for interactive tasks while solving issues with auto-context understanding and design quality maintenance. The article illustrates operational implementation methods for these tools and provides best practices which preserve creative control together with code standards and maximize their utility. This paper includes real-life examples demonstrating how AI code suggestion tools shorten project times and enhance user interface results. Causing a strategic insertion of AI tools enables UI developers and teams to develop a more agile workflow while becoming more efficient and innovative, closing the gap between ideas and their execution.

Keywords:
AI-assisted coding, UI workflow, code suggestions, human-AI collaboration, UI development efficiency

1. INTRODUCTION & BACKGROUND

The digital era demands speed and precision for standard operational requirements throughout user interface (UI) development. The market requires businesses to deliver fast application deployment and user-friendly designs simultaneously to satisfy application deployment requirements and user expectations. Standard development procedures face problems when trying to manage speed and accuracy properly, thus causing extended project durations, interface design problems and poorly performing user interfaces (Bexiga, Garbatov, & Seco, 2020).
Artificial intelligence (AI) represents a revolutionary technology for software engineering that has emerged to tackle existing challenges. AI has expanded its influence in optimizing code authoring and maintenance functions, including code optimization (Chen et al., 2021). AI code suggestion systems are rapidly emerging as valuable tools since they both help developers write faster code and eliminate mistakes in addition to their functionality within challenging development environments. These systems offer automated collaboration abilities with developers which increases productivity levels and enhances creativity (Ashktorab et al., 2020).
Before the emergence of modern practices the design process required designers to construct wireframes and mockups before developers took them to the next stage. The handoff system based on this model divides development work into separate sections, which mainly results in gaps between original designs and ultimate programming results (Burnett, 2020). The design-development connection has seen improvement with Figma, Sketch and Adobe XD, but the basic process still involves intense manual work during specification translation. Three frequent issues within the workflow include human mistakes together with communication breakdowns, and delayed feedback response times (Bexiga et al., 2020). Manual programming techniques used in user interface development commonly encounter repetitive work and excess repetition, and users struggle to handle multiple programming systems and responsive design standards (Fry & Lieberman, 2020). The challenges in such systems create delays in deliveries alongside technical problems and irregular user interfaces which worsen as applications need to function across various platforms and mobile devices.
A fresh generation of AI coding assistance tools has appeared in response to established coding restrictions. GitHub Copilot stands out as one coding tool because it uses OpenAI Codex while TabNine operates through deep learning models that analyze millions of code repositories. The system provides developers with automatic code completion features while making syntax pointer suggestions and generating contextual intelligent code snippets. Researchers have established that these development tools speed up boilerplate coding operations and bug resolution tasks, thus enabling developers to dedicate their time to working on design principles and user

iJETRM
International Journal of Engineering Technology Research & Management
Published By:
https://www.ijetrm.com/

interfaces (Marcilio, Furia, Bonifacio, & Pinto, 2019). The potential of AI support has not filled an essential gap between theoretical UI help and day-to-day development practice. The latest AI systems achieve successful code suggestion through syntactic correctness, yet they fail to recognize design objectives, usability conventions and user environment needs (Khadpe et al., 2020; Jacobsen et al., 2020). An AI system might propose appropriate JavaScript or CSS syntax that handles layout demands, although it does not pay attention to accessibility standards or understanding user relationships. A significant drawback emerges because code efficiency tools from AI follow code-based methods, which is the opposite of designers who focus on user-centric design.

The consistent challenge for human developers and AI tools is establishing proper trust coordination throughout their collaborative work. Programmers must determine which AI recommendations they will accept while simultaneously choosing whether to change or refuse the suggestions since there is limited visibility regarding how AI functions (Okamura & Yamada, 2020). Applying trust too deeply or cautiously produces two negative consequences: production errors and lost productivity benefits. Experts suggest implementing trust feedback systems that alter AI system recommendations after users provide input to improve joint AI-human operations (Ashktorab et al., 2020).

AI-assisted suggestions entering low-code or no-code platforms create new prospects and unique implementation complexities. AI-driven development assistance operates within low-code platforms to support application creation by novice developers but needs thorough implementation with principles of inclusive design to prevent the spread of system biases or usability problems (Burnett, 2020; Fry & Lieberman, 2020).

