DAYANANDA SAGAR UNIVERSITY

Devarakaggalahalli, Harohalli
Kanakapura Road, Ramanagara - 562112, Karnataka, India

Bachelor of Technology
in
COMPUTER SCIENCE AND ENGINEERING

Major Project Phase-II Report

VIRTUAL STORYBOARDING AND SCENE GENERATION

Batch: 14

By
Disha K Nanjunda - ENG21CS0120
Diya Sujil - ENG21CS0125
Harish Sasikumar - ENG21CS0147
Harsh Jolania - ENG21CS0148

Under the supervision of
Prof. Shilpa Sudheendran
Assistant Professor, Dept. of CSE

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING,
SCHOOL OF ENGINEERING
DAYANANDA SAGAR UNIVERSITY

(2024-2025)

DAYANANDA SAGAR UNIVERSITY

School of Engineering
Department of Computer Science & Engineering
Devarakaggalahalli, Harohalli, Kanakapura Road, Ramanagara - 562112
Karnataka, India

CERTIFICATE

This is to certify that the Phase-II project work titled **"VIRTUAL STORYBOARDING AND SCENE GENERATION"** is carried out by **Disha K Nanjunda(ENG21CS0120), Diya Sujil (ENG21CS0125), Harish Sasikumar(ENG21CS0147), Harsh Jolania(ENG21CS0148)** bonafide students of Bachelor of Technology in Computer Science and Engineering at the School of Engineering, Dayananda Sagar University, Bangalore in partial fulfillment for the award of degree in Bachelor of Technology in Computer Science and Engineering, during the year **2024-2025**.

Prof. Shilpa Sudheendran	**Dr. Girisha G S**	**Dr. Udaya Kumar Reddy K R**
Assistant Professor Dept. of CS&E, School of Engineering Dayananda Sagar University	Chairman CSE School of Engineering Dayananda Sagar University	Dean School of Engineering Dayananda Sagar University
Date:	Date:	Date:

Name of the Examiner **Signature of Examiner**

1.

2.

DECLARATION

We, **Disha K Nanjunda (ENG21CS0120), Diya Sujil (ENG21CS0125), Harish Sasikumar (ENG21CS0147), Harsh Jolania (ENG21CS0148)** are students of eighth semester B. Tech in **Computer Science and Engineering**, at School of Engineering, **Dayananda Sagar University**, hereby declare that the Major Project Stage-II titled **"VIRTUAL STORYBOARDING AND SCENE GENERATION"** has been carried out by us and submitted in partial fulfilment for the award of degree in **Bachelor of Technology in Computer Science and Engineering** during the academic year **2024-2025**.

Student **Signature**

Name 1: Disha K Nanjunda
USN : ENG21CS0120

Name 2 : Diya Sujil
USN : ENG21CS0125

Name 3: Harish Sasikumar
USN : ENG21CS0147

Name 4: Harsh Jolania
USN : ENG21CS0148

Place : Bangalore
Date :

ACKNOWLEDGEMENT

It is a great pleasure for us to acknowledge the assistance and support of many individuals who have been responsible for the successful completion of this project work.

First, we take this opportunity to express our sincere gratitude to School of Engineering & Technology, Dayananda Sagar University for providing us with a great opportunity to pursue our Bachelor's degree in this institution.

We would like to thank ***Dr. Udaya Kumar Reddy K R, Dean, School of Engineering, Dayananda Sagar University*** *for his constant encouragement and expert advice.*

It is a matter of immense pleasure to express our sincere thanks to ***Dr. Girisha G S, Department Chairman****,* ***Computer Science and Engineering****,* ***Dayananda Sagar University,*** *for providing the right academic guidance that made our task possible.*

We would like to thank my guide ***Prof. Shilpa Sudheendran, Assistant Professor****,* ***Dept. of Computer Science and Engineering****,* ***Dayananda Sagar University****, for sparing her valuable time to help us with every step of our project work. This helped pave the way for smooth progress and fruitful culmination of the project.*

We also thank my ***Project Coordinators: Dr. Meenakshi Malhotra, Dr. Kumar Dilip, and Dr. Sivananda Reddy E,*** *and all the staff members of Computer Science and Engineering for their support.*

We are also grateful to our family and friends who provided us with every requirement throughout the course.

We would like to thank one and all who directly or indirectly helped us in the Project work.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
Gen AI	Generative AI
LLM	Large language models
NLP	Natural Language Processing
GPT	Generative Pre-Trained Transformer
GAN	Generative Adversarial Network
CLIP	Contrastive Language-Image Pre-training.
NER	Named Entity Recognition
SSD	Solid State Drive

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Virtual Storyboarding and Scene Generation addresses the challenges of storyboarding, a vital yet time-consuming process in visual storytelling that heavily relies on artistic expertise. Independent creators, educators, and small teams often face barriers in producing high-quality storyboards due to limited resources or skills, restricting their ability to visualize and communicate creative ideas effectively. Existing research highlights the potential of generative AI in automating visual content creation but shows limited application in the sequential art of storyboarding, leaving gaps in accessibility and efficiency.

This project aims to investigate and evaluate the use of diffusion-based text-to-image models to automate storyboard generation from textual descriptions. The objective is to test the effectiveness of AI in reducing time and skill requirements while maintaining creative flexibility for users.

To achieve this, the research will focus on training and fine-tuning generative models on domain-specific datasets, integrating text processing and visual rendering tools, and designing an interactive system for user-friendly customization. This approach seeks to analyze how effectively the system can overcome traditional barriers and transform the storyboarding process, making it accessible to a broader audience.

CHAPTER 1

INTRODUCTION

CHAPTER 1
INTRODUCTION

1.1 INTRODUCTION

Virtual Storyboarding and Scene Generation represent a cutting-edge approach to visual storytelling, utilizing digital tools and technologies to create detailed and interactive storyboards. These innovative methods are highly valuable for filmmakers, educators, game developers, and content creators, as it allows them to visualize scenes and narratives before the actual production process begins. By creating sequences of images or frames that depict different scenes, creators can make necessary adjustments and refinements to ensure a cohesive story.

Our project, FrameWeaver, explores leveraging Generative AI to automate the storyboard creation process from textual descriptions, overcoming the skill and resource barriers of traditional storyboarding. Our primary focus is on optimizing AI to enhance storytelling by ensuring narrative flow, character consistency, and effective scene composition throughout the storyboard frames. By integrating these capabilities into a user-friendly tool, this Virtual Storyboarding and Scene Generation tool aims to make storyboard creation more accessible and efficient for filmmakers, animators, educators, and other creators. This innovation can empower users, regardless of their artistic skills or resources, to produce high-quality storyboards that effectively communicate their creative vision.

1.2 OBJECTIVE

The primary goal of this project is to develop an AI-powered system that automates the creation of storyboards from textual descriptions. The key objectives include:

- **Enhancing Narrative Flow**: By ensuring coherence in scene transitions and character interactions.
- **Optimizing Scene Composition**: By utilizing AI-generated visuals to maintain consistency in design and storytelling.

- **Reducing Production Time**: By automating the storyboard creation process to accelerate pre-production workflows.
- **User-Friendly Interface**: By developing an intuitive platform that allows creators to input textual descriptions and generate structured storyboard sequences effortlessly.
- **Improving Accessibility**: By providing a tool that simplifies storyboarding for users with limited artistic skills.

1.3 SCOPE

The scope of this project covers the following aspects:

- **AI-Driven Storyboard Generation**: Implementing Generative AI models capable of translating textual descriptions into illustrated frames.
- **Character and Scene Consistency**: Ensuring visual elements remain consistent across storyboard frames to maintain storytelling integrity.
- **Target Audience**: Designed for filmmakers, animators, educators, and content creators who require a streamlined storyboard creation process.

This project seeks to revolutionize how storyboards are created, making the process more efficient, accurate, and accessible to a broad range of users.

CHAPTER 2

PROBLEM DEFINITION

CHAPTER 2
PROBLEM DEFINITION

Problem Statement

Storyboarding is a crucial step in visual storytelling, helping creators plan out scenes and sequences for projects like films, animations, marketing, and games. However, it's a challenging and time-consuming process that often requires artistic skills many people don't have. Independent creators, educators, and small teams might struggle to produce high-quality storyboards because they lack access to trained artists or professional tools. Even digital tools that exist today still require a lot of manual work and creativity, which can be overwhelming for beginners or people with limited time. This makes it hard for many great ideas to be visually expressed, limiting creativity and collaboration.

Solution

This project focuses on creating a generative text-to-storyboard interface that uses AI technologies like diffusion models and related tools. The idea is to let users describe scenes in text, and the system will automatically generate storyboards based on their input. This tool will reduce the need for artistic skills, making the process faster and more accessible for everyone. It will also include options for customizing the output, exporting the storyboard, and c making it useful for both beginners and professionals.

CHAPTER 3

LITERATURE SURVEY

CHAPTER 3
LITERATURE SURVEY

The intersection of NLP and generative AI has come with new possibilities in the areas of creative writing and creation of visual content. In this regard, in our case "FrameWeaver - Virtual Storyboarding and Scene Generation System" is representative of the integration of textual narration with computer-generated images to illustrate the narrative. An examination of related literature provides not only the status of existing improvements but also the deficiencies and prospects that our system intends to fill in.

Prior studies in Story Generation as shown in "StoryGenAI: An Automatic Genre-Keyword Based Story Generation" elaborate on how AI can assist in storytelling by automating the creation of narratives using keywords from specific genres [1]. This is reflected in our use of the GPT-4o model to enhance stories by demonstrating how appropriate inputs can lead to improved narratives. Similarly, techniques focused on user guidance in creating interactive content [2] align with one of the primary objectives of our system design.

The current technical improvements and scalability of generative models [3] correspond well to our goal of using the gpt-image-1 model in a resource-efficient manner for the creation of comic style imagery. Additionally, research on how latent models improve visuals in a stylistic, cultural sense [4] further supports our storyboard generator's enhancement goals.

The transformation of text into images in "Script-to-Storyboard-to-Story Reel Framework"[5] gives us more insight on our system's implementation of spaCy as a tool for key element detection and scene analysis. The role of prompt engineering in ensuring high-quality and consistent outputs [6] is also a vital part of our framework to guarantee the achievement of reproduced images of high standard and consistent quality.

Research on the relevance of user experience in AI storytelling suggests the importance of automated grid-based storyboard assembly and PDF export functionality [7]. Similarly, the distribution of creative tools like "Artistic Fusion: AI Powered Artistry for Story Boarding" which meet our guidelines for reach and effectiveness on different platforms [8] align with our goals for accessibility and effectiveness. Studies like “Storytelling in the Metaverse: From Desktop to Immersive Virtual Reality Storyboarding,” on integrating storytelling into immersive environments [9] also suggest extensions for our system.

The educational and professional applications of such AI tools are explored in works focusing on enhancing users’ storytelling skills [10], while others demonstrated the importance of designing user-facing platforms with reliability and user needs in mind [11]. At the same time, research into interactive storytelling frameworks [9] also emphasizes the potential for improving scalability and performance in AI-driven creative systems.

Despite these advancements, there are gaps in existing AI storytelling tools. Manual storyboarding tools, while flexible, require significant expertise and lack automation, which limits accessibility. Text-to-image generators like DALL-E and Stable Diffusion fail to ensure continuity across scenes or articulate multi-scene narratives effectively.The ability of generative AI to produce semantically coherent visual outputs continues to improve, as demonstrated by recent advancements in text-to-image diffusion models [12]. This aligns with our system’s goal of maintaining scene consistency in multi-panel storyboards. The integration of NLP into generative frameworks [13] further enhances text-to-image accuracy.

Research on object-level consistency and compositional accuracy in text and visual outputs [14] support our aims to enhance narrative coherence of the system. These principles also emphasize the relationship between textual descriptions and image generation, which is critical for our multi-scene storytelling approach.

We explored studies on prompt engineering challenges to address the need for interactive methods for improving text-to-image outputs [15]. These insights guided

our efforts to design prompts that include sufficient contextual and factual information to create meaningful and stylistically uniform storyboards tailored to user needs.

User experience and iterative enhancements, essential for generative AI systems, are highlighted in works focusing on enabling users to refine their creations through feedback loops [16, 17]. These approaches foster collaborative content creation that is more meaningful, engaging, and suited to its intended purpose and audience.

These studies place our system at the intersection of innovative generative approaches to narrative design. By addressing the text-to-image gap in features, our project aims to advance existing theories of visual storytelling and push the boundaries of creative possibilities in sequential narrative applications.

CHAPTER 4

PROJECT DESCRIPTION

CHAPTER 4
PROJECT DESCRIPTION

FrameWeaver - A Virtual Storyboarding and Scene Generation System utilizes advanced Natural Language Processing (NLP) and Generative AI to transform story outlines into comic-style storyboards. The system begins with user input, where a basic plot outline is provided. Users can enhance their story using the **OpenAI GPT-4o model** for a more polished and detailed narrative. Once the story is refined, it proceeds to the **Key Element Extraction** stage.

Using **spaCy's NLP capabilities**, the story is broken into scenes or sentences, focusing on structure and flow. These individual elements then move to the **image generation phase**, where the **gpt-image-1** model on Open AI generates visuals. The prompts incorporate artistic styles like hand-drawn sketches, futurism, and cultural nuances. To avoid an **API overload**, we employ an algorithm we devised where a 60-second cooldown is applied after every three scenes, for smooth performance.

The next step is **Storyboard Assembly** where the generated images are formatted in a grid layout with captions for clarity and audio narration. The final storyboard is saved as a PDF using the Pillow library and as a PowerPoint file using the python-pptx library, ensuring easy sharing and presentation.

This system focuses on functionality, usability, and performance, enabling creators to visualize ideas quickly. It is secure, scalable, and cross-platform compatible, making it a robust tool for creative professionals. By bridging textual storytelling and visual representation, this project showcases the potential of NLP and Generative AI in education ,entertainment and content creation.

