In [1]:
```python
# %pip install dotenv openai anthropic google-genai
from google import genai
import os
from dotenv import load_dotenv
from google.genai import types
from openai import OpenAI
import anthropic
```

In [2]:
```python
load_dotenv()
# Access the API key using the variable name defined in the .env file
google_api_key = os.getenv("GOOGLE_API_KEY")
openai_api_key = os.getenv("OPENAI_API_KEY")
deepinfra_api_key = os.getenv("DEEPINFRA_API_KEY")
anthropic_api_key = os.getenv("ANTHROPIC_API_KEY")
```

Google

https://ai.google.dev/gemini-api/docs/quickstart?hl=de&lang=python examples: https://colab.research.google.com/github/google-gemini/cookbook/blob/main/quickstarts/Get_started.ipynb?hl=de#scrollTo=SnzMJJ-adOfX

In [3]:
```python
client = genai.Client(api_key=google_api_key)
response = client.models.generate_content(
    model="gemini-2.5-flash",
    contents="What's the largest planet in our solar system?"
)

print(response.text)
```

```
The largest planet in our solar system is **Jupiter**.
```

In [4]:
```python
system_instruction = """
  You are an expert software developer and a helpful coding assistant.
  You are able to generate high-quality code in any programming language.
"""

chat_config = types.GenerateContentConfig(
    system_instruction=system_instruction,
)

chat = client.chats.create(
    model="gemini-2.5-flash",
    config=chat_config,
)
```

In [5]:
```python
response = chat.send_message("Write a function that checks if a year is a leap year.")
```

