Generative AI Interview Questions and Answers

## 1. What is Generative AI, and how is it different from traditional AI?

Generative AI focuses on creating new content (text, image, code, etc.) based on patterns learned from training data, using models like GPT or diffusion models. In contrast, traditional AI often focuses on prediction or classification (e.g., detecting spam or recognizing faces). Generative AI learns distributions and can synthesize human-like output, whereas traditional AI maps input to predefined output.

## 2. What is a RAG pipeline, and why is it used?

A Retrieval-Augmented Generation (RAG) pipeline combines retrieval (fetching relevant documents from a knowledge base like Qdrant or FAISS) with generation (using LLMs to synthesize answers). It enhances LLMs by grounding responses in real, up-to-date information, reducing hallucination and improving domain-specific accuracy.

## 3. How do you prevent hallucinations in LLM-based applications?

- Use a RAG pipeline to ground outputs in external factual data  
- Use prompt engineering with clear instructions and context  
- Apply LLM post-verification using rule-based filters or secondary models  
- Fine-tune models on domain-specific QA pairs  
- Evaluate with tools like RAGAs, DeepEval, or ground-truth checks

## 4. What are embeddings, and how are they used in vector databases?

Embeddings are dense vector representations of content (text, image, etc.) that capture semantic meaning. In GenAI, they're used to index and retrieve semantically similar items using cosine similarity or Euclidean distance in vector databases like Qdrant, Pinecone, or FAISS.

## 5. How do you deploy a GenAI solution with Qdrant and an LLM?

1. Extract embeddings using a model like all-MiniLM  
2. Store vectors in Qdrant with metadata  
3. Accept a user query → convert to embedding  
4. Search Qdrant for top-K similar docs  
5. Use LLM (e.g., GPT-4) to generate response using retrieved context  
6. Deploy backend with Flask/FastAPI and optionally use Streamlit for UI