MIT | 6.S978 AI Engineering

Quiz 3

Wednesday, October 21, 2026 – 20 minutes

Name: _____

Kerberos: _____

Student ID: _____

Question	Score	Out of
1		3
2		3
3		3
Available		**9**

- The quiz is closed book and closed notes.
- Write your answers in the spaces provided. Short answers are fine.
- The quiz has 9 points. 8 points count as 100%, so the maximum score is 112.5%.

Please do not turn this page until you are told to begin.

1. Answering from a forum (3 points)

TripTalk is a travel forum. Travelers post questions, and other users reply with advice. Anyone can post, so some advice is mistaken or outdated. Each post has an ID, content, timestamp, and the number of upvotes it got from other users. Posts are always below 200 tokens in length.

TripTalk recently added an AI assistant that answers traveler questions from the forum. In their current implementation, the system starts by retrieving the top-5 posts most similar to the question and then asks an LLM to generate a reply from them.

Travelers have reported problems. One missed a flight after the assistant repeated popular airport advice that was out of date. Another got a confident answer about needing a visa due to a few incorrect posts, even though some other posts were correct.

Design a better system for TripTalk, with the signature:

```
answer(question: str, posts: list[Post]) -> str
```

Your system's inputs are the question and the **250 posts** most similar to it. Besides any AI instructions you specify, each LLM call can read up to **2,000 tokens** of content. No other sources of information are available.

In 3–4 sentences, describe your system design, including what each LLM call reads and returns, and how your design produces a reply that gives useful advice while conveying uncertainty from the forum back to the user.

2. Fixing the right problem (3 points)

Below are three cases. For each, write the letter (A–F) of the change that best addresses it. Letters may repeat or go unused.

(A) Fine-tune the encoder on the ten reviewed useful documents.

(B) Give each manual a single vector, instead of one vector per token.

(C) Score smaller, coherent passages, each kept with its document's title.

(D) Probe more ANN clusters, keeping the encoder fixed.

(E) Hide any note flagged as irrelevant by either model call.

(F) Hide only notes flagged as irrelevant by both model calls.

(a) _____ We use an approximate nearest neighbor index by searching one cluster. Our search finds two of ten reviewed useful documents. However, we learn that scoring every document with the same encoder finds nine of the ten!

(b) _____ A user asks how to reset a printer called the X200. A late-interaction retriever ranks a 300-page manual first by matching "X200" on its cover page and reset instructions for a different printer on page 212. However, we realize that the manual never explains how to reset the X200 anywhere.

(c) _____ A tool retrieves 30 incident notes for an engineer investigating an outage. It calls an LLM twice to flag irrelevant notes. The engineer can quickly skip an irrelevant note, but a hidden useful note may contain the only known fix. Which rule better fits this use?

3. A shared drive (3 points)

Your company's shared drive has **1 million documents**. Your teammate used the very best embedding model out there, and even fine-tuned it for your application, but learned that it does not work well here due to the complexity of the questions. Your new engineering manager hears this and decides to ban embedding models entirely.

You decide to rely on generative LLMs instead. Through a small evaluation, you find convincingly that an LLM reading every document finds all relevant information. Unfortunately, the team tells you immediately that this system costs too much to put in production.

Given this, you decide to use LLM calls in a program to prepare some kind of stored "index" before questions arrive. For each new question, your system must return ten potentially useful document IDs while reading only about as much text as **a few hundred documents**.

Describe your design in the two stages below. Short prose is fine.

Preparation: How is the index built, and what does it store?

Search: What does the LLM read, in what order, and how many items?