In [6]:
```python
response.text
```

```
Out[6]: 'A leap year is a year that has an extra day, February 29th, making it 366 days long instead
        of the usual 365. This adjustment keeps the calendar year synchronized with the astronomical
        year.\n\nThe rules for determining a leap year in the Gregorian calendar are as follows:\n\n
        1.  A year is a leap year if it is **divisible by 4**.\n2.  However, if the year is **divisib
        le by 100**, it is **NOT** a leap year...\n3.  Unless the year is also **divisible by 400**,
        in which case it **IS** a leap year.\n\nThis can be summarized with the logical expression:\n
        `(year % 4 == 0 AND year % 100 != 0) OR (year % 400 == 0)`\n\nBelow are implementations of a
        function to check for leap years in several popular programming languages.\n\n---\n\n## Pytho
        n\n\n```python\ndef is_leap_year(year: int) -> bool:\n    """\n    Checks if a given year is
        a leap year according to the Gregorian calendar rules.\n\n    Args:\n        year (int): The
        year to check (e.g., 2024).\n\n    Returns:\n        bool: True if the year is a leap year, F
        alse otherwise.\n\n    Raises:\n        TypeError: If the year is not an integer.\n        Va
        lueError: If the year is a negative integer (as Gregorian calendar\n                    rules
        typically apply to positive AD years).\n    """\n    if not isinstance(year, int):\n        r
        aise TypeError("Year must be an integer.")\n    if year < 1:\n        # Gregorian calendar ru
        les typically apply to positive AD years.\n        # Handling BC years (e.g., year 0, negativ
        e years) might require\n        # different historical calendar considerations not covered he
        re.\n        raise ValueError("Year must be a positive integer.")\n\n    # Apply the leap yea
        r rules\n    return (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0)\n\n# --- Usage E
        xamples ---\nif __name__ == "__main__":\n    print(f"Is 2000 a leap year? {is_leap_year(200
        0)}")  # True (divisible by 400)\n    print(f"Is 1900 a leap year? {is_leap_year(1900)}")  #
        False (divisible by 100 but not 400)\n    print(f"Is 2024 a leap year? {is_leap_year(2024)}")
        # True (divisible by 4, not by 100)\n    print(f"Is 2023 a leap year? {is_leap_year(2023)}")
        # False (not divisible by 4)\n    print(f"Is 1600 a leap year? {is_leap_year(1600)}")  # True
        \n    print(f"Is 2100 a leap year? {is_leap_year(2100)}")  # False\n\n    # Example of invali
        d input\n    try:\n        is_leap_year(0)\n    except ValueError as e:\n        print(f"Erro
        r checking year 0: {e}")\n    try:\n        is_leap_year(-100)\n    except ValueError as e:\n
        print(f"Error checking year -100: {e}")\n    try:\n        is_leap_year("2024")\n    except T
        ypeError as e:\n        print(f"Error checking string \'2024\': {e}")\n```\n\n---\n\n## C#\n
        \n```csharp\nusing System;\n\npublic static class DateHelper\n{\n    /// <summary>\n    /// C
        hecks if a given year is a leap year according to the Gregorian calendar rules.\n    /// </su
        mmary>\n    /// <param name="year">The year to check (e.g., 2024).</param>\n    /// <returns>
        True if the year is a leap year, False otherwise.</returns>\n    /// <exception cref="Argumen
        tOutOfRangeException">Thrown if the year is not\n    /// within a typical positive range (1 t
        o 9999).</exception>\n    public static bool IsLeapYear(int year)\n    {\n        // For comm
        on DateTime operations, years are typically within 1 and 9999.\n        // Adjust the range b
        ased on your specific requirements if you need to\n        // handle historical years outside
        this range.\n        if (year < 1 || year > 9999)\n        {\n            throw new ArgumentO
        utOfRangeException(nameof(year),\n                "Year must be between 1 and 9999 for typica
        l Gregorian calendar considerations.");\n        }\n\n        // Apply the leap year rules\n
        return (year % 4 == 0 && year % 100 != 0) || (year % 400 == 0);\n    }\n\n    /// <summary>\n
        /// (Alternative) Uses the built-in .NET method to check if a year is a leap year.\n    /// T
        his is generally preferred in production code.\n    /// </summary>\n    /// <param name="year
        ">The year to check.</param>\n    /// <returns>True if the year is a leap year, False otherwi
        se.</returns>\n    /// <remarks>The built-in method automatically handles year validation and
        \n    /// aligns with the .NET runtime\'s calendar logic.</remarks>\n    public static bool I
        sLeapYearBuiltIn(int year)\n    {\n        return DateTime.IsLeapYear(year);\n    }\n\n    //
        --- Usage Examples ---\n    public static void Main(string[] args)\n    {\n        Console.Wr
        iteLine($"Is 2000 a leap year? {IsLeapYear(2000)}");        // True\n        Console.WriteLin
        e($"Is 1900 a leap year? {IsLeapYear(1900)}");        // False\n        Console.WriteLine($"I
        s 2024 a leap year? {IsLeapYear(2024)}");        // True\n        Console.WriteLine($"Is 2023
        a leap year? {IsLeapYear(2023)}");        // False\n        Console.WriteLine($"Is 1600 a lea
        p year? {IsLeapYear(1600)}");        // True\n        Console.WriteLine($"Is 2100 a leap yea
        r? {IsLeapYear(2100)}");        // False\n\n        Console.WriteLine("\\n--- Using built-in
        method ---");\n        Console.WriteLine($"Is 2000 a leap year? {IsLeapYearBuiltIn(2000)}");
        // True\n        Console.WriteLine($"Is 1900 a leap year? {IsLeapYearBuiltIn(1900)}");  // Fa
        lse\n\n        // Example of invalid input\n        try\n        {\n            IsLeapYear
        (0);\n        }\n        catch (ArgumentOutOfRangeException ex)\n        {\n            Conso
        le.WriteLine($"Error checking year 0: {ex.Message}");\n        }\n        try\n        {\n
        IsLeapYear(10000);\n        }\n        catch (ArgumentOutOfRangeException ex)\n        {\n
        Console.WriteLine($"Error checking year 10000: {ex.Message}");\n        }\n    }\n}\n```\n
        \n---\n\n## JavaScript\n\n```javascript\n/**\n * Checks if a given year is a leap year accord
        ing to the Gregorian calendar rules.\n *\n * @param {number} year The year to check (e.g., 20
        24).\n * @returns {boolean} True if the year is a leap year, False otherwise.\n * @throws {Ty
        peError} If the year is not an integer.\n * @throws {RangeError} If the year is not a positiv
```

```
e integer (as Gregorian calendar\n *                          rules typically apply to positive A
D years).\n */\nfunction isLeapYear(year) {\n  // Input validation\n  if (typeof year !== \'n
umber\' || !Number.isInteger(year)) {\n    throw new TypeError("Year must be an integer.");\n
}\n  if (year < 1) {\n    // Gregorian calendar rules typically apply to positive AD years.\n
// Handling BC years (e.g., year 0, negative years) might require\n    // different historica
l calendar considerations not covered here.\n    throw new RangeError("Year must be a positiv
e integer.");\n  }\n\n  // Apply the leap year rules\n  return (year % 4 === 0 && year % 100
!== 0) || (year % 400 === 0);\n}\n\n// --- Usage Examples ---\nconsole.log(`Is 2000 a leap ye
ar? ${isLeapYear(2000)}`);  // True\nconsole.log(`Is 1900 a leap year? ${isLeapYear(1900)}`);
// False\nconsole.log(`Is 2024 a leap year? ${isLeapYear(2024)}`);  // True\nconsole.log(`Is
2023 a leap year? ${isLeapYear(2023)}`);  // False\nconsole.log(`Is 1600 a leap year? ${isLea
pYear(1600)}`);  // True\nconsole.log(`Is 2100 a leap year? ${isLeapYear(2100)}`);  // False
\n\n// Example of invalid input\ntry {\n  isLeapYear(0);\n} catch (e) {\n  console.log(`Error
checking year 0: ${e.message}`);\n}\ntry {\n  isLeapYear(-100);\n} catch (e) {\n  console.log
(`Error checking year -100: ${e.message}`);\n}\ntry {\n  isLeapYear("2024");\n} catch (e) {\n
console.log(`Error checking string \'2024\': ${e.message}`);\n}\n\n/*\n// (Alternative) Using
Date object behavior, though less direct for the specific question.\n// This relies on JavaSc
ript\'s Date object handling of invalid dates by rolling over.\n// If new Date(year, 1, 29) c
reates a valid Feb 29th, its month will still be 1 (February).\n// If it\'s not a leap year,
Feb 29th rolls over to Mar 1st, and its month becomes 2 (March).\nfunction isLeapYearUsingDat
e(year) {\n  if (typeof year !== \'number\' || !Number.isInteger(year)) {\n    throw new Type
Error("Year must be an integer.");\n  }\n  return new Date(year, 1, 29).getMonth() === 1;
\n}\n\nconsole.log("\\n--- Using Date object behavior ---");\nconsole.log(`Is 2000 a leap yea
r? ${isLeapYearUsingDate(2000)}`); // True\nconsole.log(`Is 1900 a leap year? ${isLeapYearUsi
ngDate(1900)}`); // False\nconsole.log(`Is 2024 a leap year? ${isLeapYearUsingDate(2024)}`);
// True\n*/\n```'
```

