Creating customer call transcripts

MULTI-MODAL SYSTEMS WITH THE OPENAI API

James Chapman
Curriculum Manager, DataCamp

datacamp

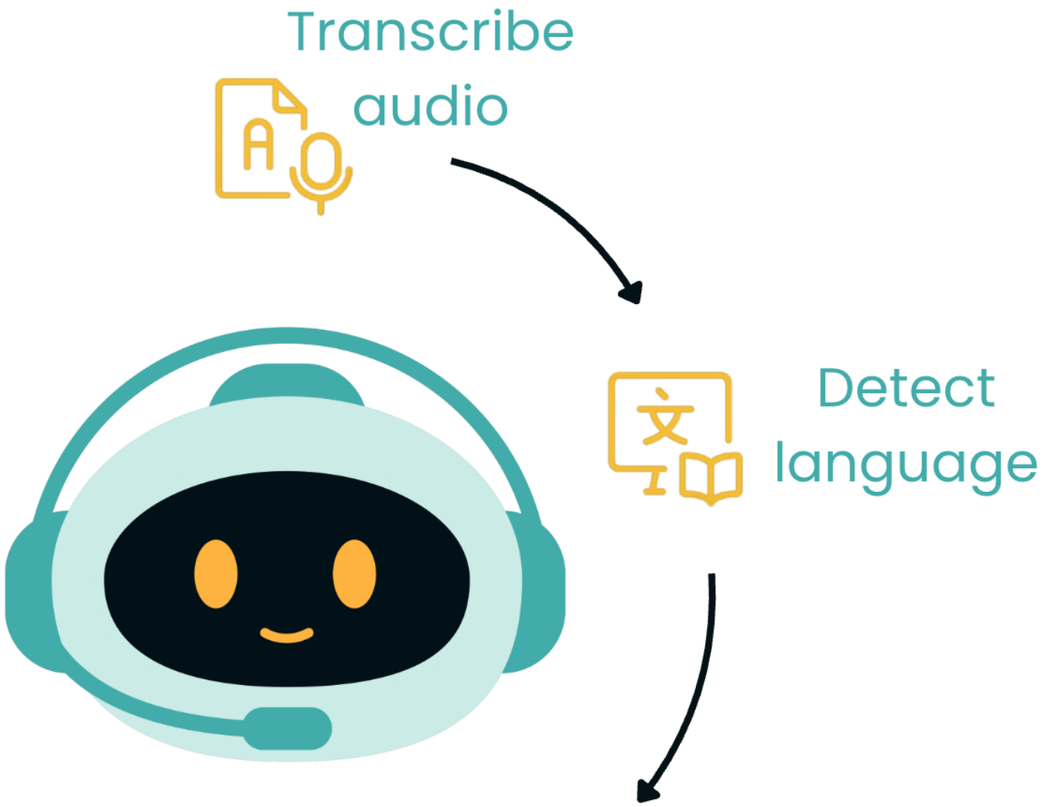
Case study introduction

- AI Engineer at DataCamp
- Handles voice messages
- Speech customer support chatbot

