Prompt engineering for chatbot development

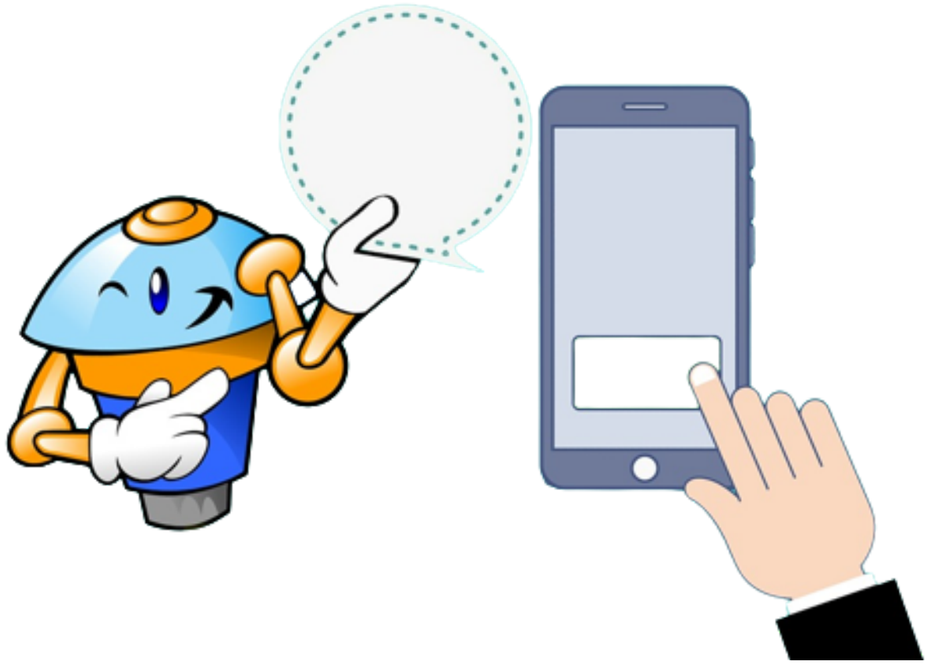
PROMPT ENGINEERING WITH THE OPENAI API

Fouad Trad
Machine Learning Engineer

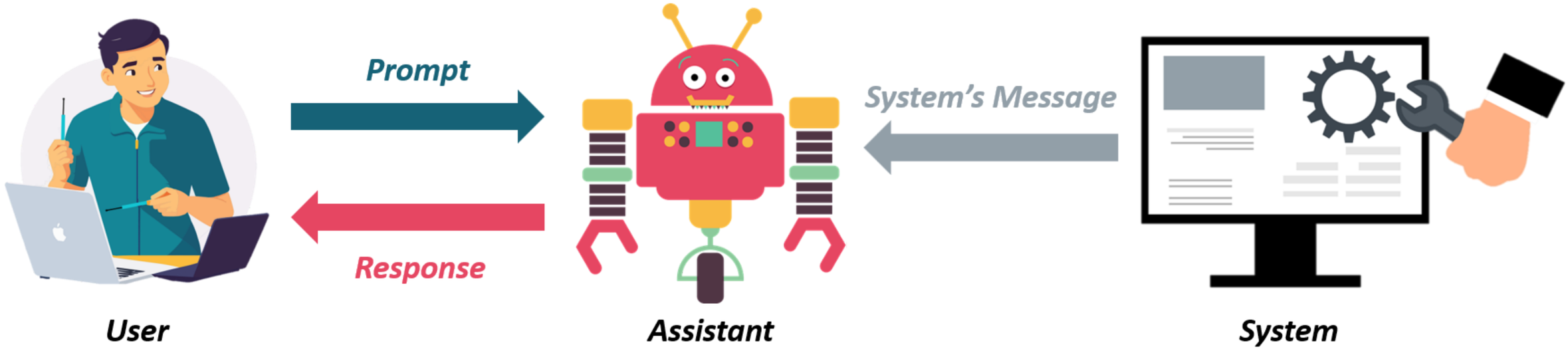
datacamp

The need for prompt engineering for chatbots

- Difficult to predict user questions
- Challenge to guarantee effective responses
- Prompt engineering guides chatbot behavior

