**Story to Image Generator Documentation**

**Software Name:** Story to Image Generator  
**Date:** 9 September 2025

**TABLE OF CONTENTS**

* [Software Summary](https://claude.ai/chat/cffe39b4-824d-42f1-a127-3e2791cc419f#software-summary)
* [Requirements](https://claude.ai/chat/cffe39b4-824d-42f1-a127-3e2791cc419f#requirements)
* [How-to Guide](https://claude.ai/chat/cffe39b4-824d-42f1-a127-3e2791cc419f#how-to-guide)
  + [Step 1: Setup & Installation](https://claude.ai/chat/cffe39b4-824d-42f1-a127-3e2791cc419f#step-1-setup--installation)
  + [Step 2: Project Creation](https://claude.ai/chat/cffe39b4-824d-42f1-a127-3e2791cc419f#step-2-project-creation)
  + [Step 3: Image Generation](https://claude.ai/chat/cffe39b4-824d-42f1-a127-3e2791cc419f#step-3-image-generation)
* [Tutorials](https://claude.ai/chat/cffe39b4-824d-42f1-a127-3e2791cc419f#tutorials)
* [FAQs](https://claude.ai/chat/cffe39b4-824d-42f1-a127-3e2791cc419f#faqs)
* [Additional Resources](https://claude.ai/chat/cffe39b4-824d-42f1-a127-3e2791cc419f#additional-resources)
* [Customer Support](https://claude.ai/chat/cffe39b4-824d-42f1-a127-3e2791cc419f#customer-support)

**Software Summary**

The **Story to Image Generator** is a comprehensive AI-powered application that transforms written stories and scripts into stunning visual images. Built with FastAPI backend and Streamlit frontend, it leverages advanced AI models to analyze narratives, generate scene descriptions, and create high-quality images that bring stories to life.

**Key Features**

* **Intelligent Script Analysis**: Automatically analyzes word count, complexity, and recommends optimal scene numbers
* **Multi-Provider AI Integration**:
  + **OpenAI GPT-4** for script analysis and scene prompt generation
  + **Runware API** for high-quality image generation
  + **Together AI** for alternative image generation options
* **Visual Style Options**: Choose from cinematic, cartoon, realistic, or artistic styles
* **Real-time Progress Monitoring**: Track generation status with live updates
* **Project Management**: Organize and manage multiple story projects
* **Batch Processing**: Generate multiple scenes simultaneously
* **Approval Workflow**: Preview and select images before final saving

**Architecture**

* **Backend**: FastAPI-based REST API with async processing
* **Frontend**: Streamlit web interface with real-time updates
* **Storage**: File-based project management with organized directory structure
* **Session Management**: In-memory session tracking for concurrent operations

**Requirements**

**API Keys Required**

You'll need API keys from the following services:

1. **OpenAI API Key**
   * Sign up at [OpenAI Platform](https://platform.openai.com/)
   * Navigate to API Keys section
   * Create new secret key
   * Pricing: Pay-per-use (GPT-4 models)
2. **Runware API Key**
   * Register at [Runware](https://runware.ai/)
   * Access your dashboard
   * Copy API key from settings
   * Pricing: Credit-based system
3. **Together AI API Key** (Optional)
   * Create account at [Together AI](https://together.xyz/)
   * Generate API key in dashboard
   * Pricing: Usage-based billing

**Python Dependencies**

The following packages are automatically installed:

fastapi>=0.104.1

uvicorn[standard]>=0.24.0

streamlit>=1.29.0

pydantic>=2.5.0

requests>=2.31.0

python-dotenv>=1.0.0

aiohttp>=3.9.0

Pillow>=10.0.0

**How-to Guide**

**Step 1: Setup & Installation**

**1.1 Download and Extract**

1. Download the Story to Image Generator package
2. Extract to your desired directory (e.g., C:\StoryToImage or ~/StoryToImage)
3. Open terminal/command prompt in the extracted directory