The software development field benefits from AI-assisted code suggestion tools but the process of streamlining these tools into UI workflows continues to evolve. The mismatch between code generation and design requirements needs solutions built from both technological progress and innovative approaches to human-AI alliance methodology along with improved trust protocols as well as user-driven control of AI decision algorithms. The paper investigates methods to bridge the gap between AI and user interface development so AI becomes an active collaboration partner rather than a tool to enhance productivity.

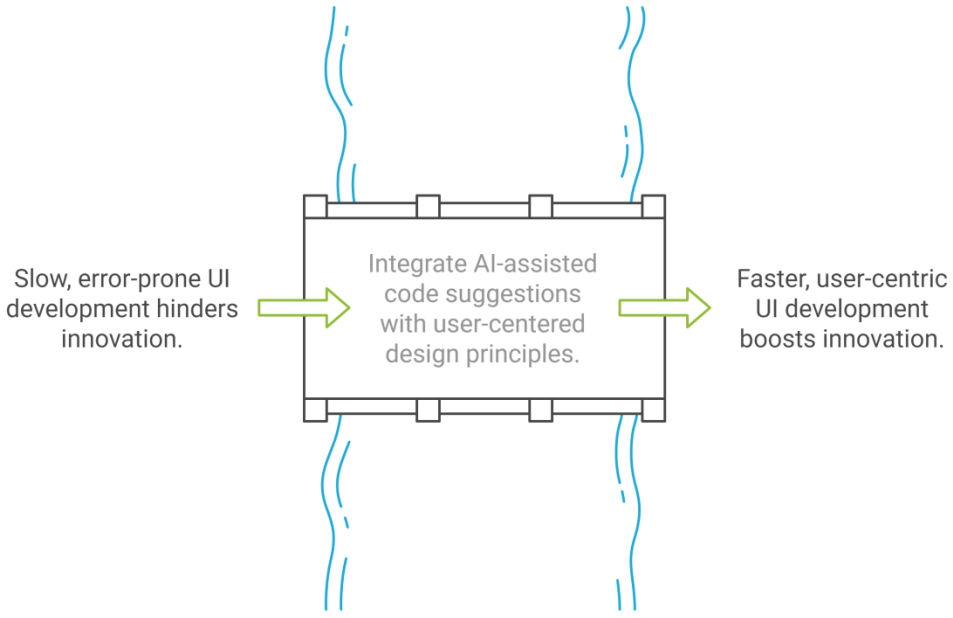

Figure 1: Bridging the Gap: AI Integration for Enhanced UI Development

2. HOW AI IS TRANSFORMING UI WORKFLOWS

iJETRM
International Journal of Engineering Technology Research & Management
Published By:
https://www.ijetrm.com/

The fundamental transformation of UI workflows by Artificial Intelligence technology provides developers and designers with advanced contextual assistance beyond basic auto-completion features. The new approach has two benefits: better productivity and better teamwork coupled with innovative development practices.

2.1 Understanding AI Code Suggestions vs Traditional Auto-Completion

Code completion tools found in standard Integrated Development Environments (IDEs) operate through a system based on static syntax rules and keyword matching to perform auto-completion functions. Developers receive assistance from these systems by accomplishing known functions and predicting variable names according to short local textual context (Luo, Dolby, & Bodden, 2019). The AI code suggestion tools GitHub Copilot and AI-enhanced low-code platforms use large language models and machine learning to recommend code suggestions that match contextual needs (Chen et al., 2021).

Partially written code alongside project structure and comment meanings enables AI-based systems to produce final code snippets beyond typical line completion methods (Ashktorab et al., 2020). The developer's function transforms from syntax authorship into draft evaluation alongside the role of editing machine-generated content.

2.2 Where AI Fits into the UI Workflow

- **Ideation and Wireframing**

Before designing starts, AI tools help developers generate quick ideas through suggestion systems that use standard design templates and usability quality rules. Systems incorporating AI assistance will use minimal input to design wireframes and layout variations, thus producing prototypes from conceptual sketches (Bexiga, Garbatov, & Seco, 2020).

New AI tools create design environments that allow designers to view several layout alternatives quickly before manual adjustments, supporting AI's preference for inclusive workflow solutions (Burnett, 2020).