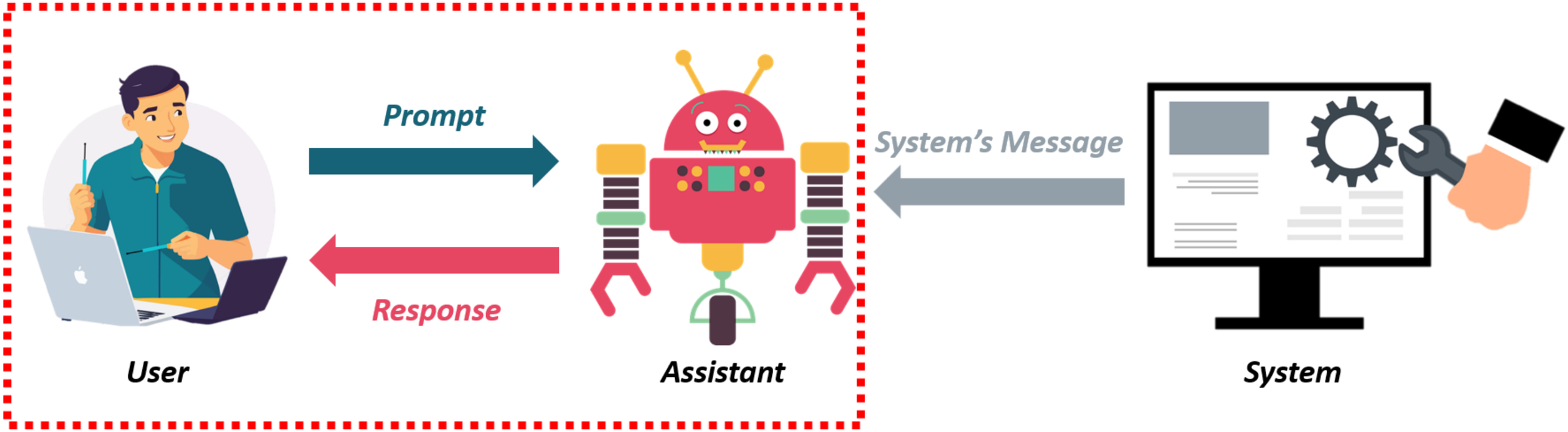


Chatbot prompt engineering with OpenAI API



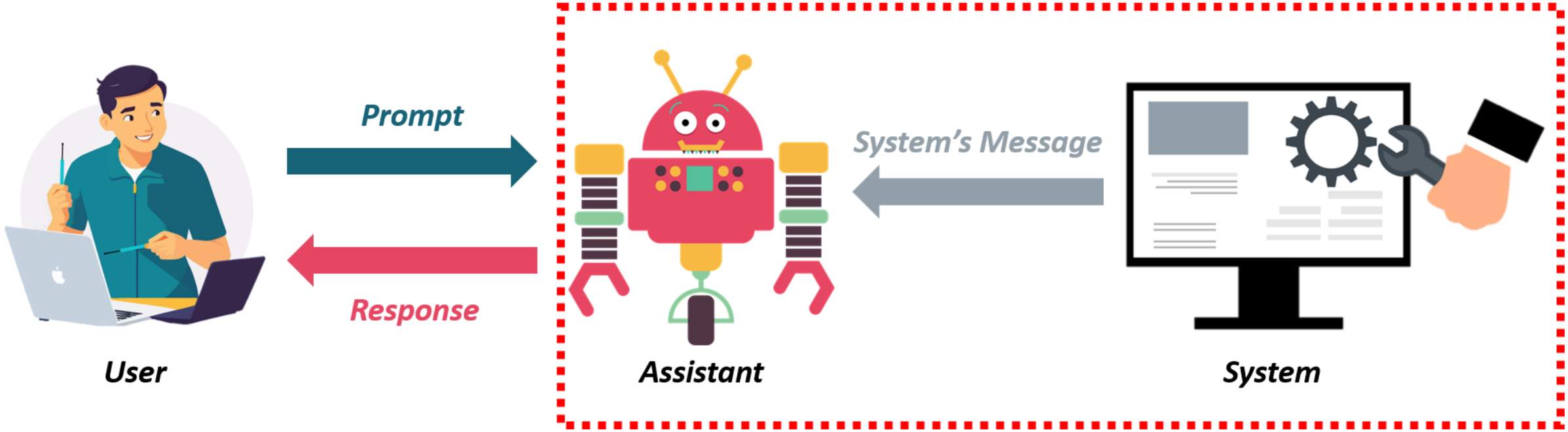
- Each message has a designated role

Chatbot prompt engineering with OpenAI API



- Each message has a designated role
- Focus has been on user prompts

Chatbot prompt engineering with OpenAI API



- Each message has a designated role
- Focus has been on user prompts
- System prompts guide chatbot's behavior

Chat completions endpoint for chatbot development

- Sends series of messages to model as a list

```
response = client.chat.completions.create(
  model="gpt-3.5-turbo",
  messages=[{"role": "system",
             "content": "You are an expert data scientist that explains complex concepts in simple terms"},
            {"role": "user",
             "content": "What is prompt engineering?"}]
)
print(response.choices[0].message.content)
```

```
Imagine you're giving instructions to a computer program, like teaching a robot to make a sandwich.
Prompt engineering is all about crafting those instructions, or "prompts," in a way that helps the
computer understand and perform the task better.
```

Changing get_response() for chatbot

- Sending one prompt

```
def get_response(prompt):
    messages = [
      {"role": "user", "content": prompt}
    ]
    response = client.chat.completions.create(
        model="gpt-3.5-turbo",
        messages=messages,
        temperature=0,
    )
    return response.choices[0].message.content

prompt = "<PROMPT>"
print(get_response(prompt))
```

- Sending two prompts

```
def get_response(system_prompt, user_prompt):
    messages = [
      {"role": "system", "content": system_prompt},
      {"role": "user", "content": user_prompt}]
    response = client.chat.completions.create(
        model="gpt-3.5-turbo",
