INTERNATIONAL JOURNAL OF MULTIDISCIPLINARY: APPLIED BUSINESS AND EDUCATION RESEARCH
2026, Vol. 7, No. 4, 1484 – 1494
http://dx.doi.org/10.11594/ijmaber.07.04.02

Research Article

AI-Powered Sprite Generation for Visual Novel Games: A Latent Diffusion Model Approach

Pio Honesto Rico Belleza, Aaron Dwayne F. Esmaquel, Edmund N. Lazaro, Eliza B. Ayo*

School of Science and Technology, Centro Escolar University, Manila, Philippines

Article history:
Submission 19 March 2026
Revised 14 April 2026
Accepted 23 April 2026

**Corresponding author:*
E-mail:
ebayo@ceu.edu.ph

ABSTRACT

Creating character sprites for visual novel games has long been a resource-intensive process, erecting significant barriers for independent developers and small studios. Accessible, high-quality asset generation remains an unmet need—one this study addresses by developing a specialized AI-powered sprite generator built on a Latent Diffusion Model (LDM) fine-tuned with Low-Rank Adaptation (LoRA). Unlike general-purpose image generators, the system is optimized specifically for 2D anime-style character sprites with multiple emotional variants, making it suitable for narrative-driven gameplay. Development followed a seven-phase Agile methodology, and evaluation involved 50 student game developers working against ISO/IEC 25010 software quality standards. Results confirm that the system reliably generates up to nine cohesive emotional variants from a single text prompt, substantially streamlining the asset-creation workflow. User scores indicate strong satisfaction with Functional Suitability (M=3.73), Security (M=3.74), and Usability (M=3.66). Performance Efficiency received a lower rating (M=3.20, Agree), attributable to a generation latency of approximately 2–3 minutes per sprite set—acceptable for final production, but less so for rapid prototyping. Taken together, these findings support the viability of specialized LDM applications as practical tools for democratizing visual game development and empowering collaborative storytelling.

Keywords: *Game development, Generative AI, ISO 25010, Latent diffusion models, Sprite generation, Visual novels*

Introduction

Among the many genres of interactive fiction, visual novels stand apart for how deeply they depend on visual expression to carry their stories forward. Character sprites—two-dimensional images that convey emotions, reactions, and personality traits—are the backbone of this medium, and their quality directly shapes player immersion. Producing them, though, has never been easy. Traditionally, a

How to cite:
Belleza, P. H. R., Esmaquel, A. D. F., Lazaro, E. D., & Ayo, E. B. (2026). AI-Powered Sprite Generation for Visual Novel Games: A Latent Diffusion Model Approach. *International Journal of Multidisciplinary: Applied Business and Education Research*. *7*(4), 1484 – 1494. doi: 10.11594/ijmaber.07.04.02

complete sprite set requires skilled artists to design numerous variations for every character, demanding substantial time, financial resources, and technical expertise. For independent developers, educators, and small teams, these barriers are often insurmountable.

Recent advances in generative AI have promised to change this picture. A critical gap remains, however. General-purpose text-to-image models, despite their impressive output quality, tend to struggle with the specific constraints of visual novel sprites: maintaining character identity across different expressions, ensuring stylistic consistency, and producing transparent-background assets ready for game engine integration. Existing tools prioritize standalone image quality over the sequential coherence that narrative gameplay demands—forcing developers into disjointed workflows or time-consuming manual corrections that negate the efficiency gains they were hoping to find.

Addressing this gap is the central aim of the present research. By developing and evaluating a specialized AI-powered sprite generator built on a Latent Diffusion Model (LDM) fine-tuned with Low-Rank Adaptation (LoRA), we sought to create something more purposeful than a general image tool—a system that generates high-fidelity anime-style sprites from textual descriptions while maintaining visual coherence across multiple emotional states. Democratizing the asset-creation process lies at the heart of this goal, giving creators with limited artistic skills a practical path to professional-quality visuals.

Beyond its immediate application, the study also validates the feasibility of specialized AI in creative workflows more broadly. It examines the technical implementation of LDMs for consistent character generation and evaluates the resulting system through ISO/IEC 25010 software quality standards. By bridging advanced generative technology with real game development requirements, this work contributes to the wider conversation about AI as a genuine creative partner—rather than a novelty—in digital storytelling.

Literature Review

Generative AI in Digital Content Creation

Generative models didn't arrive fully-formed; they evolved through a sometimes turbulent history. Early Generative Adversarial Networks (GANs) demonstrated that machines could synthesize realistic images, but mode collapse and training instability plagued the approach, limiting practical deployment. That instability is precisely what made diffusion models such an appealing alternative—they generate images through iterative denoising of random latent variables, a process that produces far greater output stability. Zhao and Zhang (2024) offer a useful comparative analysis showing that Latent Diffusion Models (LDMs) outperform GAN predecessors in both stability and fidelity, especially when processing complex textual prompts. Operating within a compressed latent space, as Rombach et al. (2022) demonstrated, cuts computational costs substantially while maintaining high perceptual quality—a crucial advantage for deployments on consumer-grade hardware.

