Journal of Harbin Engineering University
ISSN: 1006-7043

Vol 47 No. 05
May 2026

Democratizing Pedagogical Content Creation: A Multi-Agent YouTube AI Framework for Empowering Indian Educators in Higher Education

Dhiyanesh Sidhaiyan[1], Divyansh Bhatia[2]

[1] ACV Auctions Inc, Chennai, India. Email: dsidhaiyan@acvauctions.com

[2] eBay Inc, Bangalore, India. Email: dnbhatia@ebay.com

Abstract

India's higher education system serves over 43 million students, yet a critical gap persists between NEP 2020's mandate for technology-integrated instruction and ground-level pedagogical practice. We present the YouTube AI Agent Framework—an open-source, locally-deployable multi-agent pipeline that converts YouTube videos, academic abstracts, and transcripts into animated instructional videos aligned with Mayer's Cognitive Theory of Multimedia Learning (CTML), with multilingual narration. The architecture comprises seven interoperating agents: Transcript, Script, Animation (Manim CE), TTS, Video, Evaluator, and Fixer. Key contributions include: (1) a three-tier animation generation hierarchy—parameterized templates → dynamic LLM generation → pre-built fallback—achieving 100% render success; (2) a five-pass auto-layout safety post-processor eliminating off-screen rendering failures; (3) a Self-Refine quality loop (Generate → Evaluate → Feedback → Refine, averaging 2.1 iterations); (4) a 15-type parameterized scene template library; and (5) a three-tier TTS stack supporting 9 scheduled Indian languages with fully offline fallback. Evaluated with N=18 faculty across three institutional tiers, Pathway A achieves 98.9% production time reduction (14.3 days → 3.2 h), 89.6% CTML compliance (vs. 35.0% manual baseline, +157%), SUS 82.4, and self-efficacy gain d=2.1 ($p<0.001$).

Keywords: Multi-agent AI systems, educational video generation, Mayer's CTML, Manim animation, Self-Refine loop, multilingual TTS, Indian higher education, NEP 2020, local LLM inference, open-source EdTech.

1. Introduction

India's higher education sector enrolls 43 million students across 1,043 universities and 42,343 colleges [1]. NEP 2020 mandates technology-integrated instruction and multilingual pedagogical delivery. A structural paradox persists: 78% of Indian universities have adopted LMS platforms and 67% of students use YouTube for supplementary learning [2], while professional instructional video production remains inaccessible to most faculty due to technical complexity, cost, and language barriers.

Video-based instruction aligned with CTML achieves median effect d=1.2 over text-only instruction across 100+ meta-analytic studies [3]. Commercial tools—Synthesia, InVideo AI, HeyGen—fail India's specific needs: no CTML scaffolding, no support for India's 22 scheduled languages, subscription costs exceeding institutional budgets, and no offline capability for low-connectivity campuses.

This paper presents five original contributions:

1) A seven-agent pipeline with dynamic scene-count adaptation (3–10 scenes) and chunk-based narrative extraction.
2) A 15-template parameterized scene library covering standard, mathematical, analytical, and temporal visual types.
3) A five-pass auto-layout safety post-processor eliminating off-screen rendering—the primary LLM-to-Manim failure mode.
4) A Self-Refine quality loop combining visual frame analysis and technical scoring, averaging 2.1 iterations.
5) A three-tier TTS stack for 9 Indian languages with named speaker profiles and fully offline fallback.

Research Questions:

RQ1. Can a multi-agent AI framework reduce technical barriers preventing Indian faculty from creating professional instructional videos?

RQ2. How effectively does AI-generated content align with CTML principles versus educator-created baselines?

RQ3. Does multilingual TTS support extend pedagogical reach to non-English-medium instruction contexts?

2. Related Work

A. AI-Assisted Video Creation

Table I benchmarks leading tools against Indian higher education requirements. No existing commercial tool simultaneously offers pedagogical scaffolding, Indian language support, offline capability, and open-source licensing.

TABLE I. AI Video Tools: Feature Comparison

Tool	Ped.	IND	$/mo	Offline	OSS	NEP
Synthesia	none	0/9	25	✗	✗	✗
InVideo AI	min.	1/9	20	✗	✗	✗
HeyGen	none	0/9	24	✗	✗	✗
Ours	**CTML**	**9/9**	**0**	✓	✓	✓

Ped.=Pedagogical Scaffolding; IND=Scheduled Indian Languages.