        messages=messages,
        temperature=0,
    )
    return response.choices[0].message.content

system_prompt = "<SYSTEM_PROMPT>"
user_prompt = "<USER_PROMPT>"
print(get_response(system_prompt, user_prompt))
```

System message: define purpose

```
system_prompt = "You are a chatbot that answers financial questions."

user_prompt = "Who are you?"

print(get_response(system_prompt, user_prompt))
```

```
I'm a financial chatbot that answers financial questions. How can I help you?
```

- Allow chatbot to offer domain-accurate assistance
- Not defining purpose might lead to contextually irrelevant answers

System message: response guidelines

- Specify audience, tone, length, structure

```
system_prompt = """You are a chatbot that answers financial questions.
Your answers should be precise, formal and objective"""

user_prompt = "What do you think about cryptocurrencies?"

print(get_response(system_prompt, user_prompt))
```

System message: response guidelines

```
Cryptocurrencies are digital or virtual currencies that use cryptography for security and operate on
decentralized networks based on blockchain technology.
[...]

Advantages of cryptocurrencies include:
- Decentralization: [...]
- Security:[...]
- Global Accessibility: [...]

However, there are also notable concerns:
- Volatility: [...]
- Regulatory Uncertainty: [...]
- Lack of Consumer Protection: [...]

In summary, cryptocurrencies have the potential to offer various benefits, but their adoption and impact
on the financial landscape are still evolving [...]
```

System message: behavior guidance

- Conditional prompts to respond to questions

```
system_prompt = """You are a chatbot that answers financial questions.
Your answers should be precise, formal and objective.



"""
```