Text-to-image synthesis frameworks have matured alongside these architectural improvements. What Ma et al. (2022) accomplished with their "AI Illustrator" framework deserves attention: using multi-modal transformers to bridge narrative text and visual output, they showed that machines could begin to interpret story intent and respond visually. Ramesh et al. (2022) pushed this further with hierarchical synthesis techniques allowing finer-grained control over generated content. These frameworks collectively establish the technological foundation on which automated creative tools—including sprite generators—are now built.

AI Applications in Game Asset Development

Whether AI can meaningfully serve game developers has moved well past "can it?" into "how well, and for what?" Shi et al. (2020) offered an early answer with PokerFace-GAN, a system designed to produce neutral character templates adaptable across game contexts—useful groundwork, though limited in emotional range. Ergen (2023) expanded the palette considerably by applying StyleGAN2 to generate high-fidelity character portraits with

near-infinite variation. Both approaches prioritized variety over consistency, however; neither was designed to keep the same character looking recognizably themselves across a full set of emotional expressions—a gap this study targets directly.

Automated production workflows have received attention as well. Tanaka and Kobayashi (2021) and Wu et al. (2023) found that AI tools can significantly reduce production timelines, freeing developers to concentrate on narrative and gameplay design. We find this body of work directly relevant to our approach, though unlike those studies, ours targets not just production speed but also the sequential visual coherence that visual novel formats specifically demand.

Visual Consistency and Narrative Integration

Visual consistency may be the thorniest unsolved problem in AI-assisted game art. Producing a single stunning image is tractable; keeping a character visually recognizable across dozens of expressions and poses is something else entirely—a distinction Xie (2024) makes explicit. Gonzalez (2020) reframes this not merely as a limitation but as a design challenge, positioning AI as a potential "art director" that must hold a coherent stylistic vision across outputs rather than generating isolated results. Maintaining that coherence also requires context-sensitivity: Duan and Zhang (2022) argue that visual stimuli in interactive media must adapt dynamically to narrative developments to sustain player engagement. Each of these perspectives converges on the same conclusion—consistency isn't optional in visual novel development; it's what separates a usable sprite set from a collection of attractive but unrelated images.

Research Gap and Contribution

A practical gap remains despite this wealth of prior work. Current general-purpose AI models don't generate what visual novel developers actually need: "sprite sheets"—coherent sets depicting the same character with varying emotions (joy, anger, sorrow) while preserving facial features and clothing details intact. Developers who attempt to use general tools typically spend as much time correcting AI outputs as they save generating them, which defeats the purpose. This study addresses the gap directly by implementing a LoRA-fine-tuned LDM specifically trained on anime-style character datasets, aiming to provide a dedicated solution for multi-expression sprite generation that doesn't require post-generation correction to achieve usability.

