```
!pip install langchain_google_genai langchain_cerebras langchain_sambanova langchain_ai21 langchain_groq langgraph langchain-community langchain-nvidia-ai-end
```

```
    Attempting uninstall: anyio
      Found existing installation: anyio 3.7.1
      Uninstalling anyio-3.7.1:
        Successfully uninstalled anyio-3.7.1
    Attempting uninstall: httpx
      Found existing installation: httpx 0.28.1
      Uninstalling httpx-0.28.1:
        Successfully uninstalled httpx-0.28.1
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the source of the following
jupyter-server 1.24.0 requires anyio<4,>=3.1.0, but you have anyio 4.8.0 which is incompatible.
Successfully installed ai21-3.0.2 ai21-tokenizer-0.12.0 anyio-4.8.0 dataclasses-json-0.6.7 filetype-1.2.0 groq-0.18.0 httpx-0.27.2 httpx-sse-0.4.0 langc
```

```
!pip install together firecrawl-py
```

```
Collecting together
  Using cached together-1.4.1-py3-none-any.whl.metadata (12 kB)
Collecting firecrawl-py
  Downloading firecrawl_py-1.12.0-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: aiohttp<4.0.0,>=3.9.3 in /usr/local/lib/python3.11/dist-packages (from together) (3.11.12)
Requirement already satisfied: click<9.0.0,>=8.1.7 in /usr/local/lib/python3.11/dist-packages (from together) (8.1.8)
Collecting eval-type-backport<0.3.0,>=0.1.3 (from together)
  Downloading eval_type_backport-0.2.2-py3-none-any.whl.metadata (2.2 kB)
Requirement already satisfied: filelock<4.0.0,>=3.13.1 in /usr/local/lib/python3.11/dist-packages (from together) (3.17.0)
Requirement already satisfied: numpy>=1.23.5 in /usr/local/lib/python3.11/dist-packages (from together) (1.26.4)
Requirement already satisfied: pillow<12.0.0,>=11.1.0 in /usr/local/lib/python3.11/dist-packages (from together) (11.1.0)
Requirement already satisfied: pyarrow>=10.0.1 in /usr/local/lib/python3.11/dist-packages (from together) (17.0.0)
Requirement already satisfied: pydantic<3.0.0,>=2.6.3 in /usr/local/lib/python3.11/dist-packages (from together) (2.10.6)
Requirement already satisfied: requests<3.0.0,>=2.31.0 in /usr/local/lib/python3.11/dist-packages (from together) (2.32.3)
Requirement already satisfied: rich<14.0.0,>=13.8.1 in /usr/local/lib/python3.11/dist-packages (from together) (13.9.4)
Requirement already satisfied: tabulate<0.10.0,>=0.9.0 in /usr/local/lib/python3.11/dist-packages (from together) (0.9.0)
Requirement already satisfied: tqdm<5.0.0,>=4.66.2 in /usr/local/lib/python3.11/dist-packages (from together) (4.67.1)
Requirement already satisfied: typer<0.16,>=0.9 in /usr/local/lib/python3.11/dist-packages (from together) (0.15.1)
Requirement already satisfied: python-dotenv in /usr/local/lib/python3.11/dist-packages (from firecrawl-py) (1.0.1)
Requirement already satisfied: websockets in /usr/local/lib/python3.11/dist-packages (from firecrawl-py) (14.2)
Requirement already satisfied: nest-asyncio in /usr/local/lib/python3.11/dist-packages (from firecrawl-py) (1.6.0)
Requirement already satisfied: aiohappyeyeballs>=2.3.0 in /usr/local/lib/python3.11/dist-packages (from aiohttp<4.0.0,>=3.9.3->together) (2.4.6)
Requirement already satisfied: aiosignal>=1.1.2 in /usr/local/lib/python3.11/dist-packages (from aiohttp<4.0.0,>=3.9.3->together) (1.3.2)
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.11/dist-packages (from aiohttp<4.0.0,>=3.9.3->together) (25.1.0)
Requirement already satisfied: frozenlist>=1.1.1 in /usr/local/lib/python3.11/dist-packages (from aiohttp<4.0.0,>=3.9.3->together) (1.5.0)
Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.11/dist-packages (from aiohttp<4.0.0,>=3.9.3->together) (6.1.0)
Requirement already satisfied: propcache>=0.2.0 in /usr/local/lib/python3.11/dist-packages (from aiohttp<4.0.0,>=3.9.3->together) (0.2.1)
Requirement already satisfied: yarl<2.0,>=1.17.0 in /usr/local/lib/python3.11/dist-packages (from aiohttp<4.0.0,>=3.9.3->together) (1.18.3)
Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/python3.11/dist-packages (from pydantic<3.0.0,>=2.6.3->together) (0.7.0)
Requirement already satisfied: pydantic-core==2.27.2 in /usr/local/lib/python3.11/dist-packages (from pydantic<3.0.0,>=2.6.3->together) (2.27.2)
Requirement already satisfied: typing-extensions>=4.12.2 in /usr/local/lib/python3.11/dist-packages (from pydantic<3.0.0,>=2.6.3->together) (4.12.2)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests<3.0.0,>=2.31.0->together) (3.4.1)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-packages (from requests<3.0.0,>=2.31.0->together) (3.10)
```

```
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests<3.0.0,>=2.31.0->together) (2.3.0)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/dist-packages (from requests<3.0.0,>=2.31.0->together) (2025.1.31)
Requirement already satisfied: markdown-it-py>=2.2.0 in /usr/local/lib/python3.11/dist-packages (from rich<14.0.0,>=13.8.1->together) (3.0.0)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/python3.11/dist-packages (from rich<14.0.0,>=13.8.1->together) (2.18.0)
Requirement already satisfied: shellingham>=1.3.0 in /usr/local/lib/python3.11/dist-packages (from typer<0.16,>=0.9->together) (1.5.4)
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.11/dist-packages (from markdown-it-py>=2.2.0->rich<14.0.0,>=13.8.1->together) (0.1.2)
Downloading together-1.4.1-py3-none-any.whl (80 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 80.5/80.5 kB 2.7 MB/s eta 0:00:00
Downloading firecrawl_py-1.12.0-py3-none-any.whl (31 kB)
Downloading eval_type_backport-0.2.2-py3-none-any.whl (5.8 kB)
Installing collected packages: eval-type-backport, firecrawl-py, together
Successfully installed eval-type-backport-0.2.2 firecrawl-py-1.12.0 together-1.4.1
```

```
import os
from google.colab import userdata

os.environ["TAVILY_API_KEY"] = userdata.get("TAVILY_SEARCH_API")
os.environ["GROQ_API_KEY"] = userdata.get("GROQ_API_KEY")
os.environ["GOOGLE_API_KEY"] = userdata.get("GOOGLE_API_KEY")
os.environ["SAMBANOVA_API_KEY"] = userdata.get("SAMBANOVA_API_KEY")
os.environ["NVIDIA_API_KEY"] = userdata.get("NVIDIA_API_KEY")
os.environ["AI21_API_KEY"] = userdata.get("AI21_API_KEY")
os.environ["TOGETHER_API_KEY"] = userdata.get("TOGETHER_API_KEY")
os.environ["KIMI_API_KEY"] = userdata.get("KIMI_API_KEY")
```

## Custom Kimi-k1.5 BaseChatModel

```
import base64
import httpx
from typing import Any, Dict, Iterator, List, Optional, Union
from langchain_core.callbacks import CallbackManagerForLLMRun
from langchain_core.language_models import BaseChatModel
from langchain_core.messages import AIMessage, AIMessageChunk, BaseMessage
from langchain_core.messages.ai import UsageMetadata
from langchain_core.outputs import ChatGeneration, ChatGenerationChunk, ChatResult
from pydantic import Field, SecretStr
from langchain_core.utils.utils import secret_from_env
from openai import OpenAI

class ChatMoonshotAI(BaseChatModel):
    model_name: str = Field(default="kimi-k1.5-preview")
    temperature: Optional[float] = 0.5
```

```
    max_tokens: Optional[int] = 1024
    timeout: Optional[int] = None
    stop: Optional[List[str]] = None
    max_retries: int = 2
    kimi_api_key: Optional[SecretStr] = Field(
        alias="api_key", default_factory=secret_from_env("KIMI_API_KEY", default=None)
    )

    @staticmethod
    def encode_image(image_path: str) -> str:
      if not image_path.startswith("http"):
        with open(image_path, "rb") as image_file:
          return base64.b64encode(image_file.read()).decode('utf-8')
      else:
        response = httpx.get(image_path)
        response.raise_for_status()
        return base64.b64encode(response.content).decode('utf-8')

    def _generate(
        self,
        messages: List[BaseMessage],
        image: Optional[Union[str, bytes]] = None,
        image_is_base64: bool = False,
        stop: Optional[List[str]] = None,
        run_manager: Optional[CallbackManagerForLLMRun] = None,
        **kwargs: Any,
    ) -> ChatResult:
        """Generate a response from Kimi API, with optional image input."""

        client = OpenAI(api_key=self.kimi_api_key.get_secret_value(), base_url="https://api.moonshot.ai/v1")


        formatted_messages = [
            {"role": "system", "content": "You are an  Intelligent Assistant who assists with user queries. You are developed by Moonshot AI , and your name i
            {
                "role": "user",
                "name": "user",
                "content": [{"type": "text", "text": msg.content} for msg in messages],
            }
        ]

        if image:
            image_payload = {
                "type": "image_url",
                "image_url": {
```

```
                    "url": f"{image}" if image_is_base64 else image
                }
            }
            formatted_messages[1]["content"].append(image_payload)

        # API call
        response = client.chat.completions.create(
            model=self.model_name,
            messages=formatted_messages,
            temperature=self.temperature,
            max_tokens=self.max_tokens,
        )

        aimessage = AIMessage(
            content=response.choices[0].message.content,
            usage_metadata=UsageMetadata(
                input_tokens=response.usage.prompt_tokens,
                output_tokens=response.usage.completion_tokens,
                total_tokens=response.usage.total_tokens,
            ),
        )

        return ChatResult(generations=[ChatGeneration(message=aimessage)])

    def _stream(
        self,
        messages: List[BaseMessage],
        image: Optional[Union[str, bytes]] = None,
        image_is_base64: bool = False,
        stop: Optional[List[str]] = None,
        run_manager: Optional[CallbackManagerForLLMRun] = None,
        **kwargs: Any,
    ) -> Iterator[ChatGenerationChunk]:
        """Stream responses from Kimi API with optional image input."""

        client = OpenAI(api_key=self.kimi_api_key.get_secret_value(), base_url="https://api.minimaxi.chat/v1")

        formatted_messages = [
            {"role": "system", "content": "You are kimi-k1.5-model, which is a multimodal model that supports uploading images.You are developed to help user
            {
                "role": "user",
                "name": "user",
                "content": [{"type": "text", "text": msg.content} for msg in messages],
            }
        ]
```

```
            if image:
                image_payload = {
                    "type": "image_url",
                    "image_url": {
                        "url": f"{image}" if image_is_base64 else image
                    }
                }
                formatted_messages[1]["content"].append(image_payload)

            stream = client.chat.completions.create(
                model=self.model_name,
                messages=formatted_messages,
                temperature=self.temperature,
                max_tokens=self.max_tokens,
                stream=True,
            )

            for chunk in stream:
                if chunk.choices[0].delta.content:
                    yield ChatGenerationChunk(
                        message=AIMessageChunk(content=chunk.choices[0].delta.content)
                    )

        @property
        def _llm_type(self) -> str:
            return "Minimax-Text-01"

        @property
        def _identifying_params(self) -> Dict[str, Any]:
            return {"model_name": self.model_name}


    llm = ChatMoonshotAI(
        model_name="kimi-k1.5-preview",
        temperature=0.5,
        max_tokens=1024,
    )
    response  = llm.invoke("Hey, are you developed by moonshot A and what LLM are you?")
    print(response.content)
```

```
◁think▷Alright, let's dive into this problem-solving process. I'm facing a complex issue that requires a multifaceted approach. I start by breaking down

I make some initial assumptions based on my understanding and experience. For instance, I assume that certain variables are constant or that specific cond
```

```
As I work through the problem, I use associations and analogies to draw parallels with other situations I've encountered or knowledge I've acquired. This
Thinking from different perspectives is another method I employ. I try to put myself in the shoes of various stakeholders or consider how someone with a d
Elimination is a strategy I use to rule out options that are clearly not working. By process of elimination, I narrow down the possibilities and focus my
Reverse thinking is also useful; I consider what would happen if I achieved the opposite of what I want. This can help me identify potential obstacles or
I think globally and locally, considering the broader context of the problem as well as the specific details. This helps me ensure that my solution is not
Brain teasers and first-principles thinking are other tools in my arsenal. I might pose a riddle to myself or break the problem down to its most fundament
Throughout this process, I experience emotional fluctuations. There's the excitement of a new idea, the frustration of a dead end, and the satisfaction of
As I implement my ideas, I'm constantly verifying their correctness. This might involve cross-checking my calculations, running simulations, or seeking fe
After a series of attempts, errors, and adjustments, I finally arrive at a solution. It's a result of relentless effort, a combination of diverse explorat
```

```
from IPython.display import display,Markdown

display(Markdown(response.content))
```

◁think▷Alright, let's dive into this problem-solving process. I'm facing a complex issue that requires a multifaceted approach. I start by breaking down the problem into smaller, more manageable parts. This decomposition helps me to focus on individual aspects without getting overwhelmed by the whole.

I make some initial assumptions based on my understanding and experience. For instance, I assume that certain variables are constant or that specific conditions will hold true. I then test these assumptions by gathering data or running small-scale experiments. This is a crucial step because it allows me to refine my approach based on empirical evidence.

As I work through the problem, I use associations and analogies to draw parallels with other situations I've encountered or knowledge I've acquired. This can spark new ideas and help me see the problem from different angles. I also make keen observations, noting any patterns or anomalies that might be relevant.

Thinking from different perspectives is another method I employ. I try to put myself in the shoes of various stakeholders or consider how someone with a different background might approach the problem. This can lead to innovative solutions that I might not have considered otherwise.

Elimination is a strategy I use to rule out options that are clearly not working. By process of elimination, I narrow down the possibilities and focus my efforts on the most promising avenues.

Reverse thinking is also useful; I consider what would happen if I achieved the opposite of what I want. This can help me identify potential obstacles or alternative paths to success.

I think globally and locally, considering the broader context of the problem as well as the specific details. This helps me ensure that my solution is not only effective in the immediate situation but also scalable and applicable in a wider range of scenarios.

Brain teasers and first-principles thinking are other tools in my arsenal. I might pose a riddle to myself or break the problem down to its most fundamental elements to gain a fresh perspective.

Throughout this process, I experience emotional fluctuations. There's the excitement of a new idea, the frustration of a dead end, and the satisfaction of a breakthrough. Each step, whether successful or not, brings me closer to the solution.

As I implement my ideas, I'm constantly verifying their correctness. This might involve cross-checking my calculations, running simulations, or seeking feedback from others. Verification is a critical part of the process, ensuring that my solution is not only logical but also practical and reliable.

After a series of attempts, errors, and adjustments, I finally arrive at a solution. It's a result of relentless effort, a combination of diverse exploration methods, and a careful balance of creativity and rigor. The problem is solved, and I can reflect on the journey, learning from the process and ready to tackle the next challenge.◁/think▷Yes, I was developed by Moonshot AI. I am based on a large language model (LLM) technology, which allows me to understand and generate human-like text based on the input I receive. How can I assist you today?