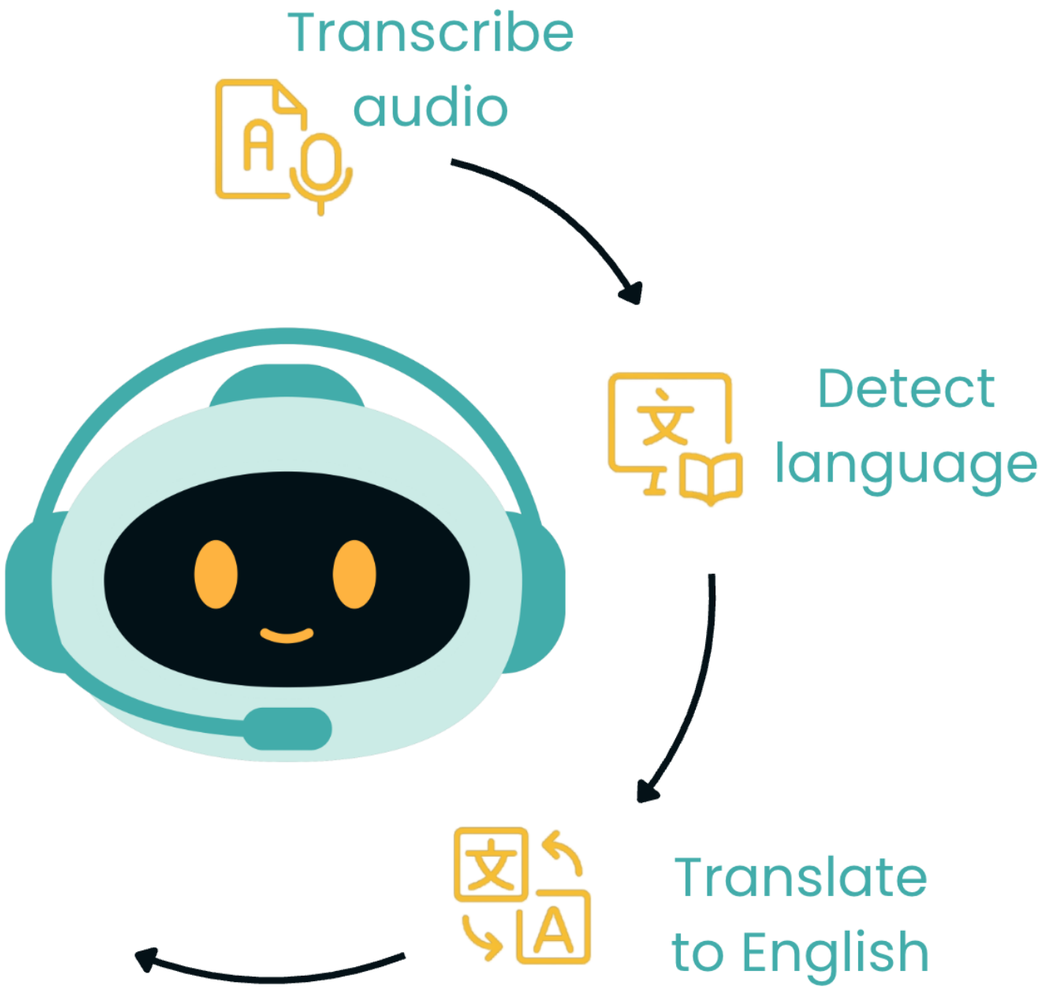
Case study introduction

Case study introduction

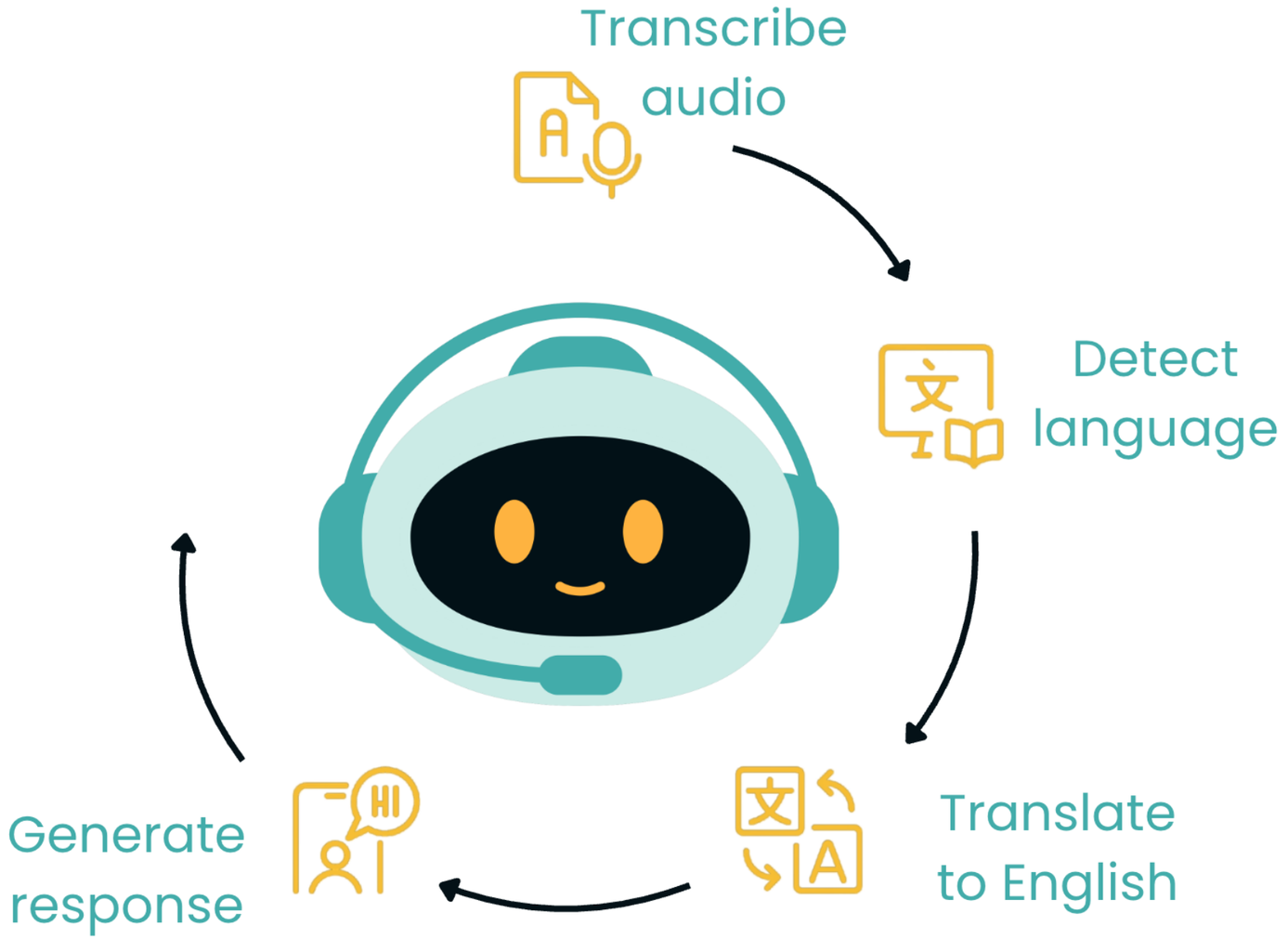
Case study introduction



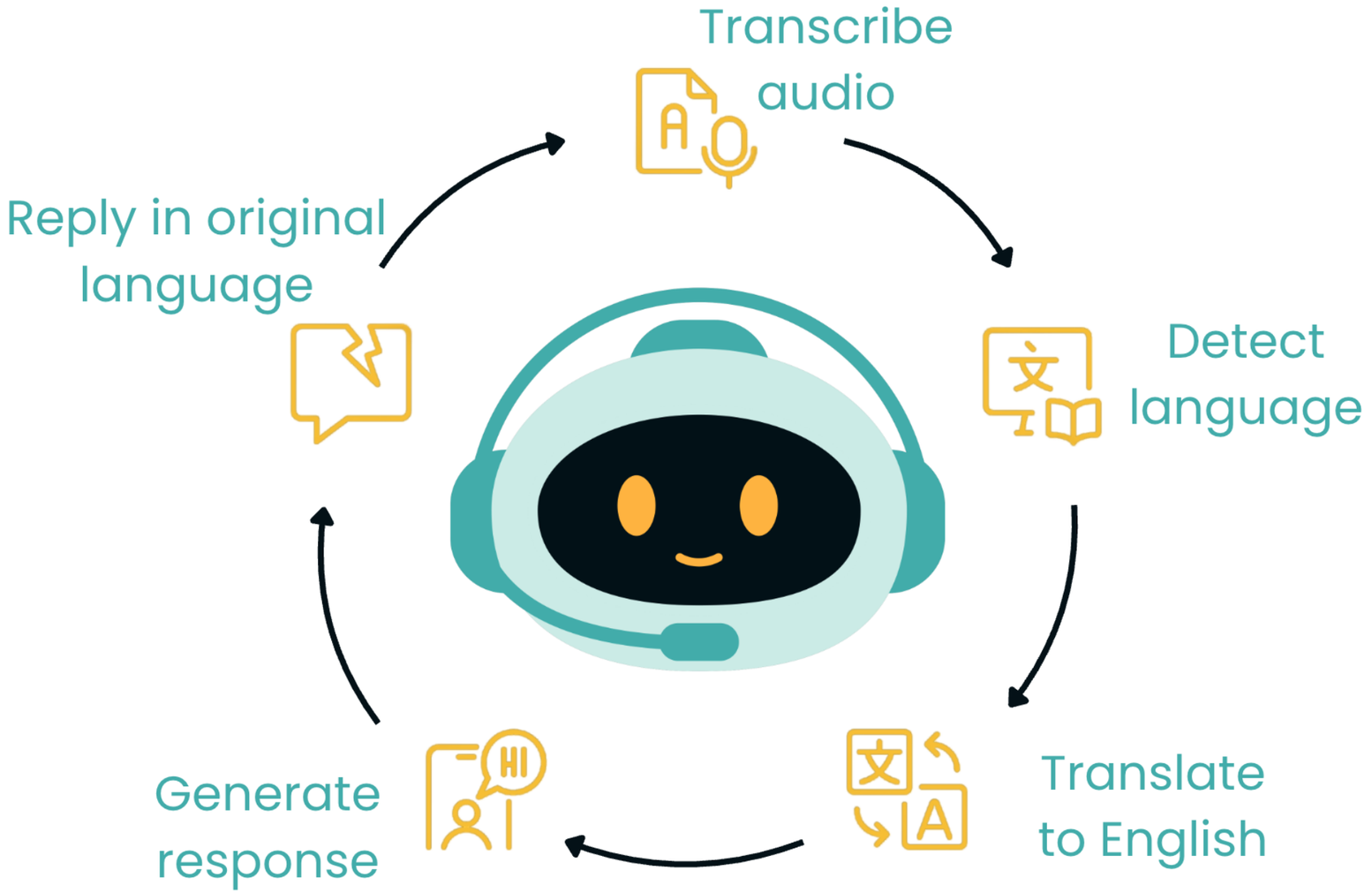
Case study introduction



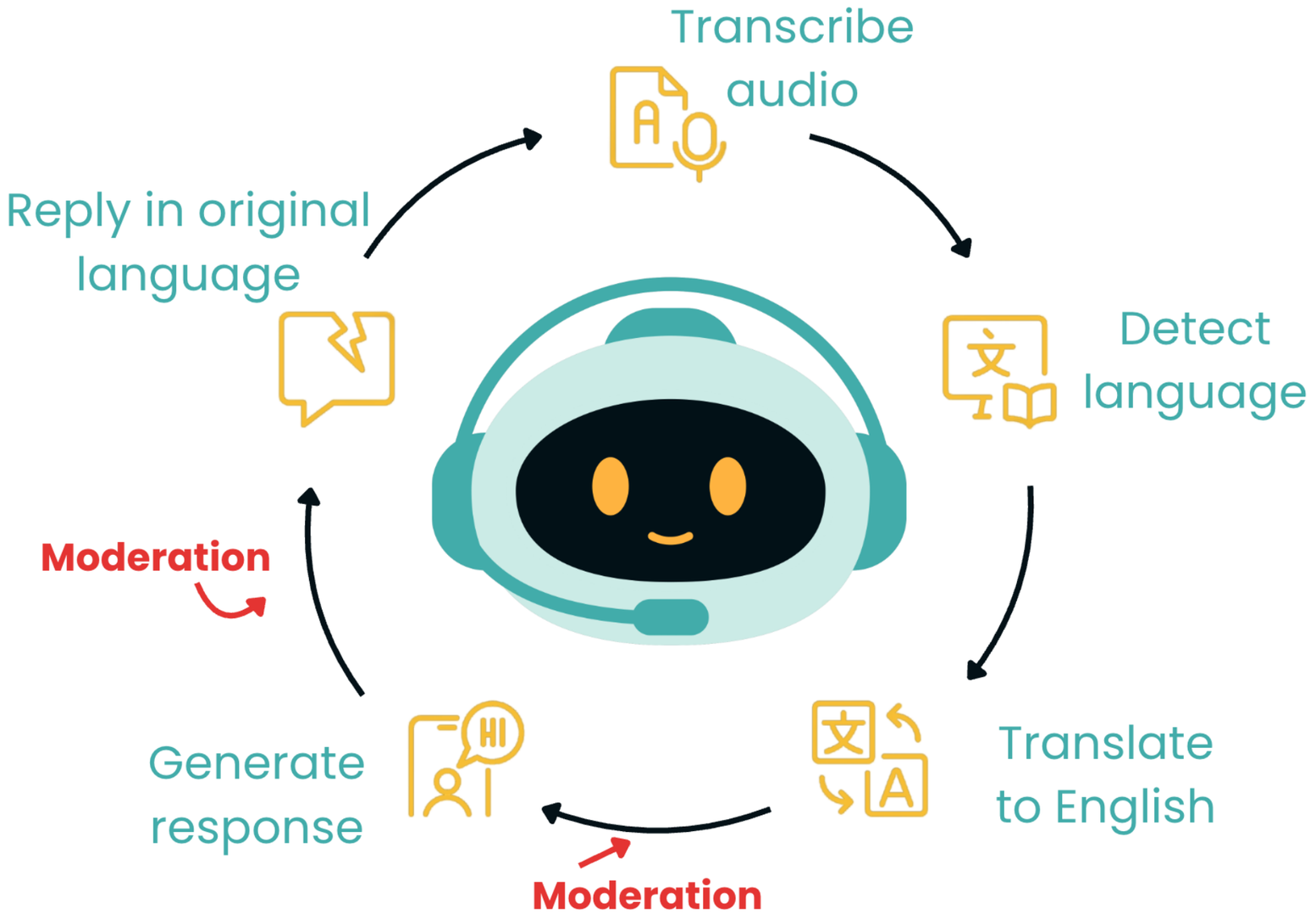
Case study introduction



Case study introduction

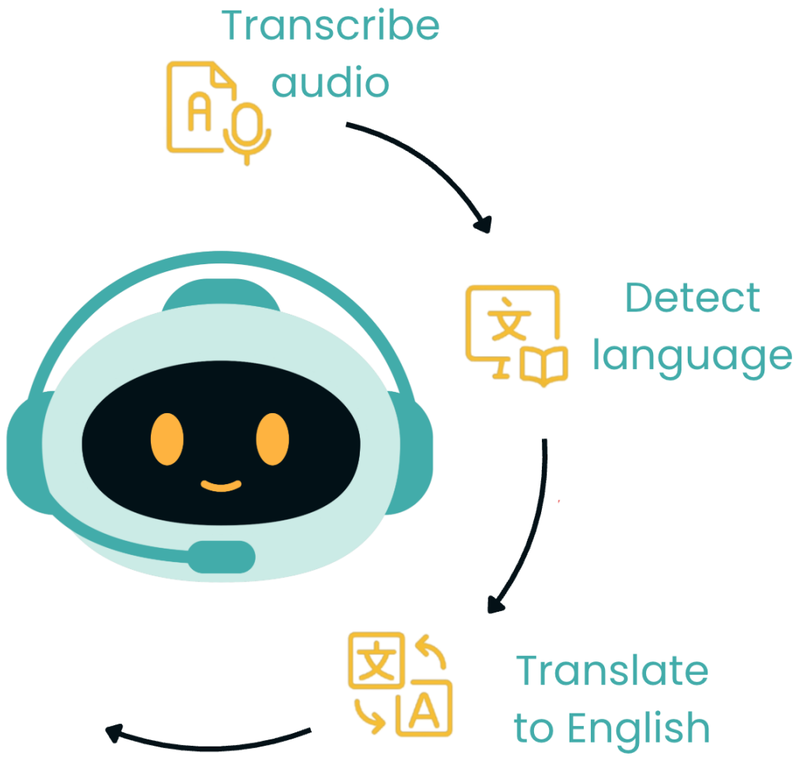


Case study introduction



Case study plan

1. Transcribe the audio into text
2. Detect the language
3. Translate into English
4. Refining the text



Step 1: transcribe audio

```
from openai import OpenAI

client = OpenAI(api_key="ENTER YOUR KEY HERE")

# Open the mp3 file
audio_file = open("recording.mp3", "rb")

# Create a transcript
response = client.audio.transcriptions.create(
                    model="whisper-1",
                    file=audio_file)
```

Step 1: transcribe audio

```
# Extract and print the transcript
transcript = response.text
print(transcript)
```

```
Вітаю! Я хочу стати AI інженером і зараз вивчаю машинне
навчання на дата кемпі. Чи варто зосередитись на OpenAI
API та long chain, якщо я хочу працювати з LLM моделями?
Або краще спершу освоїти PyTorch і навички розгортки у WS?
```

Step 2: detect language

```
response = client.chat.completions.create(
    model="gpt-4o-mini",
    max_completion_tokens=5,
    messages=[{"role": "user",
        "content": f"""Identify the language of the following text and respond
         only with the country code (e.g., 'en', 'uk', 'fr'): {transcript}"""}])

# Extract detected language
language = response.choices[0].message.content
print(language)
```

```
uk
```

