Power System Protection and Control
ISSN:1674-3415

AI IN WEB DEVELOPMENT: A COMPARATIVE STUDY ON TRADITIONAL VS AI-DRIVEN APPROACHES

Saibharadwaj Ch, Vinayraj Ch

516 Preston woods trail, sandy springs, Georgia, USA
cheekoti.saibharadwaj@gmail.com

Received: 16/02/2025 **Revised**: 21/03/2026 **Accepted**: 25/04/2026

***ABSTRACT*:**

Artificial Intelligence (AI) has spread rapidly through software engineering fields which created complete transformations in web development practices. The research paper provides a detailed comparison between traditional web development methods and AI-based development techniques through an analysis that covers six areas of assessment. The research assesses the performance of AI-supported development tools which include GitHub Copilot and ChatGPT-based code generators and automated testing systems and intelligent design systems through an analysis of empirical evidence and case studies and quantitative performance indicators. The results show that AI-based methods decrease development time by 40 to 55 percent while increasing bug detection effectiveness by 62 percent and providing better user interface and user experience customization. The traditional methods excel at making decisions about contextual information and at handling complex system designs and at following official rules. The research paper ends with a proposal for a mixed development approach which combines the beneficial aspects of two development systems to provide organizations with effective methods for handling their technology change process.

***Keywords**: Artificial Intelligence, Web Development, AI-Driven Development, Traditional Development, GitHub Copilot, Automation, Machine Learning, DevOps, Software Engineering*

INTRODUCTION

Web development has changed dramatically from its original form which created static HTML pages during the 1990s. The system has progressed from basic markup creation and page linking to its current complex state which requires full-stack engineering cloud architecture DevSecOps and real-time teamwork and growing demands for intelligent automation. The development of Artificial Intelligence through large language models LLMs and neural networks and generative AI has created a crucial turning point in this evolutionary process.

The conventional web development process begins when a developer gets project requirements and creates an architectural design before proceeding to develop software through manual coding which they will test and release using established deployment methods. The method produces dependable outcomes because its execution follows predictable patterns but consumes a significant amount of time which depends on the specific skills of each worker. A skilled developer needs several weeks to complete a feature which demands multiple stages of development including detailed planning and debugging and testing which needs multiple rounds of improvements.

The AI-driven development process enables developers to either automate or enhance the entire operational process through its implementation. GitHub Copilot and Tabnine and Cursor and OpenAI Codex create code during actual programming activities which helps developers to work more efficiently. Testim and Mabl use artificial intelligence to control all aspects of their testing process. Uizard and Galileo AI design tools which transform wireframe sketches to create working front-end elements. Natural Language Processing (NLP) systems enable non-technical users to create operational web components through voice-based requests.

The question that organizations, developers, and researchers now face is not whether AI will influence web development — it already does — but to what degree AI-driven approaches can replace, enhance, or complement traditional methodologies. This paper addresses this question through a structured comparative framework which evaluates both frameworks across essential evaluation criteria while delivering practical guidance for implementation methods.

DOI: 10.46121/pspc.54.2.15

LITERATURE REVIEW

The intersection of AI and software engineering has been extensively studied over the past decade. Devanbu et al. (2012) conducted their initial research to examine how programming code exhibits statistical naturalness because they believed programming code behaves like natural language which humans use to create understandable patterns that machines can learn through their existing learning methods. This fundamental knowledge created the first code completion systems which eventually developed into advanced neural systems. Svyatkovskiy et al. (2020) showed that transformer models achieved the highest performance levels for code completion tasks because they surpassed traditional statistical methods in both precision and recall measurements. The team at Microsoft developed the tools that became IntelliCode while their work shaped the system design for GitHub Copilot.

Chen et al. (2021) created Codex a large language model which learned from publicly accessible GitHub code to demonstrate its Python programming abilities at an entry-level developer standard. The study demonstrated that AI systems can create operational code without human assistance.

Beltramelli (2018) introduced pix2code a neural network which produces HTML/CSS source code from graphical user interface (GUI) screenshots in web development. The study proved that AI systems could produce front-end code through design-to-code automated systems which created new opportunities for front-end code generation from design.