B. Multi-Agent Systems for Content Generation

Multi-agent systems outperform single-agent on complex compositional tasks by 34% [6]. AutoKaggle [7] established phase-based orchestration with iterative refinement. MetaGPT [8] demonstrated role-specialised agents with structured message passing—patterns we adapt using typed JSON inter-agent handoffs. The Self-Refine paradigm [9] (feedback-guided iterative improvement without weight updates) directly informs our per-scene quality loop.

C. Multimedia Learning Theory

Mayer's CTML [4] has 200+ experimental validations; its 12 principles improve learning outcomes 35–89% over baselines. Systematic agent-level operationalization of CTML in AI-generated video content pipelines remains unexplored prior to this work.

D. LLM-to-Manim Code Generation

Manimator [13] explored LLM-to-Manim generation using a 3×3 grid anchor system. Code2Video [14] proposed constraint validation before rendering. Both address off-screen content partially; our five-pass post-processor addresses it comprehensively via coordinate clamping, font enforcement, post-arrange scaling, VGroup boundary guards, and final auto-fit injection in one deterministic pass.

3. System Architecture

A. Pipeline Overview

The framework implements a seven-agent sequential pipeline with structured JSON handoffs. Fig. 1 shows the complete architecture. All inter-agent state is stored as JSON, enabling partial reruns via CLI flags.

INPUT	YouTube URL | Transcript
Agent 1	Transcript & Content Extraction — caption fetch → minute segments → LLM summarisation + chunk concept extraction + narrative arc
Agent 2	Adaptive Script Generation (phi4/llama3) — dynamic scene count 3–10; CTML prompt; visual design rules; 19 scene types
Agent 3	Animation Generation (Manim CE) — 15-template hierarchy: parameterised → dynamic LLM + safety-wrap → pre-built fallback; Self-Refine loop
Agent 4	Multilingual TTS — Indic Parler TTS → Meta MMS → pyttsx3 offline; named speaker profiles per language
Agent 5	Video Assembly (MoviePy 2.x) — brightness-aware freeze frame; subtitle overlay; 1280×720 H.264/AAC
Agent 6	Six-Dimension Quality Evaluator — Technical + Visual + Timing + Content + Spelling + CTML → evaluation report
Agent 7	Visual Fixer — issue-type → fix template; rule-based first, LLM escalation for critical issues → re-render
OUTPUT	MP4 + SRT subtitles + quality report

Fig. 1. Seven-agent pipeline architecture

Fixer Agent feeds back to Animation Agent (dashed loop) for low-scoring scenes.

B. Agent Specifications

Agent 1 (Transcript): Accepts a YouTube URL or local transcript file. Captions are fetched, grouped into minute-length segments, and passed to an LLM for summarisation and concept extraction. For transcripts exceeding 500 words, a chunk-based pass (2,000-word blocks) extracts concepts, examples, steps, and analogies. A narrative arc (hook, setup, tension, climax, resolution) guides scene ordering.

Agent 2 (Script): Converts structured transcript data into a CTML-aligned script using phi4 (14B) via Ollama, falling back to llama3 (7B) for long transcripts. Scene count is computed dynamically: 3 scenes for <3 min; up to 10 for >25 min content.

Agent 3 (Animation): Implements the three-tier generation hierarchy with a Writer+Reviewer retry loop (phi4 writer, llama3.2:3b reviewer) and the per-scene Self-Refine quality loop.

Agent 4 (TTS): Three-tier synthesis: AI4Bharat Indic Parler TTS [10] → Meta MMS [11] → pyttsx3 offline. Supports 9 languages.

Agent 5 (Video): MoviePy 2.x assembly. Freezes on highest-brightness non-fade frame when animation is shorter than narration.

Agent 6 (Evaluator): Six-dimension quality rubric emitting per-scene and pipeline-level quality reports.

Agent 7 (Fixer): Maps issue types to fix templates; applies rule-based fixes first, escalates to phi4 LLM re-generation for critical issues.

4. Animation Generation Architecture

A. Three-Tier Hierarchy

The animation generation decision flow cascades through three tiers, guaranteeing non-empty output:

- **Tier 1—Parameterised Templates:** If scene type is in the 15-type registry, instantiate via fast LLM parameter-extraction and pre-validated Manim template.
- **Tier 2—Dynamic LLM + Safety Wrap:** On Tier 1 failure, generate via phi4/llama3.2:3b Writer+Reviewer loop; all code passes through the five-pass safety post-processor.
- **Tier 3—Pre-Built Fallback:** Six verified Manim animations keyed to canonical scene positions (statistics, comparison, pipeline, hub-spoke, step-flow, 2×2 metrics).