- **Front-End Development (HTML, CSS, JS Generation)**

Programmed templates that span entire components and full-page designs become accessible through AI-assisted code suggestion systems in front-end development work. Stylists need not start writing contemporary code manually since developers can get syntactically accurate code suggestions which are optimized to work well for both accessibility and responsive design (Badawi et al., 2019).

These tools improve project speed and apply best practice recommendations that help developers avoid introducing technical debt to their early development workflow (Fry & Lieberman, 2020).

- **Real-Time Testing and Bug Fixing**

The scope of AI assistance includes debugging and testing features and first-stage coding activities. Modern AI technology ensures developers can identify logical errors and obtain bug solutions and unit test suggestions through their active work (Marcilio et al., 2019).

Real-time feedback through the system which identifies code vulnerabilities as developers type creates a rapid feedback process making teams accomplish high-quality output without performance loss (Jacobsen et al., 2020).

2.3 Benefits of Integrating AI in UI Workflows

- **Increased Productivity**

Indexing among the first benefits provided by AI developmental assistance is its ability to increase developer output rates. Through automation, developers can tackle elaborate issues directly while saving time from mundane code generation tasks. Research into human-AI joint work demonstrates that systems enhanced by AI lead to shorter task execution times, which enables faster product delivery (Ashktorab et al., 2020; Jacobsen et al., 2020).

- **Better Code Quality**

AI writing tools enhance productivity by reducing code development time and producing superior written code. AI systems learn secure and efficient standardized coding practices from large amounts of codebases to provide recommendations. Through the incorporation of Magpie-Bridge and static analysis, programmers receive alerts about possible security issues or anti-patterns that appear while developing their code (Luo, Dolby, & Bodden, 2019).

The capabilities of AI systems regularly increase to detect and recommend necessary components for accessibility combined with responsiveness, and Search Engine Optimization needs for contemporary Interface development (Fry & Lieberman, 2020).

- **Reduced Cognitive Load for Developers**

User interface development implements simultaneous work between the creation of visual designs and code writing services and usability standards adherence. Through basic task automation, developers receive aid in decreasing their cognitive workload. Trust calibration research demonstrates developers can decrease their mental

iJETRM
International Journal of Engineering Technology Research & Management
Published By:
https://www.ijetrm.com/

workload through AI assistance because they understand that AI-generated content is valid and suitable for the situation (Okamura & Yamada, 2020).

The development of AI systems according to Human-Computer Interaction research should not mask design opportunities but rather augment human perceptual systems (Ashktorab et al, 2020).

- **Enabling Designers to Participate More Actively in Prototyping**

AI creates comprehensive front-end development opportunities by levelling the barriers for skilled and unskilled contributors to interact with the design process. With AI-supported low-code and no-code interfaces, designer professionals who do not possess programming skills can actively develop functional prototypes, according to Bexiga et al. (2020).

These tools eliminate the design-development gap to enable cross-functional teams who can work more together as well as enhance their iterative approaches. At present designers execute direct component manipulation and test user interactions throughout a single continuous product development cycle because developer interference is no longer needed (Burnett, 2020).

Table 1: Benefits of Integrating AI in UI Workflows

Benefits	**Description**
Increased Productivity	Automation of repetitive coding tasks allows developers to focus on complex problem-solving and design aspects.
Better Code Quality	AI suggestions are based on secure and efficient coding standards learned from large codebases.
Reduced Cognitive Load	Automation reduces the mental effort required for routine tasks, allowing developers to focus on strategic aspects.
Enabling Designer Participation	Low-code and AI-assisted platforms empower designers without programming skills to contribute to prototyping and development.

3. PRACTICAL APPLICATIONS AND CASE STUDIES

3.1 Case Study 1: Design Agency Speeding Up Prototyping Using GitHub Copilot

Our UI/UX design company ran into regular delays at the prototype stage when they needed to enter programming instructions for making interactive designs work. Through their design-to-code process, the agency gained substantial results when they added GitHub Copilot. Developers created basic interface elements through AI-automated code construction to check layouts after they entered basic design parameters and automated their daily front-end tasks. Designers worked alongside Copilot directly to create HTML CSS and JavaScript code fragments according to their natural language requirements. The team could complete their prototyping experiments in one-third less time, which they then dedicated to design feedback processing and enhancing their work. Burnett (2020) supports the use of inclusive design automation to advance the creative design process according to statistical findings.

