Practice Problem Workshop 11:
Part I - Individual Practice Questions

INFO 2310, Spring 2026

Student ID: Solution **NetID:**

Overview

Retrieval practice is the act of actively recalling information from memory, like quizzing oneself.

This workshop is designed to help you prepare and study for upcoming exams by engaging in retrieval practice.

Part I: Individual Practice Questions

Credit: 10 points. (completion)

1. Open this booklet and try your best to answer the practice problems **individually**.

   If you need to refer to your notes, you can. But you should try to answer the questions on your own first, without looking at your notes. (This will help you prepare for the upcoming exam.)

2. Once a majority of the class has answered the practice questions on their own, **form groups of 3-4 people and go over the questions together**.

   Discuss the questions and share your answers with the group.

   You should update your answers in this booklet if necessary.

   Discussing the answers will help you practice retrieving the information from memory and will also help you learn from your peers.

Submit this booklet for completion credit.

Practice Problem 1

Design a REST API for prompting an LLM, like ChatGPT. The REST API should support the following operations:

- Retrieve all **models** that are available to prompt with **text search**.
- Send a prompt to a specific model to **create** a **response**.
- Retrieve all **responses** that have been generated for a specific model with **text search**.
- Retrieve a specific **response** by its unique ID.
- Delete a specific **response** by its unique ID.

Example **model**:

```
{
  "id": "gpt-3.5-turbo",
  "name": "GPT-3.5 Turbo",
}
```

Example **response**:

```
{
  "_id": "64a2186e40e1f7d9a3f9d875",
  "model": "gpt-3.5-turbo",
  "prompt": "Create a REST API for an events collection.",
  "response": "Here is an example of a REST API for an events collection: ...",
  "created_at": "2026-04-01T12:00:00Z"
}
```

CRUD Operation	HTTP Method	URI	Query Parameters	Request Body	Response Status	Response Body
Read	GET	/api/models	q		200	JSON
Create	POST	/api/models		JSON	201 404	JSON
Read	GET	/api/models/:modelId/responses	q		200 404	JSON
Read	GET	/api/responses/:id			200 404	JSON
Delete	DELETE	/api/responses/:id			204 404	

Practice Problem 2

a. The LLM REST API from the previous problem, should support text search for both models and responses.

Create the MongoDB index(es) needed to support text search by **model name** and by **prompt**.

`init.mongodb.js`:

```
db.models.createIndex({
  name: "text"
})

db.responses.createIndex({
  prompt: "text"
})
```

b. Write the Express.js endpoint to retrieve all models unless a search parameter is provided. You may assume the database connection is available as `app.locals.db`.

`server.mjs`:

```
app.get("/api/models", async (req, res) => {
  const search = req.query.q

  const query = {}
  if (search) {
    query.$text = { $search: search }
  }
  const models = await app.locals.db.collection("models")
      .find(query)
      .toArray()
  res.status(200).json(models)
})
```