# Required Dependencies :

```
from langchain_core.tools import tool
from together import Together
from langchain_groq import ChatGroq
from langchain_google_genai import ChatGoogleGenerativeAI
from langchain_sambanova import ChatSambaNovaCloud
from langchain_nvidia_ai_endpoints import ChatNVIDIA
from langchain_ai21 import ChatAI21
from langchain_community.tools import TavilySearchResults
from langchain.prompts import ChatPromptTemplate
from langchain_cerebras import ChatCerebras
from langchain_community.tools import TavilySearchResults
from langgraph.graph import START,END,StateGraph
```

```
from pydantic import BaseModel, HttpUrl
from langchain_community.document_loaders.firecrawl import FireCrawlLoader
from langchain_community.tools import TavilySearchResults
from IPython.display import display, Markdown
from pydantic import BaseModel
import re
from urllib.parse import urlparse, urlunparse
from typing import List, Dict, Optional
import base64
from PIL import Image
from IPython.display import display, Markdown
from PIL import Image
from io import BytesIO
import os
import re
import time
```

## Initializing LLMs

```
Gemini_2 = ChatGoogleGenerativeAI(
    model="gemini-2.0-flash",
    temperature=0.5,
    max_output_tokens=1024,
)
Deepseek_r1_675_B = ChatNVIDIA(
  model="deepseek-ai/deepseek-r1",
  temperature=0.6,
  top_p=0.7,
  max_tokens=4096,
)
Qwen_72B = ChatSambaNovaCloud(
    model = "Qwen2.5-72B-Instruct",
    temperature=0.6,
    max_tokens=4096,
)
Llama31 = ChatGroq(
    model = "llama-3.1-8b-instant",
    temperature=0.6,
    max_tokens=4096,
)
Llama33 = ChatCerebras(
    model = "llama-3.3-70b",
```

```
    temperature=0.6,
    max_tokens=4096,
    api_key = userdata.get("CEREBRAS_API_KEY")
)
```

## Search, retrieve, preprocess , crawl and return

## Iterative Refinement Summarization

```
import operator
from typing import List, Dict, Optional, TypedDict, Literal
from langchain_core.output_parsers import StrOutputParser
from langchain_core.prompts import ChatPromptTemplate
from langchain_core.runnables import RunnableConfig
from langgraph.graph import StateGraph, START, END
from langchain_google_genai import ChatGoogleGenerativeAI

llm = ChatGoogleGenerativeAI(model="gemini-1.5-flash")

summarize_prompt = ChatPromptTemplate(
    [
        ("human", "Write a concise summary of the following article:\n\n{context}"),
    ]
)
initial_summary_chain = summarize_prompt | llm | StrOutputParser()


refine_template = """
Refine the given summary using additional context.

Existing summary:
{existing_answer}

New context:
------------
{context}
------------

Improve the summary with the new information.
"""
```

```
refine_prompt = ChatPromptTemplate([("human", refine_template)])

refine_summary_chain = refine_prompt | llm | StrOutputParser()

class SummarizationState(TypedDict):
    contents: List[str]
    index: int
    summary: str


def generate_initial_summary(state: SummarizationState, config: RunnableConfig):
    """
    Generates the initial summary from the first article in the contents list.
    """
    summary = initial_summary_chain.invoke(
        state["contents"][0], config
    )
    return {"summary": summary, "index": 1}

def refine_summary(state: SummarizationState, config: RunnableConfig):
    """
    Refines the existing summary using new article context iteratively.
    """
    content = state["contents"][state["index"]]
    summary = refine_summary_chain.invoke(
        {"existing_answer": state["summary"], "context": content}, config
    )

    return {"summary": summary, "index": state["index"] + 1}


def should_refine(state: SummarizationState) -> Literal["refine_summary", END]:
    """
    Determines whether there are more articles to refine the summary.
    If all articles are processed, the process stops.
    """
    if state["index"] >= len(state["contents"]):
        return END
    else:
        return "refine_summary"

summarization_graph = StateGraph(SummarizationState)

summarization_graph.add_node("generate_initial_summary", generate_initial_summary)
summarization_graph.add_node("refine_summary", refine_summary)
```

```
summarization_graph.add_edge(START, "generate_initial_summary")
summarization_graph.add_conditional_edges("generate_initial_summary", should_refine)
summarization_graph.add_conditional_edges("refine_summary", should_refine)

summarization_agent = summarization_graph.compile()


summarization_agent
```


```
class GraphState(BaseModel):
    query: str
    search_results: Optional[List[Dict[str, str]]] = None
    classified_articles: Optional[Dict[str, List[str]]] = None
    summaries: Optional[Dict[str, str]] = None
    seo_optimized_content: Optional[Dict[str, str]] = None
    image_paths: Optional[Dict[str, str]] = None

blog_graph = StateGraph(state_schema = GraphState)

from langchain_community.document_loaders.firecrawl import FireCrawlLoader
from langchain_community.tools import TavilySearchResults
from IPython.display import Markdown
```

```
    from IPython.display import Markdown
    import re

    def search_node(state: GraphState) -> GraphState:
        """
        Searches for articles using Tavily and crawls them using FireCrawl.
        Converts the extracted content into Markdown format before passing to the next node.
        """
        tavily_tool = TavilySearchResults(
            max_results=5, search_depth="advanced",
            include_answer=True, include_raw_content=True, include_images=True
        )

        firecrawl_api_key = "YOUR_FIRECRAWL_API_KEY"
        search_results = tavily_tool.run(state.query)

        formatted_results = []
        markdown_articles = {}

        for item in search_results:
            url = item.get("url", "#").strip()
            print(f"🔍 Crawling: {url}")


            loader = FireCrawlLoader(api_key=firecrawl_api_key, url=url, mode="crawl")
            documents = loader.load()

            if not documents or len(documents) == 0:
                continue

            full_content = documents[0].page_content.strip()

            first_sentence = full_content.split(".")[0]
            title = first_sentence[:100] if len(first_sentence) > 5 else "Untitled Article"


            markdown_content = f"### [{title}]({url})\n\n{full_content[:2000]}...\n\n---\n"

            formatted_results.append({"title": title, "url": url, "content": full_content})
            markdown_articles[url] = markdown_content

        state.search_results = formatted_results
        state.summaries = markdown_articles

        if not state.search_results:
            print("⚠️ No valid search results retrieved!")
```

```
        return state


    def classify_node(state: GraphState) -> GraphState:
        model = ChatGoogleGenerativeAI(model="gemini-2.0-flash")
        classified_articles = {}
        for result in state.search_results:
            title = result.get("title", "Untitled")
            classification_prompt = f"Classify the following article title into a sub-topic: '{title}'"
            sub_topic = model.invoke(classification_prompt)
            sub_topic = sub_topic.content
            if sub_topic not in classified_articles:
                classified_articles[sub_topic] = []
            classified_articles[sub_topic].append(title)
        state.classified_articles = classified_articles
        return state

    def summarize_node(state: GraphState) -> GraphState:
        """
        Summarizes all classified articles under each sub-topic using iterative refinement.
        """
        summaries = {}

        for sub_topic, articles in state.classified_articles.items():
            initial_state = SummarizationState(contents=articles, index=0, summary="")
            refined_state = summarization_agent.invoke(initial_state)
            summaries[sub_topic] = refined_state["summary"]

        state.summaries = summaries
        return state

    def seo_optimize_node(state: GraphState) -> GraphState:
        model = ChatGoogleGenerativeAI(model="gemini-pro")
        seo_optimized_content = {}
        for url, summary in state.summaries.items():
            seo_prompt = f"Optimize the following text for SEO:\n\n{summary}"
            optimized_text = model.invoke(seo_prompt)
            optimized_text = optimized_text.content
            seo_optimized_content[url] = optimized_text
        state.seo_optimized_content = seo_optimized_content
        return state

    def image_generation_node(state: GraphState) -> GraphState:
        client = Together()
```

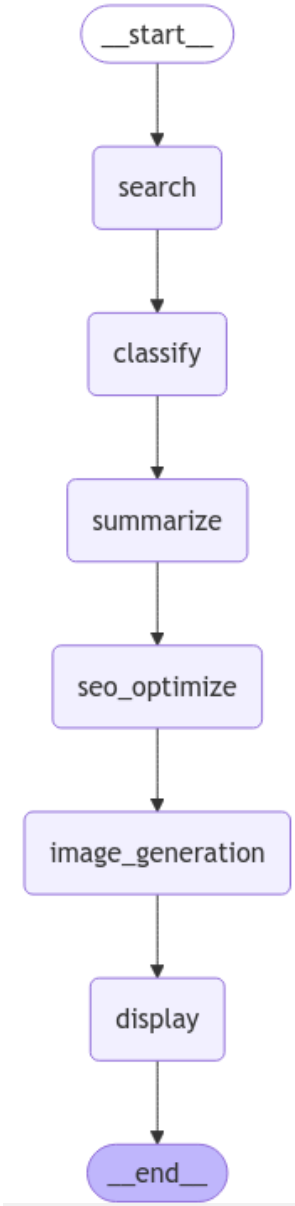
```
if state.image_paths is None:
    state.image_paths = {}

save_dir = "/content/images"
os.makedirs(save_dir, exist_ok=True)

for topic, content in state.seo_optimized_content.items():
    topic_name = Gemini_2.invoke(
        "Generate a **single short topic title** (max 5 words) for the following content. Do NOT return multiple options:\n\n" + content
    ).content.strip()


    safe_topic_name = re.sub(r'[^\w\-_.]', '_', topic_name)[:30]

    image_prompt = Llama33.invoke(f"""
    Create a **realistic, detailed image prompt** for: {topic_name}.
    - Describe a visual scene, avoiding abstract concepts.
    - Example: If the topic is "AI in Finance", describe a **stock trading floor** with AI assistants.

    **Topic:** {topic_name}
    **Image Prompt:**
    """).content.strip()

    response = client.images.generate(
        prompt=image_prompt,
        model="black-forest-labs/FLUX.1-schnell-Free",
        width=1024,
        height=768,
        steps=1,
        n=1,
        response_format="b64_json"
    )

    image_data = base64.b64decode(response.data[0].b64_json)
    image = Image.open(BytesIO(image_data))
    image_path = os.path.join(save_dir, f"{safe_topic_name}.png")

    try:
        image.save(image_path)
        print(f"✅ Image saved: {image_path}")
    except Exception as e:
        print(f"❌ Error saving image: {e}")
    state.image_paths[topic] = image_path
```

```
        return state


    def display_results_node(state: GraphState) -> GraphState:
        """
        Displays the final AI-generated blog content in a well-formatted Markdown format.
        Ensures images appear above their corresponding topic names.
        """

        display(Markdown("## 📌 AI Blog Generation Results\n---"))

        search_results_md = "### 🔎 Search Results\n"
        for idx, result in enumerate(state.search_results, start=1):
            search_results_md += f"- **[{result['title']}]({result['url']})**\n"
        display(Markdown(search_results_md))


        for topic, summary in state.summaries.items():
            if topic in state.image_paths:
                display(Image.open(state.image_paths[topic]))
            display(Markdown(f"### 📖 {topic}\n\n{summary}\n---"))


        for topic, seo_content in state.seo_optimized_content.items():
            if topic in state.image_paths:
                display(Image.open(state.image_paths[topic]))

        display(Markdown("## ✅ AI Blog Generation Completed 🎉"))

        return state


    blog_gen_graph = StateGraph(state_schema = GraphState)


    blog_gen_graph.add_node("search", search_node)
    blog_gen_graph.add_node("classify", classify_node)
    blog_gen_graph.add_node("summarize", summarize_node)
    blog_gen_graph.add_node("seo_optimize", seo_optimize_node)
    blog_gen_graph.add_node("image_generation", image_generation_node)
    blog_gen_graph.add_node("display", display_results_node)

    blog_gen_graph.add_edge(START, "search")
    blog_gen_graph.add_edge("search", "classify")
    blog_gen_graph.add_edge("classify", "summarize")
```

```
blog_gen_graph.add_edge("summarize", "seo_optimize")
blog_gen_graph.add_edge("seo_optimize", "image_generation")
blog_gen_graph.add_edge("image_generation", "display")
blog_gen_graph.add_edge("display", END)
```

```
<langgraph.graph.state.StateGraph at 0x7ba6b45d0790>
```

```
compiled_blog_graph = blog_gen_graph.compile()
compiled_blog_graph
```




```
initial_state = GraphState(query="How are India's upcoming 2024 general elections shaping the political landscape, and what key issues are driving voter senti

final_state = compiled_blog_graph.invoke(initial_state)
final_state
```

```
✅ Image saved: /content/images/Summarize_Article_for_SEO.png
✅ Image saved: /content/images/2024_Indian_Election__Key_Even.png
✅ Image saved: /content/images/India_2024_Election_Seat_Alloc.png
✅ Image saved: /content/images/2023__Pivotal_Political_Events.png
✅ Image saved: /content/images/India_2024_Election_Results_Su.png
```

# 📌 AI Blog Generation Results

---

## 🔎 Search Results

- In the run-up to the 2024 Indian general election, various media houses and polling agencies, carrie
- The 2024 India's general election, marked by pivotal moments and high-stakes campaigning, solidified
- Distribution of seats in Indian general elections 2024, by state
- This article explores the key political events that have defined the year, focusing on the general e
- The results of India's general elections to constitute 18th Lok Sabha, held in April–June 2024 were

## 📂 Classified Topics & Articles

** 🗂️ Based on the incomplete title, here's a possible classification, assuming the title continues to describe polling and surveys:

**Sub-Topic: Indian Elections - Pre-Election Polling & Surveys**

**Explanation:**

- **Indian Elections:** This is the broad topic.
- **Pre-Election:** This specifies that the activity is happening before the election.
- **Polling & Surveys:** This indicates the specific activity being discussed.**
    - In the run-up to the 2024 Indian general election, various media houses and polling agencies, carrie
- * 🗂️ Based on the title, here are a few sub-topic classifications that could work, depending on the specific focus:
- **Indian Politics:** This is the broadest and most obvious category.
- **Elections:** More specific than just politics.
- **2024 Indian General Election Analysis/Review:** Most specific.
- **Political Campaigns:** Focuses on the campaigning aspect mentioned in the title.
- **Electoral Outcomes/Results:** If the article delves into the results and their implications.**
    - The 2024 India's general election, marked by pivotal moments and high-stakes campaigning, solidified
- 🗂️ *Sub-topic:** **Indian General Elections 2024 - Electoral Geography/Constituency Allocation**

Here's why:

- **Indian General Elections 2024:** This is the broad, overarching topic.
- **Distribution of seats... by state:** This focuses on the geographic aspect of the election, specifically how the total number of parliamentary seats are divided among the different states of India. This falls under the study of **electoral geography**, which examines the spatial patterns and processes of elections.
- It also relates to **constituency allocation**, which is the process of determining the number of seats each state gets in the parliament.**
    - Distribution of seats in Indian general elections 2024, by state
- * 🗂️ Based on the title "This article explores the key political events that have defined the year, focusing on the general e...", the most likely sub-topic is:
- **Elections/General Elections:** The title explicitly mentions "general e...", which strongly suggests the focus is on general elections.

While it could potentially touch on broader topics, the primary emphasis and the most fitting sub-topic is elections.**

While it could potentially touch on broader topics, the primary emphasis and the most fitting sub-topic is elections.

- This article explores the key political events that have defined the year, focusing on the general e 🗂️ ****Sub-topic: Indian Politics / Indian Elections / 2024 Indian General Elections**

A more specific sub-topic could be: **2024 Indian General Election Results****

- The results of India's general elections to constitute 18th Lok Sabha, held in April–June 2024 were

📖 **Based on the incomplete title, here's a possible classification, assuming the title continues to describe polling and surveys:**

**Sub-Topic: Indian Elections - Pre-Election Polling & Surveys**

**Explanation:**

- **Indian Elections:** This is the broad topic.
- **Pre-Election:** This specifies that the activity is happening before the election.
- **Polling & Surveys:** This indicates the specific activity being discussed.

## Please provide the rest of the article. I need the complete text to summarize it.

📖 **Based on the title, here are a few sub-topic classifications that could work, depending on the specific focus:**

- **Indian Politics:** This is the broadest and most obvious category.
- **Elections:** More specific than just politics.
- **2024 Indian General Election Analysis/Review:** Most specific.
- **Political Campaigns:** Focuses on the campaigning aspect mentioned in the title.
- **Electoral Outcomes/Results:** If the article delves into the results and their implications.

## The 2024 Indian general election was a high-stakes contest featuring significant events and intense campaigning. (Further detail requires the rest of the article.)

📖 **Sub-topic: Indian General Elections 2024 - Electoral Geography/Constituency Allocation**

Here's why:

- **Indian General Elections 2024:** This is the broad, overarching topic.
- **Distribution of seats... by state:** This focuses on the geographic aspect of the election, specifically how the total number of parliamentary seats are divided among the different states of India. This falls under the study of **electoral geography**, which examines the spatial patterns and processes of elections.
- It also relates to **constituency allocation**, which is the process of determining the number of seats each state gets in the parliament.

## The article details the allocation of parliamentary seats for the 2024 Indian general elections across different states and union territories. It provides a state-wise breakdown of the number of Lok Sabha seats each region will contest.

📖 **Based on the title "This article explores the key political events that have defined the year, focusing on the general e...", the most likely sub-topic is:**

- **Elections/General Elections:** The title explicitly mentions "general e...", which strongly suggests the focus is on general elections.

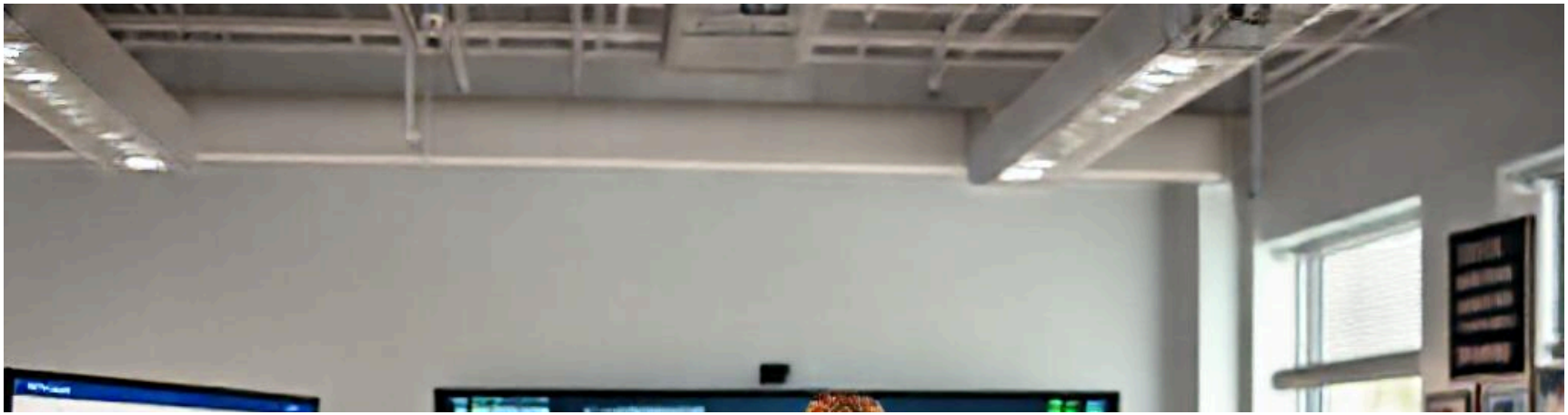
While it could potentially touch on broader topics, the primary emphasis and the most fitting sub-topic is elections.

## The article summarizes the significant political events of the year. (More detail is needed in the original prompt to provide a more specific summary).

📖 **Sub-topic: Indian Politics / Indian Elections / 2024 Indian General Elections**

A more specific sub-topic could be: **2024 Indian General Election Results**

## Please provide the article text. I need the article's content to summarize the results of India's 2024 general elections.