**1.2 Install Dependencies**

# Install required Python packages

pip install -r requirements.txt

**1.3 Configure Environment Variables**

1. Create a .env file in the root directory:

# Create .env file

touch .env # On macOS/Linux

# Or create manually on Windows

1. Add your API keys to the .env file:

# OpenAI Configuration

OPENAI\_API\_KEY=your\_openai\_key\_here

# Runware Configuration

RUNWARE\_API\_KEY=your\_runware\_key\_here

# Together AI Configuration (Optional)

TOGETHER\_API\_KEY=your\_together\_key\_here

# Optional: Performance Tuning

HTTP\_TIMEOUT=60

MAX\_RETRIES=3

RETRY\_DELAY=2

**1.4 Verify Installation**

# Test backend startup

python -m backend.main

# Should display:

# 🚀 Story to Image Generator API starting...

# ✅ OpenAI, ✅ Runware, ✅ Together AI

**Step 2: Project Creation**

**2.1 Start the Application**

**Terminal 1 - Backend Server:**

python -m backend.main

# Backend will start on http://localhost:8000

**Terminal 2 - Frontend Interface:**

streamlit run frontend/frontend.py

# Frontend will open in browser at http://localhost:8501

**2.2 Create Your First Project**

1. **Navigate to "Create Story"** section in the web interface
2. **Enter Project Details:**
   * **Title**: Descriptive name for your story
   * **Script**: Your story content (minimum 10 words)
3. **Script Writing Tips:**
   * Use descriptive language for better visual results
   * Include character descriptions, settings, and emotions
   * Separate different scenes with paragraphs
   * Mention colors, lighting, and atmosphere

**Example Script:**

The ancient lighthouse stood tall against the stormy night sky.

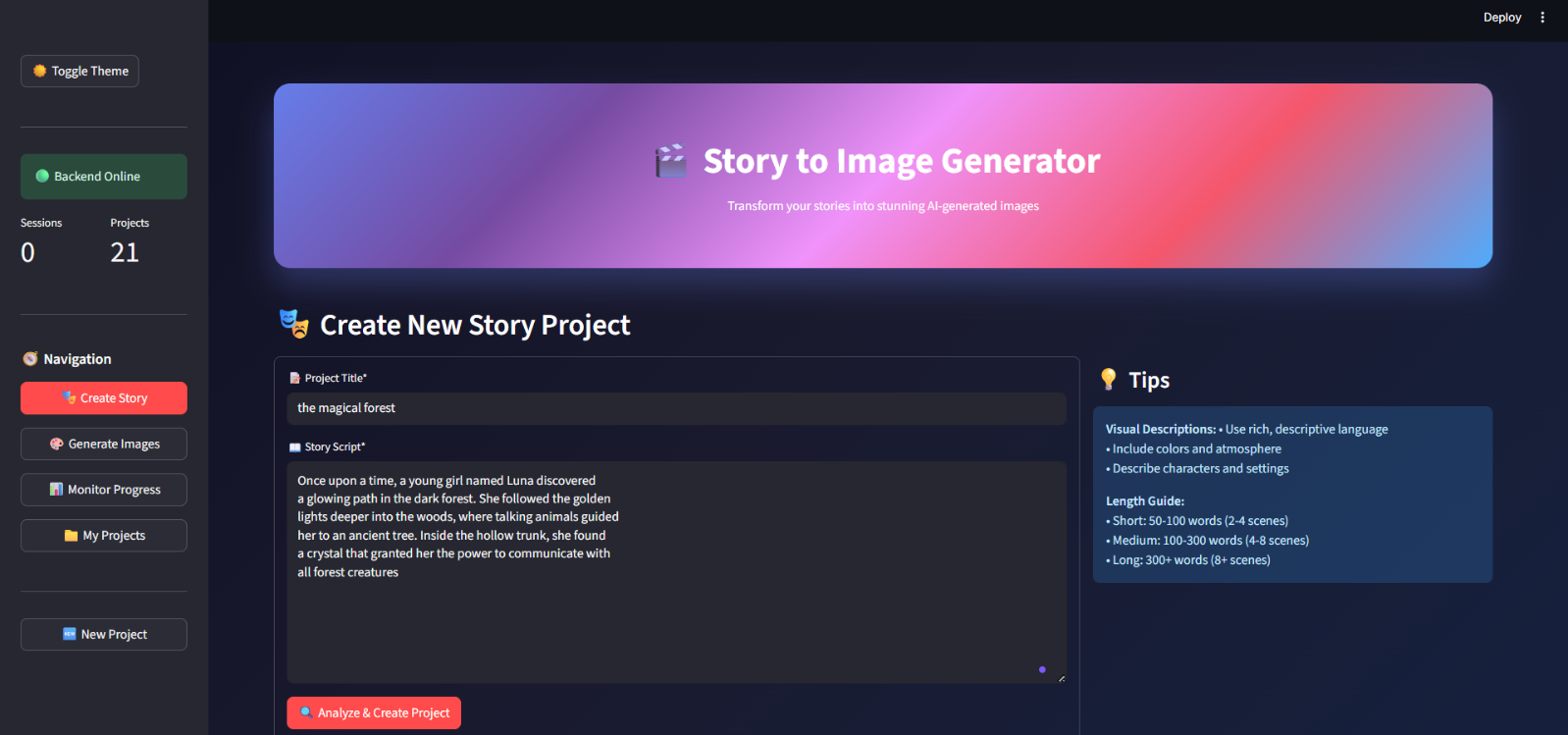
Lightning illuminated the crashing waves below as Sarah climbed