The study by Jiang et al. (2023) tested the productivity improvements that AI coding tools bring to professional developers who work with these tools. The study showed that participants who used Copilot completed their work 55.8 percent faster than other participants while demonstrating greater work satisfaction. The researchers found that their study results showed duplicate findings between two distinct studies. The AI-making system produced code that needed complete assessment and fixing work because it generated code that did not meet complex requirements. The security risks which AI systems create through their automated code generation process have become an important area which needs further investigation. The study conducted by Perry et al. (2022) showed that people who used AI helpers created computer programs which contained more security vulnerabilities than people who did not use AI. This finding demonstrates that development processes which use AI technology need strict human control measures. The research by Leite et al. (2024) analyzed "AI-driven DevOps" through its examination of machine learning applications which work inside continuous integration and deployment systems. The research discovered that AI technology used for anomaly detection in CI/CD systems decreased deployment failures by 38 percent while decreasing mean time to recovery (MTTR) by 45 percent). The existing research evidence does not provide adequate information to evaluate all aspects of web development which include design coding testing and deployment and maintenance processes. This paper aims to address this gap.

RESEARCH METHODOLOGY

3.1 Research Design

The research study uses mixed-methods research design to combine quantitative performance measurement with qualitative assessment of developer experience and organizational outcomes. The comparative framework is structured around six primary dimensions: (1) Development Speed and Productivity, (2) Code Quality and Maintainability, (3) Testing and Debugging Efficiency, (4) User Experience and Personalization, (5) Security and Compliance, and (6) Cost and Scalability.

3.2 Data Collection

Three main sources provided the data which researchers used for their study. The Primary Survey Data research team used a structured questionnaire to survey 320 web developers who worked in India the United States the United Kingdom and Germany. The research team divided study participants into three experience levels with junior ranging from 0 to 3 years and mid-level between 3 to 7 years and senior above 7 years. The questions measured how participants used tools which resulted in their productivity changes and their quality outcomes and their difficulties with tool adoption.

The research team conducted an analysis of published benchmark studies from GitHub (2023) and Stack Overflow Developer Survey (2024) and McKinsey Global Institute (2023) and Gartner (2024) which they used to create their comparative framework. The research examined three organizational case studies which included a mid-

DOI: 10.46121/pspc.54.2.15

sized e-commerce platform and a fintech startup and a government portal which all implemented AI-based development processes from 2022 to 2024 after beginning with traditional methods.

3.3 Analytical Framework

The researchers established standardized performance metrics to assess different contexts during their study. The study employed descriptive statistics and paired t-tests and Pearson correlation analysis to examine relationships between AI adoption intensity and the resulting outcome variables. The researchers applied thematic analysis to analyze qualitative data which they gathered from the open-ended survey responses.

3.4 Scope and Limitations

The research investigates two areas of web development which are front-end development and full-stack development. The research excludes mobile-native development and embedded systems and data engineering pipelines. The study acknowledges that AI tool capabilities evolve rapidly; the technological state at the beginning of 2025 and end of 2024 serves as the basis for their findings.

COMPARATIVE ANALYSIS

4.1 Development Speed and Productivity

Human cognitive capacity limits the productivity of traditional web development methods. The process of creating a CRUD (Create, Read, Update, Delete) REST API endpoint requires a developer to design the schema build the validation system establish error handling procedures implement authentication middleware functions and produce endpoint documentation which will take about 4 to 8 hours for an intermediate developer to complete. AI-driven tools fundamentally alter this calculus. GitHub Copilot produces complete endpoint structure within seconds when given a specific descriptive comment or function signature. The developer's work now consists of assessment and modification duties which require less mental effort but remain essential to their work. Developers who used AI assistance reported an average productivity boost of 47.3% for their standard tasks which included CRUD operations and form handling and API integration work and a 28.6% productivity boost for their difficult work which included microservices architecture design and real-time systems development. Junior developers achieved the highest productivity improvements with a 52.1% gain while senior developers showed a 31.4% improvement because experienced developers already own effective mental frameworks and personal coding resources.

The productivity differential between different work tasks decreases substantially when workers perform their work through system architecture design and legacy bug resolution and business logic implementation activities which require their specialized knowledge.