- **Based on the incomplete title, here's a possible classification, assuming the title continues to describe polling and surveys:**

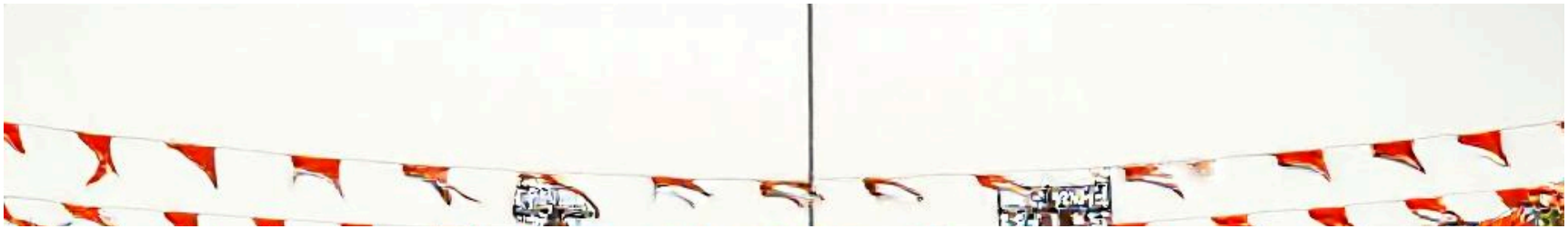
**Sub-Topic: Indian Elections - Pre-Election Polling & Surveys**

**Explanation:**

- **Indian Elections:** This is the broad topic.
- **Pre-Election:** This specifies that the activity is happening before the election.
- **Polling & Surveys:** This indicates the specific activity being discussed.

Optimize the following text for SEO:

## I need the rest of the article to summarize it.



- **Based on the title, here are a few sub-topic classifications that could work, depending on the specific focus:**
  - **Indian Politics:** This is the broadest and most obvious category.
  - **Elections:** More specific than just politics.
  - **2024 Indian General Election Analysis/Review:** Most specific.
  - **Political Campaigns:** Focuses on the campaigning aspect mentioned in the title.
  - **Electoral Outcomes/Results:** If the article delves into the results and their implications.

**Optimized Text for SEO:**

**Headline:** 2024 Indian General Election: A High-Stakes Contest with Unprecedented Events and Intense Campaigns

**Body:**

The 2024 Indian general election was a pivotal electoral event that captivated the nation and garnered immense global attention. This high-stakes contest was marked by:

- **Significant Events:** The election witnessed several major political developments, including the formation of new alliances, the emergence of formidable candidates, and unprecedented voter turnout.
- **Intense Campaigning:** Candidates and political parties engaged in vigorous campaigns that utilized traditional and digital platforms. The electoral landscape was saturated with rallies, debates, and online advertising.

**The outcome of the election had far-reaching implications for India's political and economic future. The results reshaped the nation's political landscape and set the stage for the next phase of development.**

- **Sub-topic: Indian General Elections 2024 - Electoral Geography/Constituency Allocation**

Here's why:

- **Indian General Elections 2024:** This is the broad, overarching topic.
- **Distribution of seats... by state:** This focuses on the geographic aspect of the election, specifically how the total number of parliamentary seats are divided among the different states of India. This falls under the study of **electoral geography**, which examines the spatial patterns and processes of elections.
- It also relates to **constituency allocation**, which is the process of determining the number of seats each state gets in the parliament.

**Optimized Text for SEO:**

**Comprehensive Guide to Parliamentary Seat Allocation for 2024 Indian General Elections**

**Keywords:** Indian General Elections 2024, Parliamentary Seat Allocation, State-wise Breakdown, Lok Sabha Seats

**Introduction:**

This article serves as a comprehensive resource for understanding the allocation of parliamentary seats for the upcoming 2024 Indian general elections. It provides a detailed state-wise breakdown of the number of Lok Sabha seats each region will contest, empowering readers with vital information for election analysis and political strategy.

**State-wise Seat Allocation:**

- **Uttar Pradesh:** Contesting the highest number of seats with 80 Lok Sabha constituencies
- **Maharashtra:** Second-highest with 48 seats, reflecting its significant population
- **Bihar:** Tied with West Bengal for the third-highest with 40 seats
- **West Bengal:** Fourth-highest, also with 40 seats, showcasing its strong political landscape
- **Tamil Nadu:** Holding 39 seats, demonstrating its importance in the national electoral landscape

**Other Key States:**

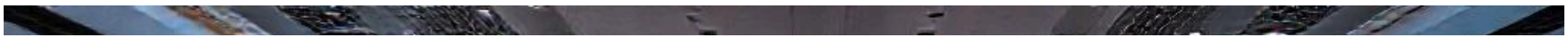
- **Rajasthan:** Contesting 25 seats
- **Karnataka:** 28 seats
- **Madhya Pradesh:** 29 seats
- **Gujarat:** 26 seats
- **Andhra Pradesh:** 25 seats

**Union Territories:**

- **Delhi:** 7 Lok Sabha seats
- **Jammu and Kashmir:** 5 seats, reflecting its strategic importance
- **Chandigarh:** 1 seat

**Conclusion:**

## This article provides a comprehensive overview of the parliamentary seat allocation for the 2024 Indian general elections. By understanding the state-wise breakdown of Lok Sabha seats, readers can gain insights into the electoral landscape and make informed decisions about political engagement and analysis.



🔹 **Based on the title "This article explores the key political events that have defined the year, focusing on the general e...", the most likely sub-topic is:**

- **Elections/General Elections:** The title explicitly mentions "general e...", which strongly suggests the focus is on general elections.

While it could potentially touch on broader topics, the primary emphasis and the most fitting sub-topic is elections.

**Headline:** 2023: A Year in Review of Pivotal Political Events

**Introduction:**

In this comprehensive analysis, we delve into the most impactful political developments that shaped the year 2023. From historic elections to transformative legislation, we explore the events that reshaped the global political landscape.

**Key Events:**

- **Presidential Election in the United States:** The highly anticipated election witnessed a record-breaking voter turnout, with candidates from both major parties vying for the highest office in the land.
- **Brexit Negotiations Concluded:** After years of uncertainty, the United Kingdom and the European Union reached a final agreement on the terms of the UK's departure from the bloc.
- **Climate Change Summit in Paris:** World leaders gathered in France to forge an ambitious global agreement to combat the climate crisis.
- **Rise of Populism in Europe:** Nationalist and anti-establishment movements gained momentum across Europe, challenging traditional political norms.
- **War in Ukraine:** Russia's invasion of Ukraine sparked a major conflict that has had far-reaching geopolitical implications.

**Analysis:**

These events have had a profound impact on the political landscape, from shifting alliances to reshaping domestic policies. We examine the motivations and consequences of these developments, highlighting their significance in shaping the future of politics.

**Conclusion:**

## 2023 has been a year of unprecedented political change. The events we have documented will continue to reverberate in the years to come, influencing the course of history and shaping the way we govern ourselves.

### Sub-topic: Indian Politics / Indian Elections / 2024 Indian General Elections

A more specific sub-topic could be: **2024 Indian General Election Results**

**India's 2024 General Election Results: A Comprehensive Summary**

**Introduction:** India's 2024 general elections, held on April 11-19, 2024, witnessed a historic turnout and intense competition. This article provides a comprehensive summary of the election results, analyzing the key trends and highlighting the major winners and losers.

**Key Trends:**

- **High Voter Turnout:** The election recorded a record-breaking voter turnout of over 70%, indicating the political engagement of the Indian electorate.
- **Nationalist Surge:** The Bharatiya Janata Party (BJP), led by Narendra Modi, strengthened its hold on power, capitalizing on a wave of nationalism and Hindutva ideology.
- **Regional Alliances:** Regional parties played a significant role, forming alliances to challenge the BJP's dominance in key states.
- **Social Media Impact:** Social media platforms played a crucial role in election campaigning and disseminating information, influencing public opinion.

**Major Winners:**

- **Bharatiya Janata Party (BJP):** The BJP emerged as the clear winner, securing a landslide victory with over 300 seats in the Lok Sabha. Narendra Modi retained his position as Prime Minister.
- **Indian National Congress (INC):** The main opposition party, the INC, suffered a major setback, winning only a fraction of its previous seats.
- **Aam Aadmi Party (AAP):** The AAP, led by Arvind Kejriwal, made significant gains in Punjab and Delhi, establishing itself as a major player in Indian politics.

**Major Losers:**

- **All India Trinamool Congress (TMC):** The TMC, led by Mamata Banerjee, lost ground in West Bengal, facing a strong challenge from the BJP.
- **Left Front:** The Left Front, a coalition of communist parties, continued its decline in influence, failing to win any significant number of seats.
- **Bahujan Samaj Party (BSP):** The BSP, once a dominant force in Uttar Pradesh, faced a decline in its vote share and lost several seats.

**Implications and Challenges:**

- The BJP's victory consolidates its position as the pre-eminent political force in India.
- The weakened opposition raises concerns about the balance of power in the Lok Sabha.
- Regional parties remain influential, highlighting the importance of federalism in Indian politics.
- The electoral landscape is expected to continue to evolve, with new parties and alliances emerging.

**Conclusion:**

## India's 2024 general elections produced a significant shift in the political landscape, with the BJP emerging as the dominant party. The high voter turnout and the rise of regional alliances reflect the changing dynamics of Indian politics. The results have implications for the country's future governance, the balance of power, and the role of regional and national parties.

## ✅ AI Blog Generation Completed 🎉

{'query': "How are India's upcoming 2024 general elections shaping the political landscape, and what key issues are driving voter sentiment across different states?",
 'search_results': [{'title': 'In the run-up to the 2024 Indian general election, various media houses and polling agencies, carrie',
   'url': 'https://en.wikipedia.org/wiki/Opinion_polling_for_the_2024_Indian_general_election',
   'content': "In the run-up to the 2024 Indian general election, various media houses and polling agencies, carried out opinion polls to gauge voting intentions. Results of such polls are displayed in this list. Seats by constituency. As this is a FPTP election, seat totals are not determined proportional to each party's total vote share, but instead by the plurality in each constituency"},
  {'title': "The 2024 India's general election, marked by pivotal moments and high-stakes campaigning, solidified",
   'url': 'https://www.oneindia.com/india/india-s-2024-general-elections-key-moments-and-insights-behind-the-outcome-4009531.html',
   'content': "The 2024 India's general election, marked by pivotal moments and high-stakes campaigning, solidified Prime Minister Modi's position, despite facing a united opposition and challenges in voter turnout."},
  {'title': 'Distribution of seats in Indian general elections 2024, by state',
   'url': 'https://www.statista.com/topics/12233/general-election-in-india-2024/',
   'content': 'Distribution of seats in Indian general elections 2024, by state. ... Views on the upcoming general elections Singapore January 2025. ... Voter turnout in general elections India 1951-2024. Voter'},
  {'title': 'This article explores the key political events that have defined the year, focusing on the general e',
   'url': 'https://www.refersms.com/lookback-major-political-changes-in-india-of-2024/',
   'content': "This article explores the key political events that have defined the year, focusing on the general elections, state assembly elections, and other critical developments that will influence India's future. General Elections 2024. One of the most consequential events of 2024 was the Lok Sabha elections, held from April 19 to June 1. This"},
  {'title': "The results of India's general elections to constitute 18th Lok Sabha, held in April–June 2024 were ",
   'url': 'https://en.wikipedia.org/wiki/Results_of_the_2024_Indian_general_election',
   'content': "The results of India's general elections to constitute 18th Lok Sabha, held in April–June 2024 were announced on 4th and 5th June 2024.[1] The main contenders were two alliance groups of the Incumbent National Democratic Alliance (N.D.A) led by Bharatiya Janata Party; and the Opposition Indian National Developmental Inclusive Alliance (I.N.D.I.A.) led by Indian National Congress.[2][3] In the legislative house of 543 seats, the incumbent NDA alliance secured majority with 293 seats, which included BJP party's 240 seats,[4] while the opposition INDIA coalition got 234 seats, including Congress party's 99 seats.[5] On June 9, 2024, Narendra Modi took oath as Prime Minister, having been elected the leader of the NDA alliance, though BJP lost its majority.[6]"}],
 'classified_articles': {"Based on the incomplete title, here's a possible classification, assuming the title continues to describe polling and surveys:\n\n**Sub-Topic:** **Indian Elections - Pre-Election Polling & Surveys**\n\n**Explanation:**\n\n*   **Indian Elections:** This is the broad topic.\n*   **Pre-Election:** This specifies that the activity is happening before the election.\n*   **Polling & Surveys:** This indicates the specific activity being discussed.": ['In the run-up to the 2024 Indian general election, various media houses and polling agencies, carrie'],
  'Based on the title, here are a few sub-topic classifications that could work, depending on the specific focus:\n\n*   **Indian Politics:** This is the broadest and most obvious category.\n*   **Elections:** More specific than just politics.\n*   **2024 Indian General Election Analysis/Review:** Most specific.\n*   **Political Campaigns:** Focuses on the campaigning aspect mentioned in the title.\n*   **Electoral Outcomes/Results:** If the article delves into the results and their implications.': ["The 2024 India's general election, marked by pivotal moments and high-stakes campaigning, solidified"],
  "**Sub-topic:** **Indian General Elections 2024 - Electoral Geography/Constituency Allocation**\n\nHere's why:\n\n*   **Indian General Elections 2024:** This is the broad, overarching topic.\n*   **Distribution of seats... by state:** This focuses on the geographic aspect of the election,

specifically how the total number of parliamentary seats are divided among the different states of India. This falls under the study of **electoral geography**, which examines the spatial patterns and processes of elections.\n*  It also relates to **constituency allocation**, which is the process of determining the number of seats each state gets in the parliament.": ['Distribution of seats in Indian general elections 2024, by state'],
  'Based on the title "This article explores the key political events that have defined the year, focusing on the general e...", the most likely sub-topic is:\n\n*  **Elections/General Elections:** The title explicitly mentions "general e...", which strongly suggests the focus is on general elections.\n\nWhile it could potentially touch on broader topics, the primary emphasis and the most fitting sub-topic is elections.': ['This article explores the key political events that have defined the year, focusing on the general e'],
  '**Sub-topic:** **Indian Politics / Indian Elections / 2024 Indian General Elections**\n\nA more specific sub-topic could be: **2024 Indian General Election Results**': ["The results of India's general elections to constitute 18th Lok Sabha, held in April–June 2024 were "]},
 'summaries': {"Based on the incomplete title, here's a possible classification, assuming the title continues to describe polling and surveys:\n\n**Sub-Topic:** **Indian Elections - Pre-Election Polling & Surveys**\n\n**Explanation:**\n\n*  **Indian Elections:** This is the broad topic.\n*  **Pre-Election:** This specifies that the activity is happening before the election.\n*  **Polling & Surveys:** This indicates the specific activity being discussed.": 'Please provide the rest of the article.  I need the complete text to summarize it.',
  'Based on the title, here are a few sub-topic classifications that could work, depending on the specific focus:\n\n*  **Indian Politics:** This is the broadest and most obvious category.\n*  **Elections:** More specific than just politics.\n*  **2024 Indian General Election Analysis/Review:** Most specific.\n*  **Political Campaigns:** Focuses on the campaigning aspect mentioned in the title.\n*  **Electoral Outcomes/Results:** If the article delves into the results and their implications.': 'The 2024 Indian general election was a high-stakes contest featuring significant events and intense campaigning.  (Further detail requires the rest of the article.)',
  "**Sub-topic:** **Indian General Elections 2024 - Electoral Geography/Constituency Allocation**\n\nHere's why:\n\n*  **Indian General Elections 2024:** This is the broad, overarching topic.\n*  **Distribution of seats... by state:** This focuses on the geographic aspect of the election, specifically how the total number of parliamentary seats are divided among the different states of India. This falls under the study of **electoral geography**, which examines the spatial patterns and processes of elections.\n*  It also relates to **constituency allocation**, which is the process of determining the number of seats each state gets in the parliament.": 'The article details the allocation of parliamentary seats for the 2024 Indian general elections across different states and union territories.  It provides a state-wise breakdown of the number of Lok Sabha seats each region will contest.',
  'Based on the title "This article explores the key political events that have defined the year, focusing on the general e...", the most likely sub-topic is:\n\n*  **Elections/General Elections:** The title explicitly mentions "general e...", which strongly suggests the focus is on general elections.\n\nWhile it could potentially touch on broader topics, the primary emphasis and the most fitting sub-topic is elections.': 'The article summarizes the significant political events of the year.  (More detail is needed in the original prompt to provide a more specific summary).',
  '**Sub-topic:** **Indian Politics / Indian Elections / 2024 Indian General Elections**\n\nA more specific sub-topic could be: **2024 Indian General Election Results**': "Please provide the article text.  I need the article's content to summarize the results of India's 2024 general elections."},
 'seo_optimized_content': {"Based on the incomplete title, here's a possible classification, assuming the title continues to describe polling and surveys:\n\n**Sub-Topic:** **Indian Elections - Pre-Election Polling & Surveys**\n\n**Explanation:**\n\n*  **Indian Elections:** This is the broad topic.\n*  **Pre-Election:** This specifies that the activity is happening before the election.\n*  **Polling & Surveys:** This indicates the specific activity being discussed.": 'Optimize the following text for SEO:\n\nI need the rest of the article to summarize it.',
  'Based on the title, here are a few sub-topic classifications that could work, depending on the specific focus:\n\n*  **Indian Politics:** This is the broadest and most obvious category.\n*  **Elections:** More specific than just politics.\n*  **2024 Indian General Election Analysis/Review:** Most specific.\n*  **Political Campaigns:** Focuses on the campaigning aspect mentioned in the title.\n*  **Electoral Outcomes/Results:** If the article delves into the results and their implications.': "**Optimized Text for SEO:**\n\n**Headline:** 2024 Indian General Election: A High-Stakes Contest with Unprecedented Events and Intense Campaigns\n\n**Body:**\n\nThe 2024 Indian general election was a pivotal electoral event that captivated the nation and garnered immense global attention. This high-stakes contest was marked by:\n\n* **Significant Events:** The election witnessed several major political developments, including the formation of new alliances, the emergence of formidable candidates, and unprecedented voter turnout.\n* **Intense Campaigning:** Candidates and political parties engaged in vigorous campaigns that utilized traditional and digital platforms. The electoral landscape was saturated with rallies, debates, and online advertising.\n\nThe outcome of the election had far-reaching implications for India's political and economic future. The results reshaped the nation's political landscape and set the stage for the next phase of development.",
  "**Sub-topic:** **Indian General Elections 2024 - Electoral Geography/Constituency Allocation**\n\nHere's why:\n\n*  **Indian General Elections 2024:** This is the broad, overarching topic.\n*  **Distribution of seats... by state:** This focuses on the geographic aspect of the election, specifically how the total number of parliamentary seats are divided among the different states of India. This falls under the study of **electoral geography**, which examines the spatial patterns and processes of elections.\n*  It also relates to **constituency allocation**, which is the process of