Step 3: translate to English

```
response = client.chat.completions.create(
    model="gpt-4o-mini",
    max_completion_tokens=300,
    messages=[
        {"role": "user", "content": f"""Translate this customer transcript
        from country code {language} to English: {transcript}"""}])

# Extract translated text
translated_text = response.choices[0].message.content
```

Step 3: translate to English

```
print(translated_text)
```

```
Hello! I want to become an AI engineer and I'm currently studying machine learning on data-camp. Should I focus on OpenAI API and long chain if I want to work with LLM models? Or is it better to first master PyTorch and deployment skills in WS?
```

Step 3: translate to English

```
print(translated_text)
```

```
Hello! I want to become an AI engineer and I'm currently
studying machine learning on data-camp. Should I focus on
OpenAI API and long chain if I want to work with LLM
models? Or is it better to first master PyTorch and
deployment skills in WS?
```

Step 4: refining the text

```
response = client.chat.completions.create(
    model="gpt-4o-mini",
    max_completion_tokens=300,
    messages=[
    {"role": "user",
     "content": f"""You are an AI assistant that corrects transcripts by fixing
     misinterpretations, names, and terminology. Please refine the following
     transcript:\n\n{translated_text}"""}])

# Extract corrected text
corrected_text = response.choices[0].message.content
```

Step 4: refining the text

```
print(corrected_text)
```

```
Hello! I want to become an AI engineer, and I'm currently
studying machine learning on DataCamp. Should I focus on
the OpenAI API and LangChain if I want to work with large
language models (LLMs)? Or is it better to first master
PyTorch and deployment skills in AWS?
```

Recap

- Transcribed the audio
- Detected and translated language
- Refined the text

- Called OpenAI API four times

```
Вітаю! Я хочу стати AI інженером і зараз вивчаю машинне навчання на дата кемпі. Чи варто зосередитись на OpenAI API та long chain, якщо я хочу працювати з LLM моделями? Або краще спершу освоїти PyTorch і навички розгортки у WS?
```

```
Hello! I want to become an AI engineer and I'm currently studying machine learning on data-camp. Should I focus on OpenAI API and long chain if I want to work with LLM models? Or is it better to first master PyTorch and deployment skills in WS?
```

```
Hello! I want to become an AI engineer, and I'm currently studying machine learning on DataCamp. Should I focus on the OpenAI API and LangChain if I want to work with large language models (LLMs)? Or is it better to first master PyTorch and deployment skills in AWS?
```

Time for practice!

MULTI-MODAL SYSTEMS WITH THE OPENAI API

Generating a customer response

MULTI-MODAL SYSTEMS WITH THE OPENAI API

James Chapman
Curriculum Manager, DataCamp

datacamp

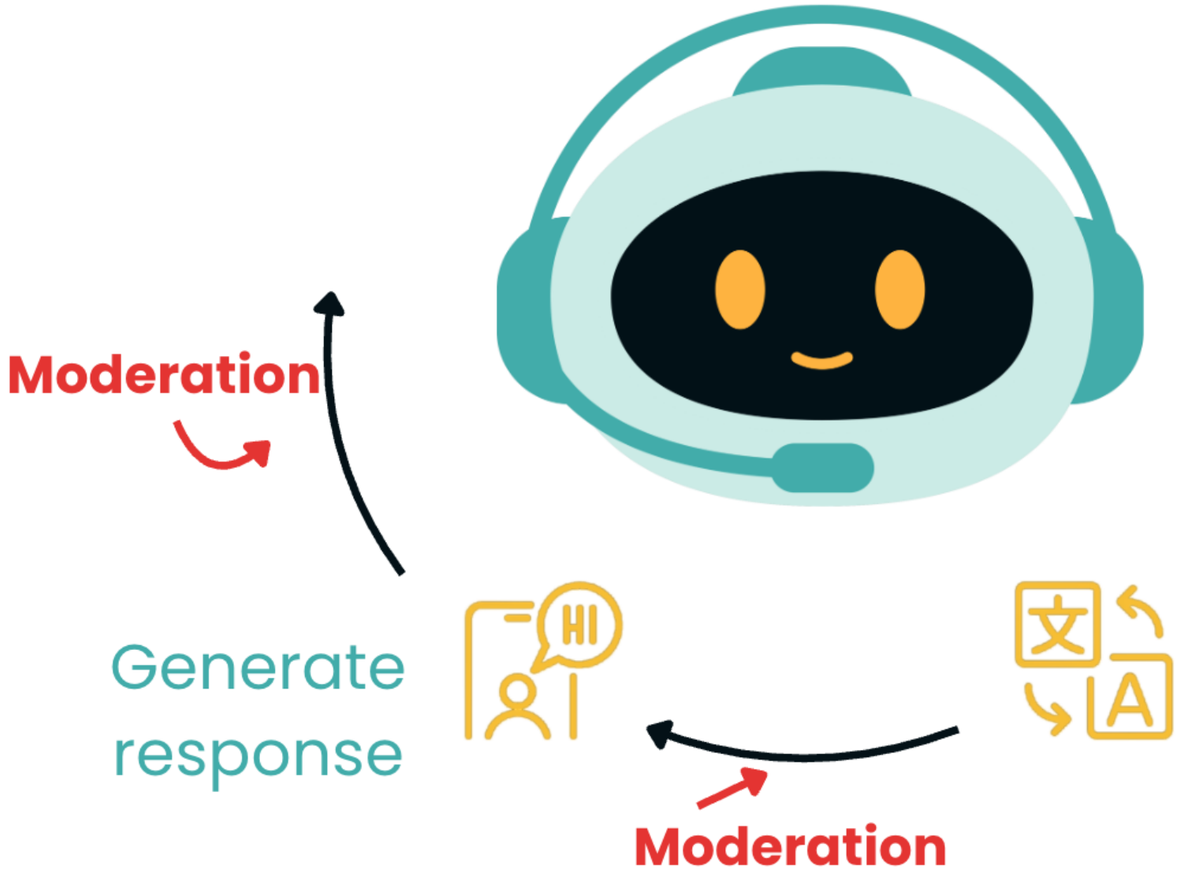
Reminder

```
print(corrected_text)
```

```
Hello! I want to become an AI engineer, and I'm currently
studying machine learning on DataCamp. Should I focus on
the OpenAI API and LangChain if I want to work with large
language models (LLMs)? Or is it better to first master
PyTorch and deployment skills in AWS?
```

Case study plan

- Customer message moderation
- Generating a response
- Response moderation



Customer message moderation

```
from openai import OpenAI

client = OpenAI(api_key="ENTER YOUR KEY HERE")

response = client.moderations.create(
    model="text-moderation-latest",
    input=corrected_text
)

# Extract scores and convert to dictionary
scores = response.results[0].category_scores.model_dump()
```

Customer message moderation

```
print(scores)
```

```
{'harassment': 1.0383088920207229e-05,
  ...
 'hate': 6.848756015642721e-07,
  ...
 'violence': 6.475193367805332e-05,
 ...}
```

Customer message moderation

```
# Extract violence score
violence_score = scores['violence']

# Check if violence score is above 0.7
if violence_score > 0.7:
    print("Content flagged for violence!")
else:
    print("Content is safe from violence.")
```

```
Content is safe from violence.
```

