Learning Objectives

Learners will be able to...

- **Use ChatGPT as collaborative tool**
- **Create a system message that sets the context of the AI**
- **Use ChatGPT as collaborative tool for brainstorming and peer review**

info

Make Sure You Know

You are familiar with Python.

Limitations

This is a gentle introduction. So there is a little bit of Python programming. The information might not be the most up to date as OpenAI releases new features.

Collaborative Programming with ChatGPT API

The ChatGPT API can be an invaluable tool for enhancing collaboration among developers, enabling team members to discuss programming concepts, work through challenges, and share ideas effectively. We will discuss techniques for using the ChatGPT API to facilitate collaboration, provide guidance on managing conversations, and explore examples that demonstrate how the API can be used to improve team productivity and communication.

Techniques for Collaborative Programming:

1. **Encourage open discussion:** Use the ChatGPT API as a neutral third party to prompt team members to share their thoughts, questions, and suggestions.
2. **Share code snippets:** Leverage the API to review, discuss, and optimize code collaboratively in real-time.
3. **Seek expert advice:** Utilize the ChatGPT API to obtain guidance on programming concepts, best practices, or debugging assistance.

Managing Conversations:

1. **Use system messages:** Set the context of the AI as a collaborative programming assistant with expertise in the relevant programming languages and domains.
2. **Incorporate user messages:** Ensure each team member is represented in the conversation using distinct user roles or identifiers.
3. **Maintain context:** Keep the conversation focused and coherent by providing appropriate context and background information as needed.

Choosing System

Lets try creating a system message that sets the context of the AI as a collaborative programming assistant with expertise in Python. The user messages represent two team members, Alice and Bob, with distinct questions related to Python decorators.

First step in our code we are going to get our libraries and API keys.

```
import os
import openai
import secret
openai.api_key=secret.api_key
```

To make visualizing easier, we are going to create a variable named `message` that will store our prompt.

```
message=[{"role": "system", "content": "You are a collaborative
        programming assistant with expertise in Python."},
        {"role": "user", "content": "Alice: I'm having trouble
        understanding how to use Python decorators. Can you
        explain them to me?"},
        {"role": "user", "content": "Bob: I think I understand
        decorators, but I could use some help with a specific
        example. Can you provide one?"}
    ]
```

After we are going to make our API call and set it equal to `response`. We will use slicing with our print statement in order to better see the information that the response generates.

```
response=openai.ChatCompletion.create(
  model="gpt-3.5-turbo",
  messages= message
)
print(response['choices'][0]['message']['content'].strip())
```

Tool to Encourage Discussion

The ChatGPT API can be a powerful tool for encouraging discussion among developers during collaborative programming. By facilitating open communication, providing neutral opinions, and fostering a supportive environment, the API can enhance teamwork and lead to more productive programming sessions.

Methods for Encouraging Discussion:

1. **Ask open-ended questions:** Use the ChatGPT API to pose open-ended questions that prompt team members to share their thoughts, ideas, or concerns.
2. **Request peer review:** Leverage the API to initiate code reviews, asking team members for feedback and suggestions for improvement.
3. **Encourage brainstorming:** Utilize the API to stimulate brainstorming sessions by providing prompts, relevant examples, or creative problem-solving techniques.

Replace the prompt inside `message` variable with the code below. Now consider the following API call to encourage a discussion among team members working on a Python project:

```
[{"role": "system", "content": "You are a collaborative
      programming assistant with expertise in Python."},
      {"role": "assistant", "content": "What are some key
      challenges you've faced while working on this Python
      project, and how have you addressed them? Share your
      experiences and any potential solutions."}
  ]
```

The system message sets the context of the AI as a collaborative programming assistant with expertise in Python. The assistant message poses an open-ended question, inviting team members to discuss their experiences and potential solutions.

Another possible approach is using the ai as an assistant that encourages team members to discuss a specific coding problem

```
[{"role": "system", "content": "You are a collaborative
        programming assistant with expertise in Python."},
        {"role": "assistant", "content": "We're currently
        working with list comprehensions in Python. Can you
        share an example of a challenging list comprehension
        you've come across and how you approached solving it?
        Feel free to discuss any alternative solutions you
        considered."}
    ]
```

The system message sets the context of the AI as a collaborative programming assistant with expertise in Python. The assistant message encourages team members to discuss a specific coding problem related to list comprehensions and share their approaches to solving it, along with any alternative solutions they considered.

By incorporating coding-related examples like this, the ChatGPT API can foster productive discussions around programming challenges and help developers learn from each other's experiences and insights. One key thing that we can make note of that is our ability to control the AI. Because of our control we can do things like adapting it to team dynamics: Customize the API prompts based on team members' preferences, communication styles, and expertise levels. Or we can control its responsiveness, Use the API to address questions, clarify misunderstandings, or provide additional information as needed.

Peer review

Peer review plays a critical role in the collaborative programming process, allowing developers to identify potential issues, optimize code, and share best practices. With the ChatGPT API, you can initiate and facilitate code reviews, making it easier for team members to provide feedback and suggestions for improvement.

Consider the following API call to request a peer review of a Python function among team members:

```
[{"role": "system", "content": "You are a collaborative
        programming assistant with expertise in Python."},
        {"role": "assistant", "content": "Here's a Python
        function we've been working on:\n\n```python\ndef
        calculate_area(width, height):\n    return width *
        height\n```\n\nCould you please review this function and
        provide any suggestions for improvement or optimization?
        Feel free to discuss any potential issues or alternative
        approaches."}
    ]
```

the system message sets the context of the AI as a collaborative programming assistant with expertise in Python. The assistant message shares a code snippet with team members and invites them to review the code and provide suggestions for improvement or optimization.

