Entertainment Content Generator — High Level Design

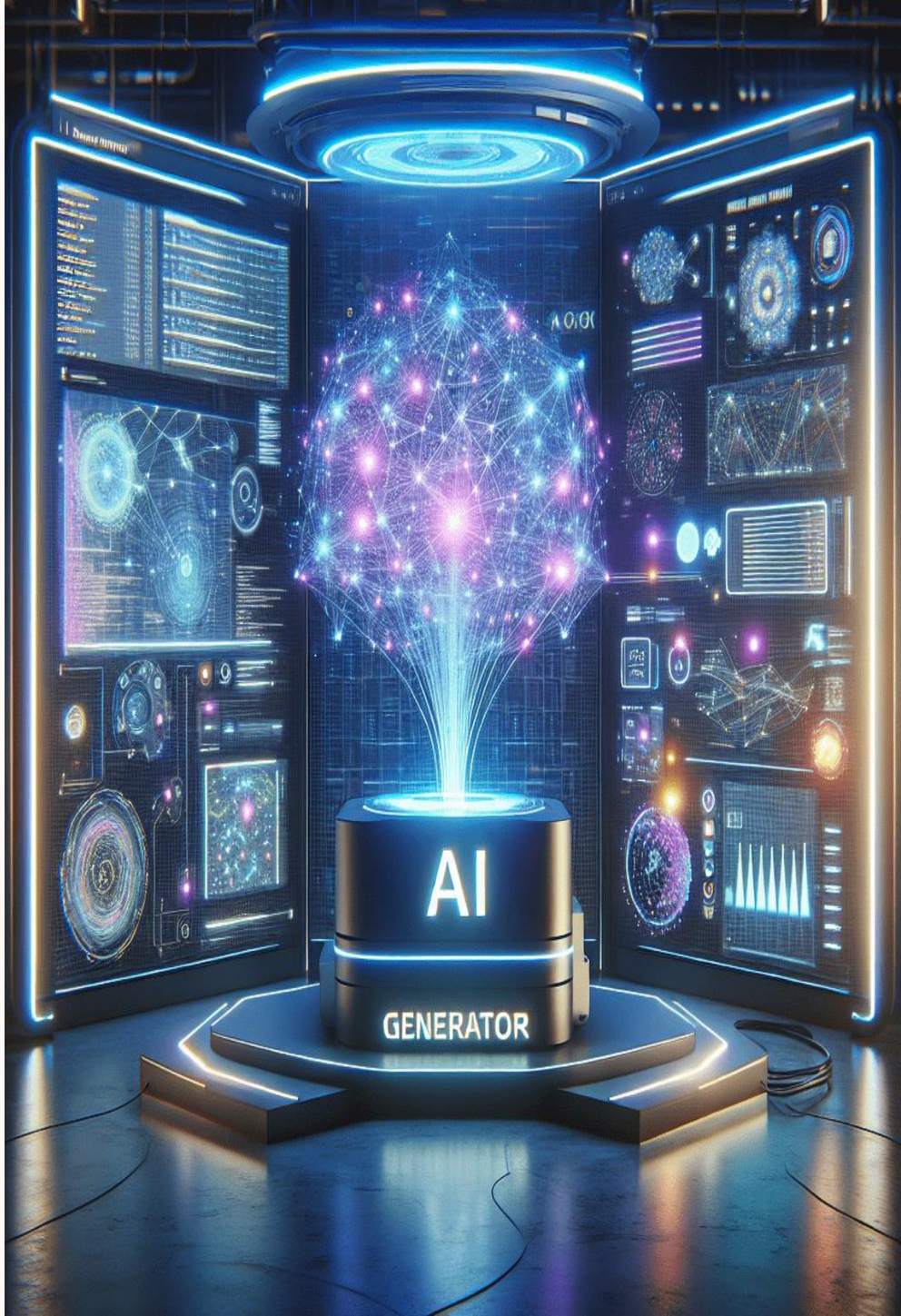
Generative AI system to automate structured entertainment content: concepts, loglines, pitches, outlines, character sketches, and scenes.

Presented By:

Aman Sharma (EN22CS301109), Anusha Singh Panwar (EN22CS301179), Ananya Bhatia (EN22CS301117), Ankit Nagar (EN22CS301134), Arpit Patidar (EN22CS301199)

Project Purpose & Scope

Convert a seed idea into professionally formatted entertainment material via a guided multi-stage pipeline. This HLD covers architecture, application design, process flow, APIs, data handling, and non-functional requirements.



Core Functionality — Multi-stage Pipeline

Concept Development

Transform seed idea into a concept.

Logline Creation

Craft concise story hooks.

Elevator Pitch

Short, persuasive summaries.

Story Outline

Structured act and beat breakdowns.