Generating a response

```
print(FAQs)
```

```
Q: How can I upgrade my subscription?
A: You can upgrade your plan anytime in your account settings under 'Billing'.
...
```

```
print(content_overview)
```

```
Content Type: Career Track // Title: Associate AI Engineer for Developers //
...
```

Generating a response

```
instruction_prompt = f"""
#### **Role**
You are a **professional AI support assistant** for DataCamp, handling:
- **Sales**: Pricing, plans, billing
- **Content**: Courses, recommendations, feedback
- **Marketing**: Partnerships, collaborations
```

Generating a response

```
instruction_prompt = f"""
#### **Role**
You are a **professional AI support assistant** for DataCamp, handling:
- **Sales**: Pricing, plans, billing
- **Content**: Courses, recommendations, feedback
- **Marketing**: Partnerships, collaborations

#### **How to Respond**
1. Review documentation: FAQs - {FAQs}, Content Overview - {content_overview}
```

Generating a response

```
instruction_prompt = f"""
#### **Role**
You are a **professional AI support assistant** for DataCamp, handling:
- **Sales**: Pricing, plans, billing
- **Content**: Courses, recommendations, feedback
- **Marketing**: Partnerships, collaborations

#### **How to Respond**
1. Review documentation: FAQs - {FAQs}, Content Overview - {content_overview}
2. Reply clearly using documented info (max 3 paragraphs)
3. If unsure, redirect to **support@datacamp.com**
"""
```

Generating a response

```
response = client.chat.completions.create(
    model="gpt-4o-mini",
    messages=[
        {"role": "system", "content": instruction_prompt},
        {"role": "user", "content": corrected_text}
    ],
    max_completion_tokens=400
)
```

Generating a response

```
# Extract chatbot response
chatbot_reply = response.choices[0].message.content
```

To become an AI engineer, focusing on both the OpenAI API and LangChain is beneficial since these tools are integral to working with large language models (LLMs). The OpenAI Fundamentals skills track on DataCamp covers creating AI applications and advanced prompting techniques, which are essential skills when working with LLMs. You can explore this track here: [OpenAI Fundamentals](https://www.datacamp.com/tracks/openai-fundamentals).

However, mastering PyTorch and deployment skills is also crucial, especially if you plan to delve deeper into the underlying mechanics of LLMs or want to implement custom solutions. The Deep Learning in Python track on DataCamp immerses you in building deep learning models using PyTorch, which is invaluable in AI engineering. You can check it out here: [Deep Learning in Python](https://www.datacamp.com/tracks/deep-learning-in-python).

In summary, I recommend a balanced approach: start with the OpenAI API and LangChain for immediate hands-on experience with LLMs while concurrently developing your skills in PyTorch and deployment techniques. This combination will give you a strong foundation as an AI engineer.

Response moderation

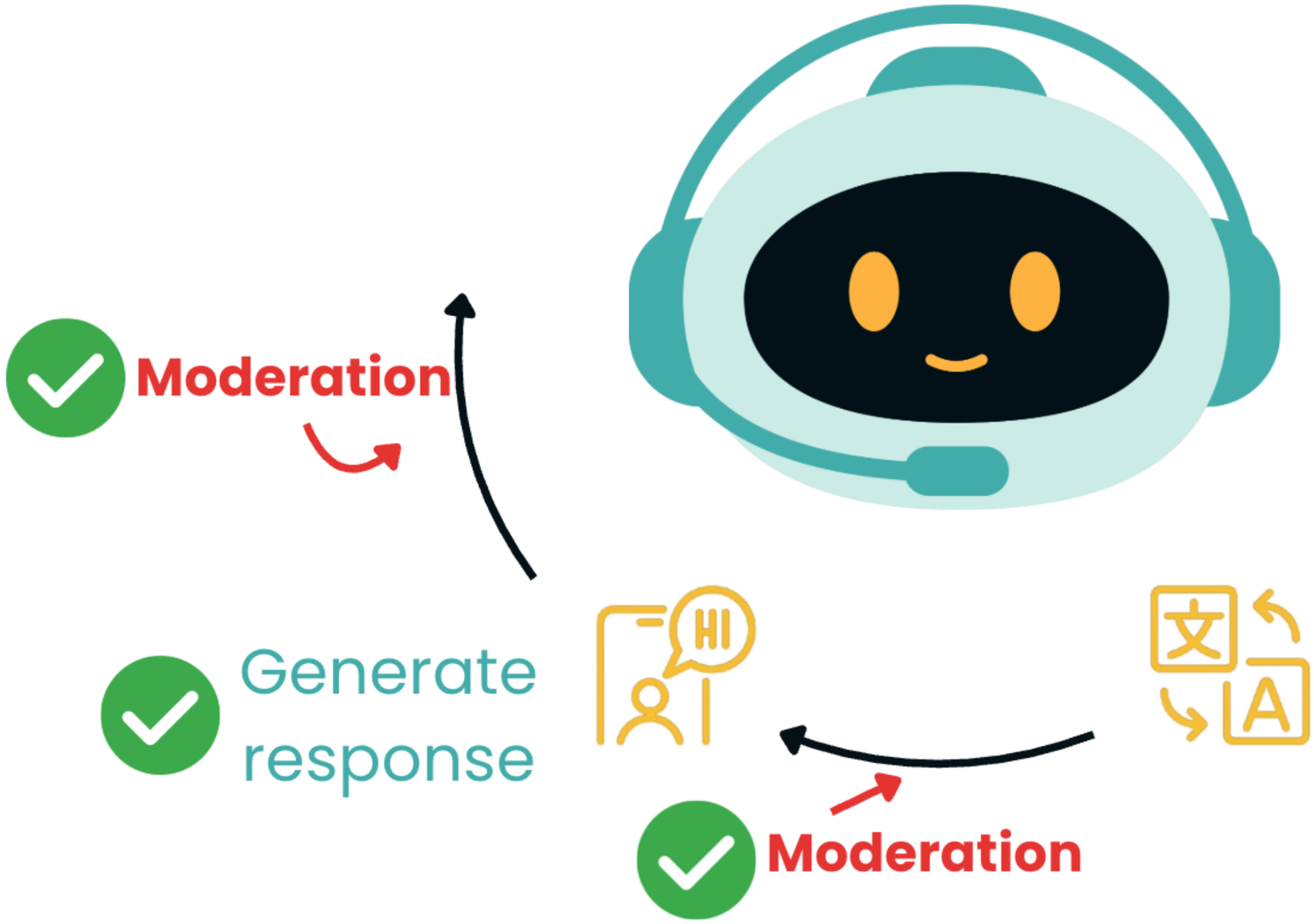
```
response = client.moderations.create(
    model="text-moderation-latest",
    input=chatbot_reply))

scores = response.results[0].category_scores.model_dump()
```

Response moderation

```
# Check if any scores exceed 0.7
if all(score > 0.7 for score in scores.values()):
    print("AI Response flagged for moderation!")
    chatbot_reply = """I'm sorry, but I can't provide a response to that request.
    Please contact support@datacamp.com for further assistance."""
else:
    print("AI Response is safe.")
```

```
AI Response is safe.
```

Recap



Let's practice!

