

Supercharge Your API Program with LLMs

Janet Wagner — Read time: 6 minutes

For more API insights, visit our other blogs.

Large language models (LLMs), a type of artificial intelligence model designed to understand and generate human-like text, have captured people's attention in dramatic fashion lately. Well-known tech companies — including Baidu, Google, Meta, and OpenAI — have made LLM-related announcements within the past few months. But it was OpenAI's release of ChatGPT this past November that propelled LLMs into mainstream news and the race to create the next best LLM. There are now an ever-increasing variety of LLMs available as powerful tools for a wide range of use cases, including designing and building APIs.

LLM AI models analyze massive volumes of text data to generate outputs such as natural language, code completions, translations, and summarizations. They work best in scenarios where it's easy to predict what comes next. And they're ideal for completing tasks for use cases involving well-defined guardrails and standard practices, like building API programs that follow an [API design-first approach](#).

This article highlights a few ways to use LLMs to supercharge your API program. Next time we'll cover using APIs to enable AI capabilities in applications — leveraging AI *with* APIs.

Using an LLM for Code

One of the ways you can boost your API program is by using an LLM to optimize and speed up API development, specifically tasks involving code. You can use an LLM for code in many ways, such as:

- Generate code for a REST API in a language like Python, C#, or JavaScript.
- Generate scripts for testing the performance of an API.
- Autocomplete API code based on input by the developer (like in an IDE).
- Detect errors or redundancies in existing API code.

LLMs work great for tasks that involve generating text for easily predictable content, like the code for a basic REST API or the text for API documentation. For example, I asked ChatGPT to write code for a REST API in Python. Here are two screenshots that show some of the model's output.



Sure! Here's an example of how you can create a basic REST API in Python using the Flask framework:

```
python Copy code  
  
from flask import Flask, jsonify, request  
  
app = Flask(__name__)  
  
# define some sample data  
fruits = [  
    {'id': 1, 'name': 'apple', 'color': 'red'},  
    {'id': 2, 'name': 'banana', 'color': 'yellow'},  
    {'id': 3, 'name': 'orange', 'color': 'orange'}  
]  
  
# define some API routes  
@app.route('/fruits', methods=['GET'])  
def get_fruits():  
    return jsonify(fruits)  
  
@app.route('/fruits/<int:id>', methods=['GET'])  
def get_fruit(id):  
    fruit = [fruit for fruit in fruits if fruit['id'] == id]  
    if len(fruit) == 0:  
        abort(404)  
    return jsonify(fruit[0])  
  
@app.route('/fruits', methods=['POST'])  
def add_fruit():  
    if not request.json or not 'name' in request.json:  
        abort(400)  
    fruit = {  
        'id': fruits[-1]['id'] + 1,  
        'name': request.json['name'],  
        'color': request.json.get('color', '')  
    }  
    fruits.append(fruit)  
    return jsonify(fruit), 201  
  
@app.route('/fruits/<int:id>', methods=['PUT'])  
def update_fruit(id):  
    fruit = [fruit for fruit in fruits if fruit['id'] == id]  
    if len(fruit) == 0:  
        abort(404)  
    if not request.json:  
        abort(400)
```

Regenerate response

```
This API provides endpoints to get a list of all fruits (^/fruits^`), get information about a specific fruit (^/fruits/<id>^`), add a new fruit (^/fruits^` with a POST request), update an existing fruit (^/fruits/<id>^` with a PUT request), and delete an existing fruit (^/fruits/<id>^` with a DELETE request).
```

Many developers reuse the same boilerplate code structure for each REST API they create and often reuse the same responses for different API endpoints. Some of them reuse code to create different scripts for testing APIs. Most, if not all, developers find all this repetitive code work *tedious and time-consuming* — making it perfect for an LLM!

Applying AI to Code Benefits API Producers

An LLM can complete mundane, repetitive API coding tasks *fast*. The less mundane and time-consuming tasks one must do, the more time they have to focus on important problem-solving. And with AI assisting with coding, developers will have more room for creative collaboration. An LLM could write the code for an API, while developers would only need to review and tweak the code as needed. AI enables developers to quickly test new ideas without having to write a lot of code upfront.