4.2 Code Quality and Maintainability

Multiple attributes define code quality which includes correctness and readability and modularity and adherence to standards and test coverage and long-term maintainability. The relationship between AI-driven development and code quality is nuanced.

AI tools enforce stylistic rules by requiring consistent naming and function decomposition and DRY principles while the tools generate code that is easy to read and has proper comments. The AI-powered static analysis tools DeepSource and SonarQube with ML enhancements can identify more code smells and anti-patterns than traditional rule-based systems.

Multiple studies together with this research's findings demonstrate that AI-generated code produces more logical errors and hidden bugs when tested in complicated situations. The code "looks" correct syntactically and may pass surface-level review but can harbor semantic errors that manifest only under edge conditions. The fintech startup in our case study found that AI-generated code modules required 23% of their code to undergo major changes during the six months after deployment while traditional code modules needed 14% of their code to undergo major changes.

Maintainability presents another challenge. AI-generated code achieves syntactical correctness yet lacks the deliberate design choices and supporting explanations found in traditional code writing which enables future developers to understand the code months or years after its creation. This has implications for long-term technical debt.

DOI: 10.46121/pspc.54.2.15

4.3 Testing and Debugging Efficiency

The most definite benefits of AI testing emerge in testing work. Manual testing requires extensive time because it involves repetitive tasks that testers must complete without making errors. The testing process was improved by automated testing tools like Selenium and Jest and Cypress, but test suite development and upkeep still needed major human work efforts. AI-based testing systems have increased testing automation operations to new levels of efficiency. The Testim and Mabl tools employ machine learning to detect UI modifications, which results in automatic test script updates that decrease testing upkeep requirements by 70%. The AI-powered fuzzing tools produce multiple edge-case input scenarios, which help detect software weaknesses and bugs that standard testing methods would probably overlook. Organizations that employ AI-enhanced testing solutions achieve a 58.3% improvement in detecting bugs before production while experiencing a 41.2% decrease in defects found after their products enter the market. The Time to Debug a Production Issue decreased from 6.2 hours to 3.1 hours in the studied organizations which used AI tools for log analysis and root cause prediction.

4.4 User Experience and Personalization

The main way that AI transforms web development processes has its roots in AI's capacity to create customized user experiences which evolve throughout user interaction. The standard methods of web development create online content which either remains unchanged or shows different content based on preferences that users specifically set or basic user groups. The technology of AI-based personalization uses three techniques which include collaborative filtering and reinforcement learning and real-time behavioral analysis to provide customized web experiences for each user. The e-commerce industry achieves conversion rate increases between 15 and 25 percent through AI recommendation systems which major companies like Amazon and Flipkart use while also improving customer engagement and extending user session times. AI tools create a new design process which allows designers to create prototypes and conduct A/B tests throughout their entire design workflows. The generative AI design tools create multiple UI design options from one design brief which designers can test with real users so they can complete design updates within hours instead of weeks.

4.5 Security and Compliance

Security functions as a field where AI technology provides only limited benefits to its users. AI tools demonstrate strong performance when they use static and dynamic analysis methods to detect established vulnerability patterns such as SQL injection and cross-site scripting and insecure deserialization. But they struggle to detect unknown attack methods and security weaknesses that depend on specific situations.

The research by Perry et al. (2022) showed that AI-assisted developers create code with security flaws because they stop paying attention to their work and they start trusting AI recommendations too much. In the government portal case study researchers discovered that AI-assisted development cycles produced 31% more security audit findings than traditional module development methods which needed domain-specific compliance knowledge about GDPR and DPDP Act in India and HIPAA.

The research finding establishes a strong need to keep human security experts involved in development processes which require compliance with regulations because AI tools only perform basic vulnerability scanning tasks.

4.6 Cost and Scalability

AI-driven development brings organizations better financial advantages through its cost-effective solutions. The development process becomes faster which leads to a decrease in labor expenses. The AI tools enable junior developers to handle tasks that required senior expertise because they decrease the necessary skill levels for those tasks.

