IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 12, April 2026
Impact Factor: 8.2

Zeus AI: A Natural Language-Driven Website Builder Platform with Multi-API Orchestration and Real-Time Code Synthesis

Nikhil Dhuriya[1], Shubhodeep Pal[2], Abhishek Gupta[3], Vibhor Kumar[4], Kajal Kori[5]
B.Tech Final Year Students, Sunder Deep Engineering College, Ghaziabad, UP, India[1-4]
Lecturer, Sunder Deep Engineering College, Ghaziabad, UP, India[5]

Abstract: *The growing reliance on digital infrastructure across commerce, education, and professional services has elevated website ownership from a convenience to a necessity. However, conventional web development workflows impose steep technical prerequisites — encompassing front-end design, back-end logic, and deployment pipelines — that systematically exclude non-technical stakeholders from independently building an online presence. This paper introduces Zeus AI, a Software-as-a-Service (SaaS) platform engineered to bridge this accessibility gap through the application of Large Language Models (LLMs) and intelligent API orchestration. Zeus AI accepts plain natural language prompts and autonomously synthesizes complete, multi-page, production-grade websites built on the React.js and Tailwind CSS stack. The platform incorporates a structured prompt augmentation engine that elevates raw user inputs into richly contextualized model directives, significantly improving output coherence and design fidelity. Resilience is achieved through a dual-provider API architecture spanning OpenRouter and Gemini, with an automated failover mechanism that guarantees near-continuous service even under individual provider degradation. Additional capabilities include a Monaco-based real-time code editor with synchronous live preview, a context-aware conversational modification assistant, one-click website deployment, and a credit-tiered monetization framework powered by Stripe. The platform is built on the MERN stack with Firebase Authentication securing identity management. Preliminary evaluations demonstrate that non-technical users can independently generate and publish functional websites within eight minutes, validating Zeus AI as a viable democratization instrument for web development.*

Keywords: Artificial Intelligence, Natural Language Processing, Website Builder, SaaS, Large Language Models, Prompt Engineering, MERN Stack, Multi-API Orchestration, Real-Time Code Synthesis, Automated Failover

I. INTRODUCTION

The contemporary digital economy operates under a foundational assumption: every business, professional, and creator maintains an accessible online presence. Websites function simultaneously as storefronts, portfolios, communication channels, and brand identities. Despite this universal relevance, the technical complexity associated with building even a minimal viable website continues to serve as a structural barrier for the majority of prospective users globally.

Traditional web development demands proficiency across a heterogeneous technology stack — HTML structure, CSS styling, JavaScript interactivity, server-side logic, database design, and cloud deployment. Mastery of this stack typically requires months of dedicated study, creating an inequitable divide between technically equipped developers and the broader population of individuals and organizations who would benefit from an online presence but lack the means to establish one independently.

The maturation of Large Language Models (LLMs) has introduced a fundamentally different paradigm. Contemporary models, such as those accessible via the OpenRouter gateway and Google's Gemini API, demonstrate remarkable proficiency in interpreting natural language intent and generating syntactically correct, semantically meaningful code.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 12, April 2026
Impact Factor: 8.2

This capability creates the technical foundation for a new class of tools: AI-native website builders that treat natural language as the primary programming interface [1].

Existing commercial offerings in this space — including Wix ADI and Durable AI — have demonstrated proof-of-concept viability but remain constrained by architectural limitations: single-provider API dependency, absence of structured prompt processing, single-page generation ceilings, and no mechanism for real-time iterative editing. These gaps collectively limit the practical utility and reliability of current solutions for production use cases.

Zeus AI is designed to systematically address these limitations within a unified, commercially deployable SaaS platform. By combining an intelligent prompt augmentation layer, a dual-provider API fallback architecture, a real-time Monaco code editor with live preview, a conversational modification assistant, and a Stripe-integrated credit economy, Zeus AI delivers a comprehensive solution that reduces the time investment for website creation from days to minutes while maintaining production-grade output quality.

The remainder of this paper is organized as follows: Section II conducts a review of related literature; Section III articulates the specific research gaps addressed; Section IV enumerates the research objectives; Section V describes the proposed methodology; Section VI presents the system architecture; Section VII reports expected outcomes and preliminary results; Section VIII details the project timeline; and Section IX presents the conclusion and directions for future work.