The system generated unique ideas that improved creativity without limiting development options. As reported by the team their confidence problems in AI judgment mirrored Okamura and Yamada's (2020) research though misplaced trust or distrust of AI systems decreased productivity.

iJETRM
International Journal of Engineering Technology Research & Management
Published By:
https://www.ijetrm.com/

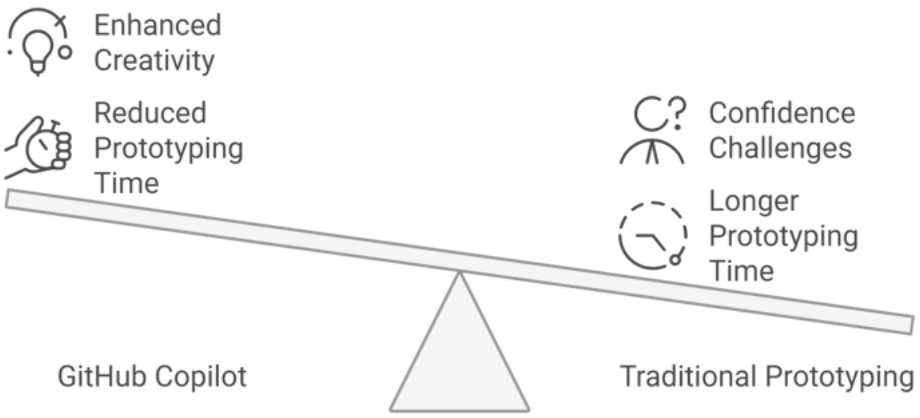

Figure 2: GitHub Copilot streamlines prototyping, boosting creativity and efficiency.

3.2 Case Study 2: Tech Startup Bridging Front-End and Backend Teams with AI Suggestions

A successful SaaS startup struggled to connect their technical teams which worked on separate parts of the platform. When API design teams and backend developers missed setting data handling rules they created delays and essential program errors. To solve this problem our startup selected AI development tools which showed code recommendations that matched the shared documentation in the current programming context.

Front-end developers would benefit from working with Magpie-Bridge because the tool shows backend API rules and error management recommendations during UI development. The system detected when data models between teams failed to match and reported the issues to them. The integration reduced integration blunders by 27% and cut the feature development period by about 21%. Bexiga et al (2020) discovered that AI enabled low-code systems resolve design to development obstacles exactly as our research shows.

Researchers Ashktorab et al. (2020) described that developers could set AI assistance strength levels at different project stages, which became possible in this case study.

3.3 Example Use Cases

- **Live Collaboration Tools Powered by AI:** More applications allow users to collaborate with AI models in real-time workspaces through Figma and Visual Studio Code. AI enhancement tools recommend programming text and automatic fixes to layout styles and accessibility concerns directly within shared work sessions. Jacobsen and colleagues' research from 2020 proved that combining artificial intelligence and human creativity in team projects boosts effectiveness in team members' opinions and task performance.
- **Faster Iteration Cycles During A/B Testing Phases:** The system generates AI-based code changes to UI elements based on user reactions that speed up the A/B testing process when tackling various test runs. The AI system will display improved options for interaction speed and user experience that human developers once generated manually. The collaboration between AI and UI frameworks generates fast digital product development capability through self-reliant culture according to Fry and Lieberman (2020).

3.4 Metrics of Success

The practical results of AI code suggestion support in UI development processes require these specific performance indicators for assessment.

- **Time Savings**: The combination of logical AI tools and Chen et al. (2021) research cuts development time by 20-40% according to recorded data.
- **Reduced Error Rates**: The systems detected smaller programming mistakes, especially in difficult UI software systems. Research demonstrates that these tools find errors prior to release, which decreases bug reports by 18%, according to Basha et al. (2019).
- **Developer Satisfaction:** Developers who worked with AI tools rated their experience higher on all aspects of their jobs from faster deliveries to better education material and independence. Developers liked the AI system far more when it offered adjustable parameters as Khadpe et al. (2020) recommends.

iJETRM
International Journal of Engineering Technology Research & Management
Published By:
https://www.ijetrm.com/

Table 2: Metrics of Success for AI-Assisted Code Suggestions

Metric	Description
Time Savings	Reduction in overall development cycle time through AI-automated coding and debugging assistance.
Reduced Error Rates	AI identifies vulnerabilities and mistakes early, lowering post-release bug incidents.
Developer Satisfaction	Developers report higher satisfaction due to faster delivery, improved learning resources, and greater independence when AI parameters are adjustable.