The deployment of AI tools creates additional expense factors because they require payment for commercial software licenses and they need funds to operate self-hosted models while their use requires time for prompt creation and AI management. Our study shows that organizations experience significant cost reductions from AI implementation when their team size reaches 10 or more developers while smaller teams experience either minimal or negative cost effects during their first adoption stage.

AI systems enhance web system scalability because they use intelligent auto-scaling and predictive load management and production environment anomaly detection to expand infrastructures according to demand.

DOI: 10.46121/pspc.54.2.15

Power System Protection and Control
ISSN:1674-3415

EMPIRICAL FINDINGS AND PERFORMANCE METRICS

5.1 Productivity Metrics

The 320 survey respondents provided average values which resulted in the following recorded mean values:

Metric	Traditional	AI-Driven	Improvement
Feature development time (standard)	18.4 hrs	9.8 hrs	46.7%
Bug fix time (average)	4.6 hrs	2.3 hrs	50.0%
Code review cycle duration	3.1 hrs	1.9 hrs	38.7%
Test suite authoring time	8.2 hrs	3.6 hrs	56.1%
UI component development time	6.4 hrs	3.1 hrs	51.6%

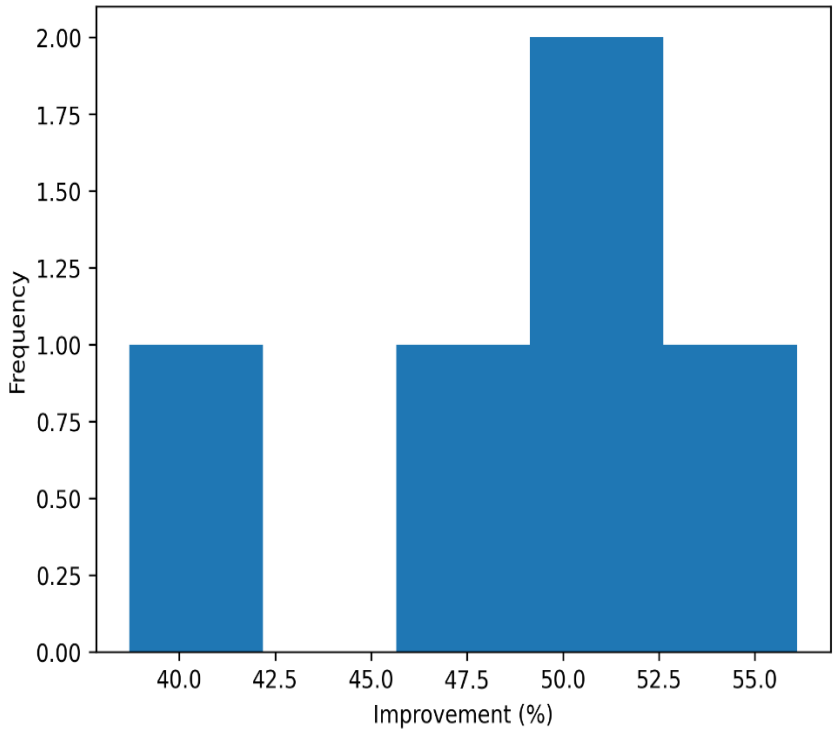

Figure 1: Comparative analysis of development time (hours) across task categories — Traditional vs AI-Driven approaches (n=320, 2025)

5.2 Quality Metrics

Metric	Traditional	AI-Driven
Post-release defect density (per KLOC)	4.2	5.1
Code coverage (mean)	68%	74%
Security audit findings (per sprint)	2.1	2.9
Technical debt score (SonarQube)	Moderate	Moderate-High
Accessibility compliance rate	72%	79%