4.1 SYSTEM DESIGN

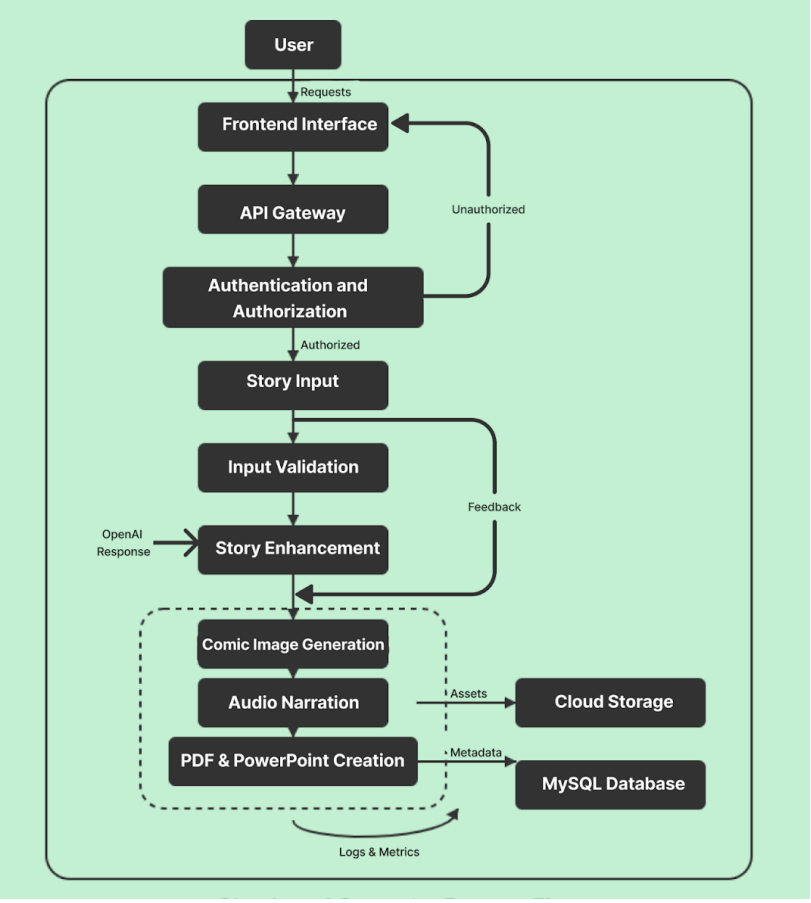

Fig. 4(a) Process Flow

The Storyboard Generator follows a structured process flow as illustrated in the diagram above. This section details each component of the system architecture and their interactions.

1. User and Frontend Interface

- **User Interaction**: Users access the system through a web-based interface to submit story inputs and retrieve generated storyboards.
- **Frontend Interface**: Provides an intuitive user experience with forms for story submission and displays for viewing results.

2. API Gateway

- **Request Management**: Routes all incoming requests to appropriate backend services.
- **Load Balancing**: Distributes incoming traffic to ensure system stability during peak usage.

3. Authentication and Authorization

- **User Verification**: Validates user credentials before allowing access to the system.
- **Token-Based Security**: Implements token mechanisms for secure sessions.
- **Unauthorized Request Handling**: Redirects unauthorized requests back to the frontend with appropriate error messages.

4. Story Input

- **Data Capture**: Collects the user's story outline in a structured format.
- **Temporary Storage**: Holds the input data in memory for immediate processing.
- **Input History**: Maintains a record of previous submissions for returning users.

5. Input Validation

- **Format Checking**: Ensures the input adheres to expected structure and format.
- **Content Validation**: Verifies that all required story elements are present.

6. Story Enhancement

- **GPT-4o Integration**: Connects with OpenAI's API to process and enhance the story.

- **Narrative Enrichment**: Expands basic story outlines into detailed scene descriptions.
- **Visual Cue Addition**: Incorporates specific details that will improve image generation quality.
- **Scene Segmentation**: Divides the enhanced story into distinct scenes for processing.
- **Template Application**: Uses the format:

  [Scene Setting] with [Key Environmental Detail]. [Main Character], described with [Clothing, Features, Expression], stands/acts [Action/Emotion]. [Background Elements or Cultural Symbols] add context. [Lighting Style], [Camera Angle], cinematic, detailed for optimal visualization.

7. Media Generation

- **Comic Image Generation**:
  - Translates textual scene descriptions into image generation prompts.
  - The **gpt-image-1** model produces visually consistent, comic-style illustrations based on user-provided prompts.
  - Implements rate limiting to manage API usage efficiently
  - Processes images for optimal quality and consistency.
  - Stores generated images with appropriate naming conventions.

- **Audio Narration**:
  - Converts scene text to speech using **GPT-4o mini TTS**.
  - Processes the audio generation asynchronously to maintain system responsiveness.
  - Applies appropriate pacing and intonation based on scene context.
  - Synchronizes audio with corresponding images.

- **PDF & PowerPoint Creation**:
  - Compiles images, text, and audio into cohesive presentation formats.

- Creates PDF storyboards with captioned images in a grid layout.
- Generates PowerPoint presentations with one scene per slide and optional narration.
- Structures content with appropriate layouts and transitions.

8. Data Storage

- **Cloud Storage**:
  - All generated media assets (such as images, audio, PDFs, and PPTXs) are securely stored using PythonAnywhere's Linux-based persistent file system.
  - Files are organized in a directory-based structure under `/home/username/project_path/`, categorized by user ID and project identifiers for efficient management. Access permissions are enforced using the Unix access control model, ensuring secure and structured content access.
  - Structured data like user details and story metadata are stored in MySQL database instances, providing persistent and reliable storage between application restarts and system updates.

- **MySQL Database**:
  - Records metadata including user IDs, timestamps, file paths, and generation parameters.
  - Maintains relationships between users, stories, and generated assets.
  - Supports auditing, version tracking, and efficient content retrieval.

9. Logging and Monitoring

- **Performance Metrics**: Tracks generation time per scene, API success/failure rates, and system performance.
- **Error Logging**: Records system errors and exceptions for troubleshooting.

- **Usage Analytics**: Collects data on feature usage and user behavior to guide improvements.

This architecture creates a seamless pipeline that transforms user story inputs into comprehensive multimedia storyboards, while maintaining data security and system reliability throughout the process.

4.2. ASSUMPTIONS AND DEPENDENCIES

4.2.1 Assumptions

- *User Inputs:*

  Users will provide a clear and concise story outline that can be enhanced or processed without ambiguity. If users choose to enhance the story, they will explicitly confirm their preference (e.g., typing "yes"). If users wish to maintain consistency across generated images, they will provide appropriate character descriptions and style preferences.

- *Model Performance:*

  The Open-AI GPT 4o model used for text enhancement is pre-trained and fine-tuned for generating concise, descriptive, and grammatically correct outputs. The **gpt-image-1** model for comic-style image generation produces visually consistent outputs based on the provided prompts. All models used are available and responsive.

- *System Resources:*

  The system will have adequate computational resources (RAM, GPU/CPU, and storage) to handle NLP processing, image generation, PDF, PPT creation seamlessly. The API rate limits will not exceed the frequency of requests made, as delays are implemented to avoid overload.

- *Frontend and Backend Integration:*

  The user interface built with Flask will function as intended, allowing users to easily input data, view results, and download outputs.

The backend logic (text enhancement, image generation, and PDF, PPTX creation) will work seamlessly with the frontend, ensuring a smooth user experience.

- *Dependencies Availability:*
  Required libraries and frameworks such as spaCy, PIL (Pillow), PPTX, FPDF, and Open AI Inference Client are correctly installed and functioning.
- *Internet Connectivity:*

  A stable internet connection is available to access the Open AI models for text enhancement and image generation.
- *File Management:*

  The output directory for storing generated images and PDFs exists, and the system has permission to read/write files.

4.2.2 Dependencies:

- *Language Models:*
  Dependency on OpenAI GPT-4o model for text enhancement, requiring proper API configuration and availability on the Open AI platform.
- *Image Generation Model:*
  Dependency on **gpt-image-1** accessed via Open AI to generate high-quality comic-style images based on prompts.
- *NLP Framework:*
  Dependency on spaCy for natural language processing tasks such as sentence segmentation and grammar correction.
- *Frontend Framework:*
  Dependency on Flask for building the user interface and ensuring smooth interaction between the frontend and backend.
- *External APIs:*
  Dependency on Open-Ai API for communication with text gpt-4o and image generation models gpt-image-1 and gpt-4o-mini-tts API for generating narrated story audio, requiring valid API keys and adherence to usage limits.
- *Libraries and Tools:*

Use of Python libraries such as Pillow for image manipulation and FPDF for generating PDF files.

- *Hardware Requirements:*

  Dependency on system hardware to handle computationally intensive tasks like NLP, image generation, and file processing.

- *Environment Configuration:*

  Dependency on a properly configured Python environment with all necessary dependencies installed (e.g., via pip).

- *User Interaction:*

  Dependency on users to provide accurate inputs and follow instructions for enhancing stories and generating outputs.

CHAPTER 5

REQUIREMENTS

CHAPTER 5
REQUIREMENTS

5.1 FUNCTIONAL REQUIREMENTS

1. *User Input Parsing*
    - The system should accept a story outline from the user.
    - It should provide an option to enhance the story using an LLM.
    - The system should ensure the text is converted to lowercase and grammar is checked for corrections.
2. *Story Enhancement*
    - The system must use the **OpenAI GPT-4o model** model to enhance the story with clear, concise, and descriptive sentences within the specified limits.
    - Parameters like temperature, max tokens, top-p, and streaming must be pre-configured to generate accurate enhancements.
3. *Key Element Extraction*
    - The system should extract individual sentences or scenes from the enhanced story using NLP for modular processing.
    - It should ensure grammatical accuracy before extracting key elements.
4. *Comic-Style Image Generation*
    - The system must use the **gpt-image-1** model to generate visually appealing comic-style images for each scene in the story.
    - Each scene should be processed with a specific prompt, saved sequentially, and follow a consistent art style.
    - The system should implement a timer of 60 seconds after every three API calls to avoid overloading.
5. *Storyboard Assembly*
    - The system should organize the generated images into a grid layout and add captions corresponding to each scene.
    - The final storyboard must be saved as a PDF file for easy sharing and storage.

5.2 NON-FUNCTIONAL REQUIREMENTS

1. *Performance:*
    - Process text enhancement and image generation within reasonable response times.
    - Support up to N scenes without significant delays.
2. *Scalability:*
    - Handle multiple users or concurrent requests without degradation.
    - Ensure the system can scale up for larger stories and higher-resolution images.
3. *Reliability*:
    - Ensure the system operates without failures under normal usage conditions.
    - Retry failed API calls or provide fallback mechanisms.
4. *Usability:*
    - Provide a user-friendly interface for input and output.
    - Ensure the system is intuitive for both technical and non-technical users.
5. *Security:*
    - Use secure API keys for accessing text and image generation models.
    - Protect user-provided data and generated outputs from unauthorized access.
6. *Portability:*
    - Ensure the system can run on multiple platforms (e.g., Windows, macOS, and Linux).
7. *Maintainability:*
    - Use modular code for easy debugging, updates, and integration with new models.
    - Provide clear documentation for developers.
8. *Compliance:*
    - Adhere to licensing terms of external libraries, APIs, and models used.

9. *Real-Time Processing:*
    - Enable streaming for tasks like text enhancement to show results incrementally.
10. *Storage Management:*
    - Optimize storage for images and PDFs.
    - Allow users to delete old files to free up space.

5.3 SOFTWARE REQUIREMENTS

1. *Programming Language:* Python 3.8 or above.
2. *Libraries/Frameworks:*
    - spaCy (for NLP tasks).
    - Pillow (for image processing).
    - FPDF (for PDF generation).
    - gpt-4o-mini-tts(for audio generation)
    - pptx (for PPT generation)
3. *Pre-trained Models:*
    - Open-AI GPT-4o (for text enhancement).
    - gpt-image-1 (for image generation).
4. *Operating System:*
    - Windows, macOS, or Linux.
5. *Cloud Platforms:*
    - The application is hosted on PythonAnywhere, a Platform-as-a-Service (PaaS) cloud environment specialized for Python-based applications. Built on top of Amazon Web Services (AWS) infrastructure, it provides a fully managed deployment model with integrated support for MySQL databases.
6. *Text Editor/IDE:*
    - Visual Studio Code, PyCharm, or Jupyter Notebook.
7. *Dependency Management:*
    - pip or Conda for package installations

8. *Browser:*
    - For accessing hosted storyboards.
9. *Database:*
    - MySQL for data storage.

5.4 HARDWARE REQUIREMENTS

1. *Processor:*
    - Intel i5 or equivalent (minimum), i7 or higher (recommended).
2. *RAM:*
    - Minimum: 8 GB.
    - Recommended: 16 GB (for smoother handling of large models and images).
3. *Storage:*
    - Minimum: 20 GB free space.
    - Recommended: SSD for faster processing and storage of images and PDFs.
4. *GPU:*
    - Recommended: NVIDIA GPU (e.g., GTX 1660 or higher) for faster model inference.
5. *Internet:*
    - Stable broadband connection (minimum 10 Mbps) for API calls and cloud interactions.
6. *Display:*
    - Minimum 1080p resolution for better UI and storyboard visualization.

CHAPTER 6

METHODOLOGY

CHAPTER 6
METHODOLOGY

6.1 MODEL SELECTION AND CONFIGURATION

6.1.1 Text Enhancement Model Selection

The initial phase of our methodology involved selecting the most appropriate model for enhancing user-provided story outlines. After conducting comparative analysis of various large language models, we chose OpenAI's GPT-4o based on its superior capabilities in:

- Maintaining narrative consistency across extended text passages
- Generating creative yet coherent expansions from pivotal plot points
- Preserving tone and stylistic elements while enhancing the content
- Processing complex story structures with minimal context

Based on our evaluation, we concluded that GPT-4o provided significantly better results for narrative enhancement tasks compared to its alternatives, particularly in its ability to maintain the original story's intent while adding depth and detail.

6.1.2 Text Enhancement Model Configuration

To optimize the text enhancement process, we calibrated the following parameters through systematic testing:

- **Role Configuration**: Specified custom system prompts to position the model as a creative writing assistant
- **Temperature**: Set to 0.5 (determined through testing across the 0.1 -0.5 range) to balance predictability with creative variation
- **Max Tokens**: 2048 tokens to allow sufficient detail while preventing excessive verbosity
- **Top P (Nucleus Sampling)**: 1 based on output quality evaluation
- **Response Format**: Structured to facilitate downstream processing

These parameters were finalized through iterative testing with diverse story inputs, to identify the optimal balance between creative enhancement and preservation of the original narrative's core elements.

6.1.3 Image Generation Model Selection

For visualizing the story scenes, we selected Open AI's **gpt-image-1** model following comprehensive comparative testing against alternatives including Stable Diffusion models. Our selection criteria prioritized:

- **Prompt Adherence**: Accuracy in translating textual scene descriptions to visuals
- **Style Consistency**: Ability to maintain uniform character appearances across multiple scenes
- **Processing Efficiency**: Generation speed appropriate for multi-scene storyboards
- **Comic Aesthetic**: Proficiency in producing illustrations with comic-book styling

The **gpt-image-1** model performed best during controlled testing with standardized scene descriptions from various narrative genres.

