Visual Learning Pattern Analysis + AI Video Synthesis System

Task 2 – Banao Tech A.I

Name: Akash Yaduwanshi
Email: aakashyaduwanshi0470@gmail.com
Availability: Part Time
Date: 03/01/2026

---

1. Introduction

This document presents the design and prototype of a **Visual Learning Pattern Analysis and AI Video Synthesis System**.
The objective of the system is to analyze the visual teaching style of a reference explainer video and automatically generate new educational videos in the same style for any given technical topic.

The system follows a **human-in-the-loop + AI automation approach**, where manual visual understanding is combined with structured AI-based generation.

---

2. PART A – Visual Learning Pattern Analysis (Manual)

2.1 Reference Video Overview

- **Video Title:** *How WebSockets Work | Deep Dive*
- **Video Type:** Technical explainer / system architecture breakdown
- **Target Audience:** Software engineers and system design learners
- **Teaching Approach:** Diagram-driven explanation with narration support

---

2.2 Visualization Style Classification

- **Primary Style:**
    - 2D schematic explainer animation
    - Vector-based diagrams
- **Secondary Style:**
    - Typing-style text animation
    - Slide-in transitions between topics
- **Excluded Styles:**
    - No character-based animation

- No whiteboard or hand-drawn visuals
- No 3D or cinematic effects

2.3 Scene Structure Pattern

Each scene in the video follows a consistent pattern:

1. A clean canvas is introduced
2. System components (client/server) appear first
3. Arrows animate to represent data flow
4. Text labels appear using typing animation
5. The scene ends with a stable visual state

2.4 Visual Elements Used

- Rectangular boxes for system components
- Arrows for communication flow
- Text blocks for headers, tokens, and data
- Color highlights for active states

2.5 Animation Rules

- Elements appear before arrows
- Arrows animate left-to-right or right-to-left
- Text appears using keyboard-style typing animation
- Animations are smooth and non-complex

2.6 Color, Typography & Timing

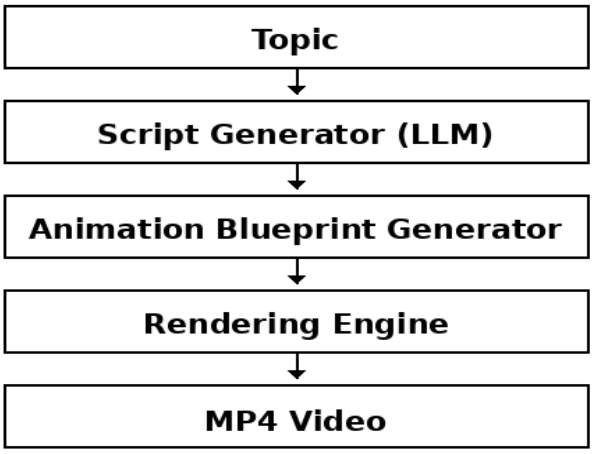
- Light background for readability
- Blue as primary accent color
- Green for success or validation states
- Sans-serif font for labels
- Monospace font for code/data
- Average scene duration: 45–60 seconds

---

3. PART B – Automatic System (Prototype)

3.1 System Architecture Overview

The system is designed as a **deterministic multi-stage AI pipeline**:



Each stage produces a structured intermediate output used by the next stage.

3.2 Stage 1 – Topic to Script Generation

Given a technical topic, the AI generates:

- Scene-by-scene narration
- Short technical explanations
- Voice-over ready text

Example Output (JSON – snippet):

```
{
  "scene_id": 2,
  "concept": "Token generation",
  "narration": "The server validates credentials and generates a JWT."
}
```

3.3 Stage 2 – Script to Animation Blueprint

The script is converted into a visual plan using the predefined style profile.

The blueprint defines:

- Scene layout
- Visual elements (boxes, arrows, text)

- Animation actions
- Timing and transitions

Example Output (JSON – snippet):

```
{
  "scene_id": 2,
  "visual_elements": ["server_box", "jwt_text", "arrow"],
  "animation": "draw_arrow"
}
```

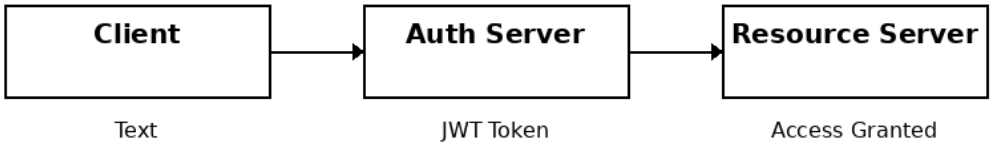


Figure 2 visually represents the animation blueprint defined in the JSON snippet above, where system components are mapped to visual boxes and communication is represented using directional arrows.

3.4 Stage 3 – Blueprint to MP4 Rendering Plan

The blueprint is translated into a rendering plan using supported tools:

- **Manim / Lottie** for animation
- **FFmpeg** for final MP4 generation
- **Optional AI TTS** for narration

This stage defines how the final video would be generated programmatically.

---

4. Role of JSON in the System

The system uses **structured JSON artifacts** as intermediate representations between pipeline stages.

These JSON files:

- Act as deterministic contracts
- Enable automation and repeatability

- Separate planning from rendering

The final MP4 video is treated as a **compiled output**, not the core system artifact.

---

5. Conclusion

This assignment demonstrates the design of a **production-style AI system** for automated educational video generation.
By combining manual visual analysis with structured AI automation, the system ensures consistency, scalability, and clarity in video synthesis.

The approach reflects real-world AI system design practices used in content automation and learning platforms.

---

6. Future Improvements

- Full Manim-based implementation
- Automatic narration synchronization
- Multi-style video generation support
- Web-based interface for topic input