System message: behavior guidance

- Conditional prompts to respond to questions

```
system_prompt = """You are a chatbot that answers financial questions.
Your answers should be precise, formal and objective.
If the question you receive is within the financial field, answer it to the best of your knowledge.

"""
```

System message: behavior guidance

- Conditional prompts to respond to questions

```
system_prompt = """You are a chatbot that answers financial questions.
Your answers should be precise, formal and objective.
If the question you receive is within the financial field, answer it to the best of your knowledge.
Otherwise, answer with 'Sorry, I only know about finance.'
"""
```

```
user_prompt = "How's the weather today?"

print(get_response(system_prompt, user_prompt))
```

```
Sorry, I only know about finance.
```

Let's practice!

PROMPT ENGINEERING WITH THE OPENAI API

Role-playing prompts for chatbots

PROMPT ENGINEERING WITH THE OPENAI API

Fouad Trad
Machine Learning Engineer

datacamp

Role-playing prompts

- Tell chatbot to play a specific role
- Chatbot -> actor in a play
- Chatbot adjusts to match role
- Tailored language and content to fit the persona
- More effective interactions

Role-playing example

Question: Could you give me technical specifications of Product X that your company offers?

Customer Support

Role-playing example

Question: Could you give me technical specifications of Product X that your company offers?

Customer Support

Product Manager

Role-playing example

Question: Could you give me technical specifications of Product X that your company offers?

Customer Support

Product Manager

Sales Engineer

Customer support agent

- Provides guidance
- Directs customers to website
- Offers assistance

Product manager

- Highlights strategic benefits
- Focuses on product alignment

Sales engineer

- Focus on technical specifics
  - Processor
  - Features
  - Security

Role-playing prompts

- Tell model to **act as** as **specific role**

```
system_prompt = "Act as an expert financial analyst."
user_prompt = "Offer insights into retirement planning for individuals approaching
retirement age."

print(get_response(system_response, user_response))
```

Expert financial analyst

```
Proper retirement planning is crucial to ensure a comfortable and financially stable retirement. Here are some key
considerations:

- Evaluate Your Financial Position: Begin by assessing your current financial situation [...]
- Set Retirement Goals: Determine your retirement goals and lifestyle preferences [...]
- Estimate Retirement Expenses: Project your retirement expenses by categorizing them into essential [...]
[...]

Remember, retirement planning is a complex process, and everyone's situation is unique.
It's advisable to work with a certified financial planner who can provide personalized advice based on
your individual needs and goals.
```

More effective role-playing

- Specific requirements within the role
- Incorporate traits like personality and expertise

```
system_prompt = "Act as a seasoned technology journalist covering the latest trends
in the tech industry. You're known for your thorough research and insightful analysis."

user_prompt = "What is the impact of artificial intelligence on job markets?"

print(get_response(system_response, user_response))
```

Technology journalist

```
Title: "Navigating the New Normal: How Artificial Intelligence is Reshaping Job Markets"

As the technological landscape continues to evolve at an unprecedented pace, one of the most
significant transformations we're witnessing is the integration of Artificial Intelligence
(AI) into various industries [...].

While AI's potential to streamline processes [...]. Let's explore the multifaceted impact
of AI on employment...

1. Automation and Job Displacement: [...]
2. Augmentation and Enhanced Creativity: [...]
[...]

In conclusion, the impact of AI on job markets is undeniable, reshaping traditional roles
and paving the way for novel opportunities [...]
```

Role-playing with requirements

- We can specify response guidelines and behavior guidance in role-playing prompts

```
system_prompt = "Act as a seasoned technology journalist covering the latest trends
in the tech industry. You're known for your in-depth research and insightful analysis."
```

Role-playing with requirements

- We can specify response guidelines and behavior guidance in role-playing prompts.

```
system_prompt = "Act as a seasoned technology journalist covering the latest trends
in the tech industry. You're known for your in-depth research and insightful analysis.
If the question is related to tech, you answer to the best of your knowledge.
Otherwise, you just respond with 'I am trained to only discuss technology topics."

user_prompt = "Which American literature books do you recommend?"

print(get_response(system_prompt, user_prompt))
```

```
I am trained to only discuss technology topics.
```

Let's practice!