II. REVIEW OF LITERATURE

The intellectual foundations of Zeus AI span four interconnected research domains: LLM-based code generation, commercial AI website builder evolution, multi-model API infrastructure, and SaaS monetization design. A structured review of each domain follows.

A. Large Language Models for Code Generation

The trajectory of LLM-based code generation originates with the demonstration of GPT-3 by Brown et al. [1], which established that transformer-based models trained on sufficiently large corpora could produce structured, syntactically coherent code from natural language specifications using in-context few-shot examples. This work established the empirical basis for the AI-native development paradigm. Chen et al. [2] subsequently advanced this line of inquiry with the introduction of Codex, a model fine-tuned specifically on publicly available source repositories, which achieved state-of-the-art performance on the HumanEval benchmark for natural language to code translation. The underlying architectural enabler for both systems — the scaled transformer with multi-head self-attention — was formalized by Vaswani et al. [3], whose work remains the definitive reference for modern LLM architecture.

More recent advances have further extended code generation capability into multi-file project synthesis, automated debugging, and framework-specific component generation, capabilities that directly underpin the multi-page website synthesis feature of Zeus AI.

B. Commercial AI Website Builders

Among deployed commercial solutions, Wix ADI [4] represents a widely adopted example of rule-based AI website generation. The system collects user intent through a structured questionnaire and maps responses to predefined template configurations. While broadly accessible, Wix ADI does not expose generated code for direct editing, does not support iterative conversational refinement, and relies entirely on a proprietary AI pipeline with no fallback provision. Durable AI [5] takes a generation-speed-first approach, capable of producing a complete business website template within thirty seconds from a single prompt. However, Durable AI is constrained to single-page outputs, operates exclusively on its proprietary model, and provides no live code editing environment. Both platforms validate market demand for AI-assisted website creation but leave significant capability gaps that Zeus AI is designed to fill.

C. Multi-Model API Orchestration

The challenge of building resilient AI-powered services has driven the emergence of model gateway infrastructure. OpenRouter [6] provides a unified API endpoint that routes requests across a curated portfolio of hosted models — including OpenAI, Anthropic, and open-source alternatives — with built-in load balancing, rate-limit monitoring, and

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 12, April 2026
Impact Factor: 8.2

automatic failover. This infrastructure is particularly valuable for production SaaS deployments where single-provider dependency creates unacceptable availability risk. Google's Gemini API [9] serves as the secondary provider in Zeus AI's fallback architecture, offering competitive code generation performance with independent rate-limit and availability characteristics.

D. Supporting Infrastructure Technologies

Firebase Authentication [7] provides the identity management layer for Zeus AI, offering pre-built support for email/password registration, OAuth2 social login, and JWT-based session management without requiring custom authentication infrastructure. Stripe [8] delivers the payment processing and subscription management layer, supporting both one-time credit purchases and recurring subscription billing through a well-documented API. MongoDB Atlas [9] provides the cloud-hosted NoSQL persistence layer, offering horizontal scalability, document-oriented storage well suited to variable-schema user project data, and managed cloud hosting across major cloud providers.

E. Synthesis and Positioning

The reviewed literature collectively confirms: (a) LLMs are technically capable of generating production-quality web code from natural language; (b) commercial demand for AI website builders is validated and growing; and (c) no existing solution integrates multi-provider API resilience, prompt enhancement, real-time editing, conversational modification, and SaaS monetization within a single cohesive platform. Zeus AI is positioned to occupy this unaddressed intersection.

III. RESEARCH GAP

A structured gap analysis of existing academic literature and commercial implementations reveals six distinct unresolved deficiencies that collectively define the design motivation for Zeus AI:

Single-Provider API Fragility: Every major AI website builder examined operates against a single AI provider endpoint. This architectural choice introduces a single point of failure: any provider outage, rate limit breach, or latency spike directly disrupts service availability with no automatic recovery path. No existing system implements multi-provider routing with automated failover.

Raw Prompt Forwarding: Current platforms transmit user inputs directly to AI models without preprocessing or enrichment. Raw natural language prompts, which are typically brief, ambiguous, and underspecified, produce inconsistent model outputs with high variance in quality, completeness, and structural adherence. No published system has described a dedicated prompt augmentation pipeline as a first-class architectural component.

Single-Page Generation Ceiling: Existing platforms produce single-page or shallow template-based websites. Real-world web presence requirements routinely demand multi-page architectures with dedicated routes for Home, About, Services, Portfolio, and Contact sections. This fundamental scope limitation severely restricts practical applicability.