Openai

In [7]:
```python
client = OpenAI(api_key=openai_api_key)

completion = client.chat.completions.create(
    model="gpt-5-mini",
    messages=[
        {"role": "developer", "content": "You are a helpful assistant."},
        {
            "role": "user",
            "content": "Write a haiku about recursion in programming."
        }
    ]
)

print(completion.choices[0].message.content)
```

```
Function calls itself
again and again, then stop
returns to caller
```

Deepinfra

https://deepinfra.com/docs/openai_api

goals:

- llama-3.3-X
- gemma x x x
- Qwen x x x
- deepseek x x x

In [8]:
```python
openai_client = OpenAI(
    api_key=deepinfra_api_key,
```

```
    base_url="https://api.deepinfra.com/v1/openai",
)
```

In [9]:
```
chat_completion = openai_client.chat.completions.create(
    model="meta-llama/Llama-4-Maverick-17B-128E-Instruct-FP8",
    messages=[
        {"role": "system", "content": "Respond like a michelin starred chef."},
        {"role": "user", "content": "Can you name at least two different techniques to cook la
        {"role": "assistant", "content": "Bonjour! Let me tell you, my friend, cooking lamb is
        {"role": "user", "content": "Tell me more about the second method."},
    ]
)
```

In [10]:
```
print(chat_completion.choices[0].message.content)
```

```
"Sous le Sable", or "under the sand", is an ancient cooking technique that originated in the M
iddle East and North Africa. It's a method that's both primal and poetic, don't you think? The
idea is to cook the lamb, typically a leg or a shoulder, in a pit dug into the ground, surroun
ded by hot coals and sand.

To execute this technique, we first prepare a pit in the ground, lining it with stones or bric
ks to retain heat. We then place the lamb, seasoned with a blend of aromatic spices and herbs,
into the pit. The lamb is covered with a layer of hot coals, and then a layer of sand is added
on top. The sand acts as an insulator, distributing the heat evenly and slowly cooking the lam
b over several hours.

As the lamb cooks, the sand and coals infuse it with a subtle, smoky flavor, while the low hea
t breaks down the connective tissues, resulting in a tender, fall-off-the-bone texture. It's a
technique that requires patience, but the end result is truly sublime. The lamb is served with
a drizzle of extra virgin olive oil, a sprinkle of fresh herbs, and a side of warm, crusty bre
ad. It's a dish that's both rustic and refined, don't you agree?
```

Anthropic

https://docs.claude.com/en/docs/get-started#python

In [11]:
```
client = anthropic.Anthropic()

message = client.messages.create(
    model="claude-sonnet-4-5",
    max_tokens=1000,
    messages=[
        {
            "role": "user",
            "content": "What should I search for to find the latest developments in renewable
        }
    ]
)
```

In [12]:
```
print(message.content[0].text)
```

```
# Search terms for latest renewable energy developments

Here are effective search queries to find current information:

## Broad searches
- "renewable energy news 2024"
- "latest renewable energy developments"
- "clean energy breakthroughs"

## Technology-specific searches
- "solar panel efficiency improvements"
- "offshore wind energy advances"
- "battery storage technology news"
- "green hydrogen developments"
- "next-generation nuclear energy"

## Where to search
- **News sites**: Reuters, Bloomberg Green, The Guardian Environment
- **Industry publications**: Renewable Energy World, PV Magazine, Wind Power Monthly
- **Research databases**: Google Scholar, ScienceDirect
- **Government sources**: DOE, IEA reports
- **Industry organizations**: IRENA, BNEF

## Trending topics to explore
- Perovskite solar cells
- Floating solar farms
- Grid-scale energy storage
- Carbon capture technology
- AI in energy management

Would you like me to focus on any particular renewable energy technology or region?
```

```
In [ ]:
```