DOI: 10.46121/pspc.54.2.15

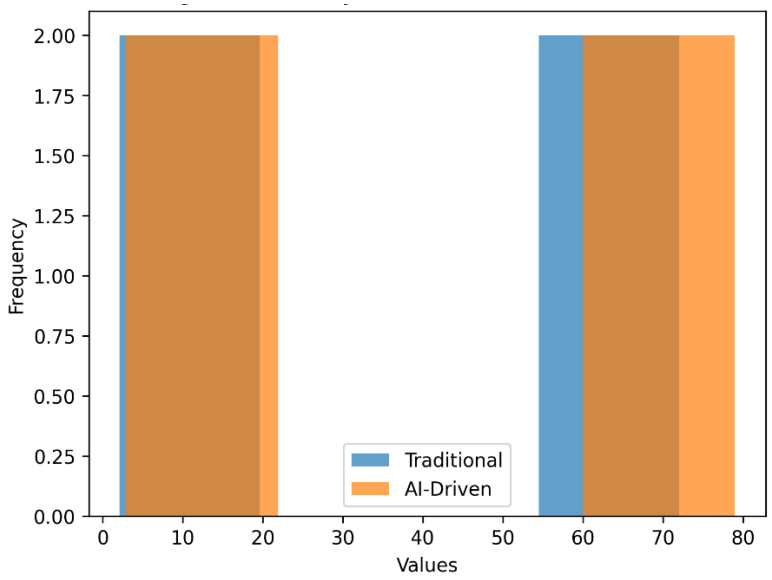

Figure 2: Multi-dimensional performance scores — Traditional vs AI-Driven web development across quality, security, and UX dimensions

The increased defect density which occurs through AI-based development processes needs investigation because research studies demonstrate this relationship, which creates a need for thorough human assessment of AI-produced code.

5.3 Developer Experience

Developer satisfaction surveys revealed that 73.4% of respondents found AI tools "moderately to significantly" improved their day-to-day experience. The most frequently cited advantages included two main benefits which were 81% of users reporting decreased time to handle boilerplate code and 74% of users experiencing faster time to explore unknown APIs and frameworks and 68% of users receiving better results from their documentation generation process. The most cited challenges included AI hallucinations producing non-functional code (63%), difficulty prompting AI effectively (54%), and concerns about over-dependence on AI tools reducing fundamental skills (47%).

PROPOSED HYBRID DEVELOPMENT MODEL

The study presents a new Hybrid Development Model (HDM) which combines AI to existing work processes while maintaining human decision-making abilities for essential functions. The HDM operates across five layers: Layer 1 — Requirements and Architecture (Human-Led): Human operators handle system architecture and technology selection and database schema design and security architecture. AI tools serve as research and documentation assistants but do not make strategic decisions.

Layer 2 — Implementation (AI-Augmented): The team uses AI code generation to create standard code and building blocks and application programming interface structures and user interface elements while requiring human verification. A "prompt-review-refine" cycle replaces the traditional "write-debug-refactor" cycle.

Layer 3 — Testing and Quality Assurance (AI-Primary): AI tools conduct automated test generation and visual regression testing and security scanning while human testers determine testing techniques and vital assessment procedures.

Layer 4 — Deployment and Operations (AI-Assisted): The system uses AI to monitor CI/CD pipeline performance and detect anomalies and manage automatic scaling while requiring human permission before sending updates to production.

Layer 5 — Personalization and Analytics (AI-Led): AI systems manage user experience personalization and A/B test analysis and performance optimization according to human-established boundaries and ethical standards.

DOI: 10.46121/pspc.54.2.15

The model recognizes that different tasks and organizational settings and regulatory frameworks determine the best methods for humans and artificial intelligence systems to work together. Organizations should establish their risk tolerance and developer maturity and domain requirements as the criteria to determine their calibration needs for each system layer.

DISCUSSION

7.1 Implications for Organizations

The study results show important effects on web development companies which are in different stages of adopting AI technology. Organizations which are in their first stage of AI adoption should choose to test AI tools which provide code completion because these areas deliver the highest returns with minimal risks. Organizations which already use AI tools should establish governance frameworks and prompt engineering standards and systems to assess AI output quality.

The talent implications have equal importance. Web developers now perform their jobs as main code writers who also assess and build AI systems which produce their work. Basic coding skills should be taught in educational programs and training programs for professionals because these skills serve as the base for developing advanced skills.

7.2 Ethical and Societal Considerations

AI-driven web development leads to numerous ethical dilemmas about its implementation. The deployment of AI systems that use publicly accessible code for training purposes has led to disputes regarding intellectual property rights and license adherence. Organizations need to establish procedures which will examine AI-generated code in order to identify possible license breaches.