MULTI-MODAL SYSTEMS WITH THE OPENAI API

Creating a speech response for customers

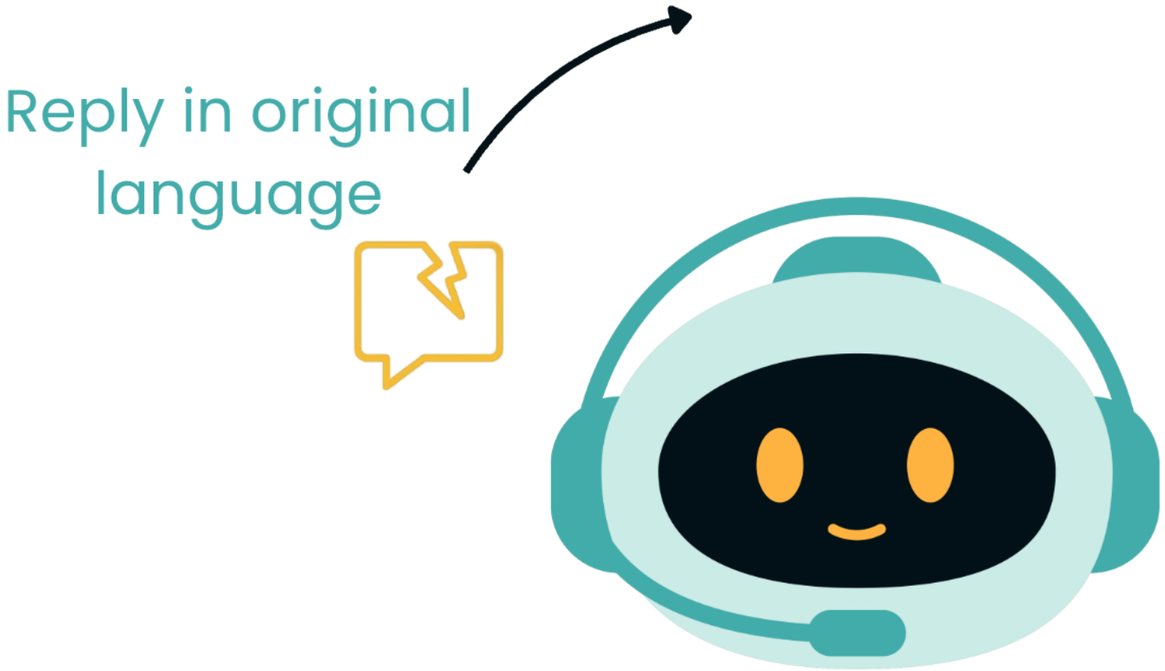
MULTI-MODAL SYSTEMS WITH THE OPENAI API

James Chapman
Curriculum Manager, DataCamp

datacamp

Case study plan

- Response translation
- Converting text in audio



Variables to use

Detected language

```
print(language)
```

```
uk
```

Generated response

```
print(chatbot_reply)
```

To become an AI engineer, focusing on both the OpenAI API and LangChain is beneficial since these tools are integral to working with large language models (LLMs). The OpenAI Fundamentals skills track on DataCamp covers creating AI applications and advanced prompting techniques, which are essential skills when working with LLMs. You can explore this track here: [OpenAI Fundamentals](https://www.datacamp.com/tracks/openai-fundamentals).

However, mastering PyTorch and deployment skills is also crucial, especially if you plan to delve deeper into the underlying mechanics of LLMs or want to implement custom solutions. The Deep Learning in Python track on DataCamp immerses you in building deep learning models using PyTorch, which is invaluable in AI engineering. You can check it out here: [Deep Learning in Python](https://www.datacamp.com/tracks/deep-learning-in-python).

In summary, I recommend a balanced approach: start with the OpenAI API and LangChain for immediate hands-on experience with LLMs while concurrently developing your skills in PyTorch and deployment techniques. This combination will give you a strong foundation as an AI engineer.

Response translation

```
response = client.chat.completions.create(
    model="gpt-4o-mini",
    messages=[
        {"role": "system", "content": f"""Translate the following text
        from English to country code {language}. Only return the translated text!"""},
        {"role": "user", "content": chatbot_reply}
             ],
    max_completion_tokens=500)
```

Response translation

```
# Extract and print the translated response
translated_reply = response.choices[0].message.content
print(translated_reply)
```

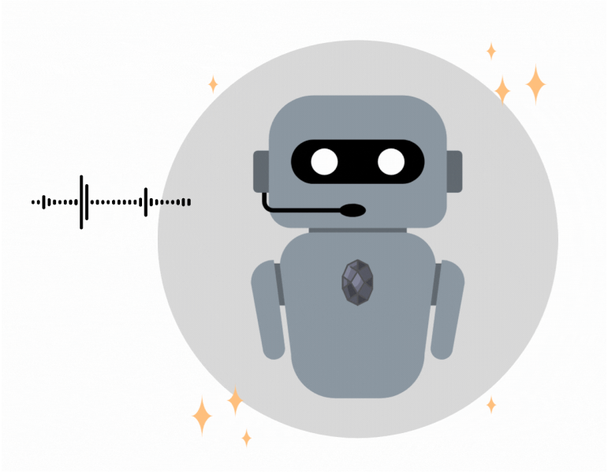
Щоб стати інженером штучного інтелекту, корисно зосередитися як на OpenAI API, так і на LangChain, оскільки ці інструменти є невід'ємними елементами роботи з великими мовними моделями (LLM). Трек навичок OpenAI Fundamentals на DataCamp охоплює створення AI-додатків та розвинуті техніки підказок, які є важливими навичками при роботі з LLM. Ви можете ознайомитися з цим треком тут: [OpenAI Fundamentals](https://www.datacamp.com/tracks/openai-fundamentals).

Проте володіння PyTorch і навичками розгортання також є вирішальним, особливо якщо ви плануєте зануритися глибше в основні механізми LLM або хочете реалізувати індивідуальні рішення. Трек Deep Learning in Python на DataCamp занурює вас у створення моделей глибокого навчання з використанням PyTorch, що є безцінним у сфері інженерії штучного інтелекту. Ви можете перевірити його тут: [Deep Learning in Python](https://www.datacamp.com/tracks/deep-learning-in-python).