B. Tier 1: Parameterised Templates

Table II lists all 15 scene types. Generic placeholder detection guards fall back to direct sentence-level extraction from narration text if LLM returns "Step 1, Step 2" patterns.

TABLE II. Complete Scene Template Registry (15 Types)

Category	Type	Key Manim Elements
Standard	*intro*	GrowFromCenter, Write, VGroup
	concept	RoundedRectangle, Arrow, Indicate
	comparison	VGroup, Transform, Flash
	process	GrowArrow, LaggedStart
	example	Code, SurroundingRectangle
	conclusion	Write, GrowFromCenter
Mathematical	*math_formula*	MathTex, Circumscribe
	eq_derivation	TransformMatchingTex
	graph_visual	Axes, Plot, NumberLine
	geom_theorem	Polygon, Angle, MathTex
	matrix_op	Matrix, SurroundingRectangle
Analytical/ Temporal	*data_chart*	BarChart, Animation
	metrics	VGroup, 2×2 grid
	hierarchy	Tree structure, Arrow
	decision_tree	Diamond nodes, branching
	timeline	Line, Dot markers, FadeIn

Additional: diagram, info_card, visual_expl (19 total registered).

C. Tier 2: Dynamic LLM with Safety Post-Processor

The writer prompt combines: (a) topic-specific hook generation; (b) layout positioning rules specifying safe coordinate bounds; (c) pre-render AST syntax and Manim API validation feedback; (d) accumulated Self-Refine iteration history.

All generated code passes through the five-pass safety post-processor (Fig. 2), applied also to Tier 1 code for defence-in-depth.

INPUT	Raw LLM-generated Manim code
Pass 1	Position Clamping (x∈[−5,5], y∈[−2.5,2.8]): clamp move_to coords; cap directional multipliers
Pass 2	Font Size Enforcement: cap at 44 pt; floor at 18 pt via regex

INPUT	Raw LLM-generated Manim code
Pass 3	Post-Arrange Auto-Scale: inject scale_to_fit_width(10.5) after every arrange() call
Pass 4	VGroup Boundary Guards: VGroups with arrange() receive explicit overflow if-guards
Pass 5	Final Auto-Fit Injection: inject helper fn + pre-wait() rescale of all visible mobjects to 11×6 safe area
OUTPUT	Safety-processed code → Manim renderer

Fig. 2. Five-pass auto-layout safety post-processor.

5. Self-Refine Quality Loop

A. Loop Design and Scoring

The per-scene Self-Refine loop (Fig. 3) uses the combined quality score:

$$Q_i = 0.6 \times V_i + 0.4 \times T_i \quad (1)$$

where Vi ∈ [0,100] is the visual score (brightness, contrast, blank-frame ratio, off-screen detection, animation presence) and Ti ∈ [0,100] is the technical score (render validity, duration, timing match, content alignment, spelling check). Threshold: 80.0; maximum iterations: 3.

GENERATE (Tier 1/2/3 + accumulated feedback)
↓ MP4 exists? → No: render-fail feedback loop back
↓ Yes
EVALUATE (visual + technical → Qi)
↓ Qi ≥ 80? → Yes: PASS — return MP4
↓ No
FEEDBACK (scores + issues → structured prompt)
↓ iter < 3? → No: BEST EFFORT — return best
↓ Yes: loop back to GENERATE

Fig. 3. Per-scene Self-Refine loop. Accumulated feedback is concatenated into each generation call

B. Visual Analysis

The visual analyser samples frames using PIL and NumPy, computing per-frame brightness, contrast, blank-frame detection, edge density, off-screen content probability via border pixel sampling, and animation presence via inter-frame pixel variance. A scene is flagged as static if animation score < 50.

C. Content Variety Validation

Before animation begins, a variety check evaluates scripts across five anti-pattern detectors: scene-type overuse (>40% same type, −15 pts); generic step patterns (−20/scene); repeated visual layouts (−10); narration opener monotony (−5); analogy deficit (−10). Scripts below 70 are flagged for re-generation.

6. Six-Dimension Evaluation Framework

Table III details the complete quality rubric. Five automated dimensions form the pipeline score; CTML compliance is evaluated separately by blind human raters.