the spiral staircase. Her red coat fluttered in the wind as she

reached the beacon room, where a mysterious glowing orb awaited

her arrival.

1. **Click "Analyze & Create Project"**
   * The system will analyze word count, complexity, and recommend scene numbers
   * Project will be created with a unique ID



**2.3 Review Analysis Results**

The system provides:

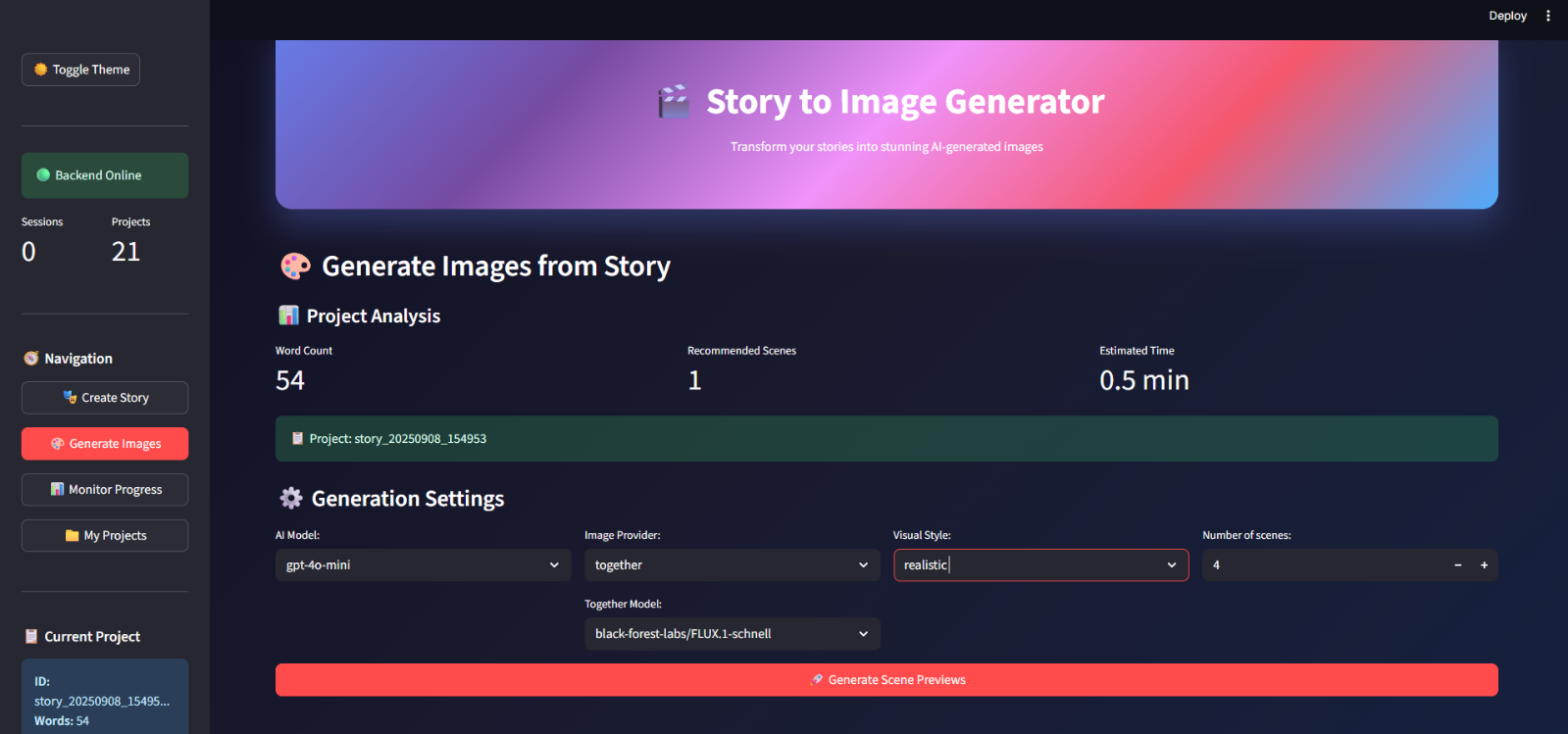
* **Word Count**: Total words in your script
* **Recommended Scenes**: Optimal number of images (typically 1 scene per 20-50 words)
* **Estimated Duration**: Reading time estimation
* **Complexity Score**: Simple, Moderate, or Complex based on vocabulary