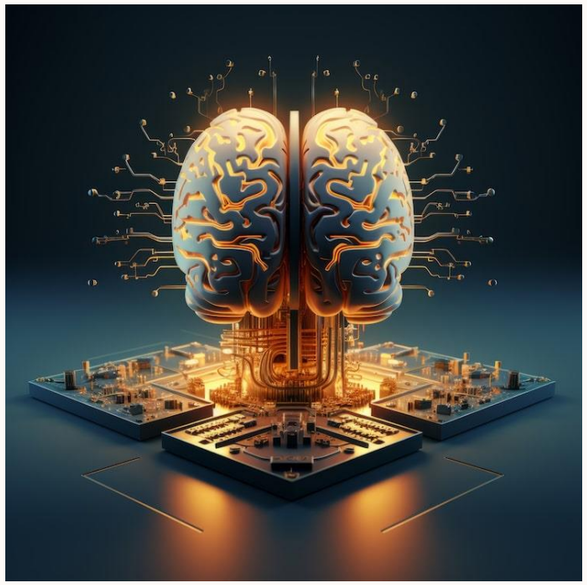
Character Profiles

Detailed character sketches.

Scene Writing

Formatted screenplay scenes.

Key Features



- Context-aware generation using semantic vector memory
- Retrieval-based continuity across stages
- Regeneration for iterative refinement
- Structured, industry-aligned output formatting
- Support for full or partial pipeline runs

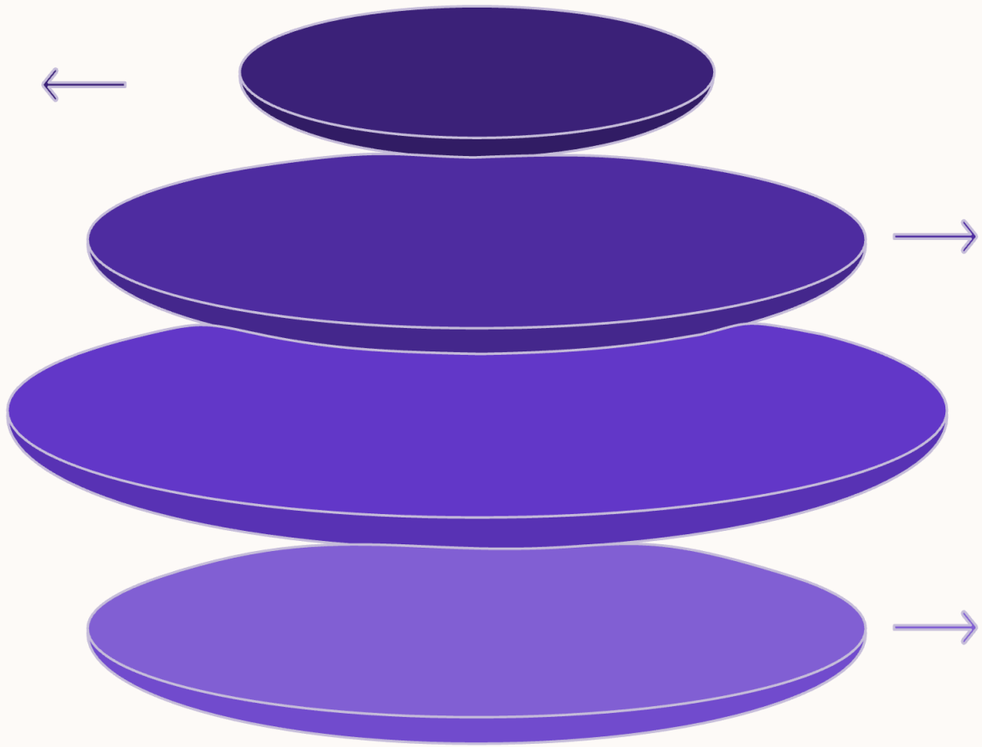
High-Level Architecture

Presentation

Streamlit UI for user interaction

AI Layer

Google Gemini LLM for reasoning



Application

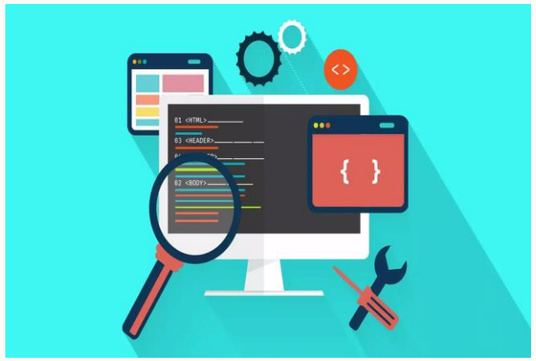
Orchestrator managing pipelines

Memory

FAISS index + vector embeddings

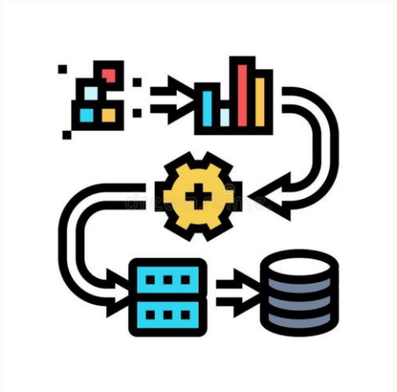
This layered design ensures modularity, scalability, and clear separation of concerns.