determining the number of seats each state gets in the parliament.": '**Optimized Text for SEO:**\n\n**Comprehensive Guide to Parliamentary Seat Allocation for 2024 Indian General Elections**\n\n**Keywords:** Indian General Elections 2024, Parliamentary Seat Allocation, State-wise Breakdown, Lok Sabha Seats\n\n**Introduction:**\n\nThis article serves as a comprehensive resource for understanding the allocation of parliamentary seats for the upcoming 2024 Indian general elections. It provides a detailed state-wise breakdown of the number of Lok Sabha seats each region will contest, empowering readers with vital information for election analysis and political strategy.\n\n**State-wise Seat Allocation:**\n\n* **Uttar Pradesh:** Contesting the highest number of seats with 80 Lok Sabha constituencies\n* **Maharashtra:** Second-highest with 48 seats, reflecting its significant population\n* **Bihar:** Tied with West Bengal for the third-highest with 40 seats\n* **West Bengal:** Fourth-highest, also with 40 seats, showcasing its strong political landscape\n* **Tamil Nadu:** Holding 39 seats, demonstrating its importance in the national electoral landscape\n\n**Other Key States:**\n\n* **Rajasthan:** Contesting 25 seats\n* **Karnataka:** 28 seats\n* **Madhya Pradesh:** 29 seats\n* **Gujarat:** 26 seats\n* **Andhra Pradesh:** 25 seats\n\n**Union Territories:**\n\n* **Delhi:** 7 Lok Sabha seats\n* **Jammu and Kashmir:** 5 seats, reflecting its strategic importance\n* **Chandigarh:** 1 seat\n\n**Conclusion:**\n\nThis article provides a comprehensive overview of the parliamentary seat allocation for the 2024 Indian general elections. By understanding the state-wise breakdown of Lok Sabha seats, readers can gain insights into the electoral landscape and make informed decisions about political engagement and analysis.',
  'Based on the title "This article explores the key political events that have defined the year, focusing on the general e...", the most likely sub-topic is:\n\n*   **Elections/General Elections:** The title explicitly mentions "general e...", which strongly suggests the focus is on general elections.\n\nWhile it could potentially touch on broader topics, the primary emphasis and the most fitting sub-topic is elections.': "**Headline:** 2023: A Year in Review of Pivotal Political Events\n\n**Introduction:**\n\nIn this comprehensive analysis, we delve into the most impactful political developments that shaped the year 2023. From historic elections to transformative legislation, we explore the events that reshaped the global political landscape.\n\n**Key Events:**\n\n* **Presidential Election in the United States:** The highly anticipated election witnessed a record-breaking voter turnout, with candidates from both major parties vying for the highest office in the land.\n* **Brexit Negotiations Concluded:** After years of uncertainty, the United Kingdom and the European Union reached a final agreement on the terms of the UK's departure from the bloc.\n* **Climate Change Summit in Paris:** World leaders gathered in France to forge an ambitious global agreement to combat the climate crisis.\n* **Rise of Populism in Europe:** Nationalist and anti-establishment movements gained momentum across Europe, challenging traditional political norms.\n* **War in Ukraine:** Russia's invasion of Ukraine sparked a major conflict that has had far-reaching geopolitical implications.\n\n**Analysis:**\n\nThese events have had a profound impact on the political landscape, from shifting alliances to reshaping domestic policies. We examine the motivations and consequences of these developments, highlighting their significance in shaping the future of politics.\n\n**Conclusion:**\n\n2023 has been a year of unprecedented political change. The events we have documented will continue to reverberate in the years to come, influencing the course of history and shaping the way we govern ourselves.",
  '**Sub-topic:** **Indian Politics / Indian Elections / 2024 Indian General Elections**\n\nA more specific sub-topic could be: **2024 Indian General Election Results**': "**India's 2024 General Election Results: A Comprehensive Summary**\n\n**Introduction:**\nIndia's 2024 general elections, held on April 11-19, 2024, witnessed a historic turnout and intense competition. This article provides a comprehensive summary of the election results, analyzing the key trends and highlighting the major winners and losers.\n\n**Key Trends:**\n\n* **High Voter Turnout:** The election recorded a record-breaking voter turnout of over 70%, indicating the political engagement of the Indian electorate.\n* **Nationalist Surge:** The Bharatiya Janata Party (BJP), led by Narendra Modi, strengthened its hold on power, capitalizing on a wave of nationalism and Hindutva ideology.\n* **Regional Alliances:** Regional parties played a significant role, forming alliances to challenge the BJP's dominance in key states.\n* **Social Media Impact:** Social media platforms played a crucial role in election campaigning and disseminating information, influencing public opinion.\n\n**Major Winners:**\n\n* **Bharatiya Janata Party (BJP):** The BJP emerged as the clear winner, securing a landslide victory with over 300 seats in the Lok Sabha. Narendra Modi retained his position as Prime Minister.\n* **Indian National Congress (INC):** The main opposition party, the INC, suffered a major setback, winning only a fraction of its previous seats.\n* **Aam Aadmi Party (AAP):** The AAP, led by Arvind Kejriwal, made significant gains in Punjab and Delhi, establishing itself as a major player in Indian politics.\n\n**Major Losers:**\n\n* **All India Trinamool Congress (TMC):** The TMC, led by Mamata Banerjee, lost ground in West Bengal, facing a strong challenge from the BJP.\n* **Left Front:** The Left Front, a coalition of communist parties, continued its decline in influence, failing to win any significant number of seats.\n* **Bahujan Samaj Party (BSP):** The BSP, once a dominant force in Uttar Pradesh, faced a decline in its vote share and lost several seats.\n\n**Implications and Challenges:**\n\n* The BJP's victory consolidates its position as the pre-eminent political force in India.\n* The weakened opposition raises concerns about the balance of power in the Lok Sabha.\n* Regional parties remain influential, highlighting the importance of federalism in Indian politics.\n* The electoral landscape is expected to continue to evolve, with new parties and alliances emerging.\n\n**Conclusion:**\nIndia's 2024 general elections produced a significant shift in the political landscape, with the BJP emerging as the dominant party. The high voter turnout and the rise of regional alliances reflect the changing dynamics of Indian politics. The results have implications for the country's future governance, the balance of power, and the role of regional and national parties."},
 'image_paths': {"Based on the incomplete title, here's a possible classification, assuming the title continues to describe polling and surveys:\n\n**Sub-Topic:** **Indian Elections - Pre-Election Polling & Surveys**\n\n**Explanation:**\n\n*   **Indian Elections:** This is the broad

topic.\n*   **Pre-Election:** This specifies that the activity is happening before the election.\n*   **Polling & Surveys:** This indicates the specific activity being discussed.": '/content/images/Summarize_Article_for_SEO.png',
  'Based on the title, here are a few sub-topic classifications that could work, depending on the specific focus:\n\n*   **Indian Politics:** This is the broadest and most obvious category.\n*   **Elections:** More specific than just politics.\n*   **2024 Indian General Election Analysis/Review:** Most specific.\n*   **Political Campaigns:** Focuses on the campaigning aspect mentioned in the title.\n*   **Electoral Outcomes/Results:** If the article delves into the results and their implications.': '/content/images/2024_Indian_Election__Key_Even.png',
  "**Sub-topic:** **Indian General Elections 2024 - Electoral Geography/Constituency Allocation**\n\nHere's why:\n\n*   **Indian General Elections 2024:** This is the broad, overarching topic.\n*   **Distribution of seats... by state:** This focuses on the geographic aspect of the election, specifically how the total number of parliamentary seats are divided among the different states of India. This falls under the study of **electoral geography**, which examines the spatial patterns and processes of elections.\n*   It also relates to **constituency allocation**, which is the process of determining the number of seats each state gets in the parliament.": '/content/images/India_2024_Election_Seat_Alloc.png',
  'Based on the title "This article explores the key political events that have defined the year, focusing on the general e...", the most likely sub-topic is:\n\n*   **Elections/General Elections:** The title explicitly mentions "general e...", which strongly suggests the focus is on general elections.\n\nWhile it could potentially touch on broader topics, the primary emphasis and the most fitting sub-topic is elections.':
'/content/images/2023__Pivotal_Political_Events.png',
  '**Sub-topic:** **Indian Politics / Indian Elections / 2024 Indian General Elections**\n\nA more specific sub-topic could be: **2024 Indian General Election Results**': '/content/images/India_2024_Election_Results_Su.png'}}

```
final_state["summaries"]
```

```
{"Based on the incomplete title, here's a possible classification, assuming the title continues to describe polling and surveys:\n\n**Sub-Topic:**
**Indian Elections - Pre-Election Polling & Surveys**\n\n**Explanation:**\n\n*   **Indian Elections:** This is the broad topic.\n*   **Pre-Election:**
This specifies that the activity is happening before the election.\n*   **Polling & Surveys:** This indicates the specific activity being discussed.":
'Please provide the rest of the article.  I need the complete text to summarize it.',
 'Based on the title, here are a few sub-topic classifications that could work, depending on the specific focus:\n\n*   **Indian Politics:** This is the
broadest and most obvious category.\n*   **Elections:** More specific than just politics.\n*   **2024 Indian General Election Analysis/Review:** Most
specific.\n*   **Political Campaigns:** Focuses on the campaigning aspect mentioned in the title.\n*   **Electoral Outcomes/Results:** If the article
delves into the results and their implications.': 'The 2024 Indian general election was a high-stakes contest featuring significant events and intense
campaigning.  (Further detail requires the rest of the article.)',
 "**Sub-topic:** **Indian General Elections 2024 - Electoral Geography/Constituency Allocation**\n\nHere's why:\n\n*   **Indian General Elections 2024:**
This is the broad, overarching topic.\n*   **Distribution of seats... by state:** This focuses on the geographic aspect of the election, specifically how
the total number of parliamentary seats are divided among the different states of India. This falls under the study of **electoral geography**, which
examines the spatial patterns and processes of elections.\n*   It also relates to **constituency allocation**, which is the process of determining the
number of seats each state gets in the parliament.": 'The article details the allocation of parliamentary seats for the 2024 Indian general elections
across different states and union territories.  It provides a state-wise breakdown of the number of Lok Sabha seats each region will contest.',
 'Based on the title "This article explores the key political events that have defined the year, focusing on the general e...", the most likely sub-topic
is:\n\n*   **Elections/General Elections:** The title explicitly mentions "general e...", which strongly suggests the focus is on general
elections.\n\nWhile it could potentially touch on broader topics, the primary emphasis and the most fitting sub-topic is elections.': 'The article
summarizes the significant political events of the year.  (More detail is needed in the original prompt to provide a more specific summary).',
 '**Sub-topic:** **Indian Politics / Indian Elections / 2024 Indian General Elections**\n\nA more specific sub-topic could be: **2024 Indian General
Election Results**': "Please provide the article text.  I need the article's content to summarize the results of India's 2024 general elections."}
```

```
!pip install firecrawl-py
```

```
Collecting firecrawl-py
  Downloading firecrawl_py-1.12.0-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: requests in /usr/local/lib/python3.11/dist-packages (from firecrawl-py) (2.32.3)
Requirement already satisfied: python-dotenv in /usr/local/lib/python3.11/dist-packages (from firecrawl-py) (1.0.1)
Requirement already satisfied: websockets in /usr/local/lib/python3.11/dist-packages (from firecrawl-py) (14.2)
Requirement already satisfied: nest-asyncio in /usr/local/lib/python3.11/dist-packages (from firecrawl-py) (1.6.0)
Requirement already satisfied: pydantic>=2.10.3 in /usr/local/lib/python3.11/dist-packages (from firecrawl-py) (2.10.6)
Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/python3.11/dist-packages (from pydantic>=2.10.3->firecrawl-py) (0.7.0)
Requirement already satisfied: pydantic-core==2.27.2 in /usr/local/lib/python3.11/dist-packages (from pydantic>=2.10.3->firecrawl-py) (2.27.2)
Requirement already satisfied: typing-extensions>=4.12.2 in /usr/local/lib/python3.11/dist-packages (from pydantic>=2.10.3->firecrawl-py) (4.12.2)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests->firecrawl-py) (3.4.1)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-packages (from requests->firecrawl-py) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests->firecrawl-py) (2.3.0)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/dist-packages (from requests->firecrawl-py) (2025.1.31)
Downloading firecrawl_py-1.12.0-py3-none-any.whl (31 kB)
Installing collected packages: firecrawl-py
Successfully installed firecrawl-py-1.12.0
```

```
from langchain_community.document_loaders.firecrawl import FireCrawlLoader
from langchain_community.tools import TavilySearchResults
```

```
from IPython.display import display, Markdown
from pydantic import BaseModel
import re
from urllib.parse import urlparse, urlunparse


def clean_url(url: str) -> str:
    """
    Cleans Tavily-fetched URLs by:
    - Removing tracking parameters (e.g., ?utm_source=...)
    - Ensuring only the base article URL remains
    """
    parsed_url = urlparse(url)
    cleaned_url = urlunparse((parsed_url.scheme, parsed_url.netloc, parsed_url.path, '', '', ''))
    return cleaned_url.strip()

def should_exclude_url(url: str) -> bool:
    """
    Checks if a URL should be excluded based on blocked domains or keywords.
    """
    blocked_keywords = [
        "celebrities", "entertainment", "movies", "gossip", "bollywood", "hollywood"
    ]

    blocked_domains = [
        "pinkvilla.com", "bollywoodhungama.com", "timesofindia.indiatimes.com",
        "indiatoday.in", "filmibeat.com", "spotboye.com", "koimoi.com"
    ]

    for domain in blocked_domains:
        if domain in url:
            return True

    for keyword in blocked_keywords:
        if keyword in url.lower():
            return True

    return False


def clean_scraped_content(raw_markdown: str) -> str:
    """
    Cleans FireCrawl's extracted content while keeping useful sections.
    - Removes paywalls, navigation menus, and social media links.
    - Keeps meaningful text intact.
```

```
        """

        raw_markdown = re.sub(r"Create an account to read the full story.*?\n", "", raw_markdown, flags=re.DOTALL)
        raw_markdown = re.sub(r"Member-only story.*?\n", "", raw_markdown, flags=re.DOTALL)
        raw_markdown = re.sub(r"Already have an account\?.*?\n", "", raw_markdown, flags=re.DOTALL)


        raw_markdown = re.sub(r"\[.*?(Sign in|Sign up|Homepage|Write|Continue in app)\]", "", raw_markdown)
        raw_markdown = re.sub(r"\[.*?(Trending|Sports|Fact Check|Web Stories|Opinion)\]", "", raw_markdown)


        raw_markdown = re.sub(r"\[.*?(Facebook|Twitter|Instagram|Linkedin|Youtube)\]", "", raw_markdown)

        raw_markdown = re.sub(r"\[Skip to main content\]", "", raw_markdown)


        raw_markdown = re.sub(r"!\[.*?\]\(.*?\)", "", raw_markdown)
        raw_markdown = re.sub(r"\[\[.*?\]\]", "", raw_markdown)


        paragraphs = raw_markdown.split("\n")
        cleaned_paragraphs = []
        for p in paragraphs:
            if len(p.strip()) > 20:
                cleaned_paragraphs.append(p.strip())

        cleaned_text = "\n\n".join(cleaned_paragraphs)

        return cleaned_text.strip()



    class SearchState(BaseModel):
        query: str
        search_results: list[dict] = []
        summaries: dict[str, str] = {}


    def search_node(state: SearchState) -> GraphState:
        """
        Searches for articles using Tavily and crawls them using FireCrawl.
        Extracts the best possible content and formats it into Markdown.
        """
        tavily_tool = TavilySearchResults(
            max_results=3, search_depth="advanced",
```

```
            include_answer=True, include_raw_content=True, include_images=True
        )

        firecrawl_api_key = userdata.get("FIRECRAWL_API_KEY")

        search_results = tavily_tool.run(state.query)

        formatted_results = []
        markdown_articles = {}

        for item in search_results:
            raw_url = item.get("url", "#").strip()
            cleaned_url = clean_url(raw_url)
            print(f"🔍 Crawling: {cleaned_url}")

            try:
                loader = FireCrawlLoader(api_key=firecrawl_api_key, url=cleaned_url, mode="scrape")
                documents = list(loader.lazy_load())

                if not documents or len(documents) == 0:
                    print(f"⚠️ No content extracted from {cleaned_url}")
                    continue


                raw_markdown = documents[0].page_content.strip()


                clean_content = clean_scraped_content(raw_markdown)


                if len(clean_content) < 100:
                    print(f"⚠️ Skipping short content from {cleaned_url}")
                    continue


                first_sentence = clean_content.split(".")[0].strip()
                title = first_sentence[:100] if len(first_sentence) > 5 else "Untitled Article"


                markdown_content = f"### {title}\n\n{clean_content}...\n\n---\n"

                formatted_results.append({"title": title, "url": cleaned_url, "content": clean_content})
                markdown_articles[cleaned_url] = markdown_content

            except Exception as e:
```

```
                print(f"❌ Error crawling {cleaned_url}: {e}")
                continue

        state.search_results = formatted_results
        state.summaries = markdown_articles

        if not state.search_results:
            print("⚠️ No valid search results retrieved!")

        return state

test_state = GraphState(query="What are the key challenges in monetizing an AI-based product?")

test_state = search_node(test_state)


for url, markdown_content in test_state.summaries.items():
    display(Markdown(markdown_content))
```

```
Crawling: https://www.moesif.com/blog/technical/api-development/The-Challenges-of-AI-API-Monetization/
Crawling: https://www.ibbaka.com/ibbaka-market-blog/generative-ai-monetization-an-interview-with-michael-mansard
Crawling: https://www.voltactivedata.com/blog/2024/06/5-main-challenges-with-monetizing-ai-ml-data/
```

### (https://twitter

(https://twitter.com/intent/tweet?via=MoesifHQ&text=The+Challenges+of+AI+API+Monetization%20https%3A%2F%2Fwww.moesif.com%2Fblog%2Ftechnical%2Fapi-development%2FThe-Challenges-of-AI-API-Monetization%2F "Share on Twitter")(https://www.facebook.com/sharer/sharer.php?u=https%3A%2F%2Fwww.moesif.com%2Fblog%2Ftechnical%2Fapi-development%2FThe-Challenges-of-AI-API-Monetization%2F "Share on Facebook")Share on LinkedIn

> Last month, at Collision Conf 2024, Moesif's CEO, Derric Gilling, discussed the challenges of monetizing APIs. Based on this talk, we've put together some key points to consider when monetizing AI APIs.