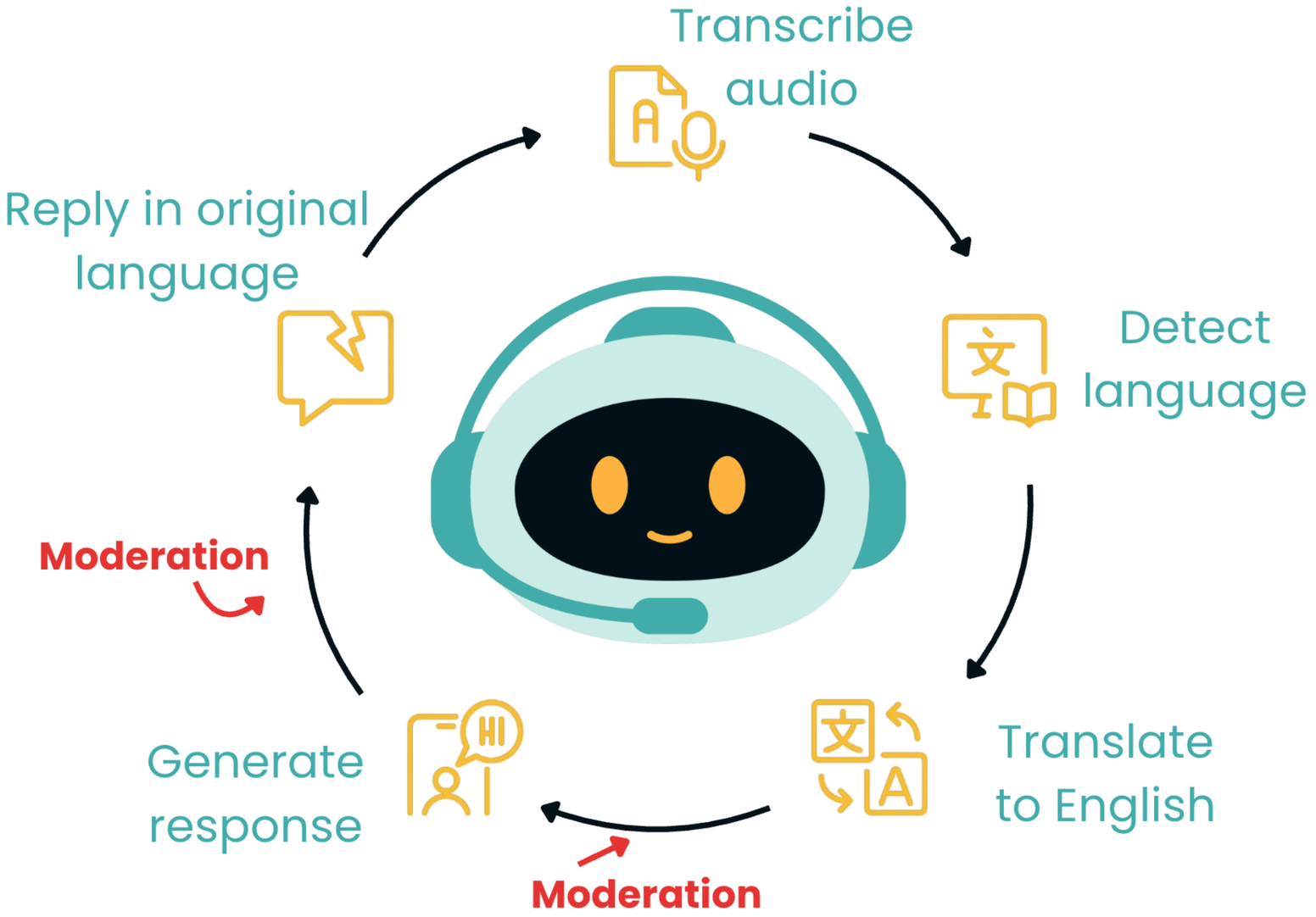
Підсумовуючи, я рекомендую збалансований підхід: почніть з OpenAI API та LangChain для негайного практичного досвіду з LLM, одночасно розвиваючи свої навички у PyTorch та техніках розгортання. Ця комбінація надасть вам міцну основу як інженера штучного інтелекту.

Text-to-speech

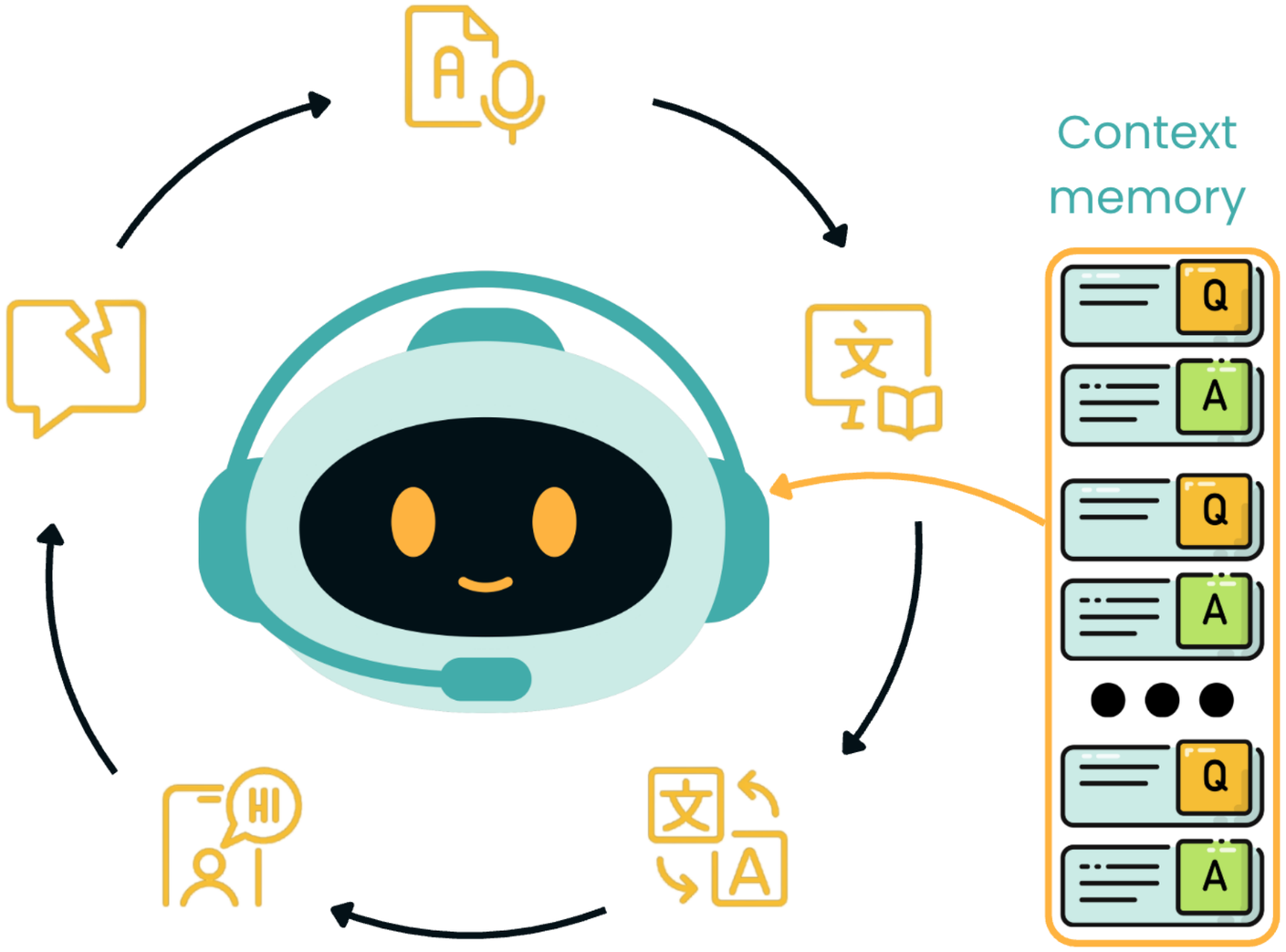
```
response = client.audio.speech.create(
    model="gpt-4o-mini-tts",
    voice="onyx",
    input=translated_reply)

response.stream_to_file("audio_reply.mp3")
```



Case study recap



Next steps



Let's practice!