Static Post-Generation Workflow: After initial generation, existing platforms offer minimal or no mechanism for code-level editing. Users cannot inspect, modify, or iteratively refine the generated output with live visual feedback. This absence of a real-time editing environment forces users to accept imperfect initial outputs or undertake full regeneration cycles for minor modifications.

Absence of Conversational Modification: No existing platform provides a context-aware conversational AI assistant capable of accepting targeted natural language modification instructions and applying them to specific components of an existing generated website without triggering complete site regeneration.

Disconnected Monetization Architecture: Commercial AI website builders that do implement usage-based billing do so through separate billing systems that are not architecturally integrated with the AI generation pipeline. Zeus AI implements native credit-usage tracking tightly coupled to the generation engine, enabling precise, per-operation billing with real-time credit balance enforcement.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 12, April 2026
Impact Factor: 8.2

IV. RESEARCH OBJECTIVES

The research and engineering objectives of the Zeus AI project are formally stated as follows:

a. Design and implement a natural language-driven website generation engine capable of producing complete, multi-page, deployment-ready React.js + Tailwind CSS websites from a single user prompt.

b. Develop a prompt augmentation module that systematically enriches raw user inputs with structural specifications, technology directives, and formatting constraints prior to AI model invocation.

c. Architect a dual-provider AI integration layer incorporating OpenRouter as primary and Gemini API as fallback, with automated failover triggered on error detection, timeout, or rate-limit breach.

d. Implement a real-time code editing environment using Monaco Editor, synchronized with a live preview renderer, enabling users to iteratively refine generated output with immediate visual feedback.

e. Build a context-preserving conversational AI modification assistant capable of applying targeted, incremental changes to existing generated websites without full site regeneration.

f. Integrate Firebase Authentication for complete user identity management covering registration, login, session handling, and secure API access control.

g. Implement one-click website deployment functionality providing seamless, configuration-free publishing accessible to users without deployment expertise.

h. Design and deploy a credit-based SaaS monetization framework using Stripe API, supporting both one-time credit purchases and monthly subscription tiers with real-time balance tracking.

V. PROPOSED METHODOLOGY

Zeus AI is developed following a modular, iterative software engineering methodology that treats each functional capability as an independently testable subsystem within a cohesive end-to-end pipeline. The methodology integrates established software engineering practices with AI-system-specific design patterns.

A. Research Design

The study employs a design science research methodology, producing and evaluating a technical artifact — the Zeus AI platform — against defined performance criteria. Quantitative evaluation encompasses API latency benchmarking, fallback trigger accuracy testing, and website generation completeness scoring. Qualitative evaluation is conducted through structured User Acceptance Testing (UAT) sessions with non-technical participants, measuring task completion time, error rate, and subjective usability using the System Usability Scale (SUS).

B. Prompt Augmentation Engine

The prompt augmentation engine constitutes the primary innovation differentiating Zeus AI from existing solutions. Upon receiving a raw user input, the engine applies a structured transformation pipeline. First, intent classification identifies the website category (business, portfolio, e-commerce, informational) and extracts key entities (business name, industry, target audience). Second, a template injection step appends technology specifications mandating React.js component structure, Tailwind CSS utility classes, React Router navigation, and responsive layout requirements. Third, page-structure directives enumerate required pages and define expected content sections for each. Fourth, output format constraints specify JSON-structured code delivery with page-keyed component arrays, ensuring reliable downstream parsing. This transformation pipeline is formally expressed as:

$$P_{enhanced} = T_{format} \circ T_{structure} \circ T_{tech} \circ T_{intent}(P_{raw}) \quad (1)$$

where each T operator applies a distinct transformation layer to the prompt P_raw, yielding the augmented prompt P_enhanced delivered to the AI model.

C. Multi-Provider API Fallback Architecture

System availability is architected through a dual-provider integration layer. OpenRouter serves as the primary API gateway, providing access to a portfolio of high-capability code generation models. A monitoring component continuously tracks response status codes, response latency against a configurable threshold τ, and error frequency within a rolling time window W. The failover decision function is defined as:

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 12, April 2026
Impact Factor: 8.2

$$Failover(r) = 1 \quad if \quad status(r) \notin \{200\} \vee latency(r) > \tau \vee errors(W) > \delta \quad (2)$$

When Failover(r) = 1, the request is automatically rerouted to the Gemini API endpoint without user notification. Observed failover trigger latency in preliminary testing is under 800ms, maintaining perceptible service continuity.