**Step 3: Image Generation**

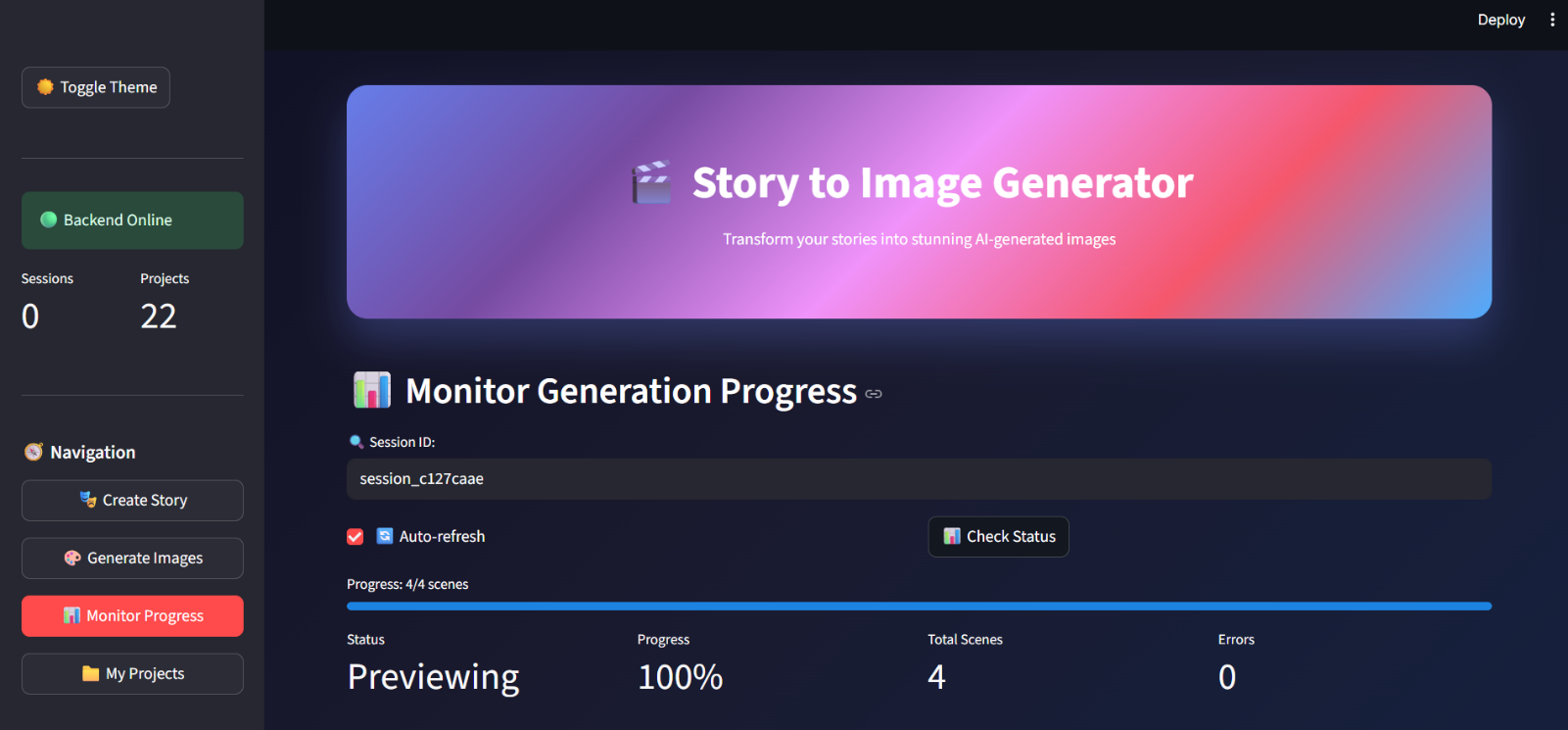
**3.1 Configure Generation Settings**

1. **Navigate to "Generate Images"** (automatically after project creation)
2. **Choose AI Model:**
   * gpt-4o-mini: Fast, cost-effective (recommended)
   * gpt-4.1-mini: Enhanced reasoning capabilities
3. **Select Image Provider:**
   * **Runware**: High-quality, artistic images
     + runware:101@1: General purpose
     + runware:102@1: Enhanced detail
     + runware:103@1: Artistic style
   * **Together AI**: Alternative provider with FLUX models
     + FLUX.1-schnell: Fast generation
     + FLUX.1-standard-Pro: Higher quality
4. **Choose Visual Style:**
   * **Cinematic**: Movie-like dramatic scenes
   * **Cartoon**: Vibrant animated style
   * **Realistic**: Photorealistic images
   * **Artistic**: Painterly, creative interpretation
5. **Set Scene Count:** Use recommended number or customize (1-50 scenes)