PROMPT ENGINEERING WITH THE OPENAI API

Incorporating external context

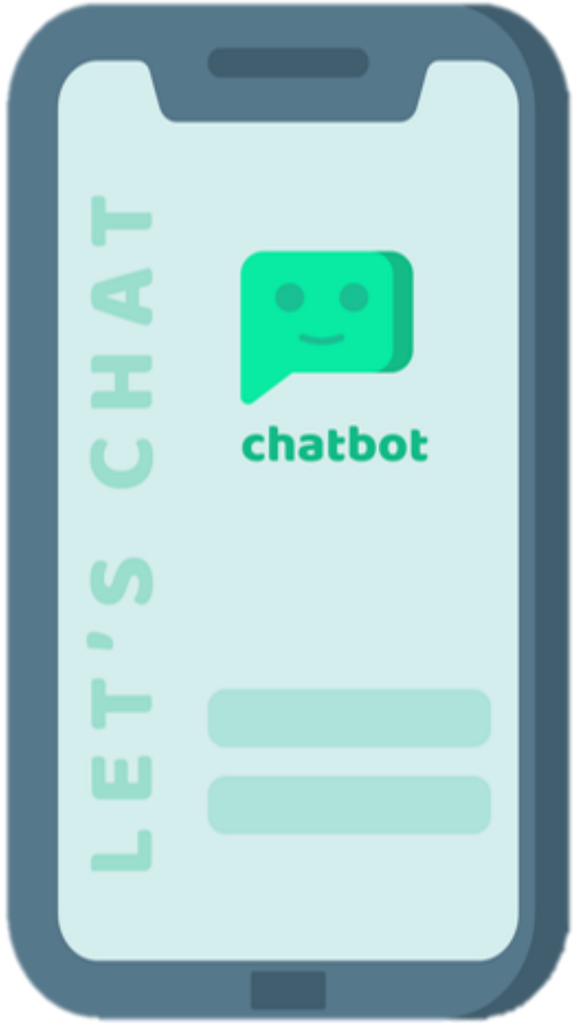
PROMPT ENGINEERING WITH THE OPENAI API

Fouad Trad
Machine Learning Engineer

datacamp

The need for external context

- Pre-trained language models recognize information they are trained on
- Need to provide more context
- More accuracy and effectiveness



Lack of information in LLMs

- Knowledge cut-off

```
system_prompt = "Act as a financial expert that knows about the latest trends."

user_prompt = "What are the top financial trends in 2023?"

print(get_response(system_prompt, user_prompt))
```

```
I apologize for any inconvenience, but as of my last knowledge update in
September 2021, I don't have information about financial trends in 2023.
```

Lack of information in LLMs

- Requested non-public information

```
system_prompt = "Act as a study buddy that helps me with my studies to succeed
in exams."

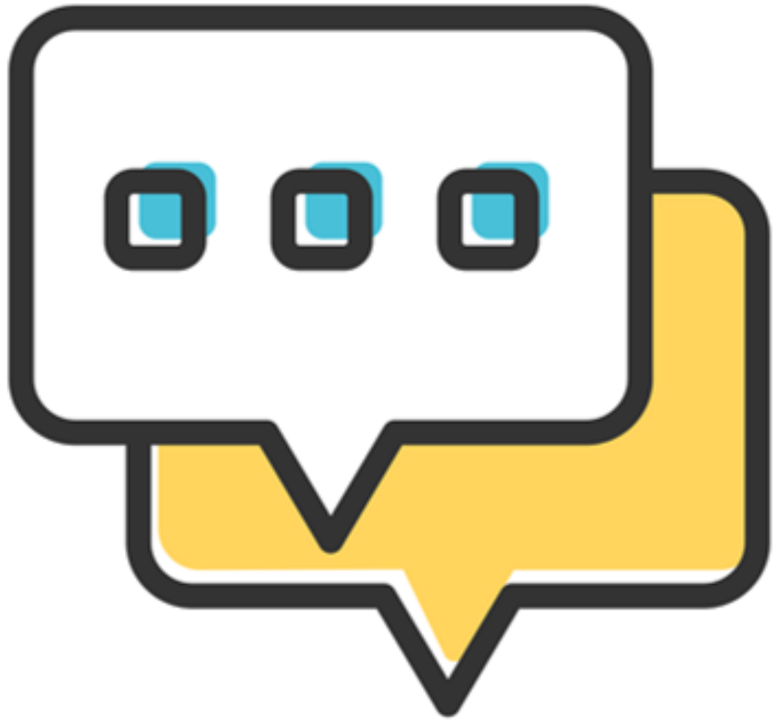
user_prompt = "What is the name of my favorite instructor?"

print(get_response(system_prompt, user_prompt))
```

```
I don't have personal information about you, including the name of your
favorite instructor.
```