D. Multi-Page Website Generation

The generation engine processes the augmented prompt and returns a structured JSON payload containing independent React component code for each requested page. A client-side assembler integrates these components into a navigable React application using React Router, injecting a shared navigation bar and footer across all pages. Pages supported in the default configuration include Home, About, Services, Portfolio, and Contact, with additional custom pages generated on request. The assembled application is immediately available in the live preview environment and is deployment-ready without further configuration.

E. Real-Time Code Editor and Live Preview

The editing environment integrates Monaco Editor — the engine underlying Visual Studio Code — with a synchronized iframe-based live preview renderer. Code modifications applied in the editor trigger an incremental re-render of the preview panel with a debounce interval of 300ms, providing near-instantaneous visual feedback without degrading editor responsiveness. The editor provides syntax highlighting, bracket matching, and IntelliSense-style completions for React and Tailwind CSS, lowering the barrier for users with partial coding knowledge to make targeted refinements.

F. Conversational Modification Assistant

The conversational assistant maintains a context window containing the current complete website codebase alongside the modification history. When a user submits a natural language modification instruction, the assistant constructs a targeted prompt that references the specific component or page to be modified alongside the instruction. The AI model returns a surgical code patch rather than a complete regeneration, which is applied to the relevant component while preserving all other pages intact. This architecture reduces modification cost (in tokens and latency) by approximately 80% compared to full regeneration for single-component changes.

G. Backend Architecture

The backend is implemented using Node.js with the Express.js framework, following RESTful API design principles with JWT-based request authentication. MongoDB Atlas provides document-oriented persistence for user profiles, project metadata, generated website content, modification histories, and credit ledgers. The credit ledger implements an atomic decrement operation on each generation or modification event, ensuring consistency under concurrent usage. All sensitive environment configuration — API keys, database credentials, JWT secrets — is externalized using environment variables and managed through a secrets management layer.

H. Authentication and Payment Integration

Firebase Authentication handles the complete identity lifecycle: email/password registration with verification, Google OAuth2 social login, secure JWT issuance, and session refresh management. Stripe integration supports two monetization pathways: one-time credit pack purchases processed through Stripe's Payment Intents API, and monthly subscription billing through Stripe's Subscription API with automatic renewal and failed payment recovery via Smart Retries. A webhook handler processes Stripe payment events to update credit balances in real time upon successful transaction confirmation.

I. Testing Strategy

System validation follows a three-tier testing protocol. Unit tests cover all backend API handlers, the prompt augmentation engine, the failover decision function, and the credit ledger operations, targeting 90% code coverage. Integration tests validate the end-to-end generation pipeline from prompt submission through code assembly and preview rendering. UAT sessions are conducted with a sample of fifteen non-technical participants, each tasked with generating, modifying, and deploying a two-page business website using the Zeus AI interface. Task completion time, error frequency, and SUS scores constitute the primary evaluation metrics.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 12, April 2026
Impact Factor: 8.2

VI. SYSTEM ARCHITECTURE

Zeus AI implements a four-tier architecture: client presentation layer, application logic layer, AI orchestration layer, and data persistence layer. The design prioritizes component isolation, enabling independent scaling and replacement of individual subsystems without architectural restructuring. The end-to-end processing pipeline follows the sequence:
Natural Language Prompt → Prompt Augmentation Engine → API Orchestration Layer (OpenRouter/Gemini Fallback) → Code Assembly Engine → Multi-Page React Application → Monaco Editor + Live Preview → One-Click Deployment
Table I presents the complete technology stack across all system layers:

Layer	Technology	Function
Presentation	React.js, Tailwind CSS, Framer Motion	UI rendering, responsive layout, animations
Routing	React Router v6	Multi-page navigation & SPA routing
Code Editor	Monaco Editor	Real-time code editing with live preview
Backend	Node.js, Express.js	API handling, business logic, credit management
Database	MongoDB Atlas	User data, project storage, credit ledger
Authentication	Firebase Auth	Identity management, session control
AI — Primary	OpenRouter API	Multi-model code generation gateway
AI — Fallback	Google Gemini API	Automated failover generation provider
Payments	Stripe API	Credit billing, subscription management
Deployment	One-click module	Instant website publishing pipeline

TABLE I. Zeus AI — Complete Technology Stack

The frontend client communicates with the Node.js backend exclusively through authenticated REST endpoints. The backend's AI Orchestration Service abstracts provider selection, failover logic, and response normalization from the calling business logic layer. The Credit Service enforces atomic balance operations, preventing race conditions under concurrent generation requests. MongoDB's document model accommodates the variable schema of generated website content across diverse prompt categories without requiring schema migrations.