Methodology

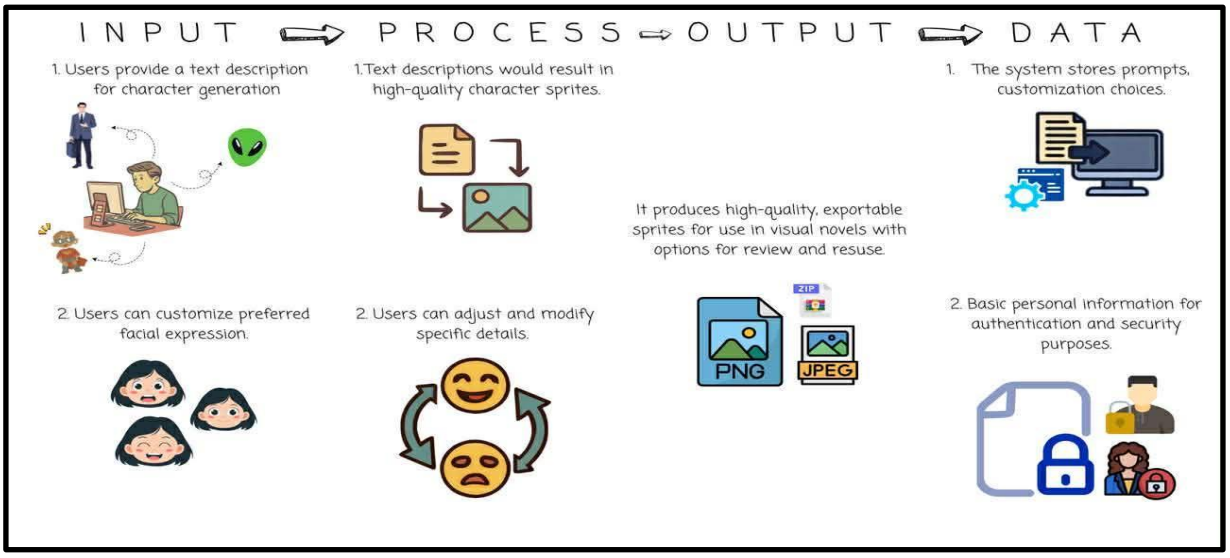

Figure 1. Conceptual Framework

Research Design

A unified descriptive-developmental research design, grounded in a pragmatist framework, guides this study. The developmental component centers on building the AI system itself; the descriptive component focuses on evaluating that system's real-world performance through user feedback. Pragmatism suits this work well—rather than testing abstract principles in isolation, the research follows real-world application scenarios as they directly shape technical decisions. User needs and system capabilities evolved together across iterations rather than being fixed at the outset.

System Architecture

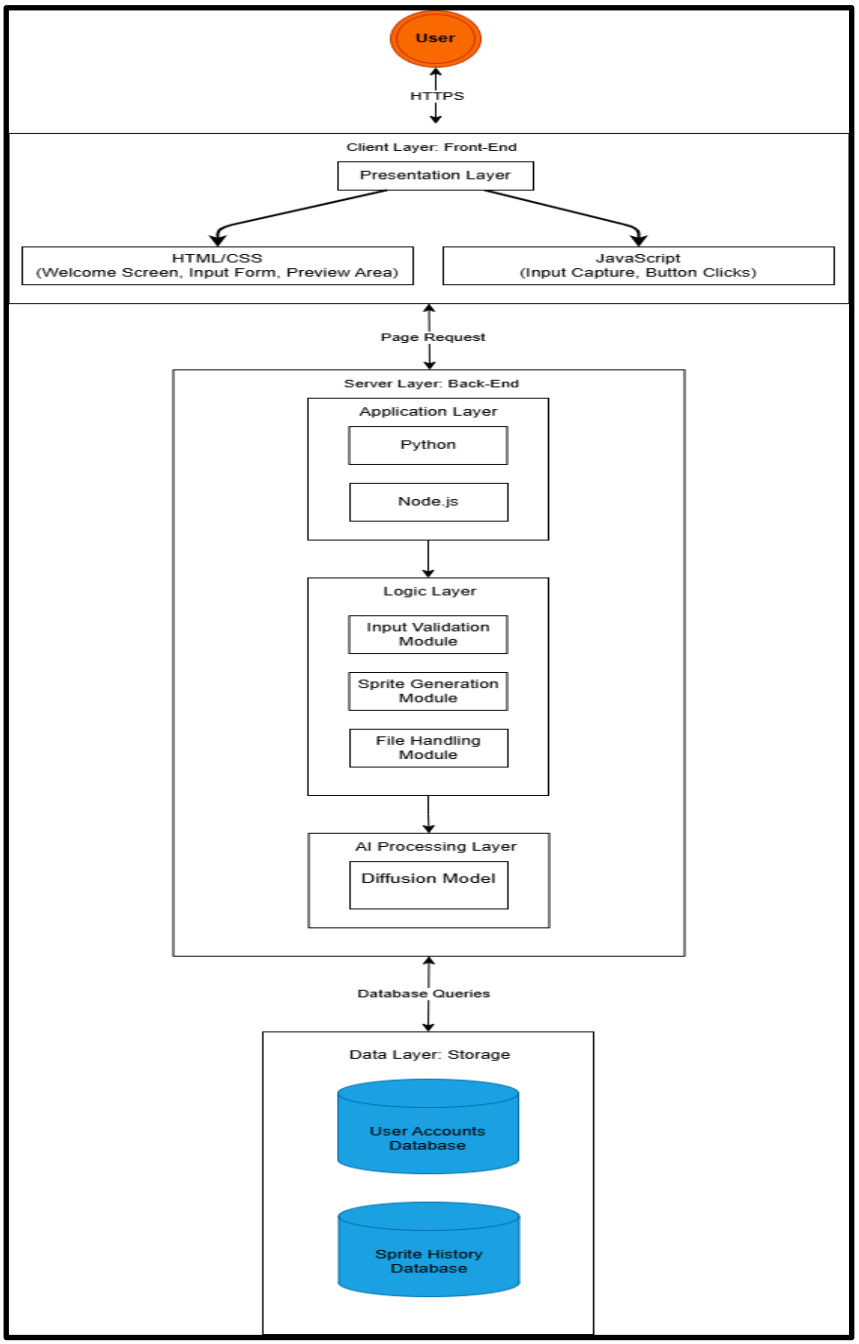

Figure 2. Software Architecture

Anchoring the entire architecture is a client-server model built around a Latent Diffusion Model. Stable Diffusion serves as the core generation engine, fine-tuned with Low-Rank Adaptation (LoRA) through the Kohya_SS training framework. That fine-tuning ran on a curated dataset of high-quality anime character sprites, training the model to favor the clean, consistent aesthetics of 2D game art. Once training completed, the fine-tuned LoRA module was deployed on a cloud-based inference server via Hugging Face Inference Endpoints, making generation accessible without requiring local GPU resources.