6.1.4 Image Generation Configuration

Our image generation pipeline was configured with the following specifications:

- **Base Resolution**: 1024×1024 pixels for optimal detail retention
- **Negative Prompts**: Custom prompts engineered to avoid common visual artifacts
- **Style Prefixes**: " comic-book style" prefix standardized across all generations
- **Character Consistency Parameters**: Custom settings to maintain visual continuity of characters

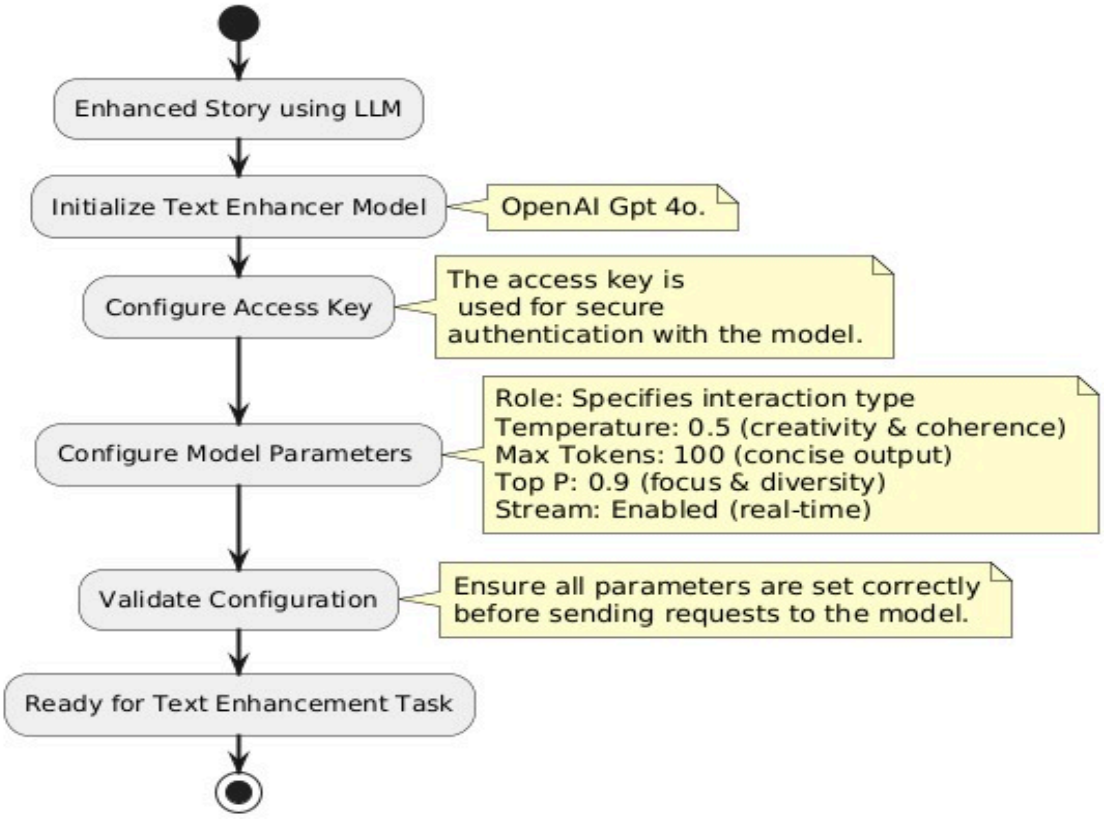

Fig 6(a) Enhancing Story using LLMs

6.2 TEXT PROCESSING

6.2.1 Input Standardization

To ensure consistent processing, all user-provided text undergoes a standardization process:

- **Case Normalization**: Converting text to consistent lower case format
- **Special Character Handling**: Standardizing quotation marks, dashes, and other typographic elements
- **Structural Formatting**: Normalizing paragraph breaks and sentence structures

This preprocessing ensures that subsequent AI processing receives consistently formatted input, reducing variability in outputs.

6.2.2 Grammatical Analysis and Correction

We implemented a two-stage approach to grammatical processing:

1. Initial assessment using rule-based grammar checking algorithms
2. Contextual correction using the GPT-4o model with specific grammar correction instructions

This step ensures the input is clean and error-free, which improves the quality of the enhanced output.

6.2.3 Conditional Enhancement Logic

Our enhancement process follows a decision-tree methodology:

1. User selects enhancement preference (yes/no)
2. If enhancement is selected:
    - Input text is analyzed for key themes and narrative elements
    - GPT-4o generates enhanced narrative with scene delineation
    - Output is validated for length constraints and narrative continuity
3. If enhancement is declined:
    - Original text undergoes minimal processing for standardization
    - Scene breaks are algorithmically determined based on paragraph structure

This approach respects user preferences while ensuring consistent downstream processing regardless of enhancement choice.

6.3 KEY ELEMENT EXTRACTION

We developed a structured approach to element extraction using natural language processing techniques:

- **Scene Segmentation**: Detecting explicit scene markers or inferring boundaries using semantic analysis
- **Character Identification**: Using Named Entity Recognition (NER) to extract characters
- **Setting Extraction**: Identifying location entities and contextual cues to determine settings
- **Timeline Mapping**: Analyzing temporal markers to establish narrative chronology

We extended the spaCy NLP pipeline with custom components to improve detection of narrative-specific elements, increasing extraction accuracy compared to standard NER approaches.

6.4 IMAGE AND AUDIO GENERATION

6.4.1 Prompt Engineering Framework

We developed a systematic prompt engineering methodology:

- **Base Template**: "[Style] + [Scene Description] + [Character Details]"
- **Style Consistency**: Using consistent artistic direction terms across all prompts
- **Character Description Injection**: Including identifying details to maintain visual consistency
- **Contextual Enrichment**: Adding environment and mood indicators based on narrative analysis

This structured approach to prompt creation resulted in better scene-to-image relevance compared to using raw scene text alone.

6.4.2 Sequential Image Generation Pipeline

Our image generation process follows a sequential workflow:

1. Process scene text to identify key visual elements

2. Construct optimized prompt using the engineering framework
3. Generate initial image with **gpt-image-1** model
4. Evaluate image-to-scene correspondence
5. Regenerate with refined prompts if needed

6.4.3 Rate Limiting and Resource Management

To ensure system stability and API compliance, we implemented:

- Dynamic rate limiting based on API response times
- Batch processing with 60-second pauses after every three images
- Concurrent request management to optimize throughput
- Error handling with exponential backoff for failed generations

6.4.4 Character Consistency

6.4.4.1 Character Style Definition

We implemented a systematic approach to maintaining character visual consistency:

- **Character Attribute Extraction**: For each identified character:
    - Physical characteristics are cataloged (height, build, facial features)
    - Clothing and distinguishing accessories are documented
    - Emotional expressions and typical poses are defined
- **Style Sheet Creation**: A formal character style definition is created containing:
    - Detailed textual descriptions for prompt inclusion
    - Key visual traits to maintain across scenes
    - Character relationships and relative scaling
- **Consistency Parameter Control**: A variable consistency level (0.0-1.0) determines:
    - How strictly character descriptions are enforced across scenes

- Balance between scene-specific requirements and character consistency
- Weight given to character attributes in the prompt engineering

6.4.4.2 Character-Aware Prompt Construction

Character descriptions are integrated into image generation prompts using:

- **Contextual Relevance**: Only including character details relevant to the current scene
- **Prominence Weighting**: Adjusting emphasis based on character's importance in the scene
- **Relationship Positioning**: Including spatial relationship cues between characters
- **Progressive Refinement**: Adding more character detail in subsequent scenes to reinforce visual identity

6.4.4.3 Visual Consistency Validation

Our evaluation methodology for character consistency includes:

- **Feature Comparison**: Automated detection of key character features across scenes
- **Visual Similarity Scoring**: Calculating consistency scores between character appearances
- **Deviation Thresholds**: Defining acceptable variance in character representation
- **Regeneration Triggers**: Establishing criteria for when to regenerate images with refined prompts

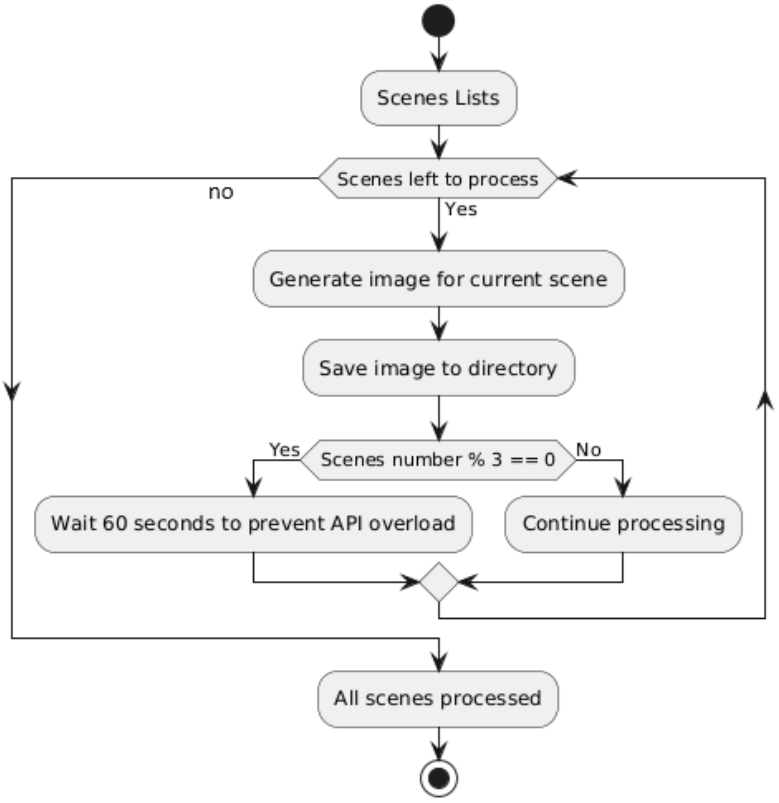

Fig 6(b) Working of Scene Generation

6.4.5 Audio Narration

6.4.5.1 Text-to-Speech Model Selection

After evaluating multiple text-to-speech solutions, gpt-4o-mini-tts was selected based on:

- Superior "voice naturalness" compared to alternatives in blind listening tests
- Multi-language support enabling future internationalization
- Consistent pronunciation of complex words and proper nouns
- Efficient processing suitable for multi-scene narratives
- Reliable API performance with minimal latency

6.4.5.2 Audio Segmentation Approach

We implemented a scene-based audio segmentation methodology:

- **Textual Preprocessing**:
    - Punctuation Normalization to improve speech pacing
    - Phonetic spelling adjustments for problematic words
    - Speech markup for emphasis and pacing
- **Per-Scene Processing**:
    - Audio generated independently for each scene rather than as a single file
    - Enables precise synchronization with visual elements
    - Allows scene-specific voice characteristic adjustments
- **Parameterization**:
    - Language selection defaulting to 'en' (English)
    - Speech rate calibration for optimal comprehension
    - Consistent voice selection across scenes

6.4.5.3 Audio Post-Processing

A systematic approach to audio refinement was developed:

- **Normalization**: Audio levels are normalized across all scene segments
- **Silence Handling**: Leading and trailing silences are trimmed to precise durations
- **Format Standardization**: All audio is converted to MP3 format with consistent bitrate
- **Quality Assurance**: Automated checks for audio clarity and completeness

6.5 STORYBOARD ASSEMBLY

6.5.1 Visual-Textual Integration

Our structured approach to storyboard assembly includes:

- **Image-Text Alignment**: Pairing generated images with corresponding scene text

- **Layout Optimization**: Determining optimal image-text placement for readability
- **Visual Flow Analysis**: Ensuring natural progression through the narrative
- **Typography Selection**: Choosing fonts and text styling for optimal readability

6.5.2 Document Format Standardization

We standardized multi-format output through:

- PDF generation with embedded text for accessibility
- PowerPoint creation with synchronized audio-visual elements
- Image sequence export with metadata for further editing

Each format underwent quality assurance testing across different viewing environments to ensure consistent presentation.

6.6 CLOUD INFRASTRUCTURE

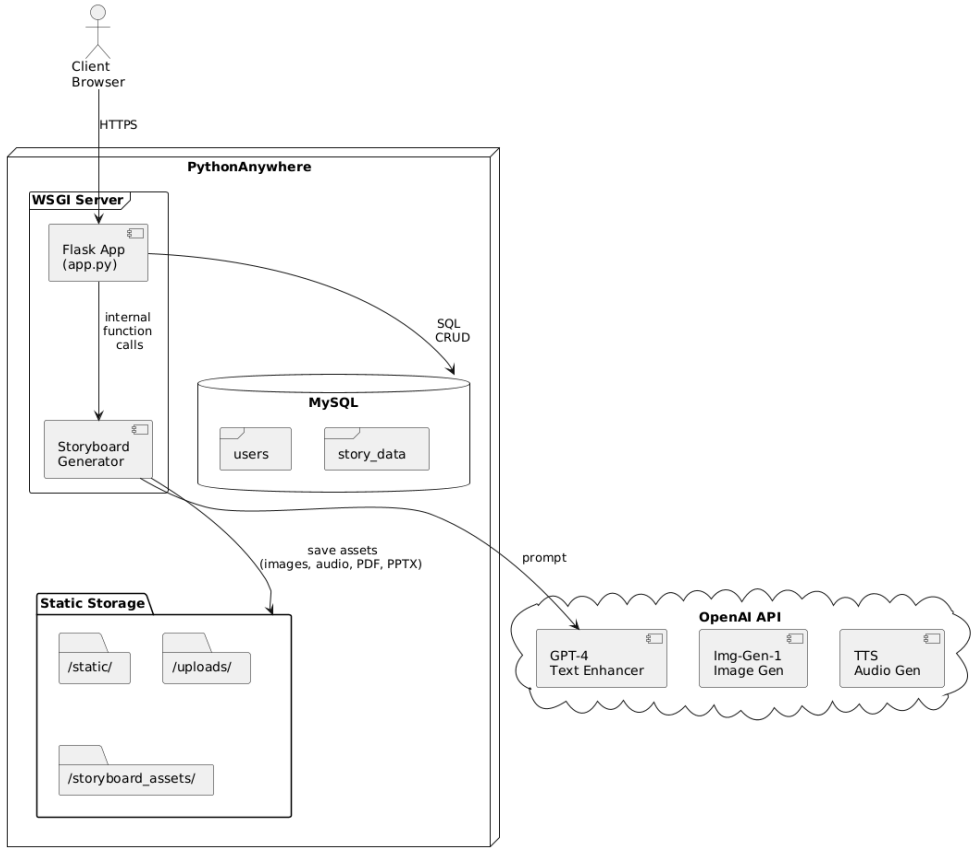

Fig 6(c) FrameWeaver Deployment Architecture on PythonAnywhere

6.6.1 Scalable Resource Allocation

- PythonAnywhere provides scalable resource management through account-based tiers such as Free, Hacker, and Web Developer, each offering increasing levels of CPU time, memory allocation, and storage.
- This enables efficient scaling of computing resources based on project demands.
- Additional storage and resource upgrades can be provisioned through account enhancements.
- Memory usage is controlled to prevent resource overconsumption, and persistent file storage ensures continuity across application sessions.

