🧪 Human-in-the-Loop Sandbox Generator - Comprehensive Documentation

Table of Contents

Project Overview

The **Human-in-the-Loop Sandbox Generator** *is an innovative educational content creation system that leverages* **LangGraph** *for workflow management and* **AI models** *for content generation. This system enables educators and researchers to create comprehensive educational sandboxes with interactive simulations, assessments, and theoretical content through a collaborative human-AI workflow.*

Key Features

- **Multi-Model AI Support***: Integration with Gemini and OpenAI models*
- **Human-in-the-Loop***: Interactive feedback and content refinement*
- **Progress Tracking***: Real-time progress monitoring and state management*
- **Step-by-Step Generation***: Structured content creation workflow*
- **Interactive Simulations***: Web-based simulations for each sandbox*
- **Dual Interface***: Both CLI and Streamlit GUI options*
- **Complete Package***: All necessary files generated automatically*

Generated Content Structure

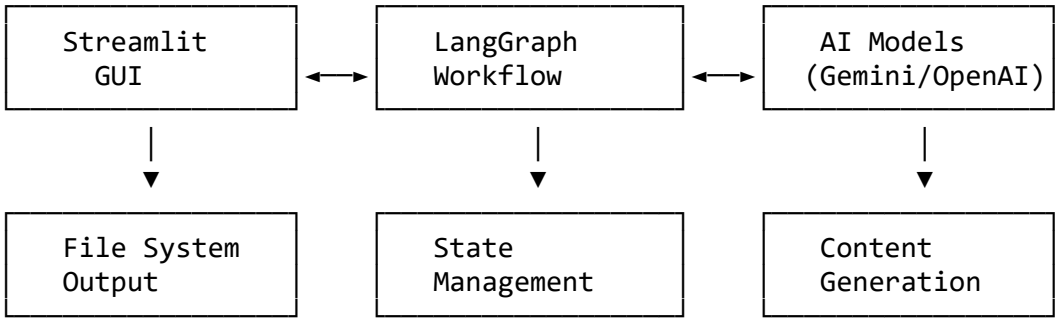
```
sandbox-name/
├── aim.md                  # Learning objectives and goals
├── theory.md               # Theoretical background
├── procedure.md            # Step-by-step instructions
├── pretest.json            # Pre-assessment questions
├── posttest.json           # Post-assessment questions
├── reference.md            # Academic references
├── sandbox-name.md         # Clean file-system name
└── simulation/             # Interactive web simulation
    ├── index.html
```

```
└── src/
    ├── main.js
    └── style.css
```

Architecture & Design

System Architecture

The project follows a **modular, event-driven architecture** *with the following key components:*



Design Patterns

1. **State Machine Pattern***: Uses LangGraph's StateGraph for workflow management*
2. **Observer Pattern***: Real-time UI updates based on state changes*
3. **Factory Pattern***: Dynamic AI model selection and initialization*
4. **Template Method***: Structured content generation with customizable prompts*

Data Flow

1. **Initialization***: User provides sandbox topic and selects AI model*
2. **Workflow Execution***: LangGraph orchestrates step-by-step content generation*
3. **Human Feedback***: Users review and provide feedback at each step*
4. **Content Refinement***: AI models update content based on feedback*
5. **File Generation***: Complete sandbox package is created with all components*
6. **Simulation Creation***: Interactive web simulation is generated*

Installation & Setup

Prerequisites

- *Python 3.8 or higher*
- *pip package manager*
- *API keys for AI models (Google Gemini and/or OpenAI)*

Step 1: Clone the Repository

```
git clone <repository-url>
cd tool-ai-lab-generation-iiith/Project
```

Step 2: Create Virtual Environment

```
python -m venv myenv
source myenv/bin/activate  # On Windows: myenv\Scripts\activate
```

Step 3: Install Dependencies

```
pip install -r requirements.txt
```

Step 4: Environment Configuration

Create a `.env` *file in the Project directory:*

```
# Google Gemini API Key
GOOGLE_API_KEY=your_gemini_api_key_here

# OpenAI API Key (optional)
OPENAI_API_KEY=your_openai_api_key_here
```

Step 5: Verify Installation

```
python -c "import streamlit, langgraph, google.generativeai, openai;
print('All dependencies installed successfully!')"
```

Usage Guide

Option 1: Streamlit GUI (Recommended)

The Streamlit GUI provides an intuitive, web-based interface for sandbox generation.

