**Project Summary: Luxury Decor Assistant (RAG)**

**Goal:**  
To build an intelligent, conversational decor assistant that suggests creative interior design ideas based on real product descriptions, using **RAG (Retrieval-Augmented Generation)** with advanced LLMs.

**✅ Phases of the Project**

**🔹 Phase 1: Data Preparation & Embedding**

* **Objective:** Embed product descriptions and store in a FAISS index for retrieval.
* **Tech stack:**
  + **Model**: microsoft/deberta-v3-base (for semantic embeddings)
  + **Tool**: FAISS (for similarity search)
  + **Output**: deberta\_faiss.index and deberta\_text\_data.csv

**🔹 Phase 2: Basic RAG Pipeline**

* **Query embedding:** Convert user queries into vectors using DeBERTa.
* **Retrieval:** Use FAISS to get top-k relevant product texts.
* **Prompt creation:** Construct a contextual prompt with retrieved texts.
* **Response generation:** Initially used **Falcon-1B** and **Flan-T5**, but faced performance issues.

**🔹 Phase 3: UI & Deployment on Hugging Face Spaces**

* **Frontend tool:** Gradio
* **Features:**
  + Custom logo + styled UI
  + Input/output boxes with titles
  + Logging of user queries
* **Deployment:** Public Hugging Face Space:  
  👉 <https://dine24-luxury-decor-rag.hf.space>

**🔹 Phase 4: Enhancements**

* Switched from single-turn input to **multi-turn conversation** using chatbot interface.
* Added **follow-up question generation**.
* Implemented a **chat log memory**.
* Finally upgraded to **OpenAI GPT-3.5 Turbo** for:
  + Better multi-turn handling
  + Richer responses
  + Reliable generation from complex prompts

**⚠️ Key Challenges Faced**

**1. Model Size and Memory Limits**

* ❌ Falcon-7B/Mistral were large and often exceeded Hugging Face’s 16GB memory cap.
* ✅ Switched to Flan-T5 then GPT-3.5 for smaller memory footprints + better results.

**2. OpenAI Migration Errors**

* ChatCompletion interface deprecated in openai >= 1.0
* ✅ Solution: Used new openai.OpenAI().chat.completions.create() style.
* 🛠️ Environment variables (OPENAI\_API\_KEY) had to be defined as **secrets** in Hugging Face.

**3. Broken UI with Logo + Image Rendering**

* ❌ Initial logo display was oversized and clunky.
* ✅ CSS adjusted for polished, responsive branding.

**4. Chatbot History Errors**

* ❌ Initially tried modifying tuple-style chat history (TypeError).
* ✅ Fixed by maintaining a list of (user, assistant) message pairs immutably.

**✨ Unique Outcomes**

| **Feature** | **Description** |
| --- | --- |
| ✅ Retrieval + Generation | Combines factual grounding + creativity |
| ✅ Multi-Turn Chat | Context-aware responses across follow-ups |
| ✅ Follow-Up Suggestion | Engaging and interactive decor journey |
| ✅ Hugging Face Integration | Entirely hosted with secure API token handling |
| ✅ Branding & Logging | User query logging and professional UI (logo, theme, button stylings) |

**🌱 Future Enhancements**

1. **🎯 Personalization Engine**
   * Detect user’s room type, style preference, and mood
   * Tag preferences like "Scandinavian", "minimalist", etc.
2. **📦 Product Linking**
   * Add hyperlinks to actual product listings from retrieved content
3. **📁 Export / Save Sessions**
   * Option to download decor suggestions + chat logs as PDF
4. **🖼️ Visual Suggestions**
   * Integrate DALL·E or Midjourney API to suggest mood boards
5. **📈 Analytics Dashboard**
   * View user trends: most searched rooms, styles, etc.
6. **🗣️ Voice Integration**
   * Use Speech-to-Text to enable verbal inputs and audio responses

**💡 Final Thoughts**

You built an **end-to-end intelligent RAG assistant**, complete with:

* Fine-grained semantic retrieval
* Generative AI with chat memory
* Professional UI
* Deployed securely on Hugging Face

This project blends **Data Science, NLP, Prompt Engineering, Full Stack UI, and Deployment** — a solid portfolio piece with real-world utility!