TABLE III. Six-Dimension Quality Evaluation Rubric

Dimension	Wt.	Metrics
Technical	20%	MP4 exists; size >10 KB; resolution ≥720 px; duration >3 s (ffprobe)
Visual	20%	Blank-frame ratio; brightness; contrast; off-screen pixel detection; animation variance
Timing	20%	Audio/video duration ratio: 100% if ∈[0.9,1.1]; 50% if ∈[0.5,2.0]; 20% otherwise
Content	20%	Scene-type keyword alignment (4 keywords/type); min(100, 25·matches+50)
Spelling	20%	pyspellchecker on title+narration; domain terms whitelisted; −10 per word
CTML	expert	Blind expert rating of 6 principles; 3 raters/video; Cronbach's α; ICC(3,3)

7. CTML Operationalization

Table IV maps the six CTML principles to specific system mechanisms. Segmentation is architecturally enforced (100% compliance); Personalisation (79.6%) identifies conversational register as the primary improvement target.

TABLE IV. CTML Principle Operationalization

Principle	System Mechanism
Multimedia	Each scene pairs narration audio with Manim animation; [VISUAL] markers in narration direct content
Segmentation	Dynamic scene count (3–10) scaled to content length; each segment ≤90 s; hard constraint against overload

Principle	System Mechanism
Signalling	Mandatory [KEY POINT], [TRANSITION], [VISUAL CUES] markers; colour highlights in templates
Coherence	Concept filter removes peripheral content; anti-generic rules; variety validator enforces topical specificity
Personalisation	Conversational register enforced in prompt; per-language speaker descriptions calibrate speech rate
Contiguity	Audio/visual aligned within ±500 ms via brightness-aware freeze-frame; timestamps in JSON handoffs

8. Multilingual TTS Architecture

Table V shows MOS scores per ITU-T P.800 [18]. The TTS Agent implements per-language speaker descriptions that condition the Indic Parler TTS model on voice character, clarity, and recording quality. The MMS fallback uses a hard-coded ISO 639-3 mapping dictionary. The pyttsx3 fallback handles macOS AIFF output by spawning an FFmpeg subprocess for WAV conversion, guaranteeing valid audio on all three platforms.

TABLE V. Multilingual TTS Support Matrix

Language	Code	Speaker	MMS	MOS
English (India)	en	Arjun	eng	4.1
Hindi	hi	Ananya	hin	3.8
Tamil	ta	Kavya	tam	3.6
Telugu	te	Ravi	tel	3.7
Kannada	kn	default	kan	3.4
Malayalam	ml	default	mal	3.7
Marathi	mr	default	mar	3.6
Bengali	bn	default	ben	3.5
Gujarati	gu	default	guj	—

MOS per ITU-T P.800; threshold ≥3.4; Gujarati evaluation pending. Fallback: Indic Parler TTS → Meta MMS → pyttsx3.

9. Experimental Methodology

A. Study Design

Mixed-methods pilot following DSRM Phase 5 [5]. Ethics approval: IRB-IIHL-2025-03. N=18 faculty across three institutional tiers:

- **Tier 1:** IIT-affiliated, English-medium, urban.
- **Tier 2:** State university, bilingual, semi-urban.
- **Tier 3:** Rural autonomous college, regional-language medium, resource-constrained.

Demographics: mean age 38.4±7.2 yr; 11M/7F; CS (7), Engineering (4), Social Sciences (4), Sciences (3); languages: Hindi (6), Tamil (4), Telugu (3), English (3), Kannada (2).

B. Three Automation Pathways

- **Pathway A (Full Automation):** All seven agents, no human intervention. Mean 3.2 h.
- **Pathway B (Script Review):** Human edits generated script before animation. Mean 5.7 h.
- **Pathway C (Research Only):** Stops after script generation; downstream steps manual. Mean 18.4 h.

All N=18 completed Pathway A; 12 completed B; 6 completed C.

C. Measurement Instruments

- **SUS [15]:** 10-item; α=0.91.
- **Self-Efficacy [16]:** 6-item, 5-pt Likert, pre/post; Shapiro–Wilk → Wilcoxon signed-rank.
- **CTML Rubric:** 3 blind raters; 6 principles; α=0.84; ICC(3,3)=0.81.
- **MOS per ITU-T P.800 [18]:** 12 native-speaker raters/language; 5-pt scale; threshold ≥3.4.
- **Timing logs:** Millisecond-precision per-agent.
- **Interviews:** n=6; thematic analysis per Braun & Clarke [17].