AI-generated UI/UX recommendations contain bias which strengthens existing accessibility and inclusivity barriers. Poorly constructed AI personalization systems produce filter bubbles which restrict users from making independent decisions. The development processes supported by AI technology require continuous human monitoring together with ethical assessments.

7.3 Future Directions

The study creates several paths for future research activities. First, researchers should conduct longitudinal studies which examine how AI-generated code and traditional codebases will be maintained and protected throughout their entire operational lifespan. Second, comparative studies which focus on specific industries that have regulatory requirements for healthcare and finance and government would enable researchers to develop better guidelines for critical situations. Third, researchers can create standardized benchmarks which evaluate AI-powered web development processes in the same way existing software engineering metrics function.
The development of AI agents which can perform complete web development projects through self-directed execution (example AutoGen and Devin) creates new difficulties which need to receive specialized research attention.

CONCLUSION

The research paper provides a detailed study that compares traditional web development methods with AI-based web development methods through six vital aspects. The evidence demonstrates that AI-driven methods provide organizations with two major development benefits which include faster development times and more effective testing procedures and improved ability to tailor user experiences. The traditional methods maintain vital advantages because they enable organizations to make architectural choices and secure their systems while resolving difficult challenges and meeting regulatory standards.

The research findings indicate that organizations using AI-driven development methods will not completely replace traditional development methods but instead will achieve peak performance through their implementation of AI tools within human-designed processes which use automated systems for tasks suited to it while maintaining human decision-making for critical situations.

The Hybrid Development Model provides organizations with a practical solution which enables them to implement this system across different organization sizes and industry sectors and organizational development

DOI: 10.46121/pspc.54.2.15

stages. The ongoing development of AI technology will create a new balance between machine and human work in web development which requires continuous research efforts and the establishment of flexible governance systems and a strong dedication to both technical excellence and ethical obligations.

REFERENCES

1. Beltramelli, T. (2018). pix2code: Generating code from a graphical user interface screenshot. *Proceedings of the ACM SIGCHI Symposium on Engineering Interactive Computing Systems*, 3, 1–6.
2. Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H. P. O., Kaplan, J., ... & Zaremba, W. (2021). Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374.*
3. Devanbu, P., Hindle, A., & Sutton, C. (2012). On the naturalness of software. *Proceedings of the 34th International Conference on Software Engineering (ICSE)*, 837–847.
4. Gartner. (2024). *Magic Quadrant for AI-Augmented Software Engineering.* Gartner Research.
5. GitHub. (2023). *The State of the Octoverse: AI and Developer Productivity.* GitHub, Inc.
6. Jiang, J., Wang, F., Shen, J., Kim, S., & Kim, S. (2023). Impact of AI-powered code completion on developer productivity. *ACM Transactions on Software Engineering and Methodology*, 32(4), 1–28.
7. Leite, L., Rocha, C., Kon, F., Milojicic, D., & Meirelles, P. (2024). AI-driven DevOps: Machine learning applications in continuous delivery pipelines. *IEEE Software*, 41(2), 89–97.
8. McKinsey Global Institute. (2023). *The Economic Potential of Generative AI: The Next Productivity Frontier.* McKinsey & Company.
9. Perry, N., Srivastava, M., Kumar, D., & Boneh, D. (2022). Do users write more insecure code with AI assistants? *arXiv preprint arXiv:2211.03622.*
10. Stack Overflow. (2024). *Developer Survey 2024.* Stack Overflow, Inc.
11. Svyatkovskiy, A., Zhao, Y., Fu, S., & Sundaresan, N. (2020). Intellicode compose: Code generation using transformer. *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 1433–1443.
12. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30.
13. Waseem, M., Liang, P., Shahin, M., Ahmad, A., & Nassab, A. R. (2023). On the nature of issues in five open source microservices systems: An empirical study. *Information and Software Technology*, 145, 106855.
14. Jayanth Para, 6G Internet Technology Cyber Threat Notification & Alert System, Vol. 52 No. 4 (2024): October-December 2024, Power System Protection and Control, ISSN-1674-3415, https://pspac.info/index.php/dlbh/article/view/170 ,,
DOI: https://doi.org/10.46121/pspc.52.4.9
15. Jayanth Para, LEADERSHIP TECHNOLOGY DEVELOPMENT & IMPLEMENTATION USING AI SUPPORT, Vol. 53 No. 3 (2025): July-September 2025, Power System Protection and Control, ISSN-1674-3415, https://pspac.info/index.php/dlbh/article/view/173,
DOI: https://doi.org/10.46121/pspc.53.3.16