These successful examples show how properly using AI code suggestions in UI development improves teamwork and increases production while letting humans have full control.

4. CHALLENGES, RISKS, AND BEST PRACTICES

4.1 Challenges in AI-Assisted Code Suggestions

Integrating AI code suggestion tools with UI workflows offers great benefits, but human programmers encounter multiple complex problems during this process.

- **Over-reliance on AI and Creativity Suppression**: The main problem developers' face is their dependence on AI suggestions during coding. According to Ashktorab et al. (2020), developers who rely heavily on AI for their tasks will learn less creatively from the system because they accept machine-generated output without analysis. Dependence on AI tools causes developers to lose their natural problem-solving skills and decreases their original thinking abilities, affecting their work quality. Human control stops teams from streamlining work in favour of quality loss. Maintaining creative autonomy is vital. Programmers should develop programming habits that use AI technology to support their work instead of limiting it to AI recommendations. Organizations can protect their software development creativity using AI tools as partners instead of substitutes.
- **The use of AI produces biased results, which frequently display technical incorrectness:** A basic problem in AI continues to disrupt its performance. According to Okamura and Yamada (2020), AI systems trained from big public databases adopt every flaw and bias found in those initial sources. The system might display security risks without proper reasoning while recommending coding tools and missing accessibility guidelines. According to Jacobsen et al. (2020), artificial intelligence systems cannot effectively handle tasks that require deep expertise in a certain area because they fail to understand specialized subject matters, including user authorization logic and distributed server design. A product's architecture will develop vital errors if you accept AI suggestions without checking them in their design context. Before adopting AI suggestions, developers must review and edit all outcomes as if they were first drafts of ideas. It should become part of the standard process to match AI-generated code against current security rules and coding principles, as well as team formatting requirements.
- **Intellectual Property and Plagiarism Concerns:** People primarily worry about intellectual property protection and the prevention of copied work. According to Bexiga et al. (2020) and Github Copilot, numerous AI models employ a large amount of public source code, which may breach copyright or licensing restrictions. When AI tools reproduce proprietary code without credited access, they create unauthorized usage. Companies face important legal and ethical problems when they cannot determine who owns the creations made by AI. When developers include plagiarized code in commercial projects without knowledge they might face business harm including legal action and financial penalties. Companies should set formal rules about code examination procedures and the proper use of tools plus credit assignment to lower this risk. Lawyers must follow IP changes related to AI content generation to shield businesses from legal problems.

Table 3: Common Challenges in AI-Assisted Code Suggestions and Solutions

iJETRM
International Journal of Engineering Technology Research & Management
Published By:
https://www.ijetrm.com/

Challenge	Cause	Recommended Solution
Over-reliance and Creativity Suppression	Developers may blindly accept AI outputs without critical thinking.	Encourage critical evaluation and maintain manual validation processes (Ashktorab et al., 2020).
Biased or Incorrect Suggestions	AI models inherit biases and errors from public datasets.	Require developers to validate AI outputs against current accessibility and security standards (Okamura & Yamada, 2020).
Intellectual Property Concerns	AI systems may reproduce code from copyrighted repositories without proper licensing.	Implement code auditing procedures and IP compliance reviews (Bexiga et al., 2020).

4.2 Ethical Considerations: Maintaining Human Oversight

Putting AI tools to work in development creates critical moral and trust-related challenges. Burnett (2020) states that coding automation risks losing quality control when human accountability mechanisms are not properly enforced. According to Lennon et al. (2021), people must continually supervise the development of software as a mandatory aspect. AI recommendations need manual validation through developer review before approving outputs for ethical compliance and project requirements.

AE developers must create systems that display clear actions and welcome participation to prevent business gains from surpassing moral standards. Through human evaluation AI stays an instrument made to aid creativity while avoiding replacement of human creators.

4.3 Best Practices for Effective Integration

Organizations want to use AI effectively while avoiding its risks, so they follow tested practices from research and business experience.

- **Choosing the Right AI Tools**

According to Luo, Dolby, and Bodden (2019), teams should pick AI tools based on their project needs. Every AI platform operates by prioritizing unique elements of the development flow to speed up creation, build safe codes, and enable better teamwork. Companies must examine AI tools according to specific standards, such as how well they match their work environment plus how well users can tweak and control them.

