Task Closure: Karan – Gurukul Seed Tier: Vedic Knowledge Lessons Script

Status: Ready for Closure

Summary:

Dev 4 successfully created the first 5 foundational lessons for the Gurukul “Seed Tier” under the subject Vedic Wisdom & Life Foundations. Each lesson included:

- Title and Theme
- Core Message
- Opening Shloka or Verse
- Activities/Story Element
- Explanation rooted in both ancient and modern wisdom

Next Step: This static syllabus now transitions into a dynamic RAG-based Knowledge Engine that scales and adapts based on the core Vedic corpus + current knowledge.

New Task: Dev 4 – Gurukul AI Knowledge Engine – RAG Syllabus Integration

Role: Sutradhar of Knowledge Flow

You now become the builder of the Gurukul’s living brain — connecting Vedic wisdom texts and our dynamic syllabus to an AI engine that answers questions, adapts lessons, and supports the Gurukul’s evolving curriculum.

Objective:

Build the Knowledge Engine Pipeline for dynamic Q&A and lesson generation using RAG (Retrieval Augmented Generation). Integrate it with the Gurukul lesson format you’ve already developed.

Tools & Tech:

- LangChain for orchestration
- FAISS / Chroma / Weaviate for vector search
- Unstructured or PyMuPDF / pdfminer for ingesting ancient texts
- MongoDB for indexing metadata
- OpenAI / Ollama for generation

- Supabase Storage (optional) for media file mappings

Phase 1 Tasks (Week 1)

Task 1: Ingest and Index Seed Knowledge Base

- Collect a small set of 5–10 key Vedic texts or verses (PDF/DOCX/TXT)
- Chunk them using LangChain text splitter (with shloka context preserved)
- Store chunks in vector DB (start with FAISS for local dev)
- Tag each chunk with metadata (scripture, chapter, topic, etc.)

Deliverables:

- Working script to ingest and index at least 3 core sources
- Vector store saved and callable via LangChain retriever

Task 2: Build a Basic RAG Pipeline

- Query input → Retrieve relevant chunks → Answer in lesson format
- Format answer to return:
  - Theme
  - Reference verse(s)
  - Explanation
  - Activity/story-style prompt (from source or generated)
- Enable debug mode to return source chunk IDs + similarity score

Deliverables:

- rag_lesson_engine.py with basic RAG QA + lesson format output
- Sample responses for 3 user questions (e.g. “What is the role of discipline?”)

Task 3: API Layer for RAG Engine

- Expose a POST endpoint /ask-vedas via FastAPI
- Accept query + optional metadata (user level, subject tag)
- Return lesson-format JSON
- Add logging to MongoDB (user ID, query, timestamp, retrieved docs)

Deliverables:

- /ask-vedas FastAPI route
- Example query and JSON output
- Mongo logging in user_lessons or interaction_logs

To be integrated

– Seed Tier Curriculum Auto-Extension

- Add a /generate-lesson endpoint to auto-suggest next lesson based on theme
- Chain from existing user history + Vedic corpus
- Add fallback defaults if RAG fails (e.g. lesson templates from Task 1)

Spiritual Reminder

You are now the bridge between Shruti (heard knowledge) and Yukti (reasoned understanding). This is not just a Q&A engine — it's a living Saraswati that responds with compassion, depth, and clarity.

When your code breathes, wisdom flows.