[PPT-06] APPLICATION PROGRAMMING INTERFACES

Application programming interface (APIs)

- An **application programming interface** (API) is a way for two or more computer programs or components to communicate with each other.
- The main players in the LLM arena (OpenAI, Anthropic, Meta, Cohere, etc) offer API platforms which allow us to use their models in a remote way.
- For closed models like those of the GPT, Gemini or Claude collections, this is the only way you have to include the tasks performed by the model in your programs.
- The remote connection with an LLM API is managed by means of an API key and a **software development kit** (SDK). The API key is given by the provider through its website. Before asking for the API key, you must register, providing a user name and a password. The SDK can be a Python package, but other languages like Java or node.js are available in most cases.

Signing up

- E-mail account.
- Password.
- API keys.
- Terms of use.

Technicalities

- Messages list.
- Roles: system, user, assistant and tools.
- Response structure.

The OpenAI API

The OpenAI API provides an interface to **OpenAI** models. The Python SDK is a package called `openai` that you can install, e.g. with `pip install openai`. After importing the package, you create a client using your API key:

```
! pip install openai
from openai import OpenAI
client = OpenAI(api_key=‘YOUR_API_KEY')
prompt = [{'role': 'system', 'content': instruction},
{'role': 'user', 'content': query}]
response =
client.chat.completions.create(messages=prompt,
model='gpt-4o-mini')
```

The Gemini API

The Gemini API provides an interface to **Google Gemini** models. The Python SDK is a package called `google.generativeai` that you can install, e.g. with `pip install google.generativeai`. After importing the package, you create a client using your API key:

```
import google.generativeai as genai
genai.configure(api_key='YOUR_API_KEY')
model = genai.GenerativeModel('gemini-1.5-flash')
chat = model.start_chat()
response = chat.send_message(prompt)
```

The Cohere API

The Cohere API provides an interface to **Cohere** models. The Python SDK is a package called `cohere` that you can install, e.g. with `pip install cohere`. After importing the package, you create a client using your API key:

```
! pip install cohere
import cohere
client = cohere.ClientV2(api_key='YOUR_API_KEY')
prompt = [{'role': 'system', 'content': instruction},
{'role': 'user', 'content': query}]
response = client.chat(messages=prompt, model='command-
a-03-2025')
```

The Groq API

- The Groq API provides an interface to open source models hosted by **Groq**. The Python SDK is a package called `groq` that you can install, e.g. with `pip install groq`. After importing the package, you create a client using your API key:

```
! pip install groq from groq import Groq client =
Groq(api_key='YOUR_API_KEY')
prompt = [{'role': 'system', 'content': instruction},
{'role': 'user', 'content': query}]
response = client.chat.completions.create(messages=prompt,
model='deepseek-r1-distill-llama-70b')
```