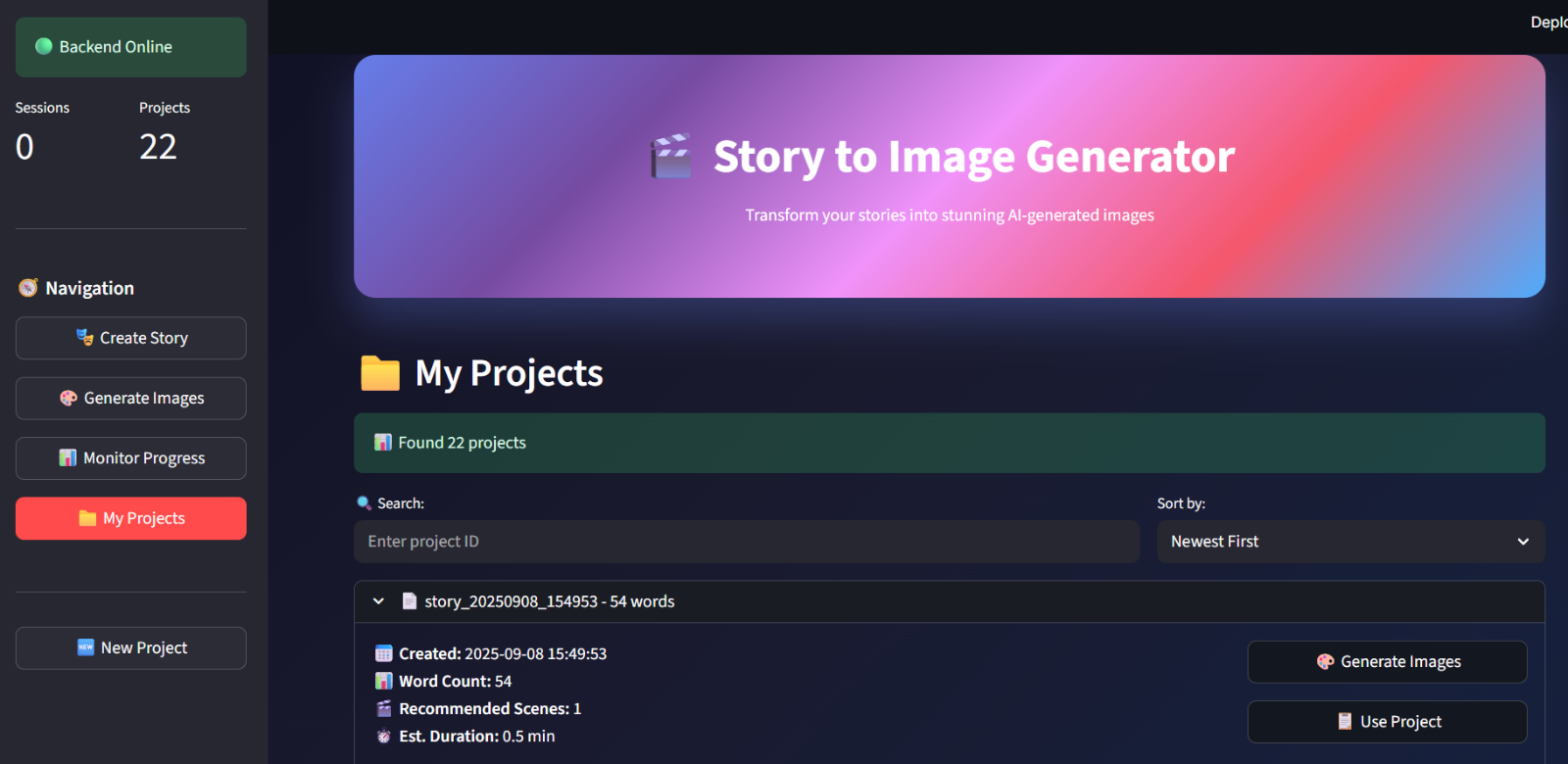
**3.2 Start Generation Process**

1. **Click "Generate Scene Previews"**
2. **Monitor Progress** in "Monitor Progress" section:
   * Real-time progress bar
   * Preview images as they're generated
   * Error notifications if any scenes fail



**3.3 Review and Approve Images**

1. **Preview Generated Scenes:**
   * Each scene displays with title and image
   * Check quality and story accuracy
   * Note any errors or failed generations
2. **Select Images to Save:**
   * Check boxes next to scenes you want to keep
   * Uncheck any you want to regenerate or skip
3. **Regenerate if Needed:**
   * Use regeneration feature for unsatisfactory images
   * Try different providers or models
   * Adjust prompts if necessary
4. **Save Final Images:**
   * Click "Save Selected Scenes"
   * Images are saved to project directory
   * Access via "My Projects" section



**Tutorials**

**Tutorial 1: Creating a Short Story Visualization**

**Objective**: Generate 4 images from a fairy tale story

**Step-by-Step:**

1. **Create Project:**

Title: "The Magic Forest"

Script: "Once upon a time, a young girl named Luna discovered

a glowing path in the dark forest. She followed the golden

lights deeper into the woods, where talking animals guided

her to an ancient tree. Inside the hollow trunk, she found

a crystal that granted her the power to communicate with

all forest creatures."

1. **Configuration:**
   * AI Model: gpt-4o-mini
   * Image Provider: runware
   * Style: artistic
   * Scenes: 4 (as recommended)
2. **Expected Results:**
   * Scene 1: Girl discovering glowing path
   * Scene 2: Following lights through forest
   * Scene 3: Meeting talking animals at ancient tree
   * Scene 4: Finding the magical crystal

**Tutorial 2: Script Formatting for Optimal Results**

**Best Practices:**

Good Example:

"The crimson sunset painted the desert landscape as Captain

Martinez adjusted her dusty hat. Her weathered leather boots

crunched against the red sand as she approached the mysterious

metal structure half-buried in the dunes."

Why it works:

✅ Specific colors (crimson, red)

✅ Character description (weathered, dusty)

✅ Environmental details (desert, dunes)

✅ Clear action and setting

Poor Example:

"She walked somewhere. It was nice. Something happened next."