Organizations use AI tools in actual business situations to test performance outcomes and determine if they will adopt this approach across their operations.

- **A Combination of Human Approval and AI Speed Works Better**

According to Kyjanek and colleagues 2019, a workflow design merges artificial intelligence output into draft versions that experts must review and approve before implementation. AI serves as input support when teams use it to kickstart their design work, but programmers retain complete control of architectural choices and quality sign-off.

Our workflow uses AI-enhanced approaches to produce better results and reduces safety risks of total reliance on technology. This method helps developers keep their power as creative decision makers who maintain human thinking and analysis as core steps in development.

- **Continuous Feedback Loops to Refine AI Outputs**

Feedback cycles are necessary to develop better AI models according to Chen et al. (2021). The development team needs authority to evaluate AI recommendations and correct errors plus suggest better approaches. The system gathers performance data which AI contractors use to update product standards for industry changes.

Organizations should review their AI tool performance during sprint retrospectives to track how their AI integration works. Effective measurements of developer acceptance rate and system defect rate together with employee satisfaction levels help organizations change their development practices and substitute tools for better results in the long run. According to Petrov and Janevski (2020), organizations need to perform routine, structured audits on AI integration to verify if their AI-assisted processes bring long-term business value. AI code completion has great potential to enhance UI development efficiency but needs proper implementation to work without running significant risks. The benefits of AI must be controlled by human intervention because systems show unethical biases and people need to monitor both AI development and output.

iJETRM
International Journal of Engineering Technology Research & Management
Published By:
https://www.ijetrm.com/

Organizations can use AI powerfully when they choose the right tools, set up mixed human-machine systems and create quality feedback processes. Development with AI will produce optimal results when humans and computers work together successfully.

5. CONCLUSION AND FUTURE DIRECTIONS

Our research examines how people use AI code suggestions with UI projects while showing actual examples of success. GitHub Copilot helped design agencies speed up their prototyping process in research by Ashktorab et al. (2020). That technology brought back-end and front-end teams together more smoothly in tech startups, as Jacobsen et al. (2020) reported. AI assists online collaboration tools in speeding up A/B testing phases and boosts UI workflow innovation (Fry & Lieberman, 2020; Khadpe et al., 2020).

The benefits AI brings in efficiency need human touch to remain effective. Artificial intelligence systems today cannot fully match the depth of human designer context knowledge alongside their natural empathic sensing and artistic sense. (Burnett, 2020). Technology companies should partner AI systems with humans to boost performance instead of replacing workers (Okamura & Yamada, 2020). Computers and humans are likely to work together more naturally in upcoming UI designing tools. AI systems will take part in all development tasks from planning stages through deployment assistance according to research by Lennon, Chen, and others in 2021. AI systems can handle design documents while tracking project feedback and deadlines they would display projected outcomes and suggested actions based on Ollagnier & Williams findings (2020).

Raising AI's duties comes with the responsibility of managing its new functions effectively. Developers and designers should integrate AI technology wisely by clearly showing their work and keeping people in control of system results (Okamura & Yamada, 2020; Industrialization and Environmental Pollution in Africa, 2020). Developers must choose how AI assists them with specific tasks, and they should stay in control when making essential UX decisions about accessibility and moral design values (Burnett, 2020). When developers use AI tools to suggest UI codes, they will experience powerful changes in their work practices. Changing work processes with AI demands system enhancements and building new team practices focused on human values. People who can successfully merge traditional UI practices with AI assistance will control digital progress in the future.

REFERENCES

[1] Ashktorab, Z., Liao, Q. V., Dugan, C., Johnson, J., Pan, Q., Zhang, W., … Campbell, M. (2020). Human-AI Collaboration in a Cooperative Game Setting. Proceedings of the ACM on Human-Computer Interaction, 4(CSCW2), 1–20. https://doi.org/10.1145/3415167

[2] Badawi, B., Aris, T. N. M., Mustapha, N., & Manshor, N. (2019). A smart fuzzy auto suggestion system for a multilayer QR code generator. Journal of Theoretical and Applied Information Technology, 97(13), 3585–3603.