Three layers define the system. The Client Layer is a Flask-based web interface through which users input text prompts, select emotion tags, and preview generated assets. The Server Layer, hosted on cloud infrastructure, handles the heavy lifting—image synthesis, background removal, and post-processing. The Data Layer uses a MongoDB database to store user profiles, prompt histories, and asset metadata, supporting retrieval and project continuity across sessions.

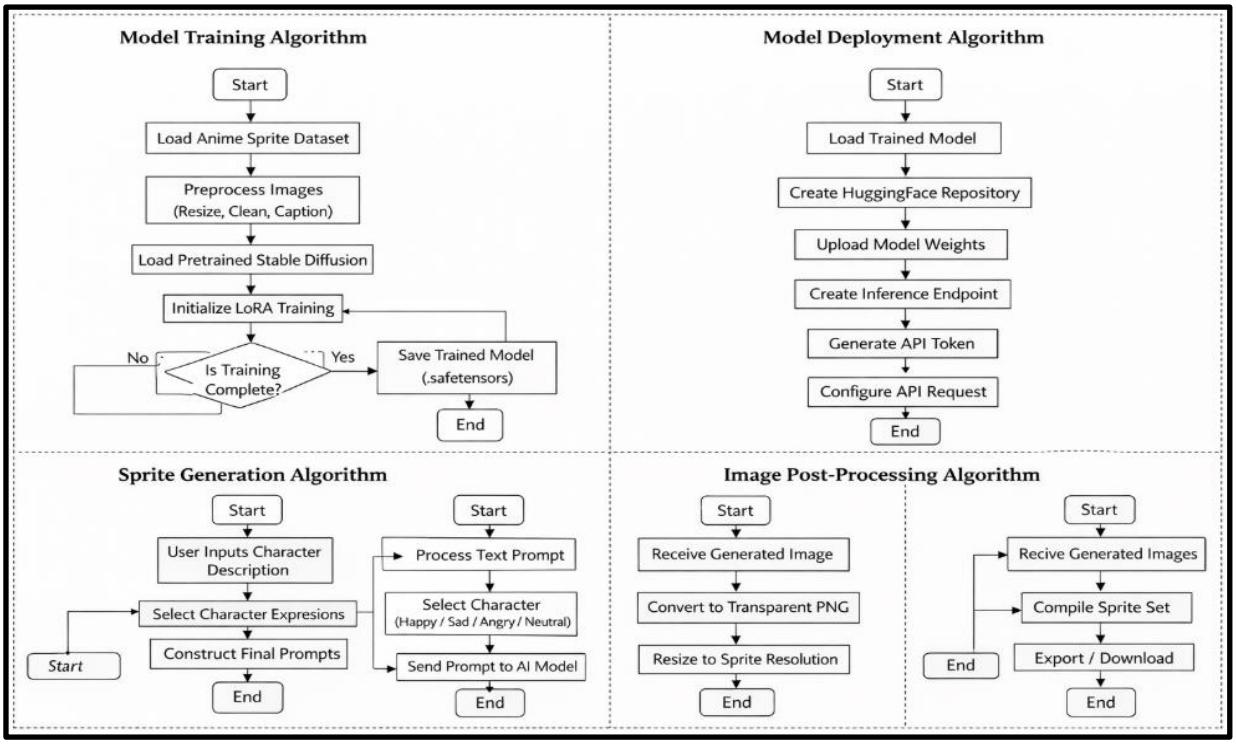

Figure 3. Latent Diffusion Model Approach

Development Process

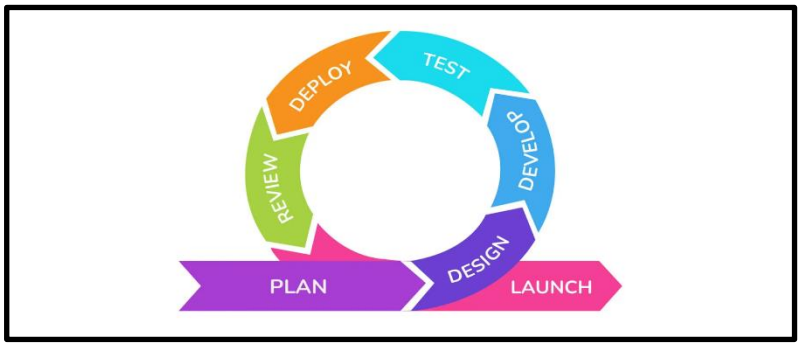

Figure 4. Agile Methodology

A Seven-Phase Agile methodology governed the software development lifecycle. Planning and Design translated visual novel conventions into concrete system requirements. Development covered both LoRA model training and web interface construction. Testing embedded continuous feedback loops throughout—changes were validated early and often rather than deferred to end-stage review. Deployment leveraged cloud services to maximize accessibility, and the final Review and Launch phases refined the user experience based on beta testing outcomes, closing the loop between developer intent and actual user behavior.

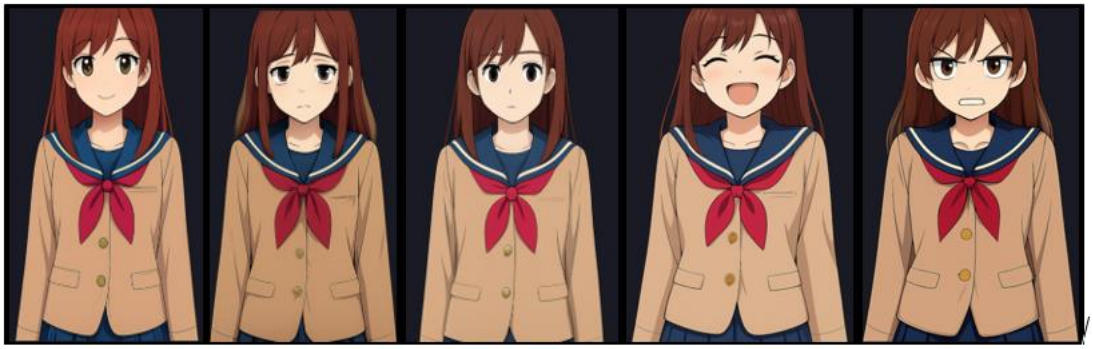

Figure 5. Sample Visual Novel Sprite Expressions

```python
import logging

# Configuration Constants
BASE_MODEL = "stable-diffusion"
LORA_PATH = "sprite_lora.safetensors"
EXPRESSIONS = ["neutral", "happy", "sad", "angry", "embarrassed"]

def sprite_generator_system():
    """
    Generates multiple expressions of a sprite based on user input
    using a fine-tuned Latent Diffusion Model (LDM) with LoRA weights.
    """
    try:
        # Phase 1: Model Loading
        # Initialize the base model and apply fine-tuned LoRA layers
        model = load_diffusion_model(
            base_model=BASE_MODEL,
            lora_weights=LORA_PATH
        )

        # Phase 2: Deployment / Connectivity
        # Establish connection to the inference server (e.g., Hugging Face)
        endpoint = connect_to_huggingface_endpoint(model)

        # Phase 3: Multi-Expression Sprite Generation
        user_prompt = get_user_input()