How to give extra information?

- Sample previous conversations
- System's prompt



Sample conversations

- Guide model to answer specific questions

```
response = client.chat.completions.create(
  model="gpt-3.5-turbo",
  messages=[{"role": "system",
             "content": "You are a customer service chatbot that responds to user queries in a gentle way"},
            {"role": "user",
             "content": "What services do you offer?"},
           {"role": "assistant",
             "content": "We provide services for web application development, mobile app development, and
            custom software solutions."},
           {"role": "user",
             "content": "How many services do you have?"}])
print(response.choices[0].message.content)
```

```
We have 3 services including web application development, mobile app development, and custom software solutions.
```

- Disadvantage: might need a lot of samples

System prompt

- Includes required context for chatbot answers

```
services = "ABC Tech Solutions, a leading IT company, offers a range of services:
application development, mobile app development, and custom software solutions."
system_prompt = f"""You are a customer service chatbot that responds to user
queries in a gentle way. Some information about our services are delimited by
triple backticks.
```{services}```"""
user_prompt = "How many services do you offer?"
print(get_response(system_prompt, user_prompt))
```

```
We have 3 services including web application development, mobile app development,
and custom software solutions.
```
```

Final note

- Previous methods work well for small contexts
- Larger contexts require more sophisticated techniques

Let's practice!

PROMPT ENGINEERING WITH THE OPENAI API

Congratulations

PROMPT ENGINEERING WITH THE OPENAI API

Fouad Trad
Machine Learning Engineer

Chapter 1

CHAPTER 1

- Key principles of prompt engineering
- Delimited prompts
- Conditional prompts
- Structured outputs

Chapter 2

CHAPTER 1

- Key principles of prompt engineering
- Delimited prompts
- Structured outputs
- Conditional prompts

CHAPTER 2

- Few-shot prompting
- Multi-step prompting
- Chain-of-thought and self-consistency
- Iterative prompt engineering

Chapter 3

CHAPTER 1

- Key principles of prompt engineering
- Delimited prompts
- Structured outputs
- Conditional prompts

CHAPTER 2

- Few-shot prompting
- Multi-step prompting
- Chain-of-thought and self-consistency
- Iterative prompt engineering

CHAPTER 3

- Text summarization and expansion
- Text transformation
- Text analysis
- Code generation and explanation

Chapter 4

CHAPTER 1

- Key principles of prompt engineering
- Delimited prompts
- Structured outputs
- Conditional prompts

CHAPTER 2

- Few-shot prompting
- Multi-step prompting
- Chain-of-thought and self-consistency
- Iterative prompt engineering

CHAPTER 3

- Text summarization and expansion
- Text transformation
- Text analysis
- Code generation and explanation

CHAPTER 4

- System prompt engineering for chatbots
- Role-playing prompts
- External context

Final note

- LLMs have limitations
- Always make sure to review the answers

Keep it up!

PROMPT ENGINEERING WITH THE OPENAI API