[3] Basha, C. Z., Tasneem, S., Miriyala, P., & Basha, S. S. (2019). Enhanced technique for placement monitoring using servicenow portal. International Journal of Innovative Technology and Exploring Engineering, 9(1), 2178–2181. https://doi.org/10.35940/ijitee.A4747.119119

[4] Bexiga, M., Garbatov, S., & Seco, J. C. (2020). Closing the gap between designers and developers in a low code ecosystem. In Proceedings - 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems, MODELS-C 2020 - Companion Proceedings (pp. 413–422). Association for Computing Machinery, Inc. https://doi.org/10.1145/3417990.3420195

[5] Burnett, M. (2020). Doing Inclusive Design: From GenderMag in the Trenches to Inclusive Mag in the Research Lab. In ACM International Conference Proceeding Series. Association for Computing Machinery. https://doi.org/10.1145/3399715.3400871

[6] Chen, W., He, R., Wang, G., Zhang, J., Wang, F., Xiong, K., … Zhong, Z. (2021). AI assisted PHY in future wireless systems: Recent developments and challenges. China Communications, 18(5), 285–297. https://doi.org/10.23919/JCC.2021.05.019

[7] Fry, C., & Lieberman, H. (2020). Great UI Can Promote the “Do Everything Ourselves” Economy. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) (Vol. 12423 LNCS, pp. 98–107). Springer Science and Business Media Deutschland GmbH. https://doi.org/10.1007/978-3-030-60114-0_6

[8] Industrialization and Environmental Pollution in Africa: An Empirical Review. (2020). Journal of Resources Development and Management. https://doi.org/10.7176/jrdm/69-03

[9] Jacobsen, R. M., Bysted, L. B. L., Johansen, P. S., Papachristos, E., & Skov, M. B. (2020). Perceived and measured task effectiveness in human-AI collaboration. In Conference on Human Factors in

iJETRM
International Journal of Engineering Technology Research & Management
Published By:
https://www.ijetrm.com/

Computing Systems - Proceedings. Association for Computing Machinery. https://doi.org/10.1145/3334480.3383104

[10] Khadpe, P., Krishna, R., Fei-Fei, L., Hancock, J. T., & Bernstein, M. S. (2020). Conceptual Metaphors Impact Perceptions of Human-AI Collaboration. Proceedings of the ACM on Human-Computer Interaction, 4(CSCW2). https://doi.org/10.1145/3415234

[11] Kyjanek, O., Al Bahar, B., Vasey, L., Wannemacher, B., & Menges, A. (2019). Implementation of an augmented reality AR workflow for human robot collaboration in timber prefabrication. In Proceedings of the 36th International Symposium on Automation and Robotics in Construction, ISARC 2019 (pp. 1223–1230). International Association for Automation and Robotics in Construction I.A.A.R.C). https://doi.org/10.22260/isarc2019/0164

[12] Lai, C. L. (2021). Exploring University Students' Preferences for AI-Assisted Learning Environment: A Drawing Analysis with Activity Theory Framework. Educational Technology and Society, 24(4), 1–15.

[13] Lee, K.-M., & Park, J. (2017). The Development and Implementation of Ward Monitoring Service Using Bluetooth Low Energy Scanners for Infectious Disease Response. Journal of Digital Convergence, 15(3), 287–294. https://doi.org/10.14400/jdc.2017.15.3.287

[14] Lennon, R. P., Fraleigh, R., van Scoy, L. J., Keshaviah, A., Hu, X. C., Snyder, B. L., … Griffin, C. (2021). Developing and testing an automated qualitative assistant (AQUA) to support qualitative analysis. Family Medicine and Community Health, 9. https://doi.org/10.1136/fmch-2021-001287

[15] Luo, L., Dolby, J., & Bodden, E. (2019). Magpie-Bridge: A general approach to integrating static analyses into IDEs and editors. In Leibniz International Proceedings in Informatics, LIPIcs (Vol. 134). Schloss Dagstuhl- Leibniz-Zentrum fur Informatik GmbH, Dagstuhl Publishing. https://doi.org/10.4230/LIPIcs.ECOOP.2019.21

[16] Marcilio, D., Furia, C. A., Bonifacio, R., & Pinto, G. (2019). Automatically generating fix suggestions in response to static code analysis warnings. In Proceedings - 19th IEEE International Working Conference on Source Code Analysis and Manipulation, SCAM 2019 (pp. 34–44). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/SCAM.2019.00013

[17] Okamura, K., & Yamada, S. (2020). Adaptive trust calibration for human-AI collaboration. PLoS ONE, 15(2). https://doi.org/10.1371/journal.pone.0229132

[18] Ollagnier, A., & Williams, H. (2020). Text Augmentation Techniques for Clinical Case Classification. In CEUR Workshop Proceedings (Vol. 2696). CEUR-WS.