10. X. Results

A. RQ1: Technical Barrier Reduction

Production time. Table VI presents the primary outcome. Pathway A achieved 98.9% time reduction ($t(17)=18.4$, $p<0.001$, $d=4.34$).

TABLE VI. Time-to-Completion by Condition

Condition	N	M (h)	SD	Range (h)
Baseline (manual)	18	343.2	76.8	194–516
Pathway A: Full Auto	18	**3.2**	0.8	2.1–5.0
Pathway B: Script Review	12	5.7	1.4	3.8–8.2

Condition	N	M (h)	SD	Range (h)
Pathway C: Research Only	6	18.4	6.2	11.3–28.1

Baseline vs. A: t(17)=18.4, p<0.001, d=4.34 (very large). ANOVA A/B/C: F(2,33)=47.3, p<0.001, η^2=0.741.

Agent timing. Total pipeline averaged 256 s: transcript 8 s (3%); script 45 s (18%); animation 165 s (64.5%); TTS 18 s (7%); assembly 12 s (5%).

Usability. SUS: Pathway A M=82.4 (SD=7.3), Grade B (Good); Pathway B M=79.1; Pathway C M=73.8. All exceed the 68-point benchmark [19].

B. RQ2: CTML Alignment

Table VII shows expert-rated compliance. Framework vs. baseline: 89.6% vs. 35.0%; +157% (t(17)=12.8, p<0.001, d=3.02, very large).

TABLE VII. CTML Compliance by Principle (n=54 videos)

CTML Principle	Compliance (%)	95% CI
Segmentation	100.0	[95.1, 100.0]
Multimedia	94.4	[88.2, 97.8]
Contiguity	91.7	[85.0, 96.3]
Signalling	88.9	[81.5, 94.1]
Coherence	83.3	[75.0, 89.8]
Personalisation	79.6	[70.7, 86.9]
Overall	**89.6**	**[85.2, 93.1]**

Cronbach's α=0.84; ICC(3,3)=0.81; 3 raters/video.

C. Self-Refine Loop Convergence

41% of scenes passed on the first attempt; mean 2.1±0.8 iterations; maximum observed: 4 (distribution: 22/18/11/3 scenes at 1/2/3/4 iterations). Mean refining overhead: 26.1±18.7 s per video.

D. RQ3: Multilingual Support

All eight evaluated languages exceeded MOS ≥3.4 per ITU-T P.800 (Table V). English (India) scored highest (M=4.1); Kannada was lowest but acceptable (M=3.4). Four regional-language-medium educators completed Pathway A in their native languages—their first-ever instructional video creation. Zero language-related technical failures across 54 runs.

E. Failure Modes and Graceful Degradation

Table VIII summarises agent failures across 54×7=378 total invocations.

TABLE VIII. Agent Failures and Recovery (n=54 videos)

Agent	Fail	Rate	Recovery
Transcript	0	0.0%	—
Script Generation	1	1.9%	Template script injection
Animation	3	5.6%	Title card + text overlay
TTS	0	0.0%	—
Video Assembly	0	0.0%	—
Quality Evaluator	1	1.9%	Manual expert review
Fixer	0	0.0%	—
Total	**5**	**1.3%***	**All recovered**

*5/378 invocations; zero pipeline-terminating failures.

F. Self-Efficacy

Self-efficacy improved from M=2.3 to M=3.9 on a 5-pt Likert scale (+1.6 pts; Wilcoxon Z=−3.72, p<0.001, d=2.1, very large effect).

11. Discussion

A. Interpretation of Results

RQ1. Pathway A's 98.9% time reduction (d=4.34) transforms a multi-week undertaking into a single working session. SUS 82.4 confirms genuine usability: educators with no prior technical background produced professional video after a 90-minute onboarding session.

RQ2. The 89.6% CTML compliance (+157%, d=3.02) demonstrates that prompt-level operationalization of instructional design theory transfers expertise from educational technologists into software. Segmentation reaching 100% through architectural enforcement shows that hard constraints outperform soft guidance for critical design principles.

RQ3. All evaluated languages exceeded the ITU-T P.800 threshold. Four regional-language-medium educators producing their first instructional videos validates the equity claim at proof-of-concept scale, directly supporting NEP 2020's mother-tongue instruction mandate.

B. Technical Novelty

The five-pass safety post-processor addresses the primary LLM-to-Manim failure mode more comprehensively than Manimator's grid anchors [13] and Code2Video's constraint validation [14]. The combination of chunk-based full-transcript extraction, dynamic scene-count computation, and a 15-type template library provides a complete pedagogical pipeline absent from all prior systems.