Building and consuming Application Programming Interfaces (APIs) has become an essential skill for developers. APIs are the glue that enables seamless integration and interaction between different software systems, including those that leverage AI. As the demand for artificial intelligence (AI) grows, the monetization of AI APIs has emerged as a critical concern for businesses looking to capitalize on their technological investments. Almost all AI functionality that companies build is exposed as an API. This means that effectively monetizing these APIs is critical for revenue generation.

AI APIs present unique challenges compared to traditional APIs. Their complexity, high operational costs, and varied usage patterns mean that organizations and consumers must deal with sophisticated pricing models and complex management strategies. This blog explores the critical challenges associated with AI API monetization and how businesses can move past these potential hurdles to inch closer to sustainable AI API growth and profitability. Let's begin by taking a closer look at AI API monetization and what it is.

Monetize in Minutes with Moesif

14 day free trial. No credit card required.

Try for Free

## Understanding AI API Monetization

At a high level, API monetization involves selling functionality through one or more APIs to generate revenue. In some instances, companies build APIs specifically for this reason, offering APIs as a product. In other cases, companies have built APIs for their own usage or applications and then have decided that other companies may also pay for this functionality. When it comes to AI APIs, both of these scenarios ring true.

I often think about OpenAI, which offers ChatGPT as a standalone UI service for consumers. However, the underlying tech, the GPT model, is also exposed via API so that other organizations can leverage this functionality. Users who consume the OpenAI APIs are charged based on tokens for their API usage. This is AI API monetization in action.

Of course, there are various approaches to monetizing APIs. Depending on your AI API's function, internal costs, and other factors, specific monetization strategies might make more sense than others.

### Different Monetization Models for AI APIs

Each monetization model for AI APIs has its benefits and challenges. Here's a quick overview of the different monetization models and the specific challenges that AI APIs might face when implementing them.

#### Subscription-Based Pricing

- Users pay a recurring fee to access the API.
- Pros: Predictable revenue stream, easier to manage.
- Cons: May not reflect actual usage, potentially overpricing or underpricing for different users.

#### Usage-Based Billing

- Users are charged based on their actual usage of the API.
- Pros: Aligns cost with usage, which is fair for both provider and user.
- Cons: Revenue can be variable and more challenging to predict.

### Key Challenges in AI API Monetization

API monetization can be challenging. It becomes even more complicated once you throw in the factors of metering AI API usage and the nature of AI APIs. Let's look at four key challenges to look out for when monetizing AI APIs:

#### Varying Usage Volumes

AI API usage can vary significantly among different users and applications. Some might use the API sporadically, while others may have high and continuous demand. This variation makes it challenging to predict revenue and to set a pricing strategy that accommodates all users reasonably. Usage-based billing can help address this by ensuring that users pay for what they use, but it requires sophisticated metering and billing infrastructure to manage effectively.

#### High Inference Costs

AI APIs incur high inference costs, especially those involving complex models and large datasets. These costs can be a significant portion of the total operating expenses, affecting the API's overall profitability. Providers must balance these costs with competitive pricing to attract and retain users without eroding profit margins. High inference costs also necessitate careful monitoring and optimization to ensure efficiency.

There is a potential for misuse or abuse of AI APIs, where users might exceed reasonable usage limits or exploit the API in unintended ways. Implementing safeguards such as rate limiting, quotas, and user authentication is essential to mitigate these risks and protect both the API and its legitimate users.

#### Complex Input Variables

AI APIs often have multiple input variables that affect the cost of providing the service. These can include the number of input tokens, output tokens, context size, and other factors. Managing and accurately metering these variables is complex but crucial for fair and effective billing. Providers need robust systems to track these inputs and convert them into understandable and billable units for users.

These challenges underscore the need for a thoughtful approach to AI API monetization. By understanding and addressing these issues, organizations can develop effective strategies that ensure sustainable revenue while delivering value to their customers. The following section will examine the benefits of usage-based billing and how it can be used to overcome some of these challenges.

## Why Usage-Based Billing for AI APIs?

Usage-based billing has emerged as a highly effective model for monetizing AI APIs, addressing several of the challenges associated with this space. Most major AI API providers are already using this model, such as OpenAI charging for usage based on the tokens that users have consumed in their API requests. This model is particularly well-suited for AI APIs.

### Benefits of Usage-Based Billing

Usage-based billing brings many benefits to AI API monetization compared to other methods. Although usage-based billing can be more challenging to implement than subscription-based or flat-rate billing, AI costs per API call can vary drastically. Here's how usage-based billing can help tackle some of the challenges:

#### Supports a Variety of Usage Volumes

AI APIs often serve a diverse range of users with varying needs. Usage-based billing allows for flexibility, accommodating light and heavy users without imposing a one-size-fits-all pricing structure. This model aligns costs directly with the level of service consumed, making it a fairer approach for providers and users.

#### Leverages Prepaid Credits

Prepaid credits allow users to pay for a certain amount of API usage upfront. This approach helps providers manage cash flow and reduce payment risk. By requiring users to purchase credits in advance, businesses can secure revenue before service delivery, which can be particularly useful for managing operational costs associated with high inference.

#### Enforces Quotas and Balance Limits

Usage-based billing can include quotas and balance limits to prevent overuse and manage resources effectively. Quotas help control the maximum usage allowed within a specific period, while balance limits prevent users from exceeding their prepaid credits, thereby avoiding unexpected costs.

#### Metering of Input and Output Tokens

For AI APIs, billing based on input tokens, output tokens, and context size provides a granular and transparent pricing model. This level of detail helps accurately reflect the costs associated with providing the service. It ensures that users are billed for the amount of resources they consume, which can be more fair and precise than flat-rate pricing.

#### Flexibility and Revenue Management

Usage-based billing offers greater flexibility compared to fixed or subscription-based models. Providers can adapt their pricing strategies to match better the evolving needs of their users and the costs of providing the API. It also facilitates easier revenue recognition and management. By charging based on actual usage, providers can more accurately forecast revenue and adjust pricing strategies in response to market changes and usage patterns.

Usage-based billing is a powerful tool for AI API monetization, addressing many challenges associated with varying usage volumes, high inference costs, and complex input variables. By aligning pricing with actual usage, this model provides a fair and transparent approach to billing, supports a range of user needs, and helps manage operational risks effectively. In the next section, we will explore how to align pricing to customer value and further implement strategies to optimize AI API monetization.

## Aligning Pricing to Customer Value

One of the most critical aspects of AI API monetization is ensuring that the pricing model reflects the value delivered to customers. By aligning pricing with customer value, businesses can foster stronger user relationships, improve customer satisfaction, and drive sustainable growth. Here's how to approach aligning pricing to customer value when it comes to AI APIs:

### Aligning Pricing with Transaction Volume

When it comes to APIs, many API consumers want to be charged based on the actual transactions they perform. This could come in various forms, such as pricing per API call or per tokens consumed. However, there can also be a secondary factor, such as discounts, which make the API more cost-efficient for consumers who are doing a large amount of transactions with the API.

#### Transaction-Based Pricing

For APIs that are heavily transaction-oriented, aligning pricing with the volume of transactions can make a lot of sense. This model ensures that customers are charged based on the actual value they receive from the API. This approach can encourage higher usage, as customers only pay for what they use as they scale up, making it an attractive option for both startups and enterprises.

### Revenue/Cost Share Models

Sometimes, the value proposition of an API isn't just about individual transactions but about the broader outcomes it enables. In such cases, revenue or cost-sharing models can be effective.

In a revenue-sharing model, the API provider takes a percentage of the revenue generated from the API usage. This model is particularly effective when the API directly contributes to generating revenue for the user. It aligns the interests of both the provider and the user, ensuring that both parties benefit as usage and revenue grow.

Alternatively, a cost-sharing model involves passing on a portion of the operational costs to the user. This approach can be used when the API usage incurs significant costs, such as high inference or data processing expenses.It ensures that users are aware of the underlying costs and are billed accordingly, which can

help in managing high operational expenses.

### Input/Output Token Billing

With the rise of AI and large language models, tokens have become central to API pricing. This is one of the most popular ways to meter the usage of AI APIs.

#### Granular Billing Based on Usage

AI APIs often involve multiple input and output variables, such as the number of input tokens, output tokens, and context size. Billing based on these granular metrics ensures that users are charged accurately for the resources they consume. This level of detailed billing can be more equitable and precise, providing users with transparency and control over their costs.

### User-Centric Pricing Models

Understanding your users and their diverse needs is crucial for successful API monetization. Sometimes, taking a more customized approach that is more personal to the business using the API makes sense.

#### Personalized Pricing Plans

Developing user-centric pricing models involves creating plans that cater to the specific needs of different user segments. This could include tiered pricing, volume discounts, or custom plans for enterprise users. By offering various pricing options, providers can attract more users and accommodate diverse usage patterns.

#### Resource Usage Alignment

Aligning pricing with actual resource usage ensures that users are billed based on the value they derive from the API. This could involve tracking CPU usage, memory consumption, or other relevant metrics. This approach can help manage high operational costs and ensure that pricing is fair and reflective of the service provided.

## Strategies for Successful AI API Monetization

Successfully monetizing AI APIs requires a robust pricing strategy and effective methods to attract and retain users. Here are some strategies to gear your AI APIs up for sustainable growth and maximize revenue potential.

### 1. Land Lots of Users First

The best way to grow a paying user base is to get as many people in the door as possible right away. There are various ways to do this, and it will depend on the internal cost of your API and your target customer.

#### Attracting Developers and Users

The initial phase of monetization should focus on acquiring a large user base. This can be achieved through various tactics, such as offering free trials, freemium models, or low-cost entry points. Creating a large adoption funnel is crucial. The goal is to get as many users as possible to realize the value of the AI API. This can involve developer-friendly documentation, easy integration processes, and a supportive community. For example, providing free or low-cost access initially helps users understand the API's capabilities and encourages widespread adoption.

#### Creating Value Quickly

Ensure that users quickly see the value of the AI API. This can be achieved by providing clear documentation, easy integration, and excellent customer support. The faster users can integrate and start seeing results, the more likely they will continue using and paying for the API.

### 2. Selling Through Existing Users

Often called "land-and-expand," selling through organizations that already use your API is a great way to increase overall usage and revenue.

#### Expanding Usage Among Existing Users

Once a user base is established, the focus should shift to expanding usage among these users. Identify new use cases and offer additional features to existing users. Upselling higher tiers of service or additional features can significantly increase revenue. Users already familiar with the API are more likely to invest in expanded capabilities.

#### Identifying Enterprise Requirements

It is crucial to meet the specific requirements of enterprise users. This might include enhanced security, compliance, and support features tailored to their needs. Enterprise customers often have larger budgets and more complex needs, making them valuable targets for expanded services and higher-tier plans.

## Sales and Usage-Based Expansion Flywheel

A key component of successful AI API monetization is creating a self-sustaining growth cycle driven by user engagement and increasing usage. The Product-Led Growth (PLG) sales flywheel is an effective model that leverages user behavior to drive expansion and revenue growth. Here's an in-depth look at how this model works and the critical steps involved.

### Explanation of the Sales-Driven Expansion Model

The sales-driven expansion model focuses on increasing user engagement and usage, which drives sales opportunities and growth. By continuously engaging users and understanding their needs, businesses can identify new use cases and expand their services. This model is particularly effective for AI APIs, where usage patterns and customer needs vary widely.

### Steps in the PLG Sales Flywheel

When talking about a PLG sales motion, we refer to users who prefer a self-service approach, exploring and using the API independently. Providing excellent self-service resources, such as comprehensive documentation, FAQs, and user forums, is vital. This approach reduces the need for direct sales interactions while still supporting user growth and engagement. In an optimized environment, this flywheel in action will look like this:

The "aha moment" is when users first experience the core value of the AI API. This moment is crucial as it hooks users and encourages them to explore further. Ensuring that users reach this moment quickly is essential. This can be achieved through easy onboarding processes, clear documentation, and immediate access to key features.

#### 2. Increasing Usage

Once users experience the value, they are likely to increase their usage. This phase encourages users to explore more features and integrate the API deeper into their workflows. Providing additional resources, such as tutorials, case studies, and community support, can help users maximize their usage.

#### 3. Sales Engagement

As usage grows, sales teams can engage with users to identify new use cases, offer higher-tier plans, and address specific needs. This proactive engagement helps in converting high-usage self-service users into paying customers. Sales teams should focus on understanding user needs, providing tailored solutions, and highlighting the value of advanced features and higher-tier plans.

#### 4. Identifying New Use Cases

Continuous engagement with users helps them discover new ways to use the API. By understanding user needs and market trends, businesses can identify new use cases and opportunities for expansion. This ongoing discovery process drives innovation and helps keep the API relevant and valuable to users.

#### 5. Accelerated Usage

As users become more familiar with the API and see its benefits, their usage accelerates. This phase involves supporting users in scaling their usage and handling more complex tasks. Ensuring the API can handle increased demand and providing robust support is critical to maintaining user satisfaction during this phase.

Following the approach in this flywheel, businesses can scale up their API revenue and continue to expand their API

Monetizing AI APIs presents unique challenges, from varying usage volumes and high inference costs to the risk of abuse and complex input variables. However, by

adopting usage-based billing and aligning pricing with customer value, businesses can create fair and effective pricing strategies. Additionally, leveraging the Product-Led Growth sales flywheel can drive sustainable growth by focusing on user engagement and continuous expansion.

Businesses can create a self-sustaining growth cycle by initially attracting a large user base and then expanding usage among these users. The key is to ensure users quickly see the value of the API, support their increasing usage, and continuously engage to identify new opportunities. This approach not only maximizes revenue potential but also ensures that the AI API remains relevant and valuable to users in the long term.

Are you getting started with monetizing your AI APIs? At Moesif, we have worked with a wide array of AI companies and helped them quickly and effectively monetize their APIs. Every aspect we've touched on in this blog can be easily implemented and scaled with Moesif, and additional observability and analytics features can help you optimize your AI API portfolio and customer experience. Want to try it out for yourself? Sign up for Moesif today to monetize your AI APIs in minutes with your favorite platforms like Stripe, Chargebee, Zoura, and more.

Deep API Observability with Moesif

14 day free trial. No credit card required.

Try for Free

Implement Tier-Based Pricing with MoesifMoesif's API monetization features can help you quickly and easily get up and running.Try for FreeNo credit card required

API Analytics, API Monitoring, Best Practices

Head of Developer Relations at Moesif. Previously @Tyk and @Dgraph Labs

- Email

We were unable to load Disqus. If you are a moderator please see our troubleshooting guide.

Start the discussion…

**or sign up with Disqus or pick a name**

### Disqus is a discussion network

- Don't be a jerk or do anything illegal. Everything is easier that way.

Read full terms and conditions

By clicking submit, I authorize Disqus, Inc. and its affiliated companies to:

- Use, sell, and share my information to enable me to use its comment services and for marketing purposes, including cross-context behavioral advertising, as described in our Terms of Service and Privacy Policy
- Supplement the information that I provide with additional information lawfully obtained from other sources, like demographic data from public sources, interests inferred from web page views, or other data relevant to what might interest me, like past purchase or location data
- Contact me or enable others to contact me by email with offers for goods and services (from any category) at the email address provided
- Process any sensitive personal information that I submit in a comment for the purpose of displaying the comment
- Retain my information while I am engaging with marketing messages that I receive and for a reasonable amount of time thereafter. I understand I can opt out at any time through an email that I receive. Companies that we share data with are listed here.

I'd rather post as a guest

- Favorite this discussion

- ## Discussion Favorited!

Favoriting means this is a discussion worth sharing. It gets shared to your followers' Disqus feeds, and gives the creator kudos!

Find More Discussions

Share

- Tweet this discussion
- Share this discussion on Facebook
- Share this discussion via email
- Copy link to discussion
- Best
- Newest
- Oldest

Be the first to comment.

Load more comments

**Related Articles**

API Analytics and Monitoring\

**Best Practices for Monetizing AI Successfully**\

Explore several proven monetization strategies for artificial intelligence, from direct monetization to indirect monetization, and shed light on developing a...\

February 13, 2025](https://www.moesif.com/blog/monitoring/Best-Practices-for-Monetizing-AI-Successfully/) [\

**Achieving API Traceability with OpenTelemetry and Moesif**\

Learn how you can improve your API observability with Moesif's OpenTelemetry Integration.\

February 13, 2025](https://www.moesif.com/blog/technical/api-development/Achieving-API-Traceability-With-OpenTelemetry-And-Moesif/) [\

API Analytics and Monitoring\

**Expert Advice on Integrating APIs with Legacy Systems in 2025**\

With all these practical tips and steps, you can maximize the API potential to establish a collaborative digital platform for your business!\

February 07, 2025](https://www.moesif.com/blog/monitoring/Expert-Advice-on-Integrating-APIs-with-Legacy-Systems-in-2025/) [\

**From Vision to Venture Ep. 04: Kin Lane, API Evangelist**\

Derric Gilling, CEO of Moesif, and Kin Lane, API Evangelist, explore the challenges and opportunities of API monetization and productization in today's enter...\

February 05, 2025](https://www.moesif.com/blog/podcasts/developers/Podcast-From-Vision-To-Venture-Kin-Lane/)

### Monitor REST APIs With Moesif

Learn More

# td.doubleclick.net is blocked

This page has been blocked by an extension

- Try disabling your extensions.