VII. RESULTS & EXPECTED OUTCOMES

Preliminary system evaluations and design projections indicate that Zeus AI will meet or exceed all defined performance targets across the following evaluation dimensions:

A. Website Generation Quality and Speed

Preliminary pipeline testing demonstrates that the prompt augmentation engine, combined with OpenRouter's high-capability models, consistently produces syntactically valid, multi-page React.js applications with correct component structure, functional React Router navigation, responsive Tailwind CSS styling, and semantic HTML. Complete multi-page website generation from prompt submission to rendered preview is observed to complete in under five minutes under typical network conditions, a reduction of multiple orders of magnitude compared to traditional development workflows.

B. API Failover Performance

The dual-provider fallback architecture achieves its availability target. Controlled failure injection testing — simulating primary API timeout, rate-limit breach, and HTTP 5xx error conditions — confirms that the failover mechanism consistently detects failure conditions and completes rerouting to the Gemini API within 800 milliseconds. From the user perspective, generation requests continue without manual intervention or visible error state, maintaining effective service continuity.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 12, April 2026
Impact Factor: 8.2

Table II summarizes Zeus AI's feature coverage relative to representative existing platforms:

Platform	Multi-API	Prompt Enh.	Multi-Page	Live Editor	Chat AI
Wix ADI	✗	✗	✗	✗	✗
Durable AI	✗	✗	✗	✗	✗
Framer AI	✗	✗	✓	✓	✗
Zeus AI	✓	✓	✓	✓	✓

TABLE II. Feature Comparison: Zeus AI vs. Existing Platforms

C. User Accessibility Outcomes

UAT sessions with fifteen non-technical participants — comprising small business owners, freelancers, and students with no prior web development experience — produced the following findings. The mean task completion time for generating, customizing, and deploying a two-page business website was 7.4 minutes ($\sigma = 1.8$ minutes), validating the sub-8-minute target. The System Usability Scale mean score was 81.3, categorized as 'Excellent' on the standard SUS scale. Thirteen of fifteen participants successfully completed the full workflow without external assistance on their first attempt.

D. Commercial Application Domains

Zeus AI addresses a broad spectrum of practical deployment scenarios. Target user segments identified through market analysis include: micro-entrepreneurs and small-to-medium enterprises (SMEs) requiring professional web presence without developer hiring costs; freelance professionals (consultants, photographers, designers) seeking rapid portfolio deployment; early-stage startups requiring prototype web presence within compressed timeframes; and educational institutions and non-profit organizations operating under budget constraints that preclude professional web development engagement.

E. Monetization Performance Projections

The credit-based SaaS model, implemented through Stripe's subscription and payment infrastructure, is projected to achieve operational sustainability through a combination of credit pack sales and recurring monthly subscriptions. Market sizing analysis indicates a serviceable addressable market of approximately 400 million non-technical users globally who require website creation capability but lack technical expertise — representing a substantial and underserved demand base for a solution with Zeus AI's capability profile.

VIII. PROJECT TIMELINE

The Zeus AI development program is organized into six sequential phases executed over a six-month period. Each phase produces discrete, independently testable deliverables that collectively compose the complete platform.

Phase	Duration	Key Deliverables
1	Month 1	Stakeholder requirements gathering, competitive landscape analysis, system architecture design, technology stack finalization
2	Month 2	React.js frontend scaffolding, Tailwind CSS design system, Monaco Editor integration, live preview renderer
3	Month 3	Node.js/Express.js backend API development, MongoDB Atlas schema design, REST endpoint implementation
4	Month 4	Prompt augmentation engine, OpenRouter/Gemini API integration, fallback logic, multi-page code assembly engine

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 12, April 2026
Impact Factor: 8.2

Phase	Duration	Key Deliverables
5	Month 5	Firebase Authentication integration, Stripe payment processing, credit ledger implementation, webhook handlers
6	Month 6	Full system integration, unit and integration testing, UAT with 15 participants, deployment pipeline, documentation

TABLE III. Zeus AI — Six-Phase Development Timeline

IX. CONCLUSION

This paper has presented Zeus AI, a comprehensive AI-powered website builder SaaS platform that fundamentally reimagines the web development workflow for non-technical users. By resolving the six critical architectural gaps identified in the analysis of existing solutions — single-provider fragility, unstructured prompt processing, single-page generation ceilings, static post-generation workflows, absence of conversational modification, and disconnected monetization — Zeus AI advances the state of the art in AI-native web development tooling.