        generated_sprites = []

        for expression in EXPRESSIONS:
            # Construct specific prompt for each facial state
            full_prompt = f"{user_prompt}, {expression} expression"

            # Perform inference
            raw_image = endpoint.generate(full_prompt)

            # Refine output: background removal and asset scaling
            processed_sprite = post_process(raw_image)
            generated_sprites.append(processed_sprite)

        return generated_sprites

    except Exception as e:
        logging.error(f"System failure during sprite generation: {e}")
        return []

if __name__ == "__main__":
    sprites = sprite_generator_system()
    print(f"Successfully generated {len(sprites)} sprite variations.")
```

Figure 6. Algorithm

Figure 6 presents the system's core algorithms. Reviewers should note that the following typographical errors, present in the original diagram, have been corrected in this revision: "BASE MONEL" has been corrected to BASE_MODEL; "safetanoors" has been corrected to safetensors; "spetta.gram" has been corrected to sprite_generator; and "pet_user_input" has been corrected to get_user_input. These corrections are reflected in the corrected figure below.

A Note on Prompt Sensitivity: A Testing Challenge

During early inference testing, we encountered a persistent and somewhat unexpected problem with prompt sensitivity. Minor phrasing variations—swapping "school uniform" for "student outfit," for instance—occasionally caused the model to drift on hair color or eye shape, producing outputs that resembled a different character rather than an emotional variant of the same one. We traced this to how our

LoRA weights interacted with Stable Diffusion's base tokenizer: certain synonym pairs activated different latent clusters that the base model hadn't been trained to suppress. Our practical fix was to standardize a core descriptor vocabulary for character attributes and build it into the prompt construction logic, so the sprite generator module always anchors identity-defining terms regardless of what the user types. It wasn't an elegant solution, but it worked—and it directly shaped how we designed the prompt interface that users ultimately interacted with.