MULTI-MODAL SYSTEMS WITH THE OPENAI API

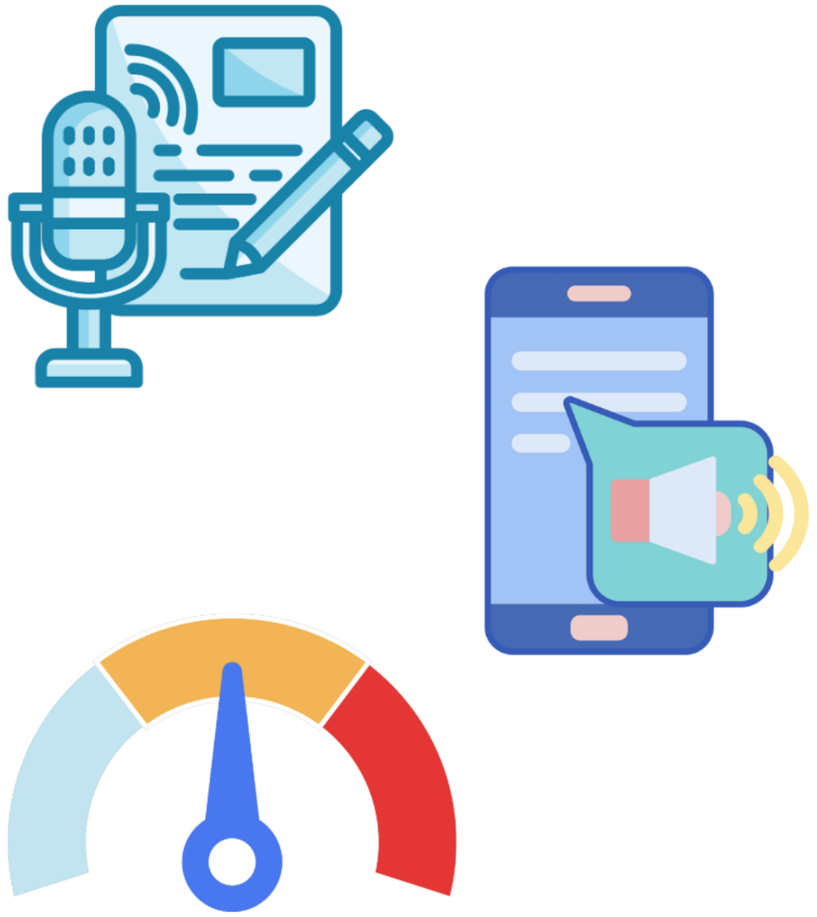
Congratulations!

MULTI-MODAL SYSTEMS WITH THE OPENAI API

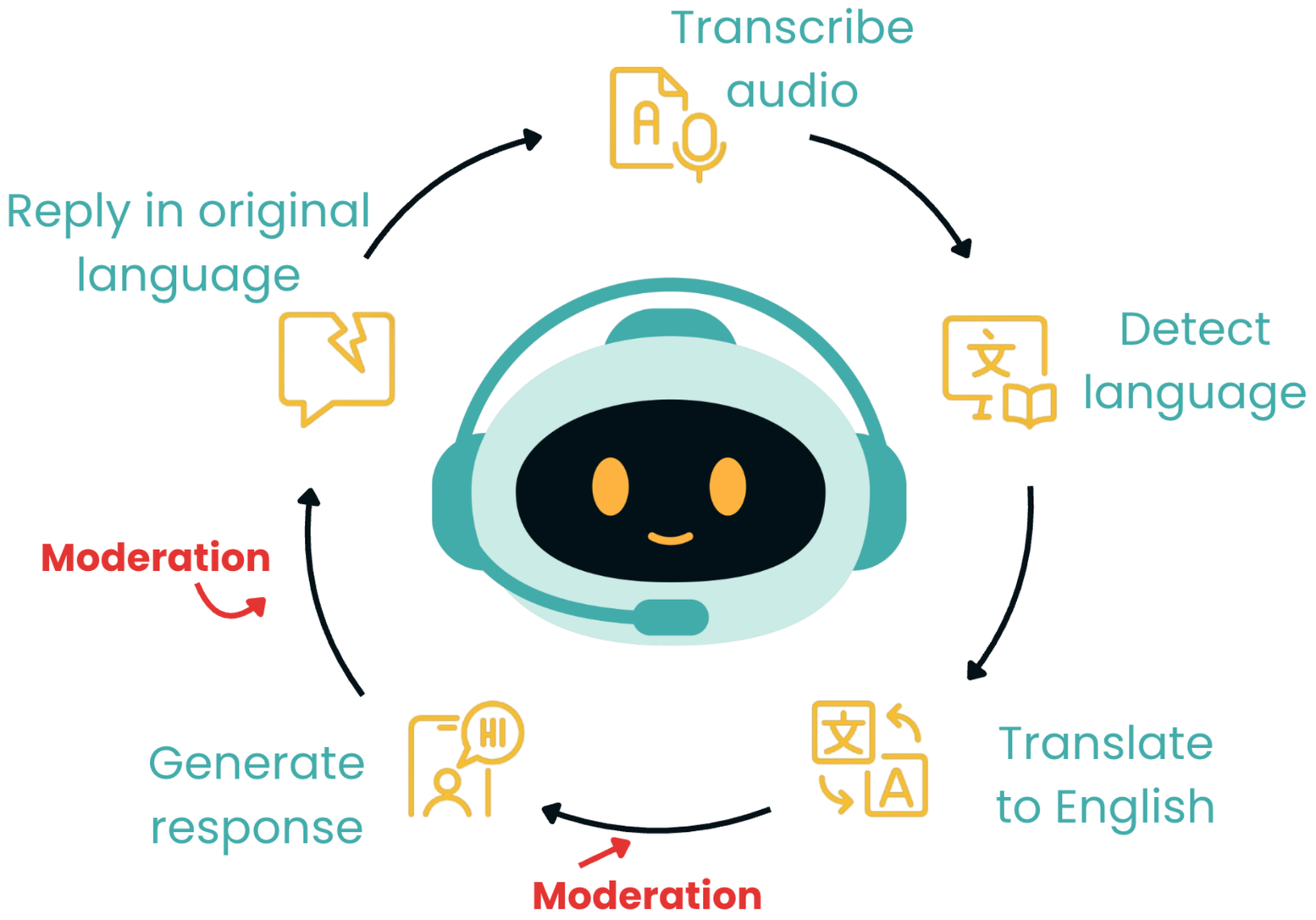
James Chapman
Curriculum Manager, DataCamp

Chapter 1

- Speech transcription and translation
- Text-to-speech
- Content moderation



Chapter 2


Transcribe audio
Detect language
Translate to English
Moderation
Generate response
Moderation
Reply in original language

What's next?

More on the OpenAI API

- Developing AI Systems with the OpenAI API
- Introduction to Embeddings with the OpenAI API

Other AI application tool stacks

- Developing LLM Applications with LangChain

Projects

- Personalized Language Tutor
- Planning a Trip to Paris with the OpenAI API

Let's practice!

MULTI-MODAL SYSTEMS WITH THE OPENAI API