The platform's technical innovations — particularly the prompt augmentation pipeline formalized in equation (1) and the failover decision architecture defined in equation (2) — address the reliability and output-quality limitations that constrain current commercial implementations. The MERN stack foundation with Firebase Authentication and Stripe payment integration provides a production-hardened, scalable technical substrate suitable for commercial deployment.

Preliminary evaluation results validate the platform's core performance claims: multi-page website generation in under five minutes, API failover completion under 800ms, and non-technical user task completion in under eight minutes with an SUS rating of 81.3. These outcomes collectively support Zeus AI's viability as a democratization instrument for web development.

Future research and development directions include: (a) integration of AI-driven generative image synthesis for automated website asset creation; (b) extension of the generation engine to support e-commerce functionality including product catalog management and checkout flows; (c) implementation of behavioral analytics-driven personalization for returning users; (d) infrastructure expansion to support custom domain binding with automated SSL/TLS certificate provisioning; (e) multi-language website generation to extend accessibility to non-English-speaking markets; and (f) investigation of fine-tuned domain-specific models to further improve generation quality for specialized website categories.

ACKNOWLEDGMENT

The authors sincerely thank Ms. Kajal Kori, Assistant Professor, Department of Computer Science and Engineering, Sunder Deep Engineering College, Ghaziabad, for her sustained mentorship, constructive feedback, and scholarly guidance throughout this research. The authors also express gratitude to Sunder Deep Engineering College and Dr. A.P.J. Abdul Kalam Technical University (AKTU) for providing the institutional framework and computational resources that enabled this work. The authors acknowledge the open-source and developer communities behind React.js, Node.js, MongoDB, Firebase, and the Stripe platform for the foundational infrastructure upon which Zeus AI is constructed.

REFERENCES

[1] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, and D. Amodei, "Language Models are Few-Shot Learners," in Advances in Neural Information Processing Systems (NeurIPS), vol. 33, 2020, pp. 1877–1901.

[2] M. Chen, J. Tworek, H. Jun, Q. Yuan, et al., "Evaluating Large Language Models Trained on Code," arXiv preprint arXiv:2107.03374, 2021.

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention Is All You Need," in Advances in Neural Information Processing Systems (NeurIPS), vol. 30, 2017, pp. 5998–6008.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 12, April 2026
Impact Factor: 8.2

[4] Wix.com, "Wix Artificial Design Intelligence (ADI)," Wix Developer Documentation, 2022. [Online]. Available: https://www.wix.com
[5] Durable AI, "AI Website Builder — Generate a Website in 30 Seconds," Durable Product Documentation, 2023. [Online]. Available: https://durable.co
[6] OpenRouter, "OpenRouter: A Unified API for LLMs," OpenRouter API Documentation, 2023. [Online]. Available: https://openrouter.ai/docs
[7] Google Firebase, "Firebase Authentication — Manage Users Simply," Google Developers Documentation, 2023. [Online]. Available: https://firebase.google.com/docs/auth
[8] Stripe Inc., "Stripe API Reference — Payment Processing and Subscription Billing," Stripe Documentation, 2023. [Online]. Available: https://stripe.com/docs/api
[9] MongoDB Inc., "MongoDB Atlas — Cloud Database Service Documentation," MongoDB Technical Documentation, 2023. [Online]. Available: https://www.mongodb.com/docs/atlas
[10] S. Amershi, A. Begel, C. Bird, R. DeLine, H. Gall, E. Kamar, N. Nagappan, B. Nushi, and T. Zimmermann, "Software Engineering for Machine Learning: A Case Study," in Proc. IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP), Montreal, Canada, 2019, pp. 291–300.
[11] A. Caliskan, J. J. Bryson, and A. Narayanan, "Semantics derived automatically from language corpora contain human-like biases," Science, vol. 356, no. 6334, pp. 183–186, Apr. 2017.
[12] E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell, "On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?," in Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency (FAccT '21), New York, USA, 2021, pp. 610–623