6.6.2 Compute Resources

PythonAnywhere supports continuous compute availability using always-on web workers that handle HTTP requests and background task queues. These queues are used for deferred processing tasks such as:

- **AI Model Inference:** Executing inference workloads like GPT-based scene processing using scheduled tasks.
- **Asset Generation:** Performing post-processing of images or other media.
- **Data Handling:** Processing user-generated scripts and story metadata asynchronously.

6.6.3 Security and Collaboration Framework

The platform ensures application security through built-in HTTPS encryption, Unix-based access controls for file-level security, and isolated MySQL database instances with credential-based access. Collaborative development is supported via a browser-based code editor, allowing remote team members to contribute in real time. Regular database backups further ensure data protection, while PythonAnywhere's simplified deployment and version control tools assist in managing updates collaboratively.

6.7 USER INTERACTION

6.7.1 Interface Design Approach

Our user interface was designed by Flask and Jinja2 framework, keeping in mind the following these principles:

- Progressive disclosure of complex options
- Real-time feedback during generation processes
- Visual previews at each stage of creation
- Consistent interaction patterns across all functions

6.7.2 User Flow Optimization

User experiences are mapped and optimized for:

- Minimal steps from input to completed storyboard
- Clear decision points with contextual guidance
- Recovery paths from common error conditions
- Seamless transitions between creation and editing modes

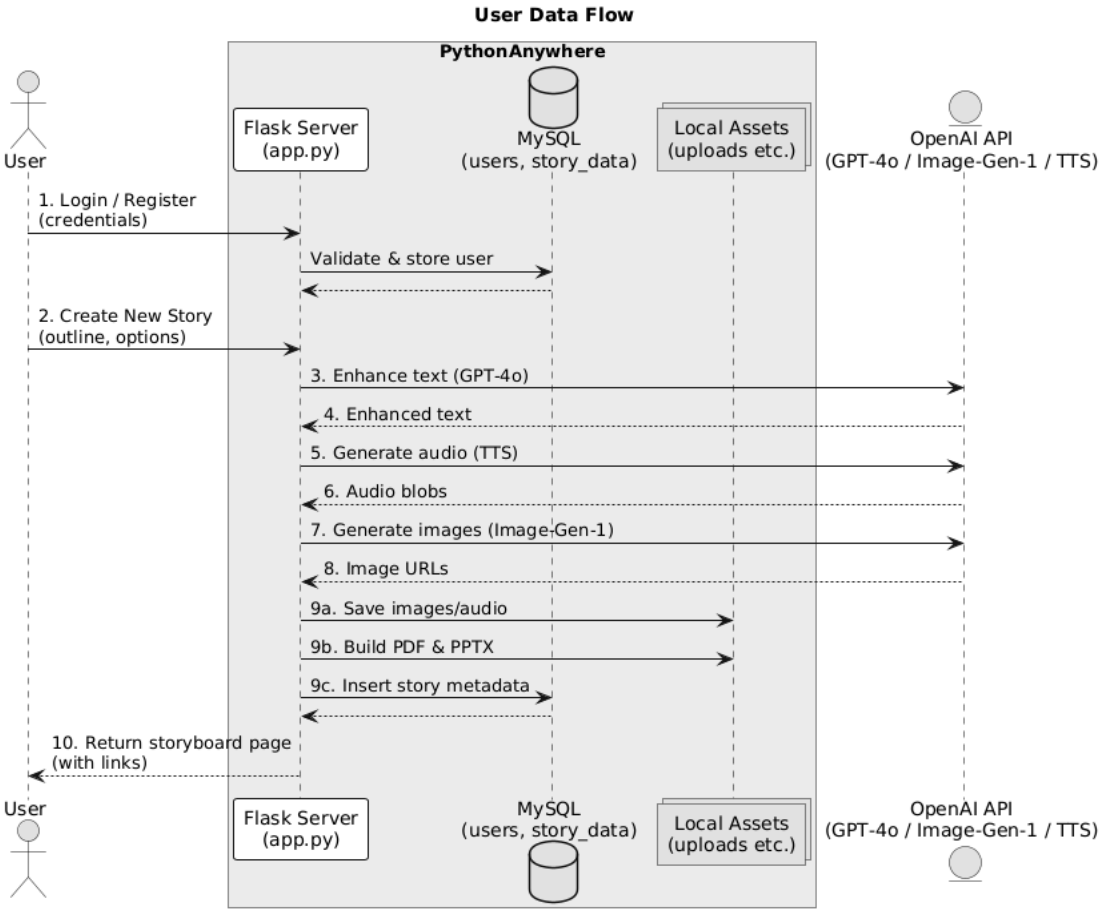

Fig 6(d) User Data Flow in FrameWeaver's System Architecture

CHAPTER 7

EXPERIMENTATION

CHAPTER 7

EXPERIMENTATION

7.1 EXPERIMENTAL SETUP AND TESTING

The storyboard generator was evaluated through systematic experiments we devised to assess both technical performance and output quality. Testing was conducted in a virtual environment local host with the following configuration:

- Ryzen 7-7400h Python 3.11 runtime
- Integration with GPT-4o and gpt-image-1 models
- MySQL database and local storage

7.2 CHARACTER CONSISTENCY ALGORITHM

The character consistency algorithm proved critical to maintaining visual continuity across scenes. This algorithm dynamically adjusts character descriptions based on a user-defined consistency level:

```
def generate_consistent_image_prompt(scene_text, character_styles,
consistency_level=0.8):
    character_descriptions = []
    scene_characters = extract_characters_from_scene(scene_text)

    for char_name in scene_characters:
        if char_name in character_styles:
            char_info = character_styles[char_name]
            traits = char_info.get("key_visual_traits", "")
            desc = char_info.get("physical_description", "")


            # Apply consistency weighting
            if consistency_level > 0.8:
```

```
            char_desc = f"{char_name}: {desc}. {traits} MUST appear exactly as
described."
        elif consistency_level > 0.5:
            char_desc = f"{char_name}: {desc}. {traits}"
        else:
            char_desc = f"{char_name} with {', '.join(traits.split()[:5])}"

        character_descriptions.append(char_desc)

    character_style_text = " ".join(character_descriptions)
    return f"Black and white comic style. {scene_text}\n\nCharacter appearances:
{character_style_text}"
```

7.3 AUDIO-VISUAL SYNCHRONIZATION ALGORITHM

Timing misalignment between images and audio narration in presentations initially posed a significant challenge. We developed a scene duration estimation algorithm that improved synchronization accuracy:

```
def calculate_scene_duration(scene_text):
    word_count = len(scene_text.split())
    base_duration = 3.0  # Base seconds

    # Reading pace (approximately 2 words per second)
    word_duration = word_count / 2.0

    # Adjust for text complexity
    complexity_factor = analyze_text_complexity(scene_text)
    complexity_duration = word_count * complexity_factor

    return base_duration + word_duration + complexity_duration
```

7.4 IMPLEMENTATION - CHALLENGES AND SOLUTIONS

7.4.1 API Rate Limiting

Problem: The Open AI gpt-image-1 API frequently returned 429 (Too Many Requests) errors during batch processing, interrupting storyboard generation.

Solution: We implemented an adaptive rate limiting system with exponential backoff:

```
def generate_with_retry(prompt, max_retries=5, initial_delay=1):
    delay = initial_delay
    for attempt in range(max_retries):
        try:
            return client.text_to_image(prompt)
        except Exception as e:
            if "429" in str(e) and attempt < max_retries - 1:
                logging.warning(f"Rate limited, retrying in {delay} seconds")
                time.sleep(delay)
                delay *= 3   # Exponential backoff
            else:
                raise
```

7.4.1 Scene Segmentation

Problem: Early testing revealed inadequate scene segmentation for unmarked scenes, resulting in disconnected narratives.

Solution: We developed a hybrid segmentation algorithm combining rule-based and NLP approaches:

```
def extract_key_elements(story_text):
    # Try explicit scene markers first
    if "scene" in story_text.lower():
        scenes = extract_explicit_scenes(story_text)
        if len(scenes) > 1:
            return scenes

    # Try paragraph breaks next
    paragraphs = [p.strip() for p in story_text.split("\n\n") if p.strip()]
```

```
if len(paragraphs) > 1:
    return paragraphs

# Use NLP to identify scene transitions
doc = nlp(story_text)
scenes = []
current_scene = []

for sent in doc.sents:
    current_scene.append(sent.text)
    if is_scene_transition(sent, doc):
        scenes.append(" ".join(current_scene))
        current_scene = []

if current_scene:
    scenes.append(" ".join(current_scene))

# Fall back to text chunking if needed
if len(scenes) <= 1 and len(story_text) > 500:
    return chunk_text(story_text)

return scenes if scenes else [story_text]
```

CHAPTER 8

TESTING AND RESULTS

CHAPTER 8
TESTING AND RESULTS

8.1 TESTING

The virtual storyboarding and scene generator underwent comprehensive testing to evaluate its functionality, performance, and output quality. Testing was conducted on a local development environment using a Ryzen 7 processor with 64GB RAM, running Python 3.11 with a local MySQL server instance. Multiple test scenarios were designed to assess different aspects of the system.

8.2 TEST CASES AND RESULTS

8.2.1 Test Case 1: Basic Story Enhancement

Objective: Verify the system's ability to enhance a basic story outline

Input:

A young wizard discovers a hidden spell book in his school's library. As he experiments with the spells, he accidentally releases a dangerous creature that begins terrorizing the school. 3 scenes for the story

Result: The system successfully enhanced the original story, adding descriptive elements and maintaining narrative coherence. The enhancement process took 1.3 seconds, and the output preserved all core plot elements while expanding the narrative.

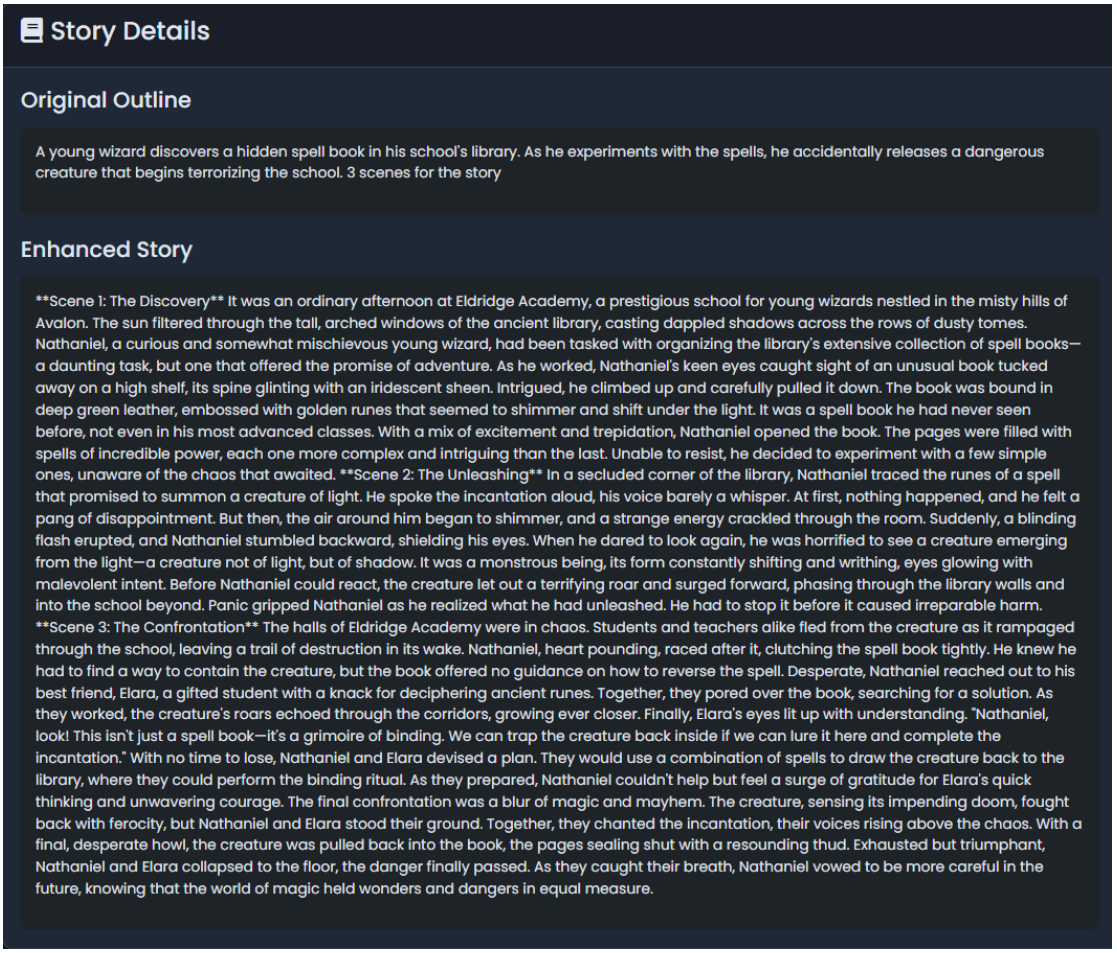

Fig 8.2(a) Input story outline and enhanced output with additional descriptive elements

8.2.2 Test Case 2: Character Consistency

Objective: Evaluate visual consistency of character appearance across multiple scenes

Input:

Scene 1: Sarah, a tall woman with curly hair, enters the abandoned building.
Scene 2: Inside, Sarah discovers a hidden passage behind a bookshelf.
Scene 3: Sarah descends the dark staircase, her red hair barely visible in the dim light.

Result: With consistency level set to 0.8, the system maintained the character's key visual traits (hair, height) across all three scenes. Minor variations in facial features were observed but did not impact character recognition.

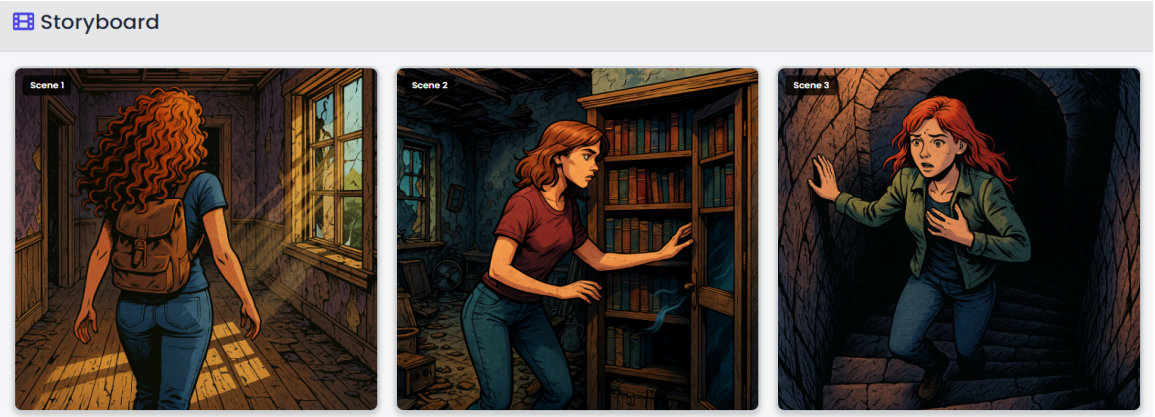

Fig. 8.2(b) Three generated images showing consistent representation of Sarah across scenes

8.2.3 Test Case 3: Complex Narrative Processing

Objective: Test system's ability to handle complex narratives with multiple characters

Input:

The diplomatic summit brought together representatives from three warring factions. General Thorne represented the Northern Alliance, his uniform adorned with medals. Ambassador Lyra from the Eastern Federation wore traditional blue robes. The Southern Republic sent Scholar Jensen, an elderly man with a mechanical arm.