Application Components



Frontend (Streamlit)

Accepts user input, displays results, manages session state, triggers regeneration, shows stored memory.



Content Pipeline

Controls stage execution, builds prompts, calls LLM, stores outputs in vector DB. Exposes run_stage() and run_full_pipeline().

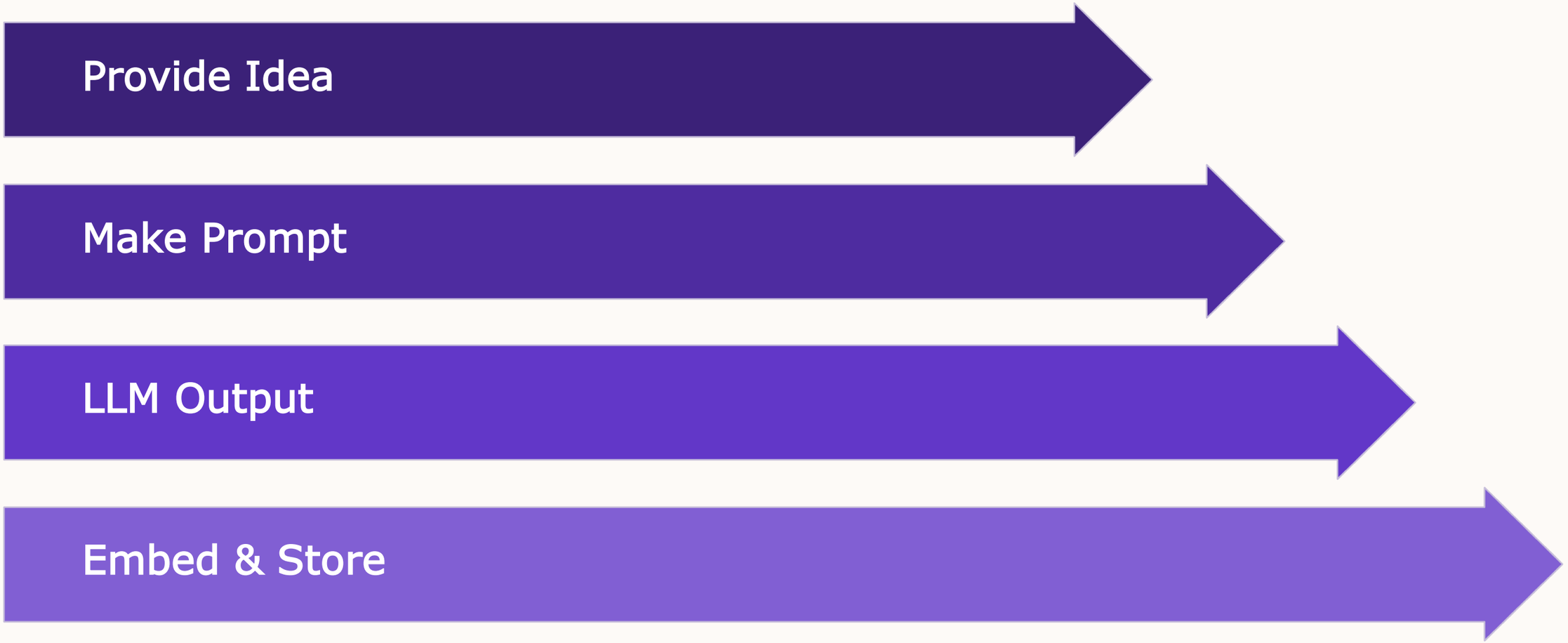
LLM Client

Connects to Google Gemini API, applies system prompts, handles retries, rate limits, and temperature settings.

Vector Store (Memory)

FAISS + SentenceTransformer embeddings; stores outputs, retrieves top-k context with metadata.

Process & Information Flow



Each stage embeds outputs and retrieves context to maintain narrative consistency across generations.

Data, Security & Non-Functional Considerations

Data Design

Embeddings, metadata (stage, content), retention policies, migration strategy, and access controls.

Performance

IndexFlatL2 for FAISS; caching and session management to reduce latency.

Security

API authentication, prompt constraints, rate-limit handling, and secure storage for sensitive data.

Reliability

Retries, graceful degradation, and monitoring for the LLM client and pipeline.

Next Steps & Audience

Intended Audience

University mentors, industry mentors, developers, architects, QA — for evaluation and development guidance.

Immediate Next Steps

Implement pipeline connectors, integrate Google Gemini, build FAISS memory, and test end-to-end scenarios.

Deliverables

Working prototype (Streamlit UI), API catalogue, data model, and performance/security validation.