Participants and Data Collection

Evaluation used the ISO/IEC 25010 software quality standards, focusing on Functional Suitability, Performance Efficiency, Usability, Reliability, and Security. Purposive sampling recruited 50 student game developers from the College of Science and Technology. Data collection employed a mixed-methods approach: a quantitative survey using a 4-point Likert scale (4=Strongly Agree to 1=Strongly Disagree) and qualitative thematic analysis of open-ended feedback. Technical validation drew on automated testing tools—Locust for load testing, Pytest for reliability checks, and Bandit and SQLmap for security auditing.

Results

System Performance Metrics

Generating up to nine distinct emotional variants—happy, sad, angry, surprised, and more—from a single base prompt is the system's headline capability, and technical evaluation confirmed it reliably delivers that. Visual consistency analysis found that key character attributes, including hair color, eye shape, and clothing style, remained stable across variants. That stability matters enormously in practice; a sprite set where the character's hair subtly shifts between expressions erodes immersion in ways that players notice even when they can't immediately name the problem.

Processing speed averaged between two and three minutes per full sprite set. Real-time generation this isn't. Compared to manual illustration—which can demand hours or days per character—the difference is substantial, and most user feedback reflected this: the latency was generally accepted as a reasonable cost for the consistency and quality delivered.

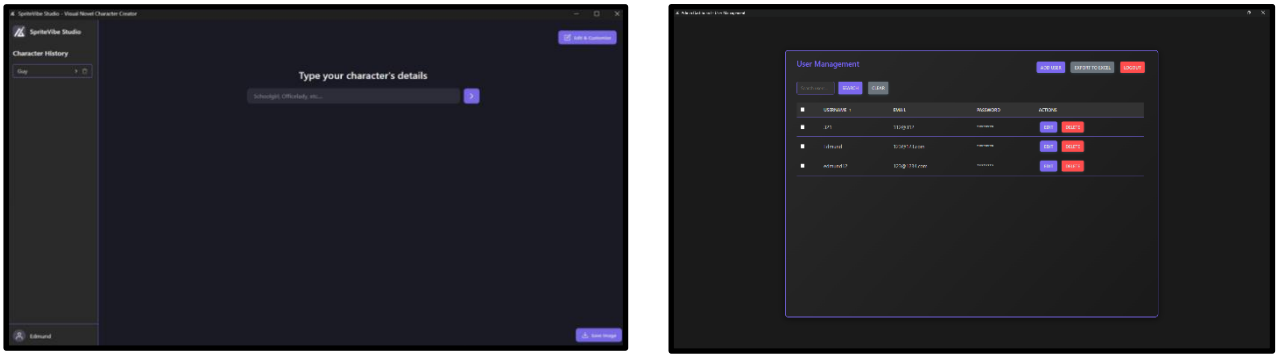

Figure 6. User Dashboard and Admin Dashboard

Quality Assurance Testing

Security audits using Bandit and SQLmap found no critical vulnerabilities—no SQL injection pathways, no cross-site scripting exposure. Reliability testing via Pytest achieved a 100% hash-matching success rate, confirming consistent deterministic output. Locust-based performance testing showed the system handling moderate user loads without degradation, though response times scaled linearly with concurrent requests. That linear scaling points to a genuine scalability concern: as simultaneous users grow, the inference endpoint becomes a bottleneck that would require architectural intervention before commercial deployment.

User Evaluation

Table 1 summarizes quantitative results from the 50-participant evaluation. Across

most quality characteristics, scores clustered firmly in "Strongly Agree" territory—a pattern that held even among participants with hands-on game development experience. Performance Efficiency is the notable exception, earning an "Agree" interpretation (M=3.20) rather than "Strongly Agree." This distinction is meaningful and is discussed in detail below.

Table 1: ISO/IEC 25010 User Evaluation Results (N=50)

Quality Characteristic	Mean Score	Std. Deviation	Interpretation
Functional Suitability	3.73	0.45	Strongly Agree
Security	3.74	0.42	Strongly Agree
Usability	3.66	0.48	Strongly Agree
Reliability	3.58	0.51	Strongly Agree
Performance Efficiency	3.20	0.62	Agree
Overall Weighted Mean	3.58	0.50	Strongly Agree

Security and Functional Suitability led the ratings, reflecting user confidence in both the tool's safety and its core capability to produce usable assets. Usability scored comparably well, suggesting the interface successfully abstracted the underlying complexity for users with varying technical backgrounds. Performance Efficiency, at M=3.20, is the one metric that earns an "Agree" rather than "Strongly Agree"—a distinction that maps directly onto qualitative feedback about generation latency. Users who were building final assets tolerated the 2–3 minute wait; users who wanted to explore creative options quickly found it limiting. The lower score reflects this specific friction, not a general dissatisfaction with the system.