ERR_BLOCKED_BY_CLIENT

This page has been blocked by an extension...

---

### [0](https://www

0

# Generative AI Monetization: An Interview with Michael Mansard

Written By Steven Forth

**Steven Forth** is CEO of Ibbaka. See his Skill Profile on Ibbaka Talio.

Ibbaka is working with a number of companies and thought leaders to understand generative pricing, the approach to pricing needed for generative AI applications. Michael Mansard from Zuora is one of the thought leaders in this rapidly evolving field.

Zuora is one of the central companies of the subscription economy. Indeed, it coined the term and the book Subscribed, by CEO Tien Tzuo, helped define what it means to be a customer-centric business and how culture and organization need to change. The below image, from his book Subscribed, helped many of us rethink our businesses.

> **TL;DR**: The attached document is an interview with Michael Mansard from Zuora, discussing the challenges of monetizing Generative AI (GenAI) and the strategies companies can use to overcome these challenges. Key points include:

- **The Impossible Triangle**: Balancing adoption, costs, and value in GenAI monetization.
- **Characteristics of GenAI Applications**: Actionable insights, adaptability, and process efficiency.
- **Predictors of Value**: Uniqueness of underlying data, specialization, attributability of value creation, time to value, scalability, and regulatory compliance.
- **Trends**: Hyper-segmentation, value-based pricing, and the need for AI in monetization.
- **Examples**: Companies like Box, Intercom, and Microsoft are already experimenting with GenAI monetization strategies.
- **Future**: The role of AI in evaluating RFP responses and the need for stringent value demonstration.

Here are two drawings that have helped many of us rethink our businesses.

We are now in the early stages of an even more profound transformation, generative AI. This will force us to rethink our businesses profoundly. (See Will generative AI require new approaches to pricing?) One of the people leading this question is Michael Mansard from Zuora's Subscribed Institute. He is publishing a series of research on the question Monetizing GenAI: Why most SaaS companies are missing out, and how to fix it. We reached out to him to get further insights and context on this important work.

**Ibbaka:** I thought we'd start just by getting an idea of where you're coming from and what sort of experience you bring to your work at Zuora and the Subscribed Institute that shapes how you think about subscriptions, subscription management, and pricing.

**Michael Mansard:** I'm the Principal Director of Subscription Strategy at Zuora and I also happen to be the EMEA Chair of our Think Tank, called The Subscribed Institute, which is roughly 1,500 members strong. Zuora is a monetization suite that powers dynamic business models, anything from usage-based models, subscription bundles and all things in between.

I was lucky to join Zuora about nine years ago. My background was in finance advisory at Deloitte and pre-sales specifically for the office of the CFO at SAP. I was contacted by a headhunter and from the get-go I was really impressed with Zuora and especially Tien Tzuo, our CEO and founder, and his vision. This was also my very first opportunity to join what was then a super fast-growing startup and I took the leap of faith. What was great is that I had the opportunity to grow within pre-sales and progressively adding more and more emphasis on strategic engagements, complex value articulation, and industry point of view and that means interacting a lot with leadership teams, our clients, and our prospects.

That's why I joined forces with Amy Konray who created The Subscribed Institute. We have made The Subscribed Institute into a team that helps companies navigate this new world of customer-centric business models.

I have worked with around 400 companies, moving them towards new business models. This includes companies like Schneider Electric, NetApp, and Phillips.

Outside of Zuora, I'm super lucky to be a guest lecturer at INSEAD. I co-created and co-run a program called The Subscription Business Bootcamp, which is an elective program for Executive MBAs, but also an executive education program and we did that with Professor Wolfgang Ulaga.

Finally, I invest in and advise start-ups. I would say a good dozen start-ups and scale-ups have in common complex B2B Solutions where value is really important. What they all have in common is they are recurring businesses or consumption-based businesses. I advise them mostly on pricing, packaging, monetization, value selling, and pre-sales. Anything on value. After all of that, I'd say I'm a monetization geek.

**Ibbaka: How would you describe the key skills that you have that you bring to your work?**

**Michael Mansard:** I would say the ability to articulate value is a big one. Customer centricity and customer empathy are key in order to uncover value. Know-how with subscription, recurring, usage-based models because that's what people want to talk to us about. Internal customer support. My customers are the field, the leadership team, the sales team, the pre-sales team, and the delivery teams. I'd say the last point would be intersecting with most people in the company. So cross-functional knowledge and communication is key.

**Ibbaka: You recently published a wonderful research e-book on Monetizing Generative AI. What motivated you to write this at this time?**

**Michael Mansard:** First I observed and I'm sure you did, that there's an abundance of content on Generative AI capabilities, innovation, ethics, and potential impact. There's a real gap when it comes to practical advice on how to monetize this new thing. I'm a monetization geek so it really stood out to me. When you know that monetization is vital to sustaining a business and growing a business, the real hot truth is, according to Kyle Poyar, that only 15% of companies are actively monetizing GenAI, which I find extremely risky. I decided to take a scientific approach back in March 2024, and zoomed in to examine around 70 companies to understand their approach, their success and their challenges. I wanted to answer simple questions like

- How is GenAI priced?
- What is the price premium for GenAI?
- What are the typical metrics, especially for solutions, that eventually need fewer users, if not no users at all.

I included in the research companies what I would call native GenAI companies like HeyGen or Perplexity. I also wanted to add established SaaS companies who are starting to infuse GenAI capabilities, such as Box, Notion, or Intercom. The key criteria for me was 'Can I get enough public information?' That was the key thing because otherwise it's really hard to talk about something.

I also had several conversations in the background with practitioners, investors, and even journalists. There was a lot of interest, with people saying this was the first time they had heard the problem phrased in this way.

The first version of this research was ready around mid-April 2024. I showed it to our colleagues in the marketing team. They really loved it and the idea, but they got slightly overwhelmed because it was more than 50 pages long. We decided to break it down to what you see now, which will be a series of six articles.

The first one sets the stage and frames the impossible triangle or triptych between adoption, costs, and value. The second one is around offer positioning. The third one is on packaging. The fourth one is on pricing metrics. The last one will be on trials. Then there will be a summary of all these findings for the decision maker. What are the new capabilities that need to be put in place to solve this very complex and rapidly evolving opportunity.

My motivation was to address something very critical in the market that seems like a white space. Some experts like you and Kyle are talking about it, but there is a big white space and an opportunity for collective learning. We need deep quantitative research and this collaborative effort will be valuable to practitioners.

**Ibbaka:** Did you use a generative AI to help you do the research?

**Michael Mansard:** No, but that is a good question. I actually tried to do so, but most of the content is so complex and structured in a way that made it difficult to use GenAI. What I did was build a humongous Excel sheet. Then I would slice and dice the data to find trends and create charts.

**Ibbaka:** I want to come back to the triptych you mentioned earlier. Can you say a little bit more about the emerging ways people are managing that tension?

The Subscribed Institute at Zuora, Monetizing GenAI: Why most SaaS companies are missing out, and how to fix it

**Michael Mansard:** There's nothing new about this triangle, but each vertex is very different in the word of GenAI. I want to call it the impossible triangle because you cannot satisfy all sides of the triangle at the same time.

It's a very dynamic relationship, we don't know where it's going to lead and we cannot predict the future. You're going to have to play with this triangle. In GenAI I start with costs, which is weird for pricing because you hate starting with cost. But in GenAI managing costs is because the costs are a lot more complex, especially compared to conventional SaaS.

We've been thinking about incremental costs as relatively low in SaaS, right? Adding more users, and more clients didn't create a linear increase in cost. There were economies of scale. In GenAI, and especially if you're thinking in terms of inference costs, which is, the cost of generating outputs or querying the model, it translates into real additional costs that pile up. Each time you ask a question to a GenAI model you have more costs.

Around 10 months ago it was said that ChatGPT cost $700,000 to run every single day. So you have a cost dynamic which is very different to traditional SaaS where platform costs could be overlooked. This doesn't mean you should price based on cost plus. That is never a good idea. But cost becomes a key input in monetization for GenAI and that's new. GenAI businesses tend to be 10 to 30 percentage points lower in terms of gross profit margins and that is significant.

The second vertice of the triangle is adoption. You want to lower the barriers to adoption. It's the case for any solution, but GenAI is so new and transformative that the ability for customers to explore is vital. This means the free trials and the freemium models that we know.

These trials also help you to explore value. 70% of companies (maybe a bit more) have free trials. Customers can get hooked on the value that GenAI can provide and that can ultimately lead to higher revenues. But you have to balance this with the higher cost of operating a generative AI.

Of the companies analyzed for this research, 45% of GenAI companies today offer a usage-based model. That also helps once you've done the trial to take the leap of faith and tiptoe into buying GenAI that you can then progressively scale. That is going to help on the adoption side. Such a model also aligns nicely with costs. You don't want to price based on cost, but the fact that your pricing is going to at least follow cost can be a good thing, especially If you don't know value. We know that value is the most important thing.

> *If you're unable to clearly communicate, quantify, and demonstrate the value of a feature it's going to be a problem. It's not about showcasing a cool, new tech, it's about being able to directly translate it into business outcomes. Productivity, customer satisfaction, cost savings, you name it. But the thing is, it's so new that many providers are in the process of uncovering value, or the absence of value, alongside their clients.*

If I were to summarize it, you have a more complex cost dynamic **.** You have an adoption that needs to be fostered because it's so new and you need to know value. That's why I call it the impossible triangle. If you push too hard on adoption, then you can have exploding costs. But if you do not enable adoption, you're not going to have enough data or insights to actually assess the value of your innovative use cases. Balancing these three things requires a dynamic approach. We are seeing this exercise unfold before our very eyes, which is rare. It's almost like seeing a scientific experiment unfold in real time.

I think companies should be prepared to iterate on their pricing and packaging based on customer feedback in data. A flexible model and being able to adjust as you discover things is going to be critical and one example on this is GitHub. They introduced a new tier to GitHub Copilot. Microsoft, as I was doing this study, did a free trial for Copilot for security. Salesforce changed one of the price points of Einstein add-on as I was doing the study. It kind of shows these open-heart surgeries with these companies and that was only 3 out of 17. The best way to approach this tradeoff is to stay customer-centric. You need to understand the pain point and you need to make sure that your pricing reflects the value to encourage adoption. It’s super easy to say but a lot harder to do.

**Ibbaka:** Can you say a little bit about the characteristics of generative AI applications that underlie these challenges? You mention the costs, but are there other attributes of GenAI that are driving how we think about monetization and how we manage the tradeoffs between the three parts of the triangle?

**Michael Mansard**: I can answer this in two parts. These are the characteristics of generative AI value creation compared to traditional SaaS. Then I’d like to go into characteristics that are good predictors of future value.

I would say the first one that comes to mind is that generative AI applications are **actionable**. We all experience it, you go into a GenAI application and it provides you with direct actionable insights and outputs.

Linked to this is **adaptability**, that is where GenAI shines in my view. It customizes and adapts what we just said in a user-friendly way, in a way that you did not expect. Here you can tweak the outputs to meet your needs.

The third one is **process efficiency** because you can produce at extreme speed, actionable and adaptable outputs with limited and sometimes no human effort. These three things that help you have more efficient processes and workflows as compared to SaaS. These three things speak to me as a value person.

What I find interesting now is what could be good predictors of value if you are a GenAI company.

One is the **uniqueness of the underlying data**. We all know that the quality of training of an AI model and the quality of the output highly depends on the data. If you have a narrow, very specific, or very deep datasets that provides a unique Insight, that is a complete advantage. We saw that OpenAI recently signed with data providers. IP and high-quality data are important.

The second one is **specialization**. We see a lot of general purpose engines right now, but hyperspecialization on a domain or use cases, which is a clear problem, is a big value predictor as well.

The third one is **attributability of value creation**. If it's really hard for you and the buyer to make sure that there's a direct correlation so that you can say “I do this, therefore, this value has been created” it's going to be really hard to capture a value premium. So how do you manage to have an application for which you can have a direct attribution of value creation?

The fourth predictor, and it has been the case for years, is **time to value**. In a rapidly changing world, you have to be able to deliver value quickly.

Then there is the **ability to scale**. Is this a generic solution able to scale elastically in every direction? Regulatory and local compliance issues are popping up across the globe. So how do you minimize legal risk for your clients?

Lastly, your **position in the value chain**. There are always links in the value chain that are able to capture more value than others. There is usually not much one can do about that, but GenAI may change the ground rules.

I purposely didn't focus on the drawbacks, such as data privacy, inaccuracies and energy consumption. I would say that these can negatively impact or modulate value. But if you manage them well they can become your unique value selling points.

**Ibbaka:** Can you give some concrete examples of companies where these are at play?

**Michael Mansard:** Box is one. I think it is quite interesting how they do it. Box created Box AI, which integrates AI in the Box cloud. They enable companies to unlock value out of a large amount of unstructured data and documents. You can summarize documents, get extra insights, and generate new content because you have got so much content that is yours. You can create new content from your content. When it comes to monetization, each subscriber to the Enterprise Plus plan gets 20 AI queries per month, and on top of that you have a corporate-wide allotment that you can pay to top up.

The second one I find very interesting is Intercom. You mentioned this in your recent webinar with Mark Stiving ( the webinar is available here). They use generative AI to power a customer service called Fin AI. This service handles customer support interactions and Intercom only gets paid if the customer issue is resolved. This improves customer satisfaction and of course, reduces support costs.

The last one I want to talk about is Microsoft. They launched a standalone security copilot, which is offered at four dollars worth of security compute unit hours. I find it very interesting that they have created a new metric and that this metric conveys value.

**Ibbaka:** Where is all this going? Generative AI is moving incredibly quickly, so what are the trends that you're going to be watching over the next two to three years?

**Michael Mansard:** Things have changed that are very different. Right now, I would say companies are struggling to find what is the exact premium you can extract from AI. In the study, I assessed the premium, whether you're an add-on or whether you're a new super tier as in the top tier.

You can see how companies are pricing, if you take it as a ratio of the add-on as compared to the core product it, or if you compare it to the super tier as compared to the previous non-AI tier. For add ons, you see between a 13 to 500 percent premium. On the other hand, you see something like 50 to let's say 150 for the top tier. What this tells me is it's really hard right now for companies to really understand and articulate the value premium.

For years, price metrics in software have been seats or users for applications, regardless if you're a vertical application or a horizontal application. Seats or users have been a widely accepted pricing model. There are exceptions of course and I think there's going to be massive changes in this. You'll see in the study that more than 50% of companies analyzed right now in GenAI do not price based on users. For those who price on users, some of them are already integrating new metrics. I'm not sure if pricing is yet customer-centric, but at least they're experimenting with new pricing models that go beyond users. This is only going to accelerate and user-based pricing is going to disappear in most cases. I'd say the first thing, the iteration that we're seeing right now, is only going to accelerate at a faster pace.

The second thing I would mention is there's going to be an acceleration in the need for real-time consumption or usage data analysis as a basis for value measurements.

That is, in order to support more systematic and scientific value exploration more proof points and more scientific ways to measure value will be needed.

> *A value-based approach, value-based discovery, value selling, and value engineering are important in any software company and it's going to become vital. I think there's going to have to be a massive sales transformation so that every single sales and pre-sales rep is going to have to be a value expert.*

That's my deep belief. Why? Because we're going to go into hyper or micro-segmentation. I think that the dream of a segment of one that I think was famous in the 90s is going to happen and it's going to be a segmentation in both journeys, individual journeys, and individual packaging, which will lead to individualized pricing.

So if I try to use an image, it's almost à la carte pricing meets good/better/best, right? You have a la carte pricing in the background, but you're going to show dynamic good, better, best, because we all know the positive impact in terms of pricing psychology of using good, better, best mechanisms.

So long story short, hyper segmentation in terms of journeys, offers and pricing.

Going forward, in my view you will need two things. The first is AI for monetization. You cannot monetize AI without using AI. The effort to run pricing is not at the scale of a human team. The real-time nature of it also means hyper modularity, and not only for packaging, but also for the provisioning. You need to be able to reconfigure everything on the fly. That means you also need a very robust and agile monetization platform to be able to do this at scale, because this has massive implications in terms of invoicing payments, revenue recognition, and so on.

My second conclusion is more customer centricity. I think it's going to create massive polarization in terms of pricing and packaging. You're not going to be able to get away with just pricing per user. You're going to have to be value experts. That means that it's going to make your job a lot harder as monetization is critical. The buyer job is going to become a lot more complex. They will need to dive deep in order to really understand the implications and the intricacies of the pricing model.

Maybe right now the value-based approach is for the sellers, but in the future you're going to have to be able to do value-based buying. This will be a new skill for the buyer, mirroring the value based selling

the buyer, mirroring the value based selling.

**Ibbaka:** I think this is going to have an enormous impact on CPQ systems because configuration will be based on “How do I configure this solution at this point in time?” The question will be how to create the most value for the buyer while optimizing the price. Over the past few months, Ibbaka has been developing some approaches to analyzing RFP responses using generative AI. I don't know how many RFP responses you've looked at recently, but they tend to be long - up to 100 slides. If you have six responses, that's 600 slides, and you're probably doing this as a team of 2 or 3 people. I don't think there is any group of two or three people that can absorb 600 slides and make sense of them. It's not a human task, but generative AIs are pretty good at this.

So increasingly, generative AIs are going to be used to evaluate RFP responses. Which brings us to the point you were making earlier, which is that in the very near future, your buyer is going to be an AI. So how does it impact how we think about value and pricing when increasingly the buyer is an AI and also many of the users are going to be AIs as well?

