

HOMWORK 3^{*}

AGENTIC FRAMEWORKS[†]

24-880 AI AGENTS FOR ENGINEERS
<https://cmuagents.github.io>

OUT: Sunday, February 8th, 2026
DUE: Sunday, February 15th, 2026
TAs: Peter Pak

Instructions

- **Collaboration Policy:** Please read the collaboration policy in the syllabus.
- **Late Submission Policy:** See the late submission policy in the syllabus.
- **Submitting your work:** You will use Gradescope to submit answers to all questions and code.
 - **Written:** You will submit your completed homework as a PDF to Gradescope. Please use the provided template for the submissions which can be written in $\text{\LaTeX}$ (via [Overleaf](#)) or directly on the PDF document. Each answer should be within the box provided (answer does not need to encompass entirety of provided space). If you do not follow the template, your assignment may not be graded correctly and there will be a **2% penalty** (e.g., if the homework is out of 100 points, 2 points will be deducted from your final score).
 - **Programming:** Gradescope's Autograder will be used to evaluate the coding portions of this assignment. In order to best obtain the points for this assignment, please follow the code upload instructions. If you do not follow the instructions correctly the autograder may not work properly.

Question	Points
LangChain (Written)	35
LangChain (Programming)	15
Model Context Protocol (Written)	30
Model Context Protocol (Programming)	15
Code Upload	0
Collaboration and Generative AI Questions	5
Total:	100

^{*}Adapted from CMU 10-623 Generative AI Homework Template

[†]Compiled on Sunday 8th February, 2026 at 18:33

Introduction

In this homework assignment we'll briefly explore some existing agentic frameworks.

At a high-level, this homework will cover the following aspects.

1. Learn about the basic modules included in LangChain.
2. Create a simple LangChain implementation with LangChain modules.
3. Learn about the features within the Model Context Protocol.
4. Create a simple example that implements some of these features.

Local Environment

For this homework we'll be using `uv` as the project and package manager. If you haven't install `uv`, you can follow the installation instructions here <https://docs.astral.sh/uv/>. You can explore all of the available commands using `uv --help` but in short, here are a couple of useful commands that you'll often use:

<code>source .venv/bin/activate</code>	(Linux and Mac) Activates python environment.
<code>source .venv/Scripts/Activate</code>	(Windows) Activates python environment.
<code>uv sync</code>	Installs / synchronizes projects dependencies.
<code>uv add <package></code>	Installs <code>pip</code> package and adds it to project dependencies.
<code>uv remove <package></code>	Removes <code>pip</code> package dependency from project.
<code>uv run <script.py></code>	Runs python script inside of <code>uv</code> environment.

1. Navigate to folder project directory and synchronize dependencies with:

```
uv sync
```

2. Activate the environment

Mac and Linux:

```
source .venv/bin/activate
```

Windows:

```
source .venv/Scripts/activate
```

3. Test `uv` works and run `main.py` file.

```
uv run main.py
```

1 LangChain (Written) (35 points)

Let's explore some of the background and different aspects of LangChain. Here are a couple of links to documentation and history that will be helpful for the following questions.

1. <https://en.wikipedia.org/wiki/LangChain>
 2. <https://reference.langchain.com/python/langchain/langchain/>
- 1.1. Within LangChain there are several modules (i.e. [Agents](#), [Middleware](#), [Models](#), [Messages](#), [Tools](#), and [Embeddings](#)) let's look into each of these and determine how they work and relate to one another.
- 1.1.a. (5 points) What is the [Agents](#) Module? What does it do and how does it relate to some of the other modules?

- 1.1.b. (5 points) What role does [Middleware](#) play in the request/response lifecycle? Give an example of when you might use middleware.

- 1.1.c. (5 points) What is the [Models](#) module? What are some of the key interfaces it exposes?

- 1.1.d. (5 points) What types of messages are supported in the [Messages](#) module? How do these map to the typical roles in a chat conversation?

- 1.1.e. (5 points) How do you define a custom tool using the [Tools](#) module? What information must a tool provide to the agent?

- 1.1.f. (5 points) What is the purpose of the [Embeddings](#) module? Describe a use case where embeddings would be essential.

- 1.2. (5 points) What's the difference between a [Model](#) and an [Agent](#)?

2 LangChain (Programming) (15 points)

In this section, you will create a LangChain application that demonstrates your understanding of the core modules discussed in the written section. Your task is to build a working LangChain example that shows your understanding of the previously discussed LangChain modules:

Note: The starter code was developed using Python 3.14 and LangChain 1.2.9. Depending on your Python version, you may need to adjust imports or install different packages. Refer to the LangChain documentation for your specific version.

