

Curl

```
curl -X 'POST' \
'http://127.0.0.1:8000/agent' \
-H 'Accept: application/json' \
-H 'Content-type: application/json' \
-d '{
  "prompt": "Calculate 10 * 32"
}'
```

Request URL

```
http://127.0.0.1:8000/agent
```

Server response

Code	Details	Links
200	<p>Response body</p> <pre>{ "response": "320" }</pre> <p>Response headers</p> <pre>content-length: 18 content-type: application/json date: Sun, 08 Feb 2026 06:12:51 GMT server: unicorn</pre>	<a href="#">Copy</a> <a href="#">Download</a>

Responses

Code	Description	Links
200	Successful Response	No links

Media type

application/json

Controls Accept header.

Example Value | Schema

```
"string"
```

POST /agent Agent

Parameters

No parameters

Request body required

application/json

Edit Value | Schema

```
{ "prompt": "Calculate 10 * 32" }
```

Execute Clear

**Response headers**

```
content-length: 2321
content-type: application/json
date: Sun, 08 Feb 2026 06:07:58 GMT
server: unicorn
```

**Responses**

Code	Description
200	Successful Response
	Media type <input checked="" type="button" value="application/json"/> Example Value   Schema <b>"string"</b>
422	Validation Error
	Media type <input checked="" type="button" value="application/json"/> Example Value   Schema <pre>{   "detail": [     {       "loc": [         "string",         0       ],       "msg": "string",       "type": "string",       "input": "string",       "ctx": {}     }   ] }</pre>

Execute
Clear

**Responses**

Curl

```
curl -X 'POST' \
  'http://127.0.0.1:8000/generate' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
    "prompt": "Explain what a GPU is"
}'
```

Request URL

```
http://127.0.0.1:8000/generate
```

Server response

Code	Details
200	Response body <pre>{   "response": "A Graphics Processing Unit (GPU) is a specialized electronic circuit designed to quickly manipulate and alter memory to accelerate the creation of images in a frame buffer intended for output to a display device. GPUs are also capable of general-purpose computing, making them increasingly used for tasks beyond just graphics rendering.\n\nGPUs were originally designed to handle the complex calculations required for 3D graphics and video games. However, as computing power and memory have increased, GPUs have become versatile enough to be used for various applications such as:\n1. Artificial Intelligence (AI) and Machine Learning (ML): GPUs are particularly well-suited for AI and ML tasks due to their massive parallel processing capabilities.\n2. Scientific Simulations: GPUs can accelerate simulations in fields like physics, chemistry, and climate modeling by leveraging their ability to perform many calculations simultaneously.\n3. Cryptocurrency Mining: GPUs are used for cryptocurrency mining due to their ability to process large amounts of data quickly.\n4. Data Analysis and Visualization: GPUs can be used to speed up data analysis and visualization tasks, especially for large datasets.\n5. General-Purpose Computing (GPGPU): This involves using GPUs for tasks that require parallel processing.\n6. Memory Interface: This is responsible for communicating with system memory.\n7. Registers: Small amounts of on-chip memory used to store data temporarily while it's being processed.\n\nGPUs have several advantages over Central Processing Units (CPUs):\n1. Parallel Processing: GPUs can execute many tasks simultaneously, making them ideal for applications that require simultaneous processing.\n2. High Memory Bandwidth: GPUs often have higher memory bandwidth than CPUs, which allows them to handle large amounts of data quickly.\n3. Efficient Power Consumption: Modern GPUs are designed to be power-efficient, making them suitable for use in mobile devices and other low-power applications.\n\nIn summary, a GPU is a specialized electronic circuit that can perform complex calculations and manipulate data efficiently, making it an essential component in various computing applications beyond just graphics."</pre>

Response headers  
content-length: 2321

← → ⌂ 127.0.0.1:8000/docs#/default/agent\_agent\_post

**FastAPI** 0.1.0 OAS 3.1

/openapi.json

**default**

**POST** /generate Generate

**Parameters**

No parameters

**Request body** required

application/json

Edit Value | Schema

```
{ "prompt": "Explain what a GPU is" }
```

Parameters

No parameters

Request body **required**

application/json

```
{
  "prompt": "Explain what a GPU is"
}
```

Execute

LOADING

Responses

Code	Description	Links
200	Successful Response	No links
422	Validation Error	No links

Media type application/json

Example Value | Schema

```
"string"
```

Media type application/json

Example Value | Schema

```
{
  "detail": [
    {
      "loc": [
        "string",
        0
      ],
      "msg": "string",
      "type": "string",
      "input": "string",
      "ctx": {}
    }
  ]
}
```

Schemas

The screenshot shows a Windows Snipping Tool window with a captured screenshot of an API documentation page. The page displays an 'Execute' button which is currently loading. Below the button, there's a table for 'Responses' with two rows: one for a successful 200 status code and one for a validation error 422. The 200 row has a 'Media type' dropdown set to 'application/json'. The 422 row also has a 'Media type' dropdown set to 'application/json'. Both rows have a 'Links' column with 'No links'. The 422 row also includes a 'Validation Error' schema example. At the bottom of the screenshot, there's a message from the Snipping Tool stating 'Screenshot copied to clipboard' and 'Automatically saved to screenshots folder'.

Similar outputs for without ollama thru groq when it renders the inference api support.