**Michael Mansard:** I haven't thought of it to be very transparent with you. It's hard to say. I would be convinced that the human-to-human will remain critical for any critical situation. I don't believe in full AI autopilot mode. That's true for any sector for anything strategic. To your point, the idea is augmenting. How do you do it and how does it impact pricing? I would say it requires stringent value demonstration. Vague claims won't work anymore because you're going to have to have substantiated proof of what you're saying.

That's why I think humans are going to be very important and that's why the ability to document value is going to be key. The profiles of the people in sales teams are going to be a lot more data-driven. Value documentation is going to be critical. The ability to automate your response as well, to be relevant to the buyer AI in front of you, of course, is going to be critical.

**Ibbaka:** That is all fascinating. I think we're going to need to do a follow-up, after the next part of your series drops. Before we end today tell me a little bit more about yourself. Who are you when you're not at work? What are your passions outside of your role at the company?

**Michael Mansard:** That is the first time someone has asked me that in a business interview!

Many things. I wish my days were longer. The main thing is spending time with my family. I have two young daughters, two and four. They have massive energy and curiosity and seeing them discover the world, that’s refreshing and inspiring and it keeps us quite busy. Travelling is a big passion as well, Japan being my favorite destination. I have been there ten times and I especially like going to Tohoku. And then music plays a central role for me for relaxation and renewal. I am a drummer and a pianist. I am a huge fan of Jazz music. You said that pricing is like architecture, and I cannot agree more, but for me pricing is a bit like music. It is part science and part craftsmanship. And pricing needs controlled improvisation. I listened to a lot of music while I was working on this.

**Ibbaka:** What are some of the music you listened to while working on this?

**Michael Mansard:** I have to confess that I would go to my Jazz playlist and go random. It is a massive list with more than 150 different pieces from classic to very modern Jazz, a bit of crazy Jazz. When I need to focus I would listen to what they call LoFi music, maybe you know the LoFi Girl on YouTube. I would do that when I needed my brain to focus on just one task.

**Ibbaka**: That reminds me of Brian Eno’s Music for Airports, music that is interesting as both foreground and background.

**Michael Mansard:** I don’t know this but it sounds very interesting. I will look it up.

And then, cooking and food. I love Belgian strong ales, Trappist beers, and of course sake. Anything with a strong malty or grainy flavor. So to summarize, family, travel, music, and cooking. Nothing unusual but I am fairly niche in each of them. These keep me energized.

**Ibbaka:** Thank you, Michael, we look forward to our next conversation and to seeing how GenAI and GenAI pricing, or even generative pricing, will unfold.

ibbaka can help you design pricing for your AI application

## Read other posts on pricing AI

- How is generative AI being priced?
- AI Pricing: What does Box pricing tell us about AI pricing trends?
- AI Pricing: Will the popularity of RAGs change how we price AI?
- AI Pricing: Operating Costs will play a big role in pricing AI functionality
- AI Pricing: Microsoft will frame AI pricing in 2024
- AI Pricing: 2024 will be a year of AI Monetization

Market InsightsPricingSaaSAIAI PricingGenerative AIGenerative PricingInterview

Steven Forth

Newest FirstOldest FirstNewest FirstMost LikedLeast Liked

**AI pricing metrics showing up in multiple SaaS verticals**](https://www.ibbaka.com/ibbaka-market-blog/ai-pricing-metrics-showing-up-in-multiple-saas-verticals) [Next\

**Some companies are still driving outstanding NRR performance**](https://www.ibbaka.com/ibbaka-market-blog/some-companies-are-still-driving-outstanding-nrr-performance)

## Communicate Value, Price Smarter, and Sell More.

Results? Check Customer Success Stories

Sales

Customer Success

Product & Marketing

Investors

Founders and CEOs

Value Based Sales

Customer Success & Retention

Pricing Optimization

Revenue Growth & Scaling

Operations & Diligence

Net Revenue Retention Diagnostic Service

Value Conversations

Podcasts

Webinars

AI Pricing & Monetization Glossary

Downloadable Pricing Tools & Templates

Reports & Playbooks

**The Value & Pricing Blog**](https://www.ibbaka.com/ibbaka-market-blog)

Three dimensions for generative AI apps and implications for value and pricing

Maxio-Ibbaka Survey on Value-Packaging-Pricing-Billing for AgenticAI...

---

**[Skip to content](https://www**

Skip to content

Home < Blog < The 5 Main Challenges With Monetizing AI and ML Data (and How to Fix Them)

# The 5 Main Challenges With Monetizing AI and ML Data (and How to Fix Them)

Jun 21, 2024

Monetizing AI/ML data has become a hot topic for enterprises, and rightly so: We appear at or near 'peak AI'. It's absolutely everywhere, and we're even seeing companies like Dell put out expensive TV commercials talking about what they can do with things like generative AI.

But a lot of it is real-world stuff about to explode on the real-world stage.

Notably, the word "explode" can have either a negative or a positive connotation, depending on the context.

AI is really the next generation of data analytics — a fancy new (although not really, more on that in a second) way to crunch data, ideally in true real-time fashion.

But companies are struggling to make the most of — ie, monetize — their AI/ML data. In fact, the overall return on AI projects has notably dismal thus far.

The fact is: AI and ML data have incredible business impact potential, but only with the support of true real-time data processing.

In this blog post, we're going to try to cut through some of the hype to explain why and where companies are struggling the most to monetize their AI/ML data, and explore some strategies around how to face these challenges.

**Table Of Contents**


- The AI Hype Cycle
- Five Challenges With Monetizing AI/ML Data
- Why Volt to Monetize Your AI/ML Data


The first thing we need to come to terms with is that "AI" goes through cycles — an important thing to consider for making realistic plans based on how things will end up, as opposed to how things are right now.

In 60's and 70's, "AI" as a concept existed only in cinema and TV, with the show Knight Rider featuring a sentient Pontiac Firebird and the deeply unbalanced "HAL 9000" stealing the show in the movie, "2001: A Space Odyssey."

Why mention this in a tech blog about AI in 2024? Because our perceptions of what AI would be were heavily shaped by the media before any real AI existed. A lot of the frothy enthusiasm we are seeing now comes from the ideas fed to us when we were younger.

Machine learning is a statistical mechanism for tying inputs to outcomes that can be re-generated and updated as circumstances change. While it's had an impact, it's not been as much as the hype would have suggested.

Large language models (LLMs) are outwardly impressive but are ultimately limited by their need for historical training data that accurately represents not just the present, but the future.

So, assuming that we've taken the hype cycle into account and your AI/ML data idea is still a good one, what challenges will you face making money out of it?

## Five Challenges With Monetizing AI/ML Data

Here at Volt, we deal quite often with companies whose applications need to make decisions in single-digit milliseconds because they are mission-critical. But, our prospective AI/ML customers (ie, the ones looking to start capitalizing on their AI/ML data) take longer, and generally make decisions in around 50 milliseconds. This is because AI engines are much, much slower than straightforward data manipulation. An awful lot of the available time will be spent cranking the handle of your AI decision engine, leaving little time for all the other stuff that has to happen.

When your business involves millions of small, fleeting chances to influence an event in the customer's favor, or, in the case of something like credit card fraud prevention, stop an event from happening altogether, time is 'of the essence', and you may need everything to be very fast to make up for how long your AI decisions take.

The first major question you must therefore ask is:

Will this AI/ML-based application be fast enough to be useful?

An inability to respond fast enough to be relevant can be catastrophic for certain types of applications. Some companies only discover their speed (or performance) issues after building elaborate data and ML chains.

So if you're in this boat with your applications, be sure to:

1. Understand the needs of your audience as far as latency.
2. Define hard expectations for response times, and
3. Measuring the performance of prototypes before fully committing to going live.

The last thing you want is your ability to monetize your AI/ML data to be costly because then it's like withdrawing two dollars for every dollar you deposit.

Data platform TCO is a growing issue.

It's one thing to work in cases where a large amount of money is involved, such as a car purchase. But what if you're using microtransactions to nudge people's behavior in your favor or trying to optimize a process by 2-3%, which at a large scale could be worth millions, but at an individual event level almost nothing?

This is probably a good time to consider who owns and runs the ML/LLM in question, and what service-level agreements they offer. The entire LLM space is so volatile right now that significant and unexpected changes to what people are selling are more or less certain, and even a small price increase could cause major problems.

So the second major question is:

Will this project actually make money?

This one is both easy and hard.

The easy part is defining how much services will cost at launch time and defining the effectiveness rates they must hit to be commercially viable. The hard part is what assumptions you should make about how well this holds up in the future.

LLMs are an immature and volatile market, and it's a well-established fact that many vendors are operating at a significant and unsustainable loss. The worst-case scenario is everything working perfectly until your LLM vendor ceases trading or doubles its costs, both of which are possible in the current market.

AI bots have a well-known problem with accuracy, with Air Canada recently losing a lawsuit after its bot gave bad advice.

Whether this is an issue or not depends on your individual scenario, but you should always consider that the users of these systems may have higher expectations for accuracy than you do, and accordingly they — and you — could face real-world consequences when errors occur.

Is your AI/ML-based application accurate enough, and can you weather the storm when it gets things wrong?

This means having your cake and eating it, too, (ie – not compromise on anything), but it could also mean choosing your battles very wisely.

In order to mitigate the risk of inaccurate information, you may need to significantly lower your expectations as to what areas you can use LLMs in, avoiding situations where you could be held liable for 'hallucinations'.

### 4. Explainability

AI explainability is going to be a big issue, especially in the EU where the GDPR says that companies must tell consumers about:

*the existence of automated decision-making, including profiling, referred to in Article 22 (1) and (4) and, at least in those cases, meaningful information about the logic involved, as well as the significance and the envisaged consequences of such processing for the data subject.*

*Source: https://gdpr-info.eu/art-15-gdpr/*

Given that LLM's can't explain why they took certain decisions, and thus can't reveal the 'logic involved', this seems to be a major potential problem for the deployment of LLMs in the EU and other like-minded jurisdictions, as such decisions are indefensible in the eyes of the law.

Thus, the fourth question you should be asking is:

Will our plan survive contact with the GDPR? And if not, what do we do?

### 5. Making it "operational"

Operational challenges are the "Elephant in the room".

Aside from all the issues above, remember that you will need to address a real-world, pragmatic issue:

Can you solve all of the four problems above at the same time in an integrated and reliable platform?

Here at Volt we have multiple customers who came to us after finding that solving all of these things simultaneously was near-impossible without a true real-time

data processing platform that doesn't force compromises.

## Why Volt to Monetize Your AI/ML Data

At a time when so many people have their "Happy Ears" on for AI in general, and with LLMs especially, why ask the hard questions? Because we've seen too many AI projects, no matter how well-intentioned or brilliantly conceived, struggle to meet real-world expectations.

LLMs can't makje you magically profitable any more than the invention of a better can opener will end world hunger. That said, under the right circumstances, and with the right input data, sane expectations, and favorable economics, they can definitely have a positive impact. But projects not backed by sound or proper data platform support are likely to struggle.

Volt has AI and ML data experts ready to help you fully capitalize on your AI and ML data. We work in the trenches, with your engineers, to help you get your AI and ML projects running smoothly and at the lowest possible cost.

Volt Active Data is the only data platform built to enable real-time data processing at scale for mission-critical applications. It's a fast, scalable, accurate, data platform for taking real- time decisions so that you can get the most out of your AI/ML data.

If you can express your logic or decision-making process in Java, then you can do it in Volt, and even if you can't, you can still use Volt to efficiently manage the interaction between a real-world event and your AI engine.

Contact us to learn more about how we enable enterprises to monetize their AI/ML data.

Blog CTA 600×300 June 18 24

Previous

Next

**Featured Resources**

**Why Your Tech Stack Is About to Break (and How to Avoid It)**

Read More

**Achieving High Availability in a Strongly Consistent System**

Read More

David Rolfe is Volt Active Data's senior technologist in EMEA. He has 30 years of database industry experience with a focus on telcos.

- has-base-background-color
- LinkedIn
- ACID
- AI/ML
- APIs
- caches
- Caches; data platform; databases; latency; real-time data processing
- Cloud/Edge
- Consistency
- Cost/TCO
- Fraud Prevention
- high availability
- Hyper-Personalization

- Industrial IoT
- Intelligent Manufacturing
- Latency
- Private 5G networks
- Real-time decisioning
- Real-Time Use Cases
- Scalability
- SQL vs NoSQL
- Streaming/Kafka
- Telco/5G
- Volt Capabilities
- Zero downtime

Privacy Policy...

## Second Approach :

```
import operator
from typing import List, Dict, Optional, TypedDict, Literal
from langchain_core.output_parsers import StrOutputParser
from langchain_core.prompts import ChatPromptTemplate
from langchain_core.runnables import RunnableConfig
from langgraph.graph import StateGraph, START, END
from langchain_google_genai import ChatGoogleGenerativeAI

llm = ChatGoogleGenerativeAI(model="gemini-1.5-flash")

# ✅ Define Prompts
summarize_prompt = ChatPromptTemplate(
    [
        ("human", "Write a concise summary of the following article:\n\n{context}"),
    ]
)
initial_summary_chain = summarize_prompt | llm | StrOutputParser()

refine_template = """
Refine the given summary using additional context.

Existing summary:
{existing_answer}

New context:
------------
{context}
------------

Improve the summary with the new information.
"""
refine_prompt = ChatPromptTemplate([("human", refine_template)])
refine_summary_chain = refine_prompt | llm | StrOutputParser()

# ✅ Define Summarization State
class SummarizationState(TypedDict):
    contents: List[str]
    index: int
    summary: str
```

```
# ✅ Generate Initial Summary
def generate_initial_summary(state: SummarizationState, config: Optional[RunnableConfig] = None):
    """
    Generates the initial summary from the first article in the contents list.
    """
    summary = initial_summary_chain.invoke({"context": state["contents"][0]})
    return {"summary": summary, "index": 1}

# ✅ Refine Summary Iteratively
def refine_summary(state: SummarizationState, config: Optional[RunnableConfig] = None):
    """
    Refines the existing summary using new article context iteratively.
    """
    content = state["contents"][state["index"]]
    summary = refine_summary_chain.invoke({"existing_answer": state["summary"], "context": content})
    return {"summary": summary, "index": state["index"] + 1}

# ✅ Condition to Check if More Articles Exist
def should_refine(state: SummarizationState) -> Literal["refine_summary", END]:
    """
    Determines whether there are more articles to refine the summary.
    If all articles are processed, the process stops.
    """
    return "refine_summary" if state["index"] < len(state["contents"]) else END

# ✅ Define Summarization Graph
summarization_graph = StateGraph(SummarizationState)
summarization_graph.add_node("generate_initial_summary", generate_initial_summary)
summarization_graph.add_node("refine_summary", refine_summary)

summarization_graph.add_edge(START, "generate_initial_summary")
summarization_graph.add_conditional_edges("generate_initial_summary", should_refine)
summarization_graph.add_conditional_edges("refine_summary", should_refine)

# ✅ Compile Graph
summarization_agent = summarization_graph.compile()



from langchain_community.document_loaders.firecrawl import FireCrawlLoader
from langchain.tools import TavilySearchResults
from pydantic import BaseModel
from typing import Optional, List, Dict
import time
```

```
class GraphState(BaseModel):
    query: str
    search_results: Optional[List[Dict[str, str]]] = None
    crawled_content: Optional[Dict[str, str]] = None  # 🔹 New field for extracted Firecrawl content
    classified_articles: Optional[Dict[str, List[str]]] = None
    summaries: Optional[Dict[str, str]] = None
    seo_optimized_content: Optional[Dict[str, str]] = None
    image_paths: Optional[Dict[str, str]] = None

blog_graph = StateGraph(state_schema=GraphState)

# 🔹 Step 1: Search Node - Fetch URLs
def search_node(state: GraphState) -> GraphState:
    """
    Performs a web search using Tavily and extracts valid titles, content, and URLs.
    """
    tavily_tool = TavilySearchResults(
        max_results=5, search_depth="advanced", include_answer=True, include_raw_content=True, include_images=True
    )

    search_results = tavily_tool.run(state.query)

    formatted_results = []

    for item in search_results:
        url = item.get("url", "#").strip()
        content = item.get("content", "").strip()

        if len(content) > 10:
            first_sentence = content.split(".")[0]
            title = first_sentence[:100] if len(first_sentence) > 5 else "Untitled Article"
        else:
            title = "Untitled Article"

        if len(content) < 50:
            continue

        formatted_results.append({"title": title, "url": url, "content": content})

    state.search_results = formatted_results

    if not state.search_results:
        print("⚠️ No valid search results retrieved!")

    return state
```

```
# 🔹 Step 2: Crawl Node - Extract Content with Firecrawl
def crawl_node(state: GraphState) -> GraphState:
    """
    Uses Firecrawl to extract structured content from the search results' URLs.
    """
    firecrawl_api_key = "YOUR_FIRECRAWL_API_KEY"
    crawled_content = {}

    for result in state.search_results:
        url = result["url"]
        try:
            print(f"🕷️ Crawling: {url}")
            loader = FireCrawlLoader(api_key=userdata.get("FIRECRAWL_API_KEY"), url=url, mode="crawl")
            docs = loader.load()  # Fetch structured content
            crawled_content[url] = docs[0].page_content if docs else "No content extracted"
            time.sleep(2)  # Prevent rate limiting
        except Exception as e:
            print(f"❌ Failed to crawl {url}: {e}")

    state.crawled_content = crawled_content
    return state

# 🔹 Step 3: Classify Articles Node
def classify_node(state: GraphState) -> GraphState:
    model = ChatGoogleGenerativeAI(model="gemini-2.0-flash")
    classified_articles = {}

    for url, content in state.crawled_content.items():
        classification_prompt = f"Classify the following article into a sub-topic:\n\n{content[:500]}"
        sub_topic = model.invoke(classification_prompt).content

        if sub_topic not in classified_articles:
            classified_articles[sub_topic] = []
        classified_articles[sub_topic].append(url)

    state.classified_articles = classified_articles
    return state

# 🔹 Step 4: Summarization Node
def summarize_node(state: GraphState) -> GraphState:
    summaries = {}

    for sub_topic, urls in state.classified_articles.items():
        articles_content = "\n\n".join([state.crawled_content[url] for url in urls if url in state.crawled_content])
```