- **Agents** - Use an agent or agent executor
- **Middleware** - Implement callbacks or other middleware functionality
- **Models** - Initialize and use an LLM or chat model
- **Messages** - Create and use message objects (HumanMessage, AIMessage, etc.)
- **Tools** - Define at least one custom tool
- **Embeddings** - Initialize and demonstrate an embeddings model

You have creative freedom in what your application does. Some ideas include:

- A conversational agent with custom tools
- A simple RAG (Retrieval Augmented Generation) system
- A math assistant that uses tools for calculations
- A Q&A bot with semantic search capabilities

2.1. (15 points) Complete the `langchain_example.py` file in the `programming/` directory. Your implementation should:

1. Include a brief comment section (towards the top of the file) of what your langchain example does.
2. Utilize at least 3 of the 6 modules
3. Include comments explaining how each module is being used
4. Demonstrate meaningful functionality (not just imports)

5 points are autograded based on file submission/completion, and **10 points** are manually graded based on your usage of each module.

3 Model Context Protocol (Written) (30 points)

Now let's explore some of features from the Model Context Protocol (MCP). Here are a couple of links to documentation that will be helpful for the following questions.

1. <https://modelcontextprotocol.io/docs/concepts/architecture>
2. <https://modelcontextprotocol.io/specification/2025-11-25>

In MCP, **servers** offer features to clients, and **clients** may offer features back to servers. Let's explore each of these capabilities.

3.1. MCP Servers can offer the following features to clients: **Resources**, **Prompts**, and **Tools**. Let's examine each of these.

3.1.a. (5 points) What are **Resources** in MCP? How do they provide context and data to clients?

- 3.1.b. (5 points) What are **Prompts** in MCP? How do they differ from simple text strings, and what makes them useful for creating templated workflows?

- 3.1.c. (5 points) What are **Tools** in MCP? How does the server expose tools for the AI model to execute?

- 3.2. MCP Clients may offer the following features to servers: [Sampling](#), [Roots](#), and [Elicitation](#). Let's examine each of these.
- 3.2.a. (5 points) What is [Sampling](#) in MCP? How does it enable server-initiated agentic behaviors and recursive LLM interactions?

- 3.2.b. (5 points) What are [Roots](#) in MCP? How do they help servers understand the boundaries (URI or filesystem) they should operate within?

- 3.2.c. (5 points) What is [Elicitation](#) in MCP? How does it allow servers to request additional information from users during execution?

4 Model Context Protocol (Programming) (15 points)

In this section, you will create an MCP server that demonstrates your understanding of the core concepts discussed in the written section. Your task is to build a working MCP server that incorporates at least the following features:

- **Tools** - Define at least one tool that the AI model can execute
- **Resources** - Expose at least one resource that provides context or data

You may optionally also include:

- **Prompts** - Templated messages and workflows

Note: You can use the [MCP Inspector](#) to test and debug your server during development. The Inspector provides a visual interface to interact with your server's tools, resources, and prompts.

You have creative freedom in what your MCP server does. Some ideas include:

- A file system server that exposes files as resources and file operations as tools
- A calculator server with math tools and configuration resources
- A weather server that provides location data as resources and forecast tools
- A notes/todo server with CRUD operations as tools and note contents as resources

4.1. (15 points) Complete the `mcp_example.py` file in the `programming/` directory. Your implementation should:

1. Include a brief comment section (towards the top of the file) describing what your MCP server does.
2. Implement at least one **Tool** that performs a meaningful action
3. Implement at least one **Resource** that provides useful context or data
4. Include comments explaining how each feature is implemented

5 points are autograded based on file submission/completion, and **10 points** are manually graded based on your implementation of tools and resources.

5 Code Upload (0 points)

To upload your code to Gradescope, run the following command to generate a zip file:

```
uv run code/zip_programming.py
```

Upload the generated zip file to the appropriate programming slot on Gradescope.

5.1. Make sure the following files are included when uploading to Gradescope

☐ langchain_example.py

☐ mcp_example.py

6 Collaboration and Generative AI Questions (5 points)

After you have completed all other components of this assignment, report your answers to these questions regarding the collaboration policy. Details of the policy can be found in the syllabus.

- 6.1. (2 points) Did you collaborate with anyone on this assignment? If so, list their name or Andrew ID and which problems you worked together on.

- 6.2. (3 points) Did you use generative AI for parts of this assignment? If so, include the input prompts along with links to the conversation.