Applying AI to Code Benefits API Consumers

API consumers want consistency across all the APIs they use. They also want APIs that help them solve specific problems. Applying AI to code-related tasks gives developers more time to innovate and work on new APIs to solve more complex issues. And an LLM can help internal stakeholders adhere to API standards and best practices, for example, by checking API code against a style guide, so that consumers ultimately get the consistent and useful APIs

they need. You can build reliable and consistent APIs faster with the help of an LLM.

Using an LLM for API Documentation

Great documentation helps users understand how to use your API and get to ‘Hello World’ faster. You can use an LLM to ensure that you provide excellent documentation for *all* your APIs. LLMs are ideal for tasks that involve digesting and summarizing text. They’re also great at outlining concepts and explaining things by analogy. Creating the [documentation for an API](#) requires all these abilities (though a tool like [Stoplight](#) is also a great option). An LLM can help you with documentation in several ways:

- Have the LLM analyze your API source code and then automatically generate API documentation, describing the functionality of its endpoints in a way humans can understand.
- Let the LLM review your existing documentation, looking for holes and then making suggestions on how to fill them. It could also analyze your documentation to ensure the content matches your current API functionality.
- If you know where your API customers are having problems, you could use an LLM to analyze code and support requests and then generate relevant guidance adding that information to your documentation or an FAQ page.

In general, using an LLM can save developers a lot of time because they don’t have to manually create the documentation for the APIs they build. And let’s be honest — most developers would rather work on designing and building APIs than write the documentation for them. AI can generate consistent, easy-to-read documentation much faster than humans. We all know API

documentation matters a lot, but few developers have enough time to work on it. With an LLM, documentation is as simple as a short natural language prompt like “create a docstring for this function” or “create an API reference for this endpoint.”

LLMs for API Design and Development: A Summary of Use Cases

- **Automatic documentation generation:** Use an LLM to automatically generate or update API documentation based on the source code and comments. This can help reduce the time and effort required for maintaining accurate and up-to-date documentation.
- **Code generation:** An LLM can generate code snippets or entire SDKs (Software Development Kits) for different programming languages, making it easier for developers to integrate with the API.
- **Error detection and resolution:** An LLM can analyze API code to identify potential errors, inconsistencies, or areas for improvement and suggest possible solutions or optimizations. This can lead to more robust and efficient APIs.
- **Natural language interfaces:** Employ an LLM to create natural language interfaces for APIs, allowing developers to interact with the API using plain text queries instead of traditional API calls. These interfaces can improve usability and accessibility for non-technical users.
- **Chatbot development:** You can use an LLM to develop chatbots that provide support and guidance for API users. These chatbots can answer questions, provide troubleshooting assistance, or offer code samples, making it easier for developers to work with the API.
- **API design optimization:** By analyzing usage patterns and developer feedback, an LLM can identify areas where the API design can be

improved, such as simplifying complex endpoints or adding new features to address common use cases.

- **Educational resources:** Utilize an LLM to create tutorials, guides, and other educational resources that help developers learn how to use the API effectively. This can lead to better adoption and user satisfaction.
- **Community engagement:** An LLM can assist you in managing and engaging with the developer community, answering questions on forums, generating content for blog posts, and participating in discussions to support and promote the API.

Every Rose Has Its Thorns: LLM Caveats

LLMs are amazing AI tools, but they make mistakes from time to time, and they don't work well for API edge cases. If you plan on experimenting with an LLM for API code or documentation, you'll need a human touch throughout the process to prompt the AI and verify its output. You'll need a skilled human editor to turn the information the LLM generates into compelling content for your API documentation. You'll need engineers to check all AI-generated code for clarity and accuracy. And if you create [APIs as Products](#), you need to focus on who uses your APIs and how. You'll need people to lead those conversations and empathize with customers, translating those interactions into ideas an LLM can execute.

Humans and AI: a Winning Combination

When you have AI models and humans working together, you can ramp up your API program, building better APIs faster while ensuring consumers have a great experience. A human writer or coder with an LLM has access to a power tool that can automate the completion of mundane tasks and accelerate the API design process. You can further boost your API program by

using a collaborative API design and documentation tool like Stoplight. Stoplight lets internal and external API stakeholders work together using a single platform. An LLM can set an API up to run the race, but only humans can get that API over the finish line.



Read More Stoplight Blogs

Subscribe for the latest in API Design

By submitting this you will be receiving our latest updates on post.

Take a Listen to API Intersection

The podcast on the intersection between API design & digital transformation