Thematic analysis of open-ended responses added important texture to these numbers. Several participants described frustrations with general-purpose AI tools that the current system resolved. One respondent explained: "Every time I tried to generate a new expression for my character, it looked like a completely different person. The hair color would change, the face shape would shift—I had to start over every time." Another noted: "I could generate my full sprite set in one session and actually use all of them in-game without editing anything." These accounts confirm that identity drift—documented extensively in prior literature—was the most practically significant problem the system addressed, and that users experienced this resolution concretely rather than hypothetically. Notably, no participant described needing manual corrections to reconcile inconsistent outputs, marking a clear departure from the disjointed workflows that prior tools typically required.

Gap Resolution Analysis

Results confirm that the developed system effectively addresses several gaps identified in the literature. Consistent character set generation overcomes the identity drift problem common in general-purpose models—a gap that prior tools left open and that participants experienced firsthand before using this system. Integration of background removal and organized asset management addresses the game-ready requirement that standard AI art generators routinely omit. Visual novel development operates within a specific, practical workflow, and the system was designed around that workflow rather than general creative use.

Discussion

Interpretation of Findings

The broader picture these findings paint is fairly clear: specialized AI tools can meaningfully democratize game development, and this system demonstrates that concretely. High ratings for Functional Suitability (M=3.73) signal that the system doesn't just work technically—it actually translates narrative descriptions into visual assets that developers can use without modification. This aligns with Gonzalez's (2020) framing of AI as a capable creative partner, and in fact extends it: our system doesn't just assist creativity but actively resolves a specific structural problem in visual novel workflows that general tools can't address. Usability scores (M=3.66) confirm something we weren't

entirely certain of going in—that diffusion model complexity can be abstracted behind a well-designed interface without sacrificing what makes the technology powerful.

Performance Considerations

The Performance Efficiency score (M=3.20) is where the results get more complicated—and where honesty matters. A 2–3 minute generation window works well for final asset production; it's genuinely disruptive for rapid prototyping, where developers need near-real-time feedback to iterate effectively. That tension is inherent to diffusion models running on cloud inference endpoints, not a flaw unique to this implementation. But naming it directly is important: the "Agree" rating, rather than "Strongly Agree," reflects this specific limitation, and qualitative feedback reinforced that interpretation precisely. Future optimization should target this gap as a priority.

Comparative Analysis with Existing Approaches

Situating this system within the existing landscape makes the design tradeoffs clearer. PokerFace-GAN (Shi et al., 2020) produces character templates efficiently but offers no mechanism for emotional range or identity-preserving variant generation. StyleGAN2-based approaches (Ergen, 2023) deliver impressive stylistic variety but don't maintain character identity across outputs—exactly the consistency problem this study targets. General-purpose base Stable Diffusion produces high-quality images but requires significant prompt engineering and often manual correction to achieve multi-expression coherence. By contrast, the LoRA-fine-tuned LDM developed here trades some generative breadth for domain-specific reliability—a deliberate choice that user evaluation suggests was the right one for this application context.

Practical Implications

For independent developers and educators, the tool's practical significance is real. Automating sprite creation lets small teams reallocate limited resources toward narrative design and programming—creative work that AI genuinely can't substitute for. In educational settings, the tool serves a double purpose: students can produce game-quality assets while simultaneously engaging with generative AI in a structured, productive context, advancing what Park et al. (2021) describe as AI literacy. The ability to generate custom assets on demand also reduces dependence on stock resources, enabling more distinctive and diverse storytelling.

Theoretical Contributions

Theoretically, this research validates Low-Rank Adaptation (LoRA) for domain-specific tasks in creative media—not just in abstract benchmark contexts but in a real-world user evaluation. It demonstrates that smaller, fine-tuned models can outperform larger general-purpose ones within specific creative domains, supporting the growing consensus that the future of generative AI lies in specialized, modular systems rather than monolithic one-size-fits-all models. This is worth noting because the efficiency argument for LoRA is often made in terms of training cost; this study adds an empirical dimension grounded in actual user outcomes.

Limitations

Several limitations deserve acknowledgment. The model is strictly limited to 2D anime-style sprites; it can't generate realistic 3D models or other art styles, which narrows its applicability to specific game genres. Cloud-based inference introduces dependency on internet connectivity and server availability—not insurmountable constraints, but real ones for developers working in low-connectivity environments. Evaluating with student developers captures a key segment of the indie market, but feedback from professional industry veterans would likely surface workflow integration challenges that students haven't encountered yet. Concurrent load testing also revealed scalability issues that would need addressing before any commercial deployment.