Starting the GUI

```
streamlit run langgraph_streamlit_gui.py
```

GUI Usage Steps

1. **Model Selection***: Choose your preferred AI model from the sidebar*
2. **Topic Input***: Enter your sandbox topic (e.g., "Newton's Laws of Motion")*
3. **Start Generation***: Click "Start Generation" to begin the workflow*
4. **Review & Feedback***: For each generated component:*
   - *Review the content*
   - *Provide feedback if needed*
   - *Choose to update, save & continue, or skip*
5. **Download***: Once complete, download individual files or the complete package*

GUI Screenshots

[SCREENSHOT PLACEHOLDER 1: Main GUI Interface] Caption: The main Streamlit interface showing the three-column layout with settings, content generation, and chat areas.

[SCREENSHOT PLACEHOLDER 2: Content Review] Caption: Content review interface showing generated aim document with feedback options.

[SCREENSHOT PLACEHOLDER 3: Progress Tracking] Caption: Progress bar and step indicators showing current generation status.

[SCREENSHOT PLACEHOLDER 4: Download Section] Caption: Download options showing individual files and complete package download.

Option 2: Command Line Interface

For users who prefer command-line interaction:

```
python langgraph_cli.py
```

CLI Features

- *Interactive step-by-step generation*
- *Real-time feedback collection*
- *Progress tracking*
- *File download options*

Core Components

1. LangGraph Experiment Generator (`langgraph_experiment_generator.py`)

The core engine that manages the entire sandbox generation workflow.

Key Classes

SandboxState

```python
class SandboxState(TypedDict):
    sandbox_topic: str
    current_step: str
    sandbox_name: str
    aim: str
    pretest: List[Dict[str, Any]]
    posttest: List[Dict[str, Any]]
    theory: str
    procedure: str
    references: str
    user_feedback: str
    user_action: Literal["update", "save", "continue"]
    progress: float
    completed_steps: List[str]
```

```
    system_message: str
    user_message: str
```

SandboxGenerator

Main class responsible for:

- *AI model management*
- *Content generation*
- *Workflow orchestration*
- *File system operations*

Key Methods

- `generate_content(prompt)`*: Generates content using selected AI model*
- `workflow_step(state)`*: Main workflow logic*
- `save_content(state)`*: Saves generated content to files*
- `build_graph()`*: Constructs the LangGraph workflow*

2. Streamlit GUI (`langgraph_streamlit_gui.py`)

Modern web interface built with Streamlit for enhanced user experience.

Features

- **Three-Column Layout***: Settings, content generation, and chat*
- **Real-time Updates***: Live progress tracking and state management*
- **Model Switching***: Dynamic AI model selection*
- **File Downloads***: Individual and package download options*
- **Chat Interface***: Interactive AI assistance*

Key Components

- **Settings Panel***: Model selection and sandbox setup*
- **Content Area***: Step-by-step content review and feedback*
- **Chat Panel***: Interactive AI assistance and guidance*

3. System Prompts (`SystemPrompts`)

Comprehensive prompt templates for each content type:

- `SANDBOX_NAME_PROMPT`*: File-system-friendly naming*
- `AIM_PROMPT`*: Learning objectives and goals*
- `PRETEST_PROMPT`*: Pre-assessment questions*
- `POSTTEST_PROMPT`*: Post-assessment questions*
- `THEORY_PROMPT`*: Theoretical background*
- `PROCEDURE_PROMPT`*: Step-by-step instructions*
- `REFERENCES_PROMPT`*: Academic references*

API Reference

SandboxGenerator Class

Constructor

```
SandboxGenerator(model_name: str = "gemini-2.5-flash-preview-05-20")
```

Methods

`update_model(model_name: str)`

Updates the AI model being used for content generation.