Result: The system correctly identified all three characters and maintained their distinct visual attributes in the generated images. Processing time hikes up compared to single-character narratives.

Fig. 8.2(c) Generated image showing distinct representations of all three characters with their defining attributes

8.3 PERFORMANCE EVALUATION

To ensure comprehensive evaluation, we created a test corpus of story outlines spanning multiple genres (science fiction, fantasy, thriller, drama) with varying complexity levels.

8.3.1 Processing Time Analysis

Table 8.3.1 Average Processing Time by Story Length (Ryzen 7, 64GB RAM)

Story Length (words)	Text Enhancement (s)	Image Generation (s) *	Audio Creation (s)	Total Processing (s)
<100	0.9	40.8	1.5	43.2
100-299	1.3	51.2	2.9	55.4
300-499	1.8	97.0	4.6	103.4
500	2.9	117.1	5.6	125.6

*generation of each scene varies based on complexity and internet speed

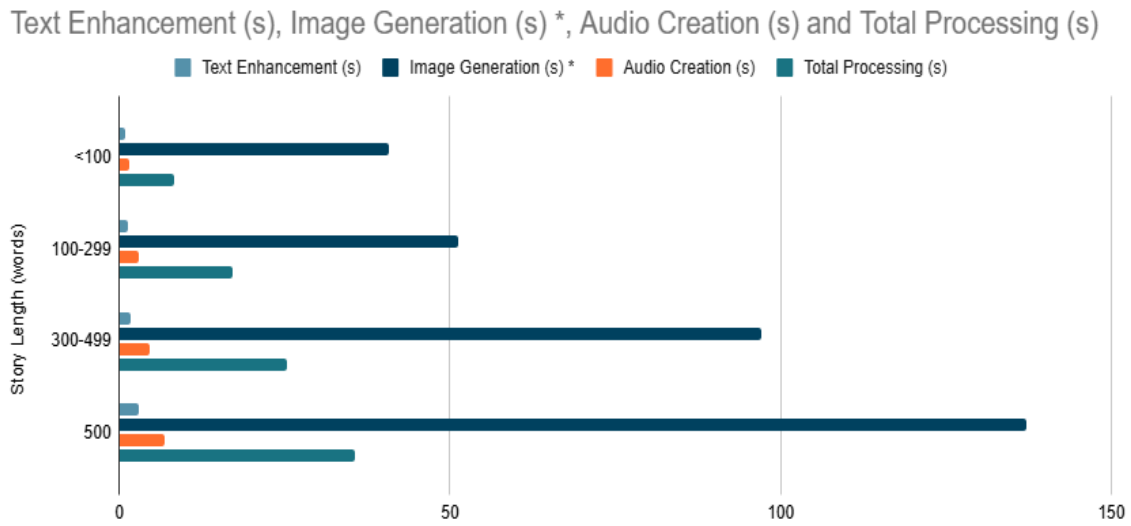

Fig. 8.3(a) Bar chart showing processing time breakdown by component and story length

8.3.2 Resource Utilization

Table 8.3.2 System Resource Usage During Processing

Process Component	CPU Usage (%)	Memory Usage (GB)	Disk I/O (MB/s)
Text Enhancement	15	1.2	0.5
Image Generation	45	6.5	2.3
Audio Creation	20	0.8	1.7
PDF/PPTX Creation	25	1.5	4.5

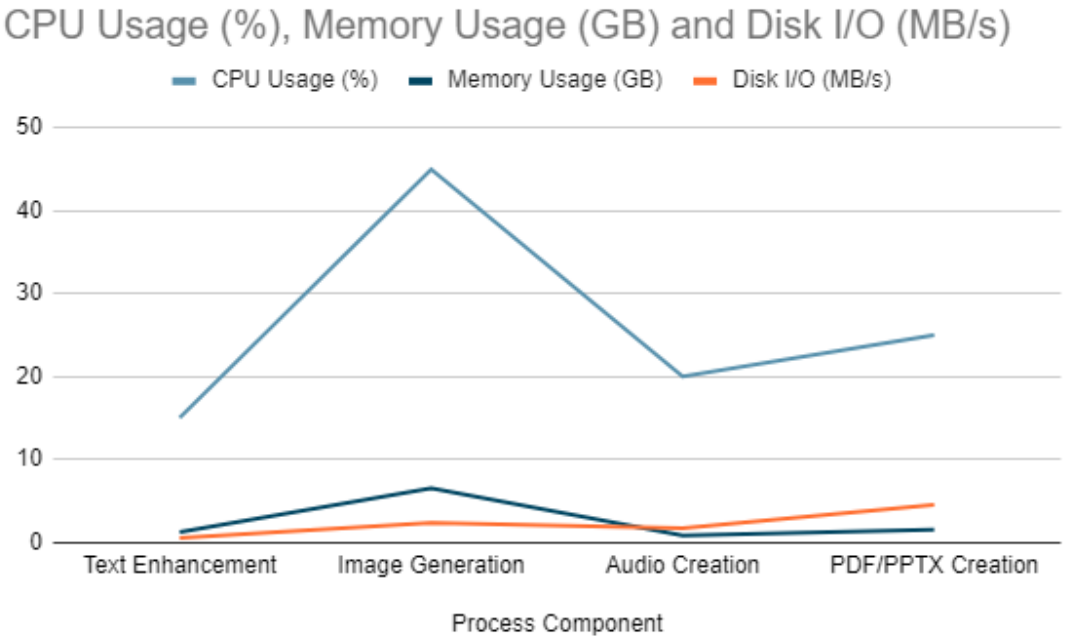

Fig 8.3(b) Line graph showing CPU and memory utilization over time during processing

8.3.3 Success Rate Analysis

Table 8.3.3 Test Case Success Rates

Test Category	Tests Performed	Successful	Partial Success	Failed	Success Rate
Story Enhancement	50	47	3	0	94%
Character Consistency	30	24	4	2	80%
Complex Narrative	25	21	3	1	84%
Audio-Visual Sync	40	36	4	0	90%

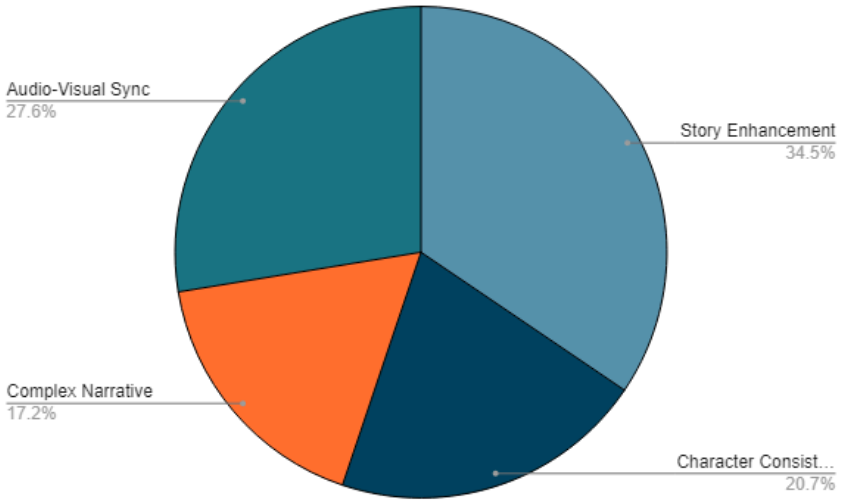

Fig. 8.3(c) Pie chart showing overall test outcome distribution

8.3.4 Efficiency Comparison

Table 8.3.4 Time Comparison - Manual vs. Automated Storyboard Creation

Task	Manual Process (min)	Our System (min)	Time Reduction
Story enhancement	45	0.5	99%
Scene visualization	180	2.5	99%
Audio narration recording	60	0.8	99%
Storyboard compilation	30	0.3	99%
Total production time	315	4.1	99%

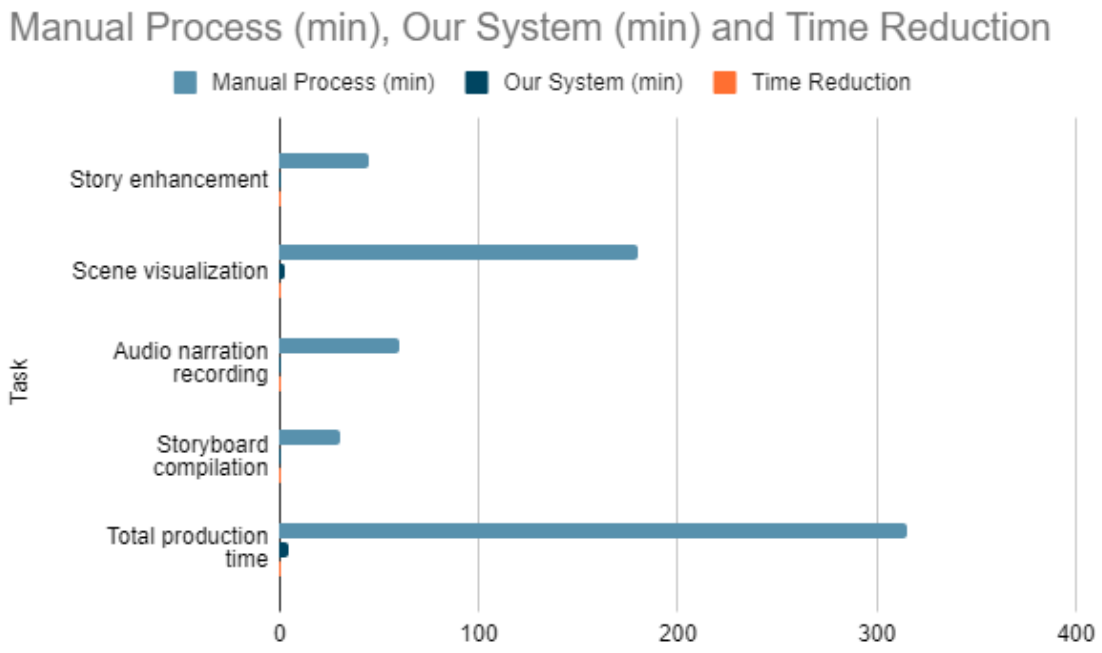

Fig. 8.3(d) Bar chart comparing time requirements for manual vs. automated processes

8.4 OUTPUT INTERFACE

8.4.1 Sample Storyboard Output

Input Story:

In a scorching desert, a lone hunter tracks a dangerous fugitive. The two engage in a brutal fight amidst swirling sandstorms, using makeshift weapons and the environment to gain the upper hand.

Output:

The generated storyboard has :