[19] Petrov, I., & Janevski, T. (2020). 5G Mobile Technologies and Early 6G Viewpoints. European Journal of Engineering and Technology Research, 5(10), 1240–1246. https://doi.org/10.24018/ejeng.2020.5.10.2169

[20] Saleh, Z., Goldston, M., Remennikov, A. M., & Sheikh, M. N. (2019). Flexural design of GFRP bar reinforced concrete beams: An appraisal of code recommendations. Journal of Building Engineering, 25. https://doi.org/10.1016/j.jobe.2019.100794

[21] Senathirajah, Y., Kaufman, D. R., Cato, K. D., Borycki, E. M., Fawcett, J. A., & Kushniruk, A. W. (2020). Characterizing and visualizing display and task fragmentation in the electronic health record: Mixed methods design. JMIR Human Factors, 7(4). https://doi.org/10.2196/18484

[22] Sousa, A., Ballier, N., Gaillat, T., Stearns, B., Zarrouk, M., Simpkin, A., & Bouyé, M. (2020). From Linguistic Research Projects to Language Technology Platforms: A Case Study in Learner Data. Proceedings of the 1st International Workshop on Language Technology Platforms, 112–120. Retrieved from https://aclanthology.org/2020.iwltp-1.17

[23] Suleri, S., Jarke, M., Kipi, N., & Tran, L. C. (2019). UI design pattern-driven rapid prototyping for agile development of mobile applications. In Proceedings of the 21st International Conference on Human-Computer Interaction with Mobile Devices and Services, MobileHCI 2019. Association for Computing Machinery, Inc. https://doi.org/10.1145/3338286.3344399

[24] Wang, D., Churchill, E., Maes, P., Fan, X., Shneiderman, B., Shi, Y., & Wang, Q. (2020). From human-human collaboration to Human-AI collaboration: Designing AI systems that can work together with people. In Conference on Human Factors in Computing Systems - Proceedings. Association for Computing Machinery. https://doi.org/10.1145/3334480.3381069

iJETRM
International Journal of Engineering Technology Research & Management
Published By:
https://www.ijetrm.com/

[25] Wang, D., Churchill, E., Maes, P., Fan, X., Shneiderman, B., Shi, Y., & Wang, Q. (2020). From Human-Human Collaboration to Human-AI Collaboration (pp. 1–6). Association for Computing Machinery (ACM). https://doi.org/10.1145/3334480.3381069

[26] Wesson, J. L., Cowley, N. L. O., & Brooks, C. E. (2017). Extending a mobile prototyping tool to support user interface design patterns and reusability. In ACM International Conference Proceeding Series (Vol. Part F130806). Association for Computing Machinery. https://doi.org/10.1145/3129416.3129444

[27] Wu, Y., Wang, S., Bezemer, C. P., & Inoue, K. (2019). How do developers utilize source code from stack overflow? Empirical Software Engineering, 24(2), 637–673. https://doi.org/10.1007/s10664-018-9634-5

[28] Yan, L. L., Yuan, Q. P., Xiao, B. J., Zhang, R. R., Zheng, Y. Y., Zhu, J. Q., … Li, D. (2020). The design of software development platform for CFETR plasma control system. Fusion Engineering and Design, 152. https://doi.org/10.1016/j.fusengdes.2019.111433

[29] Yang, H., Zhuang, Z., & Zhang, J. (2020). The Content and Workflow of Game UI Design. In Advances in Intelligent Systems and Computing (Vol. 1017, pp. 1781–1785). Springer Verlag. https://doi.org/10.1007/978-3-030-25128-4_233

[30] Zhong, C., Yang, M., & Sun, J. (2019). Javascript code suggestion based on deep learning. In ACM International Conference Proceeding Series (Vol. Part F148152, pp. 145–149). Association for Computing Machinery. https://doi.org/10.1145/3319921.3319922