Parameters:

- `model_name`*: Name of the AI model to use*

Supported Models:

- `gemini-2.5-flash-preview-05-20` *(default)*
- `gemini-1.5-flash`
- `gemini-1.5-pro`
- `gpt-4`
- `gpt-3.5-turbo`

`generate_content(prompt: str) -> str`

Generates content using the selected AI model.

Parameters:

- `prompt`*: The prompt to send to the AI model*

Returns:

- *Generated content as a string*

`parse_json_content(content: str) -> List[Dict[str, Any]]`

Parses JSON content from generated text (for quiz questions).

Parameters:

- `content`*: Raw generated content*

Returns:

- *Parsed JSON data as a list of dictionaries*

`save_content(state: SandboxState) -> str`

Saves all generated content to the file system.

Parameters:

- `state`*: Current sandbox state*

Returns:

- *Name of the created sandbox directory*

Workflow Steps

The system follows a 7-step workflow:

1. **Sandbox Name** *(14.3%): Generate file-system-friendly name*
2. **Aim** *(28.6%): Create learning objectives*
3. **Pretest** *(42.9%): Generate pre-assessment questions*
4. **Posttest** *(57.1%): Generate post-assessment questions*
5. **Theory** *(71.4%): Create theoretical background*
6. **Procedure** *(85.7%): Generate step-by-step instructions*
7. **References** *(100%): Compile academic references*

Troubleshooting

Common Issues

1. API Key Errors

Problem*: "API_KEY not found in environment variables"*

Solution*:*

```
# Check if .env file exists
ls -la .env

# Create .env file if missing
echo "GOOGLE_API_KEY=your_key_here" > .env
echo "OPENAI_API_KEY=your_key_here" >> .env
```

2. Model Loading Issues

Problem*: "Unsupported model" error*

Solution*:*

- *Verify model name spelling*
- *Check API key validity*
- *Ensure internet connectivity*

3. Streamlit Port Issues

Problem: *Port already in use*

Solution:

```
# Kill existing Streamlit processes
pkill -f streamlit

# Or specify a different port
streamlit run langgraph_streamlit_gui.py --server.port 8502
```

4. File Permission Errors

Problem: *Cannot create directories or files*

Solution:

```
# Check directory permissions
ls -la

# Fix permissions if needed
chmod 755 .
```

Debug Mode

Enable debug logging by setting environment variable:

```
export DEBUG=1
python langgraph_streamlit_gui.py
```

Areas for Improvement

1. Enhanced AI Model Integration

Current State: *Basic integration with Gemini and OpenAI models*

Improvements:

- *[ ]* **Multi-Model Ensemble**: *Combine multiple AI models for better content quality*
- *[ ]* **Model Performance Metrics**: *Track and compare model performance*
- *[ ]* **Custom Model Support**: *Integration with local models (Llama, Mistral)*
- *[ ]* **Model Fallback**: *Automatic fallback to alternative models on failure*

Implementation Priority: *High* **Estimated Effort**: *2-3 weeks*

2. Advanced Content Generation

Current State: *Basic content generation with simple prompts*

Improvements:

- [ ] **Content Templates**: *Pre-defined templates for different subjects*
- [ ] **Multilingual Support**: *Generate content in multiple languages*
- [ ] **Content Validation**: *AI-powered content quality assessment*
- [ ] **Version Control**: *Track content changes and revisions*

Implementation Priority: *High* **Estimated Effort**: *3-4 weeks*

3. Enhanced Simulation Engine

Current State: *Basic HTML/JS/CSS simulation template*

Improvements:

- [ ] **Physics Engine**: *Integration with real physics simulation libraries*
- [ ] **3D Visualizations**: *Three.js integration for 3D simulations*
- [ ] **Interactive Elements**: *Drag-and-drop, real-time parameter adjustment*
- [ ] **Data Export**: *Export simulation data for analysis*

Implementation Priority: *Medium* **Estimated Effort**: *4-6 weeks*

4. User Experience Enhancements

Current State: *Functional but basic UI*

Improvements:

- [ ] **Dark Mode**: *Toggle between light and dark themes*
- [ ] **Responsive Design**: *Better mobile and tablet support*
- [ ] **Keyboard Shortcuts**: *Power user shortcuts for faster navigation*
- [ ] **Customizable Layout**: *User-configurable interface layout*

Implementation Priority: *Medium* **Estimated Effort**: *2-3 weeks*

5. Collaboration Features

Current State: *Single-user system*

Improvements:

- [ ] **Multi-User Support**: *Team collaboration on sandbox creation*
- [ ] **Version History**: *Track changes and collaboration history*
- [ ] **Comments System**: *Inline comments and feedback*
- [ ] **Sharing**: *Share sandboxes with other users*

Implementation Priority: *Low* **Estimated Effort**: *6-8 weeks*

6. Analytics and Reporting

Current State*: No analytics or reporting*

Improvements*:*

- *[ ]* **Usage Analytics***: Track user behavior and preferences*
- *[ ]* **Content Performance***: Measure sandbox effectiveness*
- *[ ]* **Generation Metrics***: Track generation time and success rates*
- *[ ]* **Export Reports***: Generate detailed reports for stakeholders*

Implementation Priority*: Low* **Estimated Effort***: 3-4 weeks*

7. Advanced Workflow Features

Current State*: Linear workflow with basic branching*

Improvements*:*

- *[ ]* **Custom Workflows***: User-defined workflow steps*
- *[ ]* **Parallel Processing***: Generate multiple components simultaneously*
- *[ ]* **Conditional Logic***: Smart branching based on content type*
- *[ ]* **Workflow Templates***: Pre-defined workflow templates*

Implementation Priority*: Medium* **Estimated Effort***: 4-5 weeks*

8. Security and Privacy

Current State*: Basic API key management*

Improvements*:*

- *[ ]* **Encrypted Storage***: Secure storage of sensitive data*
- *[ ]* **Access Control***: Role-based access control*
- *[ ]* **Audit Logging***: Comprehensive audit trails*
- *[ ]* **Data Privacy***: GDPR compliance and data anonymization*

Implementation Priority*: High* **Estimated Effort***: 3-4 weeks*

Contributing

Development Setup

1. **Fork the Repository**
2. **Create Feature Branch**: `git checkout -b feature/amazing-feature`
3. **Make Changes***: Implement your improvements*
4. **Test Thoroughly***: Ensure all functionality works*
5. **Submit Pull Request***: Detailed description of changes*

Code Style Guidelines

- *Follow PEP 8 for Python code*
- *Use type hints for all function parameters*
- *Add docstrings for all classes and methods*
- *Write unit tests for new functionality*

Testing

```
# Run basic tests
python -m pytest tests/

# Run with coverage
python -m pytest --cov=. tests/
```

Conclusion

The Human-in-the-Loop Sandbox Generator represents a significant advancement in educational content creation, combining the power of AI with human expertise through an intuitive, interactive interface. The system's modular architecture and comprehensive feature set make it a valuable tool for educators, researchers, and content creators.

The project demonstrates the potential of LangGraph for complex workflow management and showcases how AI can enhance rather than replace human creativity in educational content development. With the planned improvements, this system has the potential to become a cornerstone tool in modern educational technology.

For questions, issues, or contributions, please refer to the project repository or contact the development team.

🧪 Happy Sandbox Generation!

Last Updated: [Current Date] Version: 1.0.0