Future Research Directions

Several directions emerge naturally from these findings. Multimodal capabilities—generating animated sprites with breathing cycles

or blinking directly from static outputs—represent the most immediately compelling extension, and advances in video diffusion make this technically plausible in the near term.

Reducing inference time is the most pressing practical need. At current latency levels, the system works well for production but imperfectly for prototyping. Bringing generation time under one minute would meaningfully change how developers interact with the tool. Three optimization pathways merit focused investigation. Model distillation compresses the trained model into a smaller, faster version with minimal quality loss—a technique that has demonstrated strong results in NLP contexts and is increasingly being applied to image generation. Quantization reduces numerical precision (typically from 32-bit to 8-bit or 4-bit floating point representations) to accelerate inference on both cloud and local hardware without proportional losses in output quality. Inference acceleration frameworks—tools like TensorRT or ONNX Runtime—optimize computational graphs for specific hardware profiles, often yielding substantial throughput improvements without architectural changes. Pursuing these strategies in combination, rather than sequentially, is likely to produce the greatest latency reductions and bring the system within reach of the sub-one-minute target.

Expanding the training dataset to cover a broader range of visual styles would extend the system's utility beyond standard anime aesthetics—an important step toward serving the full diversity of visual novel sub-genres. Large-scale beta testing with more varied user groups would also surface edge cases that the current 50-person evaluation couldn't reach, further validating the system's robustness across different development contexts.

Conclusion

What this research produced and validated is a domain-specific AI sprite generator that doesn't just generate attractive images—it maintains the character identity that visual novel storytelling requires. The fine-tuned LDM approach works. Generating up to nine emotional variants from a single prompt while keeping hair color, eye shape, and clothing consistent across the set isn't something general-purpose tools can reliably do, and demonstrating that it's achievable through domain-specific fine-tuning is the study's core empirical contribution.

User evaluation results confirm high satisfaction across most ISO/IEC 25010 quality characteristics. Performance Efficiency is the exception, and it's an honest one: the 2–3 minute latency is a real limitation, not a minor inconvenience. Future work addressing this through distillation and quantization will make the system genuinely usable for rapid prototyping workflows, not just final production. That's an engineering problem, not a conceptual one—and the concept has been validated.

What we find genuinely encouraging is the qualitative feedback. Participants described workflows that actually changed and frustrations that were actually resolved. Identity drift isn't an academic problem; it's why developers who try general AI tools often give up on them. A system that addresses it directly doesn't just improve a metric—it changes whether a certain group of creators can make the work they want to make. Independent developers, students, and small teams without dedicated artists are the ones who benefit most, and they're exactly who this tool was designed for.

The technical foundation is solid, the practical utility is demonstrated, and the path forward—toward faster inference, animated outputs, and broader stylistic range—is clearer now than it was at the outset. Technical and artistic limitations should not silence unique creative voices. With tools like this, increasingly they don't have to.

Software Demonstration Link: https://youtu.be/OuDCaiCo3ug

References

Duan, H., & Zhang, J. (2022). AI-driven environmental concept design: A framework for adaptive visual stimuli in interactive storytelling. *Journal of Digital Media Arts*, 14(2), 112-128.

Ergen, T. (2023). Procedural generation of video game characters using StyleGAN2: techniques and applications. *International Journal of Computer Games Technology*, 2023, Article 451239.

Gonzalez, M. (2020). The artificial art director: AI as a creative partner in game development. *Game Studies*, 20(1). http://gamestudies.org/2001/articles/gonzalez

Ma, X., Li, Y., & Chen, S. (2022). AI Illustrator: A cross-modal framework for narrative visualization. *IEEE Transactions on Multimedia*, 24, 2890-2902.

Park, J., Kim, S., & Yoo, D. (2021). Enhancing digital literacy in education through AI-assisted storytelling tools. *Computers & Education*, 168, 104201.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., & Chen, M. (2022). Hierarchical text-conditional image generation with CLIP latents. *arXiv preprint arXiv:2204.06125*.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 10684-10695).

Shi, Y., Wang, X., & Liu, Z. (2020). PokerFaceGAN: Generating neutral face templates for game character customization. *ACM Transactions on Graphics*, 39(4), 89:1-89:12.

Tanaka, H., & Kobayashi, S. (2021). Automating game asset creation: A survey of AI-based approaches. *Entertainment Computing*, 38, 100412.

Wu, L., Zhang, Y., & Wang, K. (2023). The impact of generative AI on video game visualization workflows. *Journal of Graphics Tools*, 27(1), 15-29.

Xie, Y. (2024). Consistency challenges in AI-generated character art for serial narratives. *Digital Creativity*, 35(1), 45-60.

Zhao, L., & Zhang, W. (2024). From GANs to Diffusion: A comparative analysis of text-to-image models in digital art. *AI & Society*, 39, 215-230.