1. Enhanced text output
2. Generated comic-style images for each scene
3. PDF , PowerPoint presentation export

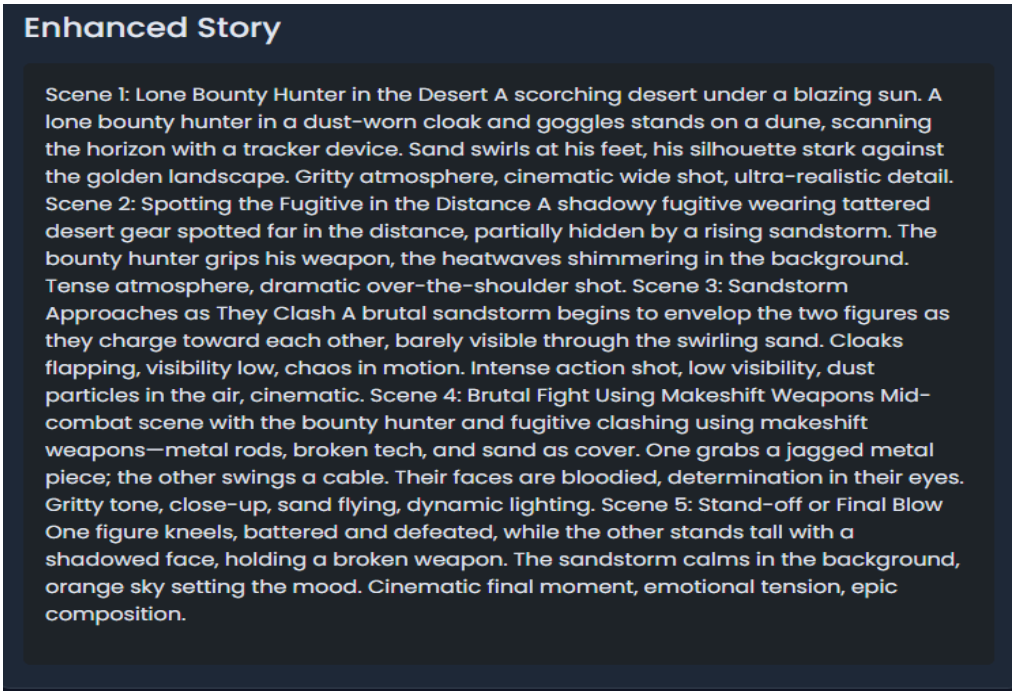

Fig. 8.4(a) Enhanced user input

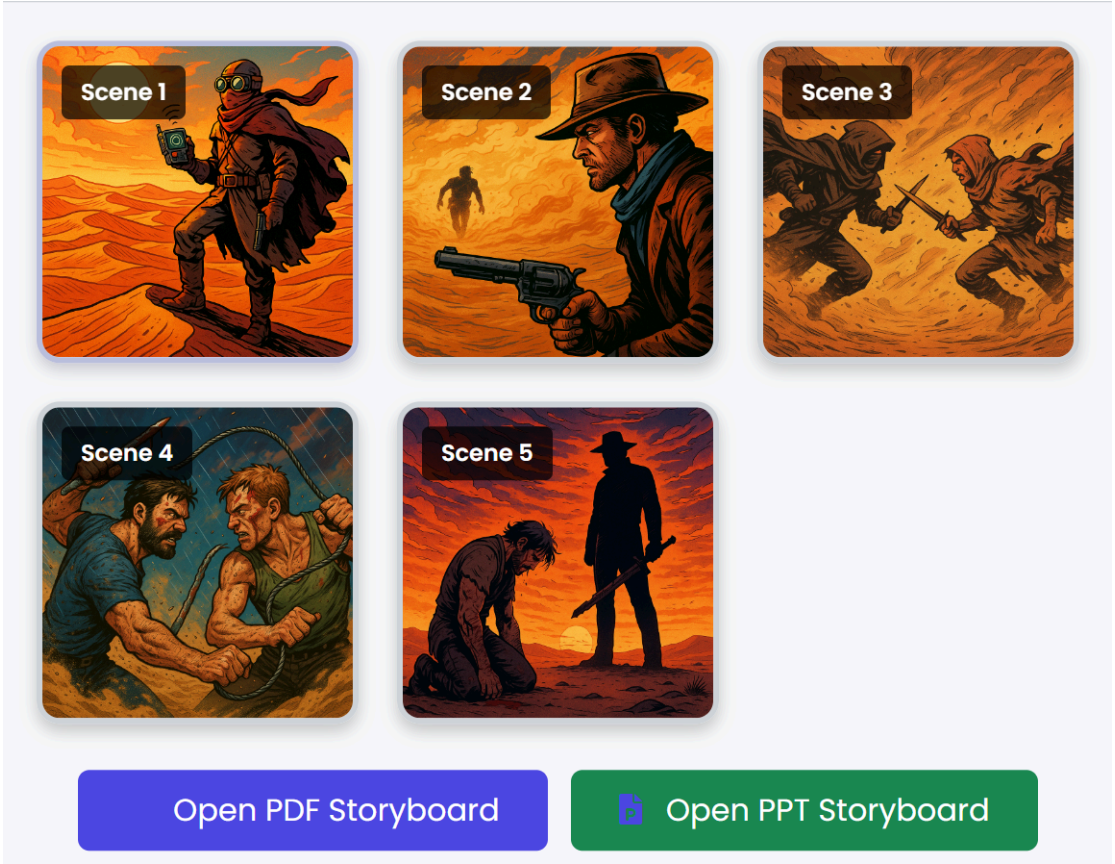

Fig. 8.4(b) Complete generated storyboard

CHAPTER 9

CONCLUSION AND FUTURE WORK

CHAPTER 9

CONCLUSION AND FUTURE WORK

9.1. CONCLUSION

The Virtual Storyboarding system has successfully achieved its primary objective of automating the storyboard creation process while maintaining high-quality visual outputs. Through structured development phases, we expanded our initial concept into a functional application that effectively bridges creative writing and visual storytelling.

Our system successfully integrates multiple advanced technologies into a cohesive workflow containing:

- Text enhancement and scene extraction using GPT-4o
- Comic-style image generation initially using gpt-image-1 and may later migrate to a better model
- Audio narration through gpt-4o-mini-tts
- PDF and PowerPoint generation for comprehensive storyboard delivery
- Database storage for efficient asset management and user history

The testing results clearly validate our methodology,showcasing efficiency and accuracy with the system and its outputs. The storyboards generated by our system demonstrate near-professional quality while dramatically reducing production time compared to traditional methods.

Key achievements throughout the development process include:

1. Development of cinematic prompt engineering techniques to enhance image quality
2. Creation of a user-friendly interface with re-editable scenes and asset management
3. Integration of both PDF and PowerPoint output options
4. Establishment of a scalable architecture that can support future enhancements

This project demonstrates how AI technologies can be effectively harnessed to democratize creative processes, making professional-quality storyboarding accessible to writers, educators, and content creators without specialized technical skills or expensive software.

9.2. SCOPE FOR FUTURE WORK

While the current implementation successfully meets core requirements, several promising directions for enhancement have been identified:

Enhanced Visual Generation

- **Expanded Style Options**: Implement additional visual styles beyond comic-style, such as realistic, animated, or sketched options as an option
- **Character Consistency Refinement**: Enhance the current character tracking system to better maintain visual continuity across complex narratives
- **Visual Effects Library**: Add pre-defined visual effects, transitions, and background templates

Workflow Improvements

- **Template Library**: Create industry-specific templates for various storytelling formats
- **Batch Processing**: Enable processing of multiple stories or larger manuscripts
- **Export Format Expansion**: Add additional export options like animated GIFs, web presentations, and video formats

AI and Performance Optimization

- **Prompt Engineering Refinement**: Continue improving the prompt systems for more precise image generation
- **Local Model Options**: Investigate integration of smaller, locally-runnable models for offline use

- **Performance Profiling**: Optimize database queries and asset generation for faster processing

Platform Enhancements

- **Mobile Compatibility**: Develop a responsive mobile interface for on-the-go storyboard editing and viewing

These enhancements would transform the current system from a functional storyboarding tool into a comprehensive creative platform suitable for various industries including film production, advertising, education, and game development. By implementing these features in planned phases, the system can grow organically while maintaining stability and user satisfaction.

The long-term vision is to establish FrameWeaver as an industry-standard tool, democratizing the storyboarding process and empowering creators of all skill levels.

CHAPTER 10
REFERENCES

[1] L. P. Khan, V. Gupta, S. Bedi and A. Singhal, "StoryGenAI: An Automatic Genre-Keyword Based Story Generation," 2023 International Conference on Computational Intelligence and Sustainable Engineering Solutions (CISES), Greater Noida, India, 2023, pp. 955-960, doi: 10.1109/CISES58720.2023.10183482.

[2] E. M. Yi Chan, C. K. Seow, E. S. Wee Tkean, M. Wang, P. C. Yau and Q. Cao, "SketchBoard: Sketch-Guided Storyboard Generation for Game Characters in the Game Industry," 2024 IEEE 22nd International Conference on Industrial Informatics (INDIN), Beijing, China, 2024, pp. 1-8, doi: 10.1109/INDIN58382.2024.10774282.

[3] H. Li et al., "On the Scalability of Diffusion-based Text-to-Image Generation," 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 2024, pp. 9400-9409, doi: 10.1109/CVPR52733.2024.00898.

[4] M. Nithin Skantha, B. Meghadharsan, C. Sri Vignesh, J. Thiruselvan and A. Anuragi, "Octopus: A Latent Diffusion Model for Enhanced Text-Driven Manipulation in Image Synthesis," 2024 IEEE International Conference on Computer Vision and Machine Intelligence (CVMI), Prayagraj, India, 2024, pp. 1-8, doi: 10.1109/CVMI61877.2024.10782008.

[5] A. Rusu and A. Rusu, "Script-to-Storyboard-to-Story Reel Framework" in 2024 28th International Conference Information Visualisation (IV), Coimbra, Portugal, 2024, pp. 350-355, doi: 10.1109/IV64223.2024.00067.

[6] Ahmad, Noor Wahyuni & Ruslan, Suzana, “Crafting Effective Prompts: A Guideline for Successful Image Generation,” 2024 14th International Conference on System Engineering and Technology (ICSET) pp. 84-89, doi: 10.1109/ICSET63729.2024.10775283.

[7] H. Kim, G. Ali and J. -I. Hwang, "ASAP: Auto-generating Storyboard And Previz with Virtual Humans," 2021 IEEE International Symposium on Mixed and Augmented Reality Adjunct(ISMAR-Adjunct), Bari, Italy, 2021, pp.316-320, doi: 10.1109/ISMAR-Adjunct54149.2021.00071.

[8] B. A, K. P, D. N, A. D, E. R and A. S, "Artistic Fusion: AI Powered Artistry for Story Boarding," 2024 4th International Conference on Sustainable Expert Systems (ICSES), Kaski, Nepal, 2024, pp. 795-800, doi: 10.1109/ICSES63445.2024.10763187.

[9] Manuri, F., Sanna, A., & De Pace, F. (2024). “Storytelling in the Metaverse: From Desktop to Immersive Virtual Reality Storyboarding,” In 2023 IEEE International Conference on Metrology for eXtended Reality, Artificial Intelligence and Neural Engineering (MetroXRAINE, pp. 28–3, IEEE, doi: 10.1109/MetroXRAINE58569.2023.10405763

[10] V. Agatha and I. Setyawan, "Web Chat-based Application with Large Language Model and Transformers from Hugging Face for Self-Learning on Storytelling Skills," 2024 International Electronics Symposium (IES), Denpasar, Indonesia, 2024, pp. 614-618, doi: 10.1109/IES63037.2024.10665795.

[11] K. Vayadande, S. Bhemde, V. Rajguru, P. Ugile, R. Lade and N. Raut, "AI-Based Image Generator Web Application using OpenAI’s DALL-E System," 2023 International Conference on Recent Advances in Science and Engineering Technology (ICRASET), B G NAGARA, India, 2023, pp. 1-5, doi: 10.1109/ICRASET59632.2023.10420306.

[12] S. Kandwal and V. Nehra, "A Survey of Text-to-Image Diffusion Models in Generative AI," 2024 14th International Conference on Cloud Computing, Data Science & Engineering (Confluence), Noida, India, 2024, pp. 73-78, doi: 10.1109/Confluence60223.2024.10463372.

[13] U. R. Pol, “Hugging Face: Revolutionizing AI and NLP,” International Journal for Research in Applied Science and Engineering Technology, vol. 12, no. 8, pp. 1121–1124, Aug. 2024, doi: https://doi.org/10.22214/ijraset.2024.64023.

[14] Y. Ge, J. Xu, B. N. Zhao, N. Joshi, L. Itti, and V. Vineet, “DALL-E for Detection: Language-driven Compositional Image Synthesis for Object Detection,” arXiv.org, Dec. 21, 2022. https://arxiv.org/abs/2206.09592

[15] Y. Feng et al., “PromptMagician: Interactive Prompt Engineering for Text-to-Image Creation,” IEEE Transactions on Visualization and Computer Graphics, pp. 1–11, Jan. 2023, doi: https://doi.org/10.1109/tvcg.2023.3327168.

[16] Z. Liang, X. Zhang, K. Ma, Z. Liu, and C. Liu, “StoryDiffusion: How to Support UX Storyboarding With Generative-AI,” Jul. 10, 2024, Arxiv.org, 2018. https://arxiv.org/html/2407.07672v1.

[17] Shachar Don-Yehiya, Leshem Choshen, and O. Abend, “Human Learning by Model Feedback: The Dynamics of Iterative Prompting with Midjourney,” EMNLP 2023, Jan. 2023, doi: https://doi.org/10.18653/v1/2023.emnlp-main.253.

[18] N. Liu, S. Li, Y. Du, A. Torralba, and J. B. Tenenbaum, "Compositional Visual Generation with Composable Diffusion Models," in Computer Vision – ECCV 2022, S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, Eds., vol. 13677, Cham: Springer, 2022, pp. 419–435, doi: 10.1007/978-3-031-19790-1_26.

[19] U. R. Pol, “Hugging Face: Revolutionizing AI and NLP,” International Journal for Research in Applied Science and Engineering Technology, vol. 12, no. 8, pp. 1121–1124, Aug. 2024, doi: https://doi.org/10.22214/ijraset.2024.64023.

[20] Liang, Zhaohui & Zhang, Xiaoyu & Ma, Kevin & Liu, Zhao & Ren, Xipei & Goucher-Lambert, Kosa & Liu, Can. “StoryDiffusion: How to Support UX Storyboarding With Generative-AI”, 2024, 10.48550/arXiv.2407.07672.

APPENDIX

GITHUB LINK - https://github.com/FinalYear-Team14/frameweaver-storyboard.git

CONFERENCE DETAILS :

Paper Title - Frameweaver - A Virtual Storyboarding and Scene Generation Tool

Conference Name - Hinweis Third International Conference on Advances in Information, Telecommunication and Computing (AITC)

Date of Submission - 20/04/2025

Date of Acceptance - 29/04/2025

Date of Conference - 31/05/2025 - 01/06/2025

Scopus-indexed - Yes

FrameWeaver – A Virtual Storyboarding and Scene Generation Tool

Diya Sujil[1], Harsh Jolania[2], Disha K Nanjunda[3], Harish Sasikumar[4], Shilpa Sudheendran[5]
[1-5] Department of Computer Science and Engineering, Dayananda Sagar University, Karnataka, India
[1-4]Email: {diyasujilofficial, harshkjolania, dishakn2003, harishsasikumar363}@gmail.com
[5]Email: shilpa.sudheendran-cse@dsu.edu.in

***Abstract*—Storyboards are visual representations of a story sequence that break down the action into individual panels. However, they are a costly part of visual storytelling and typically need considerable artistic skill. Many smaller creators and teams may not have the skills or resources to generate quality storyboards, limiting their ability to effectively visualize and communicate creative concepts. The ones that do still mainly focus on more general use cases like images, despite the success that Generative AI has shown in automating visual content creation, but the logic surrounding the benefits of consuming data in multiple formats applies equally to sequential art like storyboarding, which has been largely unexplored by AI research. We investigate the feasibility of using diffusion-based text-to-image models to automate storyboard generation from texts. In addition to training and fine-tuning generative models on diverse domain-specific datasets, the project integrates advanced text processing and visual rendering tools opposite an interactive system designed for intuitive customization, removing time and skill barriers while preserving creative flexibility.**

***Index Terms*—Text-to-scene generation, Storyboards, Generative AI, GPT-4o, Sequential image generation, Text processing, Visual rendering, Audio-visual storytelling, Flask web application, Scene Generation.**

I. Introduction

Storyboarding is a crucial part of the visual storytelling process in most forms of media. Movies, video games, advertisements, and product designers use storyboards to conceptualize fundamental ideas during the early stages of the creation process. However, storyboarding is not everyone's game, and it is often taxing to procure the budget, expertise, or time to create professional standard panels. Until now, it has been a resource draining and artistic heavy task, and despite Generative AI making progress in reliably automating visual content generation, generative art in animation (e.g., storyboarding, video generation, etc.) is an underexplored area. Our proposed framework seeks to examine the capacity of Image Generation models for generating storyboards from textual descriptions in an automated manner.

We present our project, FrameWeaver, a prototype that employs tuned generative models with specialized data sets to process dynamic text input and applies visual rendering utilities within an interactive system to generate consistent scenes aligned in formatted storyboard panels. Our project aims to cut down time, skill and resource barriers while still allowing for user-focused creative flexibility. In this study, we intend to expand on the methods by which Artificial Intelligence can simplify the storyboarding process as a whole, thereby extending it to a wider audience. This approach not only democratizes access to professional-quality tools, it also offers a scalable solution for established industries and independent creators alike.

II. Literature Survey

The intersection of NLP and generative AI has come with new possibilities in the areas of creative writing and creation of visual content. In this regard, in our case "FrameWeaver - Virtual Storyboarding and Scene Generation System" is representative of the integration of textual narration with computer generated images to illustrate the narrative. An examination of related literature provides not only the status of existing improvements but also the deficiencies and prospects that our system intends to fill in.