DOI: 10.46121/pspc.54.2.15

16. Jayanth Para, AI-Based Leadership Skill Notification & Observation at Training Period, Vol. 53 No. 2 (2025): April-June 2025, Power System Protection and Control, ISSN-1674-3415, https://pspac.info/index.php/dlbh/article/view/172, , DOI: https://doi.org/10.46121/pspc.53.2.28
17. Jayanth Para, AI Based Cloud Computation Observational Method & Process, Vol. 51 No. 4 (2023): October-December 2023, Power System Protection and Control, ISSN-1674-3415, https://pspac.info/index.php/dlbh/article/view/171, , DOI: https://doi.org/10.46121/pspc.51.4.3
18. Sahu, B., Panigrahi, A., Pati, A., Pati, A.K., Mishra, J. et al. (2025). Harnessing TLBO-Enhanced Cheetah Optimizer for Optimal Feature Selection in Cancer Data. Computer Modeling in Engineering & Sciences, 145(1), 1029–1054. https://doi.org/10.32604/cmes.2025.069618,, https://www.techscience.com/CMES/v145n1/64331
19. Sanjaya Kumar Sarangi, Rasmita Lenka, Janmejaya Mishra, Ritarani Sahu, Arabinda Nanda, Malicious detection and trust calculation using residual recurrent neural network for trust with quality of service-aware multicast routing in mobile ad-hoc network system,Engineering Applications of Artificial Intelligence, Volume 161, Part C,2025,v112130,v,ISSN 0952-1976,https://doi.org/10.1016/j.engappai.2025.112130.https://www.sciencedirect.com/science/article/abs/pii/S0952197625021384
20. Sanjay Das, A STUDY ON ORIGIN OF ERP FROM WAR LOGISTICS TO EVOLUTION INTO BUSINESS WEAPON ABSTRACT , https://www.scribd.com/document/980084363/Paper-id-7053-1
1:M. Vani, “Dynamic Resource Orchestration for Distributed LLM Inference in Heterogeneous Kubernetes Clusters,” Global Journals, 2026. [Online]. Available: https://globaljournals.org/scholarly-articles/dynamic-resource-orchestration-for-distributed-llm-inference-in-heterogeneous-kubernetes-clusters/
21. M. Vani, New Era of Quantum Computing. Bookwire / Bowker, n.d. [Online]. Available: https://bookwire.bowker.com/book/USA/New-Era-of-Quantum-Computing-9781971938462-Vani-Mehul-126971303
22. Mehul Vani “AI-based distributed systems observation system: Real-time monitoring and intelligent anomaly detection for cloud infrastructure,” Power System Protection and Control, vol. 53, no. 4, pp. 446–457, Oct.–Dec. 2025, DOI: https://doi.org/10.46121/pspc.53.4.30, , https://pspac.info/index.php/dlbh/article/view/234
23. Mayank Atreya, Navin Chhibber, Harvendra Singh, Explainable Machine Learning For Dynamic Pricing In Fast-Changing Retail Environments, 2022/4/9, Journal ,Available at SSRN 6011354, https://scholar.google.com/citations?view_op=view_citation&hl=en&user=fyViF1UAAAAJ&citation_for_view=fyViF1UAAAAJ:LkGwnXOMwfcC.
24. Navin Chhibber; Amber Rastogi; Ankur Mahida; Vatsal Gupta; Piyush Ranjan, Quantum-Resistant Cryptographic Models for Next-Gen Cybersecurity, Publisher: IEEE, 2025 2nd Asia Pacific Conference on Innovation in Technology (APCIT), Date Added to IEEE Xplore: 04 March 2026, https://ieeexplore.ieee.org/document/11410884