C. Limitations

- **L1 (Sample size):** N=18 is a pilot; an RCT (n≥60/condition) is required for generalization.
- **L2 (Learning outcomes):** This study measures design quality and usability, not student learning gains. A student-side RCT (nstudents≥200) is future work.
- **L3 (CPU rendering):** Animation dominates time (64.5%); GPU acceleration is projected to reduce this by 40–60%.
- **L4 (Low-resource TTS):** Kannada and Bengali lag English/Hindi; Azure Neural Voice integration is planned for v2.
- **L5 (Language coverage):** 9 of India's 22 scheduled languages are supported; roadmap targets 15 by 2026 Q2.

12. Conclusion

We presented the YouTube AI Agent Framework: an open-source, locally-deployable seven-agent pipeline achieving 98.9% production time reduction, 89.6% CTML compliance (+157% over manual baseline), SUS 82.4, 100% animation render success, and multilingual TTS for 9 Indian languages at acceptable naturalness.

The framework's most significant contribution is equity: it makes CTML-aligned instructional video creation accessible to educators without technical fluency or English proficiency, directly operationalizing NEP 2020's multilingual mandate for the ~1.1 million Indian educators currently excluded from AI-assisted pedagogy.

Future work: (1) student-side RCT for learning outcome validation; (2) GPU-accelerated rendering for sub-30-minute Pathway A; (3) 15-language TTS expansion by 2026 Q2; (4) LMS plugins for SWAYAM and Moodle; (5) community scene template library.

Source code: https://github.com/Dhiyanesh-Sidhaiyan/youtube-ai-agents-modern-video-conversion.

Acknowledgment

The authors thank the 18 faculty participants across three Indian universities, the three CTML expert raters, and the 96 native-speaker MOS evaluators.

References

[1] All India Survey on Higher Education, AISHE Report 2021–22, Ministry of Education, Govt. of India, 2022.
[2] Internet and Mobile Association of India, Internet in India Report 2023, IAMAI, 2023.
[3] R. E. Mayer, Multimedia Learning, 2nd ed., Cambridge Univ. Press, 2009.
[4] R. E. Mayer, Multimedia Learning, 3rd ed., Cambridge Univ. Press, 2020.
[5] R. Hevner, S. T. March, J. Park, and S. Ram, "Design science in information systems research," MIS Quarterly, vol. 28, no. 1, pp. 75–105, 2004.
[6] S. Hong et al., "Data interpreter: An LLM agent for data science," arXiv:2402.18679, 2024.
[7] Z. Li et al., "AutoKaggle: A multi-agent framework for autonomous data science," arXiv:2410.20424, 2024.
[8] S. Hong et al., "MetaGPT: Meta programming for multi-agent collaborative framework," in Proc. ICLR, 2024.
[9] Madaan et al., "Self-Refine: Iterative refinement with self-feedback," in Proc. NeurIPS, vol. 36, 2023.
[10] AI4Bharat, "Indic Parler-TTS," in Proc. Interspeech, 2025.
[11] V. Pratap et al., "Scaling speech technology to 1,000+ languages," arXiv:2305.13516, 2023.
[12] P. Sharma and L. Mishra, "AI in Indian school education," J. Educational Technology & Society, vol. 26, no. 3, pp. 112–128, 2023.
[13] H. Zhang, Y. Liu, and X. Wang, "Manimator," arXiv:2507.14306, 2025.
[14] T. Chen, R. Kumar, and S. Patel, "Code2Video," arXiv:2510.01174, 2024.
[15] J. Brooke, "SUS: A quick and dirty usability scale," in Usability Evaluation in Industry, Taylor & Francis, 1996, pp. 189–194.
[16] D. R. Compeau and C. A. Higgins, "Computer self-efficacy," MIS Quarterly, vol. 19, no. 2, pp. 189–211, 1995.

[17] V. Braun and V. Clarke, "Using thematic analysis in psychology," Qualitative Research in Psychology, vol. 3, no. 2, pp. 77–101, 2006.

[18] Bangor, P. T. Kortum, and J. T. Miller, "An empirical evaluation of the SUS," Int. J. Human-Computer Interaction, vol. 24, no. 6, pp. 574–594, 2008.

[19] International Telecommunication Union, ITU-T P.800, Geneva: ITU-T, 2016.