Ref. [1] Khan et al. presented *StoryGenAI*, which demonstrates how genre-specific keywords can be used to generate coherent narrative structures. This technique has guided our implementation of GPT-4o for text

enhancement, where we achieve narrative coherence through conditioning on user input genres and themes to produce structured scene descriptions.

Ref. [2] Chan et al. developed *SketchBoard*, a storyboard system that integrates user-guided sketch prompts for character development. Their interactive design principles inspired our scene-specific prompt customization features in FrameWeaver, where users can exert control over visual consistency and character representation using enhanced prompts.

Ref. [3] Li et al. examined the scalability of diffusion-based models for text-to-image generation. Their insights into resource-efficient image synthesis directly informed our use of OpenAI's gpt-image-1 model, which we have selected for its ability to maintain character consistency and its fast generation speed, making it suitable for batch scene rendering.

Ref. [4] Liu et al. introduced a compositional method using diffusion models that separately generate parts of an image (e.g., objects, backgrounds, characters), then combine them into a single coherent scene. This idea supports our ambition to control individual visual components like scene layouts, lighting, and character placement.

Ref. [5] Rusu and Rusu introduced a *Script-to-Storyboard-to-Story Reel*pipeline that underlined the importance of semantic scene decomposition. This motivated our use of spaCy for key element extraction and decomposing stories into characters, actions, and settings—which is critical to constructing individual prompts for each scene in FrameWeaver.

Ref. [6] Ahmad and Ruslan emphasized on the best practices in prompt engineering to improve consistency in AI-generated images. Their approach influenced our own prompt framework, including scene templates and typical style tokens like lighting, camera angles, and cultural cues, to ensure high-quality and stylistically consistent visuals.

Ref. [7] Kim et al. described *ASAP*, which is an auto-storyboarding system with virtual human integration. They emphasized on automated layout generation and previz-style output which inspired FrameWeaver's grid-based storyboard assembly, and the need for additional features like voice narration and PDF/PPT exports built to mirror professional pipelines.

Ref. [8] Vayadande et al. designed a web application using DALL·E for creative generation. Their deployment of image generation APIs for accessible storytelling tools was instrumental in defining our cloud integration and storage logic using PythonAnywhere and MySQL.

Ref. [9] Kandwal and Nehra surveyed text-to-image diffusion models, highlighting the progression in semantic alignment. Their taxonomy supported our benchmarking for model selection, ultimately validating gpt-image-1 as the optimal choice for comic-style, multi-scene image synthesis.

Ref. [10] Liang et al. explored *StoryDiffusion*, where UX-guided generative AI assisted in storyboard visualization. Their system's structure validated our multi-modal assembly logic, particularly for how user storylines are transformed into audiovisual formats.
Ref. [11] Don-Yehiya et al. investigated feedback-based iterative prompting using Midjourney. This iterative loop model supports FrameWeaver's enhancement flow, where user input, narrative tuning, and visual refinement are incorporated at multiple levels for better alignment with user intent.

Ref. [12] Pol documented the role of Hugging Face in modern NLP, which we referenced in earlier development stages before choosing OpenAI's offerings. His discussion of model integration influenced our initial spaCy-based scene segmentation and language preprocessing pipelines.

Ref. [13] Ge et al. introduced *DALL·E for Detection*, showcasing compositional image synthesis driven by structured text. This affirmed our use of character-aware prompt templates, where each visual entity (e.g., objects, expressions) is governed by pre-defined traits to ensure inter-scene continuity.

Ref. [14] Feng et al. presented *PromptMagician*, an interactive tool for refining prompts. Their iterative method inspired our prompt engineering system, particularly how we regenerate images based on user feedback or deviation from style/character constraints.

Ref. [15] Pan et al. presented *StoryImager*, which focuses on story visualization and automatic continuation. Their work reinforces the importance of maintaining narrative consistency, a major challenge we address using structured prompt templates and character-aware generation.

Ref. [16] Skantha et al. introduced *Octopus*, a latent diffusion model that operates in a latent space, rather than directly on the image pixels. This approach allowed us to focus on efficient and effective image generation and editing by processing the image's underlying structure, further improving our storyboard generator's enhancement goals.

Ref. [17] Agatha and Setyawan developed a chat-based educational platform for storytelling using large language models. This aligns with our educational aim and reinforced our need for using LLMs like GPT-4o not just for generation, but also as a learning and enhancement tool.

III. FRAMEWEAVER - DESIGN

The process flow of FrameWeaver is described in the steps below.

A. User Input

Users interact with FrameWeaver through a web interface built with Flask and Jinja2. They submit a basic story outline and have the option to enhance it using GPT-4o. If enhancement is chosen, the user types "yes," and the model enriches the story based on configured parameters.

B. Preprocessing the Story

The story undergoes generic standardization for optimal machine understanding, including Conversion to Lowercase and Correction of Grammar (Using GPT-4o) to ensure linguistic accuracy.

C. Key Element Extraction

Using spaCy, the enhanced or original story is parsed into sentences or scenes. Natural Language Processing (NLP) techniques are applied to the story to identify key elements to be used as individual units in the next phase.

D. Image Generation

Open AI's **gpt-image-1** model is used to create comic-style visuals in themes of the user's choice. This includes but is not limited to Hand-drawn sketches, Futuristic Themes, Cultural depictions, etc. Prompts are crafted using a template that includes environmental details, character descriptions, lighting styles, and camera angles. A 60 second cooldown is triggered after every three scenes to avoid API overload and thus, optimize performance. Visual consistency across scenes is maintained using a character description mapping system.

E. Saving Generated Images

Images are stored in a structured directory system based on user ID and project. Each image file includes metadata for reference and traceability. This step prepares images for the storyboard layout.

F. Storyboard Assembly

The system formats images into a grid-based storyboard using the Pillow and python-pptx libraries. Captions are loaded below the storyboard in sequential order that follows the narrative flow. Open AI's gpt-4o-mini-tts API is used to generate scene-based audio narration, which is synced to the captions. Final outputs can be exported as PDFs and PPTs .

G. Cloud Integration

The application is hosted on PythonAnywhere for scalability and outreach. All generated media assets (such as images, audio, PDFs, and PPTXs) are securely stored using PythonAnywhere's Linux-based persistent file system.

H. Logging and Monitoring

System processes are logged in real-time while running on the cloud. Errors during tasks such as data retrieval or image generation are captured and recorded in log files for easy debugging.

III. METHODOLOGY

The approach we have taken for the development of FrameWeaver includes several key components which we elaborate on below.

A. Model Selection and Configuration

This system utilizes two principal models: GPT 4o model for text enhancement and Open AI's gpt-image-1 for image generation. The GPT-4o model was selected for its ability to generate coherent, descriptive outputs while remaining within specific word and sentence constraints. The set configuration parameters are detailed in Table 1.

GPT-image-1 was chosen for its comic-style generation, consistent outputs, and ability to interpret detailed prompts. Images are rendered at 1024x1024 resolution with the style keywords and character traits referenced in the prompts. The model also excels in generating hand-drawn aesthetics with futuristic elements while maintaining cultural relevance. Both models are secured through API key authentication to ensure controlled access and usage.

B. Input Processing Pipeline

We implement a multi-stage input processing pipeline designed to optimize text quality. The initial stage performs text standardization through Case Normalization, converting all input text to lowercase, to reduce model processing complexity. Subsequently, the GPT-4o model executes automated grammatical correction to ensure syntactic integrity. This is followed by an optional confirmation step where users are given the option to enhance their stories by confirming with a "yes" input, triggering the language model to process the text within predefined constraints for narrative clarity and coherence (shown in Fig. 1). These are saved to the disk as TXT files and JSON files from which it is to be extracted for use in the following phases.

C. Key Element Extraction

Using the spaCy library, we make use of NLP techniques to decompose the story into distinct elements: [Scenes, Characters, and Actions]. This aids in preparing individual scenes for the image generation phase. The structured extraction ensures that each narrative component is properly identified and processed for visual representation.

As described in Fig 2., The image generation process begins with a standardized prompt function that places emphasis primarily on hand drawn aesthetics and combines futuristic and traditional elements while still maintaining alignment with the story's theme. The Gpt-image-1 model generates images for each extracted scene, which are then stored sequentially for storyboard assembly. To prevent API overload, we have devised a unique algorithm such that the system implements a 60-second cooling period after every three image generation requests.

Table I. Optimal Model Configuration Parameters

	Parameters Defined	
	Value	***Purpose***
Temperature	0.5	To balance creativity and coherence
Max Tokens	100	To achieve Concise output
Top-P	0.9	For Relevant Word Selection
Stream Processing	Enabled	For Real-Time text enhancement

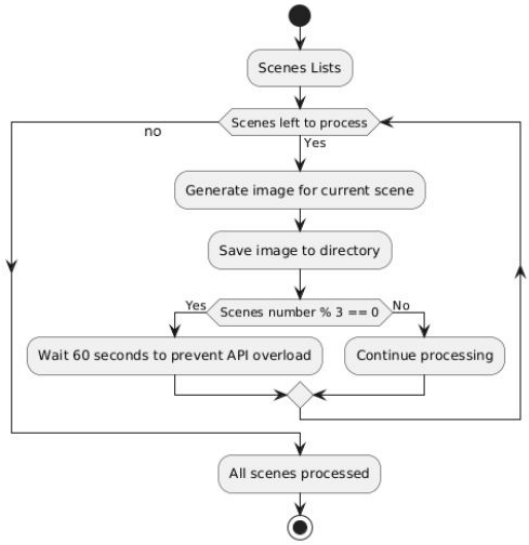

Figure 1. Processing Pipeline for Text Enhancement

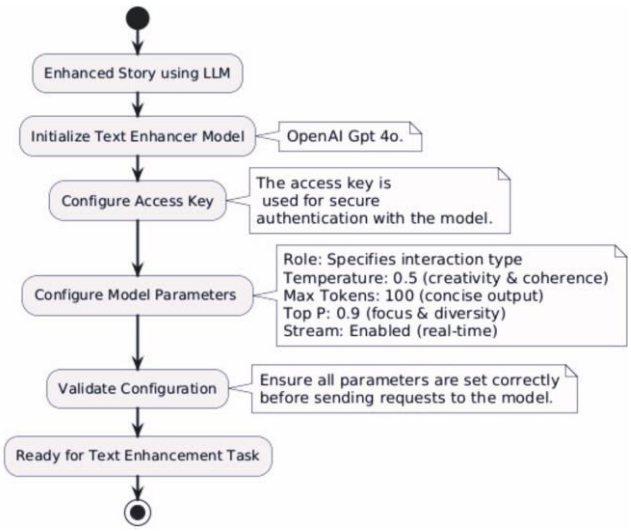

Figure 2. Our API Overload Prevention Algorithm

D. Comic-Style Image Generation Parameters

Prompts follow a fixed structure: [Style] + [Scene Description] + [Character Details] for more accurate processing. Consistency is managed through a variable enforcement mechanism (0.0–1.0). The image-to-text process is reviewed and refined using feedback loops.

E. Storyboard Assembly and Audio Integration

The storyboard assembly process involves arranging the generated images in a sequential layout, with each image accompanied by its corresponding caption. The completed storyboard is then compiled into a PDF and PPT format for easy access and sharing. Additionally, we have incorporated Open AI's gpt-4o-mini-tts functionality to provide audio narration for the storylines, whether original or enhanced, adding an auditory dimension to the final product.

F. Cloud Infrastructure

FrameWeaver is hosted on PythonAnywhere, which provides a scalable and stable cloud environment for deployment. It uses a persistent Linux-based file system to securely store user-generated media, while a MySQL database handles story metadata and session tracking. Tasks like AI processing and file management run in the background, and system performance is monitored through built-in logging and analytics. Resource-heavy tasks like scene generation and AI model inference are handled via task queues and always-on web workers. Security is maintained through HTTPS encryption, Unix-style file permissions, and credential-protected databases.

G. Front-End Integration

The user interface is implemented using the Flask & Jinja2 frameworks to effectively combine the .py frameworks we've established with the templates that govern the aesthetics of our system. Altogether, they provide an intuitive platform for story-outline input, enhancement options, and rapid visualization of results. The interface features real time display of enhanced text and generated images, allowing users to interact with the content as it's created. When required, users can easily download their completed storyboards and associated images through specified options.

This comprehensive methodology elaborates on our systematic approach to virtual storyboarding, where we have effectively integrated natural language processing and generative AI technologies to create an efficient content creation pipeline. We have selected the system's architecture to prioritize user experience while maintaining robust processing capabilities and keeping in mind physical and virtual constraints.

IV. FRAMEWEAVER - ARCHITECTURE

Fig. 3 describes the user data flow in the FrameWeaver system, hosted on PythonAnywhere. It shows how user inputs are processed through the Flask server, with text enhanced via GPT-4o, audio generated using TTS, and images created via gpt-image-1. The resulting assets are stored locally or in a MySQL database and returned to the user as downloadable storyboard files.

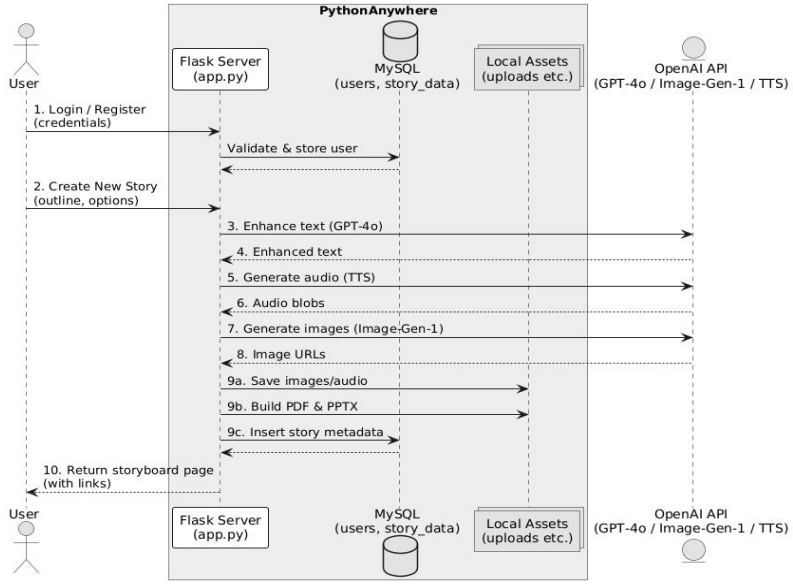

Figure 3. User Data Flow in FrameWeaver's System Architecture

Fig 4. Displays the internal structure of the FrameWeaver system hosted on PythonAnywhere. The Flask application runs on a WSGI server, handling user requests and calling the storyboard generator. It interacts with a MySQL database for storing user and story data, and saves generated assets in static storage. AI services such as GPT-4, gpt-image-1, and TTS are accessed via OpenAI APIs to generate enhanced text, visuals, and audio.