Why it doesn't work:

❌ Vague descriptions

❌ No visual details

❌ Unclear setting and characters

**Tutorial 3: Troubleshooting Failed Generations**

**Common Issues and Solutions:**

1. **"Generation Failed" Error:**
   * Check API key validity
   * Try different image provider
   * Reduce scene complexity
   * Check internet connection
2. **Poor Image Quality:**
   * Switch to different model (try runware:102@1)
   * Change visual style
   * Improve script descriptions
   * Use more specific prompts
3. **Slow Generation:**
   * Use faster models (FLUX.1-schnell)
   * Reduce number of scenes
   * Check system resources
   * Try during off-peak hours

**FAQs**

**General Usage**

**Q: How many scenes should I generate for my story?** A: The system automatically recommends optimal scene count based on script length. Generally, 1 scene per 20-50 words works well. Short stories (50-100 words) need 2-4 scenes, while longer narratives (300+ words) can have 8-15 scenes.

**Q: Can I use the same project to generate images multiple times?** A: Yes! Each generation creates a new session, so you can experiment with different settings, styles, and providers for the same story without losing previous results.

**Q: What file formats are supported for generated images?** A: All images are saved as high-quality JPG files (1024x1024 pixels) with descriptive filenames like scene\_001.jpg, scene\_002.jpg, etc.

**Technical Issues**

**Q: The backend shows "API key not found" errors** A: Ensure your .env file is in the root directory with correct API key format:

OPENAI\_API\_KEY= your-key-here

RUNWARE\_API\_KEY= your-runware-key

TOGETHER\_API\_KEY= your-together-key

Restart the backend after updating the file.

**Q: Images are not loading in the preview** A: This usually indicates network connectivity issues or API rate limits. Try:

1. Refresh the page
2. Check your internet connection
3. Wait a few minutes if you've hit rate limits
4. Try a different image provider

**Q: The application is running slowly** A: Performance can be improved by:

* Using faster models (gpt-4o-mini, FLUX.1-schnell)
* Reducing concurrent scenes
* Ensuring adequate system memory
* Using a stable internet connection

**Billing and API Usage**

**Q: How much does it cost to generate images?** A: Costs depend on your chosen providers:

* **OpenAI**: ~$0.01-0.03 per scene for prompt generation
* **Runware**: Credit-based, typically $0.05-0.15 per image
* **Together AI**: ~$0.02-0.08 per image

A typical 5-scene project costs approximately $0.50-1.00 total.

**Q: Can I use the application offline?** A: The application requires internet connectivity for AI API calls. However, once images are generated and saved, you can view them offline in your project directories.

**Additional Resources**

**Project Directory Structure**

backend/

├── image\_generation/

│ └── story\_YYYYMMDD\_HHMMSS/ # Your projects

│ ├── script.txt # Original story

│ ├── analysis.json # Analysis results

│ ├── scene\_prompts.txt # Generated prompts

│ └── images/ # Final images

│ ├── scene\_001.jpg

│ ├── scene\_002.jpg

│ └── ...

**API Endpoints (For Advanced Users)**

* GET /: Application status and health check
* POST /analyze-script: Create new project from script
* POST /generate-previews: Start image generation
* GET /generation-status/{session\_id}: Monitor progress
* POST /approve-previews: Save selected images
* GET /projects: List all projects
* GET /models: Available AI and image models

**Configuration Options**

Advanced users can customize behavior in backend/config.py:

# Timeout settings

TIMEOUT = 60 # API request timeout (seconds)

MAX\_RETRIES = 3 # Generation retry attempts

RETRY\_DELAY = 2 # Delay between retries (seconds)

# Model configurations

CONFIG = {

"runware": {

"models": ["runware:101@1", "runware:102@1", "runware:103@1"]

},

"together": {

"models": ["black-forest-labs/FLUX.1-schnell", "..."]

}

}

**Performance Optimization Tips**

1. **Batch Processing**: Generate multiple scenes simultaneously for efficiency
2. **Model Selection**: Use appropriate models for your quality vs. speed requirements
3. **Caching**: Generated images are cached in project directories
4. **Resource Management**: Monitor system memory during large batch generations
5. **API Limits**: Be aware of provider rate limits and plan accordingly