Now, Imagine that two team members are having a disagreement about whether to use exceptions or return codes for error handling in a Python project. You can use the ChatGPT API to facilitate a neutral discussion, helping them explore the pros and cons of each approach:

```
[
        {"role": "system", "content": "You are a collaborative
programming assistant with expertise in Python."},
        {"role": "user", "content": "Alice and Bob are having a
disagreement about whether to use exceptions or return codes for
error handling in our Python project. Can you help them discuss
the pros and cons of each approach?"}
    ]
```

the system message sets the context of the AI as a collaborative programming assistant with expertise in Python. The user message describes the disagreement between Alice and Bob. The assistant message provides a neutral overview of the pros and cons of using exceptions and

return codes for error handling in Python, inviting Alice and Bob to share their thoughts based on the specific requirements and constraints of the project.

The ChatGPT API can expedite the review process, allowing developers to address issues more quickly and efficiently.
By acting as a neutral party, the API can help create a more open and supportive environment for constructive feedback.

Brainstorming

Brainstorming is an essential aspect of problem-solving in software development, as it enables developers to explore different approaches and come up with innovative solutions. The ChatGPT API can be an effective tool for facilitating brainstorming sessions, offering suggestions, and helping developers consider alternative ways of tackling coding challenges

---

Methods for Using ChatGPT API for Brainstorming:

1. **Present the problem:** Use the ChatGPT API to present the coding problem to the developers, inviting them to think about potential solutions.
2. **Generate ideas:** Leverage the API to generate a list of ideas, approaches, or techniques that could be applied to the problem.
3. **Encourage discussion:** Utilize the API to create a supportive environment for open discussion and evaluation of the generated ideas.

Consider the following API call to help a team brainstorm solutions for optimizing a Python function that processes a large dataset:

```
[
        {"role": "system", "content": "You are a collaborative
        programming assistant with expertise in Python."},
        {"role": "assistant", "content": "We have a Python
        function that processes a large dataset, but its
        performance is not satisfactory. Let's brainstorm some
        ideas to optimize the function. Here are some initial
        suggestions:\n\n1. Use vectorized operations with
        libraries like NumPy.\n2. Apply parallel processing
        techniques using the multiprocessing module. \nPlease
        share your thoughts on these ideas, and feel free to
        suggest additional approaches."}
    ]
```

the system message sets the context of the AI as a collaborative programming assistant with expertise in Python. The assistant message presents a coding problem related to performance optimization and offers initial suggestions for brainstorming. The message encourages the team members to discuss these ideas and contribute their own suggestions.

Let's consider a scenario where a team of developers is working on a web application and looking for ideas to improve the user experience (UX). They can use the ChatGPT API to generate ideas for enhancing the UX. Here's an example API call:

```
[
        {"role": "system", "content": "You are a collaborative
programming assistant with expertise in Python."},
        {"role": "assistant", "content": "We have a Python
function that processes a large dataset, but its performance is
not satisfactory. Let's brainstorm some ideas to optimize the
function. Here are some initial suggestions:\n\n1. Use
vectorized operations with libraries like NumPy.\n2. Apply
parallel processing techniques using the multiprocessing
module.\n3. Profile the code to identify performance bottlenecks
and optimize them.\n4. Consider using a more efficient data
structure or algorithm.\n\nPlease share your thoughts on these
ideas, and feel free to suggest additional approaches."}
    ]
```

The system message sets the context of the AI as a collaborative assistant with expertise in web development and user experience. The assistant message asks for ideas to improve the UX of the web application.

By generating ideas, encouraging discussion, and promoting collaboration, developers can find innovative solutions to coding problems and improve the overall quality of their projects.

Sure! Let's consider a scenario where a team of developers is working on a web application and looking for ideas to improve the user experience (UX). They can use the ChatGPT API to generate ideas for enhancing the UX. Here's an example API call:

```
[
        {"role": "system", "content": "You are a collaborative
assistant with expertise in web development and user
experience."},
        {"role": "assistant", "content": "Our team is working on
a web application, and we want to improve the user experience.
Can you generate some ideas to enhance the UX of our
application?"}
    ]
```

In this example, the system message sets the context of the AI as a collaborative assistant with expertise in web development and user experience. The assistant message asks for ideas to improve the UX of the web application.

A possible response from the ChatGPT API might include:

```
{
  "role": "assistant",
  "content": "Certainly! Here are some ideas for improving the
        user experience of your web application:

1. Implement a clean and intuitive user interface that helps
        users navigate the application easily.
2. Optimize page load times to ensure a smooth and responsive
        experience.
3. Add informative tooltips and contextual help to guide users
        through the application.
4. Personalize the user experience by offering customizable
        themes, layouts, or content based on user preferences.
5. Implement clear and concise error messages that help users
        understand and resolve issues.
6. Use A/B testing to evaluate the effectiveness of different
        design elements and make data-driven decisions.
7. Ensure the application is accessible to users with
        disabilities by following accessibility guidelines and
        best practices.

Feel free to discuss these ideas with your team and consider any
        additional suggestions they might have."
}
```

In this response, the ChatGPT API generates a list of ideas to improve the user experience of the web application, providing the team with various options to explore and discuss.

Coding Exercise

In this exercise, your task is to create a simplified collaborative tool using the ChatGPT API that can help a user get answers to their questions.

You need to write a function called `collaborative_chat_gpt()`. This function should take a single user message as an argument and return a response from the AI model.

For simplicity, let's assume that the role of the AI model is always "You are a knowledgeable assistant."

Function Signature:

```
def collaborative_chat_gpt(prompt: str) -> str:
    pass
```

Example:

```
print(collaborative_chat_gpt("What is the capital of France?"))
# The output should be a response generated by the AI model
        based on the user message.
```

Notes:

Please handle the API responses carefully and consider all possible exceptions and errors.