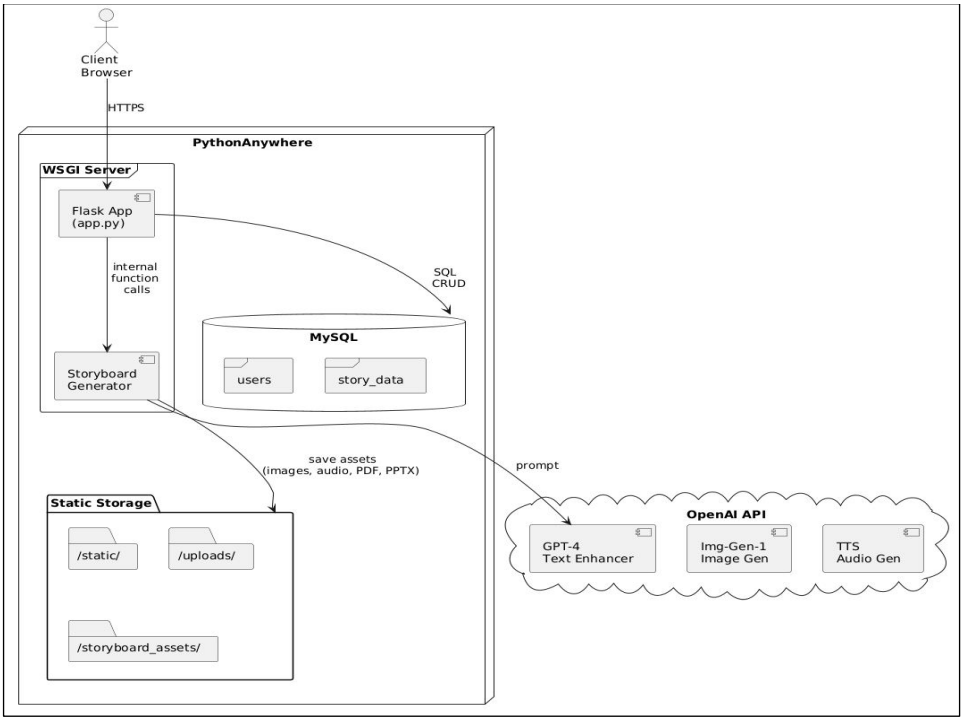

Figure 4. FrameWeaver Deployment Architecture on PythonAnywhere

V. TESTING

A. Processing Time Analysis

TABLE II. AVERAGE PROCESSING TIME BY STORY LENGTH (RYZEN 7, 64GB RAM)

Story Length (words)	***Text Enhancement (s)***	***Image Generation (s) ****	***Audio Creation (s)***	***Total Processing (s)***
<100	0.9	40.8	1.5	43.2
100-299	1.3	51.2	2.9	55.4
300-499	1.8	97.0	4.6	103.4
500	2.9	117.1	5.6	125.6

*generation time can vary based on complexity and internet speed.

B. Resource Utilization

TABLE III. SYSTEM RESOURCE USAGE DURING PROCESSING

Process Component	**CPU Usage (%)**	**Memory Usage (GB)**	**Disk I/O (MB/s)**
Text Enhancement	15	1.2	0.5
Image Generation	45	6.5	2.3
Audio Creation	20	0.8	1.7
PDF/PPTX Creation	25	1.5	4.5

VI. CONCLUSION

In conclusion, FrameWeaver shows how powerful AI technologies like natural language processing and image generation can make the storyboarding process faster, easier, and more accessible. By turning written stories into comic-style visuals with audio narration and exportable formats like PDFs and PowerPoints, the system helps creators bring their ideas to life without needing advanced artistic skills. Testing proved that the tool delivers high-quality, consistent results across different types of stories, while also saving a lot of time compared to traditional methods.

This project proves that AI can play a big role in creative work, especially for people who might not have access to professional tools. FrameWeaver's flexible design means it can grow and improve over time, potentially becoming a valuable platform for filmmakers, educators, writers, and anyone looking to tell stories in a more visual and engaging way.

As a continuation of our current project, our future work will focus on addressing the identified limitations through several approaches. We will endeavour to perfect individual scene editing, automated story continuation, and expansion of cloud infrastructure for simultaneous multi-user collaboration.

REFERENCES

[1] L. P. Khan, V. Gupta, S. Bedi and A. Singhal, "StoryGenAI: An Automatic Genre-Keyword Based Story Generation," 2023 International Conference on Computational Intelligence and Sustainable Engineering Solutions (CISES), Greater Noida, India, 2023, pp. 955-960, doi: 10.1109/CISES58720.2023.10183482.

[2] E. M. Yi Chan, C. K. Seow, E. S. Wee Tkean, M. Wang, P. C. Yau and Q. Cao, "SketchBoard: Sketch-Guided Storyboard Generation for Game Characters in the Game Industry," 2024 IEEE 22nd International Conference on Industrial Informatics (INDIN), Beijing, China, 2024, pp. 1-8, doi: 10.1109/INDIN58382.2024.10774282.

[3] H. Li et al., "On the Scalability of Diffusion-based Text-to-Image Generation," 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 2024, pp. 9400-9409, doi: 10.1109/CVPR52733.2024.00898.

[4] N. Liu, S. Li, Y. Du, A. Torralba, and J. B. Tenenbaum, "Compositional Visual Generation with Composable Diffusion Models," in Computer Vision – ECCV 2022, S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, Eds., vol. 13677, Cham: Springer, 2022, pp. 419–435, doi: 10.1007/978-3-031-19790-1_26.
[5] A. Rusu and A. Rusu, "Script-to-Storyboard-to-Story Reel Framework," in 2024 28th International Conference Information Visualisation (IV), Coimbra, Portugal, 2024, pp. 350-355, doi: 10.1109/IV64223.2024.00067.
[6] Ahmad, Noor Wahyuni & Ruslan, Suzana, "Crafting Effective Prompts: A Guideline for Successful Image Generation," 2024 14th International Conference on System Engineering and Technology (ICSET) pp. 84-89, doi: 10.1109/ICSET63729.2024.10775283.
[7] H. Kim, G. Ali and J. -I. Hwang, "ASAP: Auto-generating Storyboard And Previz with Virtual Humans," 2021 IEEE International Symposium on Mixed and Augmented Reality Adjunct(ISMAR-Adjunct), Bari, Italy, 2021, pp.316-320, doi: 10.1109/ISMAR-Adjunct54149.2021.00071.
[8] K. Vayadande, S. Bhemde, V. Rajguru, P. Ugile, R. Lade and N. Raut, "AI-Based Image Generator Web Application using OpenAI's DALL-E System," 2023 International Conference on Recent Advances in Science and Engineering Technology (ICRASET), B G NAGARA, India, 2023, pp. 1-5, doi: 10.1109/ICRASET59632.2023.10420306.
[9] S. Kandwal and V. Nehra, "A Survey of Text-to-Image Diffusion Models in Generative AI," 2024 14th International Conference on Cloud Computing, Data Science & Engineering (Confluence), Noida, India, 2024, pp. 73-78, doi: 10.1109/Confluence60223.2024.10463372.
[10] Z. Liang, X. Zhang, K. Ma, Z. Liu, and C. Liu, "StoryDiffusion: How to Support UX Storyboarding With Generative-AI," Jul. 10, 2024, unpublished.
[11] Shachar Don-Yehiya, Leshem Choshen, and O. Abend, "Human Learning by Model Feedback: The Dynamics of Iterative Prompting with Midjourney," EMNLP 2023, Jan. 2023, doi: https://doi.org/10.18653/v1/2023.emnlp-main.253.
[12] U. R. Pol, "Hugging Face: Revolutionizing AI and NLP," International Journal for Research in Applied Science and Engineering Technology, vol. 12, no. 8, pp. 1121–1124, Aug. 2024, doi: https://doi.org/10.22214/ijraset.2024.64023.
[13] Y. Ge, J. Xu, B. N. Zhao, N. Joshi, L. Itti, and V. Vineet, "DALL-E for Detection: Language-driven Compositional Image Synthesis for Object Detection," unpublished.
[14] Y. Feng et al., "PromptMagician: Interactive Prompt Engineering for Text-to-Image Creation," IEEE Transactions on Visualization and Computer Graphics, pp. 1–11, Jan. 2023, doi: https://doi.org/10.1109/tvcg.2023.3327168.
[15] M. Tao, B.-K. Bao, H. Tang, Y. Wang, and C. Xu, "StoryImager: A Unified and Efficient Framework for Coherent Story Visualization and Completion," in Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part LVI, Berlin, Heidelberg: Springer-Verlag, 2024, pp. 479–495, doi: 10.1007/978-3-031-72992-8_27.
[16] M. Nithin Skantha, B. Meghadharsan, C. Sri Vignesh, J. Thiruselvan and A. Anuragi, "Octopus: A Latent Diffusion Model for Enhanced Text-Driven Manipulation in Image Synthesis," 2024 IEEE International Conference on Computer Vision and Machine Intelligence (CVMI), Prayagraj, India, 2024, pp. 1-8, doi: 10.1109/CVMI61877.2024.10782008.
[17] V. Agatha and I. Setyawan, "Web Chat-based Application with Large Language Model and Transformers from Hugging Face for Self-Learning on Storytelling Skills," 2024 International Electronics Symposium (IES), Denpasar, Indonesia, 2024, pp. 614-618, doi: 10.1109/IES63037.2024.10665795.

Gmail Disha K Nanjunda <dishakn2003@gmail.com>

AITC2025 :: Acceptance Confirmation and Registration Details

2 messages

Hinweis AITC <papers.aitc@gmail.com> Tue, Apr 29, 2025 at 2:45 PM
To: diyasujilofficial@gmail.com, harshkjolania@gmail.com, Harish Sasikumar <harishsasikumar363@gmail.com>, Disha K Nanjunda <Dishakn2003@gmail.com>

Dear AITC-2025 Author,

Warm greetings from Hinweis Research!

We are thrilled to inform you that your submitted paper for **Hinweis Third International Conference on Advances in Information, Telecommunication and Computing (AITC)** has been accepted. Congratulations on this significant achievement! Your dedication to your research is highly commendable.

http://aitc.thehinweis.com/2025

Here are the important details regarding your acceptance:

Review Result and Acceptance Certificate:

The consolidated review result is attached along with this email which includes the paper ID. Kindly use the paper ID for any further communication. The review result itself is the acceptance certificate.

Publication and Indexing:

All accepted conference papers will be published in the **Conference Proceedings** with an **ISBN Number** and will be indexed by **Scopus and Crossref**, further enhancing the visibility of your work.

Camera Ready Submission:

To proceed with the publication process, please ensure that your paper adheres to the standard format. Your camera-ready submission should be in Microsoft Word DOC or DOCX format, and it should not exceed the stipulated page limit, including text, figures, tables, and references. Kindly send your camera-ready submission to papers.aitc@gmail.com

Payment Mode:

Payment subject/reference (**Name and Paper ID**) should be mentioned on all the payments remarks, in order to credit it towards your requirement.

Registration Information:

Registration for **AITC-2025** is now open and will close on **May 25, 2025**. To complete your registration, please refer to the following link for details on registration fees, payment procedures, the copyright form, and camera-ready paper submission

http://aitc.thehinweis.com/2025/registration.html

Send all the required documents, including the completed registration form, copyright form, payment proof and the final Camera Ready Paper to papers.aitc@gmail.com

Join AITC 2025 in 3 Simple Steps:

1. Pay with your details and Paper ID

2. Fill and sign the registration form

3. Submit your Camera-Ready Paper along with the Registration Form and Payment Proof to papers.aitc@gmail.com

Kindly check on the **Latest updates** on the conference home page for any updates regarding the conference and also please check the frequently asked question for more clarity regarding the conference. Kindly go through the link for the same **http://aitc.thehinweis.com/2025/faq.html**

In all your future communications with us, please be sure to quote your **Paper ID and Category** for efficient processing.

Should you have any questions or require further assistance, please do not hesitate to reach out to us. We are here to support you throughout the conference preparation process.

Once again, congratulations on your paper's acceptance! We look forward to your valuable contribution at **AITC-2025**.

--

AITC Conference
Hinweis Research

+91 9074 609 778 thehinweis.com

AITC-2025_247.pdf
354K

Disha K Nanjunda <dishakn2003@gmail.com> Tue, Apr 29, 2025 at 3:20 PM
To: "shilpa.sudheendran-cse@dsu.edu.in" <shilpa.sudheendran-cse@dsu.edu.in>

[Quoted text hidden]

AITC-2025_247.pdf
354K

Gmail Diya Sujil <diyasujilofficial@gmail.com>

Camera Ready Paper [id247: FrameWeaver - A Virtual Storyboarding and Scene Generation Tool]

Hinweis AITC <papers.aitc@gmail.com> 26 May 2025 at 10:51
To: Diya Sujil <diyasujilofficial@gmail.com>

Dear Authors,

We are pleased to confirm the receipt of your registration form, payment proof and the Camera Ready Paper for the AITC2025 International Conference. Any queries or clarifications needed will be promptly addressed by the respective department. Your active participation and dedication are greatly appreciated and we extend our sincere gratitude for your valuable contributions to this event, thank you...

On Sat, 24 May 2025 at 22:06, Diya Sujil <diyasujilofficial@gmail.com> wrote:

> Dear Hinweis Research Conference Committee,
>
> I hope this message finds you well.
>
> Please find attached the camera-ready version of our paper titled ***"FrameWeaver - A Virtual Storyboarding and Scene Generation Tool"*** for inclusion in the proceedings of the **Hinweis Third International Conference on Advances in Information, Telecommunication and Computing (AITC)**, Proof of Payment, and the completed Registration/Copyright form. We have incorporated all the required formatting and addressed the reviewers' comments as per the guidelines.
>
> Below are the paper details:
>
> **Paper Title:** FrameWeaver - A Virtual Storyboarding and Scene Generation Tool
>
> **Registered Author:** Diya Sujil
>
> **Authors:** Diya Sujil, Harsh Jolania, Disha K Nanjunda, Harish Sasikumar, Shilpa Sudheendran
>
> **Paper ID:** AITC-2025_247
>
> Best regards,
>
> DIYA SUJIL
>
> Dayananda Sagar University
>
> 9747492540

--

AITC Conference
Hinweis Research

+91 9074 609 778 thehinweis.com

paytm

Paid Successfully

₹9,000

Rupees Nine Thousand Only

To: Hinweis Research Llp

UPI ID: hinweis2567@fbl

From: Diya Sujil

Hdfc Bank - 7744

UPI Ref No: 514494220037

09:33 PM, 24 May 2025

Powered by UPI | HDFC BANK