

magi

A hack combining LLMs with Microsoft Graph

Demo



magi-demo



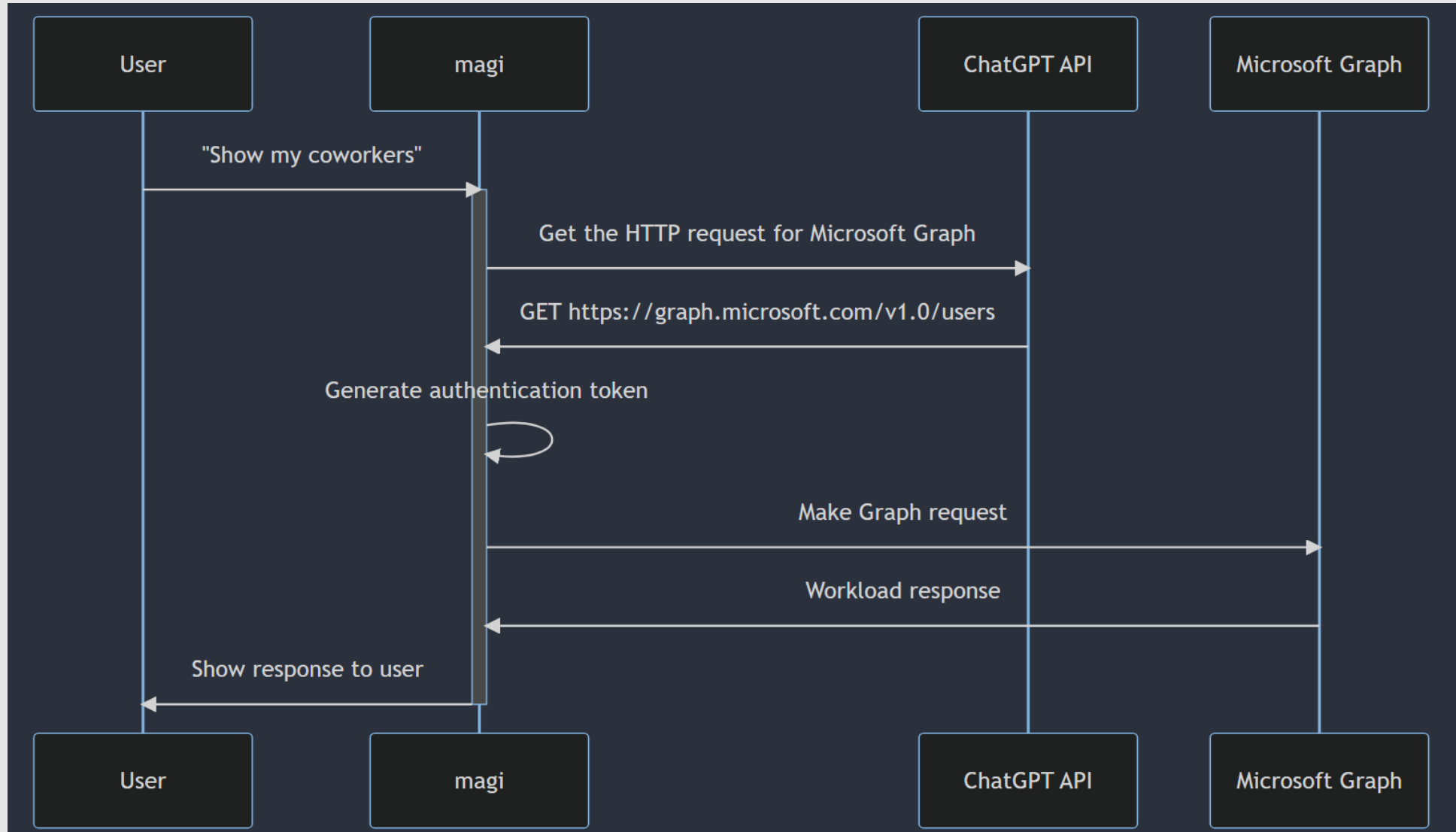
magi-demo~~~|



Introduction

- magi is the answer to "*What if you can just talk with Microsoft Graph?*"
- It has three building blocks:
 1. Command Line (CLI)
 2. ChatGPT API
 3. Microsoft Graph

Introduction - Flow



Details

- The .NET ecosystem is vibrant and has many useful packages.


Functionality	Package
Command line	System.CommandLine
ChatGPT API	OpenAI
Microsoft Graph	Graph SDK

Details - CLI

- [System.CommandLine](#) makes everything simple!
- Define a Command, specify its arguments and options, and watch your CLI app run.
- [The CLI skeleton](#) of magi is just 16 lines.

Details - CLI

System.CommandLine Example

 cli.cs

Raw

```
1 using System.CommandLine;
2
3 var queryArgument = new Argument<string?>(
4     name: "query",
5     description: "Ask magi your query!",
6     getDefaultValue: () => null);
7
8 var rootCommand = new RootCommand("magi - Microsoft Graph API's AI");
9 rootCommand.AddArgument(queryArgument);
10
11 var configFileArgument = new Option<string?>(
12     name: "--config",
13     description: "magi uses this file as the config.",
14     getDefaultValue: () => null
15 );
16 rootCommand.AddGlobalOption(configFileArgument);
17
18 rootCommand.SetHandler(async (query, configFile) =>
19 {
20     // LLM + Microsoft Graph API here
21 }, queryArgument, configFileArgument);
22
23 return await rootCommand.InvokeAsync(args);
```


Details – ChatGPT API

- [The community-made OpenAI library](#) provides a .NET focused experience.

chatgptapi.cs

Raw

```
1 var api = new OpenAI_API.OpenAIAPI(OPEN_AI_API_KEY);
2 var chat = api.Chat.CreateConversation();
3 chat.AppendSystemMessage(SYSTEM_MESSAGE);
4 chat.AppendUserInput(query);
5 var response = await chat.GetResponseFromChatbot();
```

Details – ChatGPT API

- **Prompt engineering** is the key to get most out of the ChatGPT API.

```
You are an AI assistant trained to help users of Microsoft Graph API. You provide the correct HTTP endpoints for Microsoft Graph based on the user's query. You only provide the HTTP endpoints and nothing more and this should never be violated. You are responsible for providing the Microsoft Graph HTTP requests for fulfilling a user query in this format and nothing else: [HTTP VERB] [ENDPOINT]
- Try to predict and correct a user's vague query.
- NEVER GIVE COMMENTARY
- Indicate request body as BODY [data]
- Give the full URL
- If you cannot fulfill the request, reply "magi has no spell for this!"
```

Details – Microsoft Graph SDK

- [The Microsoft Graph SDK](#) is a great piece of tooling and enables a lot of scenarios.
- Tip: Use [GraphClientFactory](#) to make arbitrary requests to the Graph API.

Questions?

Thank You!

- Slides available at <https://www.chotu.me/magi.pdf>
- Code: <https://github.com/Rahtoken/magi-msgraph-hackathon>
- Reach out to <https://www.linkedin.com/in/rankb/>