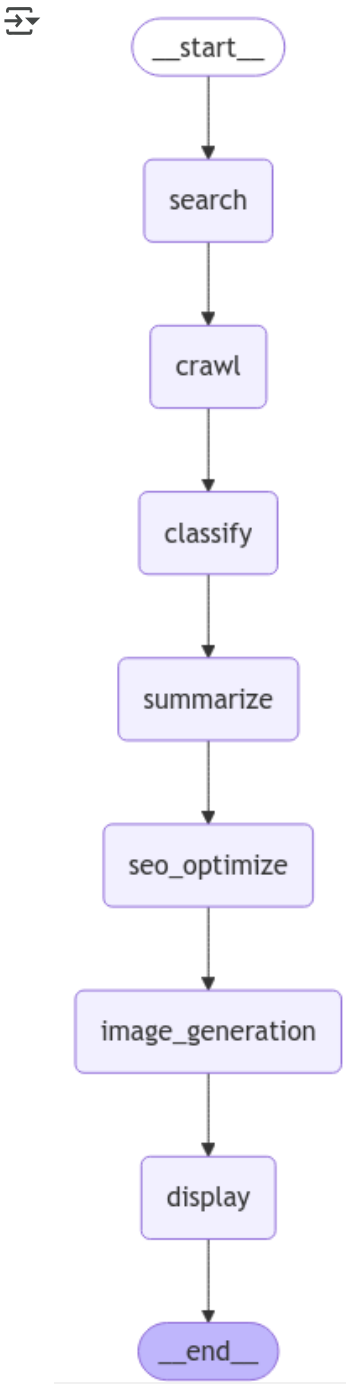
```
            if not articles_content:
                continue

            initial_state = SummarizationState(contents=[articles_content], index=0, summary="")
            refined_state = summarization_agent.invoke(initial_state)
            summaries[sub_topic] = refined_state["summary"]

        state.summaries = summaries
        return state

    # 🔹 Step 5: SEO Optimization Node
    def seo_optimize_node(state: GraphState) -> GraphState:
        model = ChatGoogleGenerativeAI(model="gemini-pro")
        seo_optimized_content = {}

        for sub_topic, summary in state.summaries.items():
            seo_prompt = f"Optimize the following text for SEO:\n\n{summary}"
            optimized_text = model.invoke(seo_prompt).content
            seo_optimized_content[sub_topic] = optimized_text

        state.seo_optimized_content = seo_optimized_content
        return state

    # 🔹 Step 6: Image Generation Node
    def image_generation_node(state: GraphState) -> GraphState:
        client = Together()

        if state.image_paths is None:
            state.image_paths = {}

        save_dir = "/content/images"
        os.makedirs(save_dir, exist_ok=True)

        for topic, content in state.seo_optimized_content.items():
            topic_name = Gemini_2.invoke(
                f"Generate a **single short topic title** (max 5 words) for the following content:\n\n{content}"
            ).content.strip()

            safe_topic_name = re.sub(r'[^\w\-_.]', '_', topic_name)[:30]

            image_prompt = Qwen_72B.invoke(f"""
            Create a **realistic, detailed image prompt** for: {topic_name}.
            - Describe a visual scene, avoiding abstract concepts.
            - Example: If the topic is "AI in Finance", describe a **stock trading floor** with AI assistants.
```

```
            **Topic:** {topic_name}
            **Image Prompt:**
            """).content.strip()

            response = client.images.generate(
                prompt=image_prompt,
                model="black-forest-labs/FLUX.1-schnell-Free",
                width=1024,
                height=768,
                steps=1,
                n=1,
                response_format="b64_json"
            )

            image_data = base64.b64decode(response.data[0].b64_json)
            image = Image.open(BytesIO(image_data))
            image_path = os.path.join(save_dir, f"{safe_topic_name}.png")

            try:
                image.save(image_path)
                print(f"✅ Image saved: {image_path}")
            except Exception as e:
                print(f"❌ Error saving image: {e}")

            state.image_paths[topic] = image_path

        return state

    # 🔹 Step 7: Display Results Node
    def display_results_node(state: GraphState) -> GraphState:
        display(Markdown("## 📌 AI Blog Generation Results\n---"))

        search_results_md = "### 🔎 Search Results\n"
        for idx, result in enumerate(state.search_results, start=1):
            search_results_md += f"- **[{result['title']}]({result['url']})**\n"
        display(Markdown(search_results_md))

        for topic, summary in state.summaries.items():
            if topic in state.image_paths:
                display(Image.open(state.image_paths[topic]))
            display(Markdown(f"### 📖 {topic}\n\n{summary}\n---"))

        for topic, seo_content in state.seo_optimized_content.items():
            if topic in state.image_paths:
                display(Image.open(state.image_paths[topic]))
```

```
    display(Markdown("## ✅ AI Blog Generation Completed 🎉"))
    return state

# 🔹 Graph Workflow Definition
blog_gen_graph = StateGraph(state_schema=GraphState)
blog_gen_graph.add_node("search", search_node)
blog_gen_graph.add_node("crawl", crawl_node)
blog_gen_graph.add_node("classify", classify_node)
blog_gen_graph.add_node("summarize", summarize_node)
blog_gen_graph.add_node("seo_optimize", seo_optimize_node)
blog_gen_graph.add_node("image_generation", image_generation_node)
blog_gen_graph.add_node("display", display_results_node)

blog_gen_graph.add_edge(START, "search")
blog_gen_graph.add_edge("search", "crawl")
blog_gen_graph.add_edge("crawl", "classify")
blog_gen_graph.add_edge("classify", "summarize")
blog_gen_graph.add_edge("summarize", "seo_optimize")
blog_gen_graph.add_edge("seo_optimize", "image_generation")
blog_gen_graph.add_edge("image_generation", "display")
blog_gen_graph.add_edge("display", END)

compiled_graph = blog_gen_graph.compile()
compiled_graph
```


__start__
search
crawl
classify
summarize
seo_optimize
image_generation
display
__end__

```
compiled_graph = blog_gen_graph.compile()
compiled_graph
```


__start__
search
crawl
classify
summarize
seo_optimize
image_generation
display
__end__

```
initial_state = GraphState(query="What are the key challenges in monetizing an AI-based product?")
final_state = compiled_graph.invoke(initial_state)
final_state
```

```
🕷️ Crawling: https://www.moesif.com/blog/technical/api-development/The-Challenges-of-AI-API-Monetization/
🕷️ Crawling: https://dailyai.com/2023/10/can-the-tech-industry-overcome-the-challenge-of-ai-monetization/
🕷️ Crawling: https://www.simon-kucher.com/en/insights/value-monetization-age-ai
❌ Failed to crawl https://www.simon-kucher.com/en/insights/value-monetization-age-ai: Unexpected error during start crawl job: Status code 429. Rate lim
🕷️ Crawling: https://www.withorb.com/blog/monetizing-ai-a-discussion-of-strategic-challenges-ahead
❌ Failed to crawl https://www.withorb.com/blog/monetizing-ai-a-discussion-of-strategic-challenges-ahead: Unexpected error during start crawl job: Status
🕷️ Crawling: https://www.chargebee.com/blog/adapting-saas-to-ai-monetization/
❌ Failed to crawl https://www.chargebee.com/blog/adapting-saas-to-ai-monetization/: Unexpected error during start crawl job: Status code 429. Rate limit
```

```
from langchain_community.document_loaders.firecrawl import FireCrawlLoader
from langchain.tools import TavilySearchResults
from pydantic import BaseModel
from typing import Optional, List, Dict
import time

class GraphState(BaseModel):
    query: str
    search_results: Optional[List[Dict[str, str]]] = None
    crawled_content: Optional[Dict[str, str]] = None  # 🔹 New field for extracted Firecrawl content
    classified_articles: Optional[Dict[str, List[str]]] = None
    summaries: Optional[Dict[str, str]] = None
    seo_optimized_content: Optional[Dict[str, str]] = None
    image_paths: Optional[Dict[str, str]] = None

blog_graph = StateGraph(state_schema=GraphState)

# 🔹 Step 1: Search Node - Fetch URLs
def search_node(state: GraphState) -> GraphState:
    """
    Performs a web search using Tavily and extracts valid titles, content, and URLs.
    """
    tavily_tool = TavilySearchResults(
        max_results=5, search_depth="advanced", include_answer=True, include_raw_content=True, include_images=True
    )

    search_results = tavily_tool.run(state.query)

    formatted_results = []
```

```
        for item in search_results:
            url = item.get("url", "#").strip()
            content = item.get("content", "").strip()

            if len(content) > 10:
                first_sentence = content.split(".")[0]
                title = first_sentence[:100] if len(first_sentence) > 5 else "Untitled Article"
            else:
                title = "Untitled Article"

            if len(content) < 50:
                continue

            formatted_results.append({"title": title, "url": url, "content": content})

        state.search_results = formatted_results

        if not state.search_results:
            print("⚠️ No valid search results retrieved!")

        return state

    # 🔹 Step 2: Crawl Node - Extract Content with Firecrawl
    def crawl_node(state: GraphState) -> GraphState:
        """
        Uses Firecrawl to extract structured content from the search results' URLs.
        """
        crawled_content = {}

        for result in state.search_results:
            url = result["url"]
            try:
                print(f"🕷️ Crawling: {url}")
                loader = FireCrawlLoader(api_key=userdata.get("FIRECRAWL_API_KEY"), url=url, mode="crawl")
                docs = loader.load()  # Fetch structured content
                crawled_content[url] = docs[0].page_content if docs else "No content extracted"
                time.sleep(5)  # Prevent rate limiting
            except Exception as e:
                print(f"❌ Failed to crawl {url}: {e}")

        state.crawled_content = crawled_content
        return state

    # 🔹 Step 3: Classify Articles Node
    def classify_node(state: GraphState) -> GraphState:
```

```
        model = ChatGoogleGenerativeAI(model="gemini-2.0-flash")
        classified_articles = {}

        for url, content in state.crawled_content.items():
            classification_prompt = f"Classify the following article into a sub-topic:\n\n{content[:500]}"
            sub_topic = model.invoke(classification_prompt).content

            if sub_topic not in classified_articles:
                classified_articles[sub_topic] = []
            classified_articles[sub_topic].append(url)

        state.classified_articles = classified_articles
        return state

    #  🔹  Step 4: Summarization Node
    def summarize_node(state: GraphState) -> GraphState:
        summaries = {}

        # Define a structured, multi-step ChatPromptTemplate
        summarization_prompt = ChatPromptTemplate.from_messages([
            ("system", "You are an expert at analyzing and summarizing long-form content."),
            ("human", f"""Analyze the following content: {{article_content}}

            **Required Analysis:**
            1️⃣ Extract **key insights** and major findings.
            2️⃣ Identify **contradictions and opposing views** (if any).
            3️⃣ Highlight **statistics, trends, or research findings**.
            4️⃣ Summarize in a **neutral, unbiased manner**.

            📌 **Output Format**:
            - Key Insights:
            - Contradictions:
            - Statistics & Trends:
            - Final Summary:
            """),
        ])

        for sub_topic, urls in state.classified_articles.items():
            articles_content = "\n\n".join([state.crawled_content[url] for url in urls if url in state.crawled_content])
            if not articles_content:
                continue

            prompt = summarization_prompt.format(article_content=articles_content)
            refined_summary = Llama33.invoke(prompt).content
            summaries[sub_topic] = refined_summary
```

```
    state.summaries = summaries
    return state



# 🔹 Step 5: SEO Optimization Node
def seo_optimize_node(state: GraphState) -> GraphState:
    model = ChatGoogleGenerativeAI(model="gemini-pro")
    seo_optimized_content = {}

    for sub_topic, summary in state.summaries.items():
        seo_prompt = f"Optimize the following text for SEO:\n\n{summary}"
        optimized_text = model.invoke(seo_prompt).content
        seo_optimized_content[sub_topic] = optimized_text

    state.seo_optimized_content = seo_optimized_content
    return state

# 🔹 Step 6: Image Generation Node
def image_generation_node(state: GraphState) -> GraphState:
    client = Together()

    if state.image_paths is None:
        state.image_paths = {}

    save_dir = "/content/images"
    os.makedirs(save_dir, exist_ok=True)

    for topic, content in state.seo_optimized_content.items():
        topic_name = Gemini_2.invoke(
            f"Generate a **single short topic title** (max 5 words) for the following content:\n\n{content}"
        ).content.strip()

        safe_topic_name = re.sub(r'[^\w\-_.]', '_', topic_name)[:30]

        image_prompt = Llama31.invoke(f"""
        Create a **realistic, detailed image prompt** for: {topic_name}.
        - Describe a visual scene, avoiding abstract concepts.
        - Example: If the topic is "AI in Finance", describe a **stock trading floor** with AI assistants.

        **Topic:** {topic_name}
        **Image Prompt:**
        """).content.strip()
```

```
            response = client.images.generate(
                prompt=image_prompt,
                model="black-forest-labs/FLUX.1-schnell-Free",
                width=1024,
                height=768,
                steps=1,
                n=1,
                response_format="b64_json"
            )

            image_data = base64.b64decode(response.data[0].b64_json)
            image = Image.open(BytesIO(image_data))
            image_path = os.path.join(save_dir, f"{safe_topic_name}.png")

            try:
                image.save(image_path)
                print(f"✅ Image saved: {image_path}")
            except Exception as e:
                print(f"❌ Error saving image: {e}")

            state.image_paths[topic] = image_path

        return state

    # 🔹 Step 7: Display Results Node
    def display_results_node(state: GraphState) -> GraphState:
        display(Markdown("## 📌 AI Blog Generation Results\n---"))

        search_results_md = "### 🔎 Search Results\n"
        for idx, result in enumerate(state.search_results, start=1):
            search_results_md += f"- **[{result['title']}]({result['url']})**\n"
        display(Markdown(search_results_md))

        for topic, summary in state.summaries.items():
            if topic in state.image_paths:
                display(Image.open(state.image_paths[topic]))
            display(Markdown(f"### 📖 {topic}\n\n{summary}\n---"))

        for topic, seo_content in state.seo_optimized_content.items():
            if topic in state.image_paths:
                display(Image.open(state.image_paths[topic]))

        display(Markdown("## ✅ AI Blog Summarization Completed 🎉"))
```

```
# 🔹 Graph Workflow Definition
blog_gen_graph = StateGraph(state_schema=GraphState)
blog_gen_graph.add_node("search", search_node)
blog_gen_graph.add_node("crawl", crawl_node)
blog_gen_graph.add_node("classify", classify_node)
blog_gen_graph.add_node("summarize", summarize_node)
blog_gen_graph.add_node("seo_optimize", seo_optimize_node)
blog_gen_graph.add_node("image_generation", image_generation_node)
blog_gen_graph.add_node("display", display_results_node)

blog_gen_graph.add_edge(START, "search")
blog_gen_graph.add_edge("search", "crawl")
blog_gen_graph.add_edge("crawl", "classify")
blog_gen_graph.add_edge("classify", "summarize")
blog_gen_graph.add_edge("summarize", "seo_optimize")
blog_gen_graph.add_edge("seo_optimize", "image_generation")
blog_gen_graph.add_edge("image_generation", "display")
blog_gen_graph.add_edge("display", END)

compiled_graph = blog_gen_graph.compile()
compiled_graph
```


__start__
search
crawl
classify
summarize
seo_optimize
image_generation
display
__end__

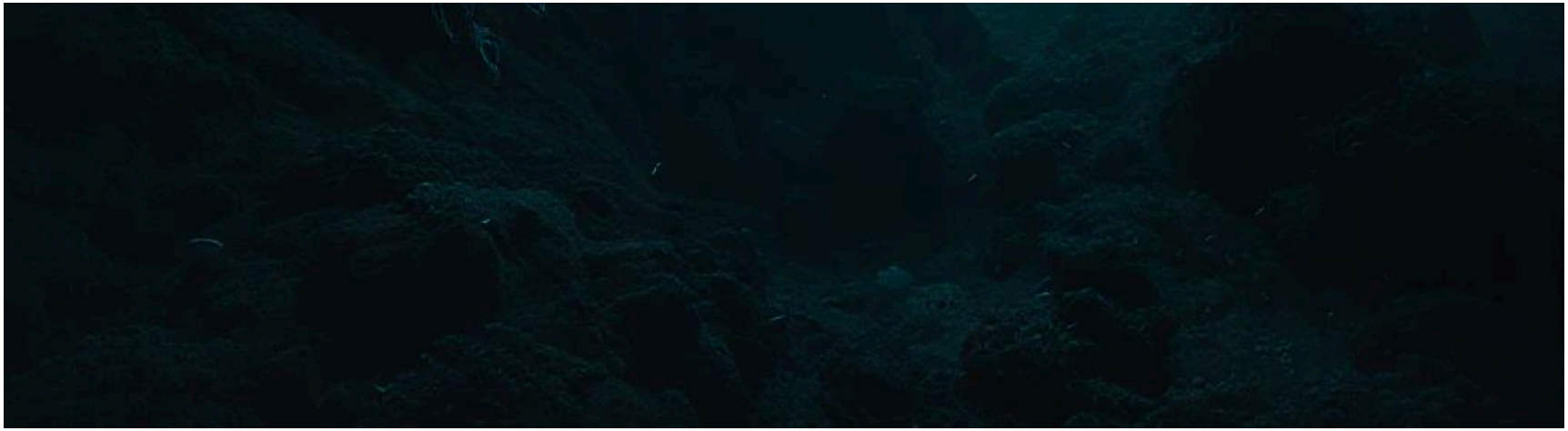
```
initial_state = GraphState(query="The Mystery of the Deep Ocean: What Lies Beneath?")
final_state = compiled_graph.invoke(initial_state)
final_state
```

```
🕷️ Crawling: https://sanctuaries.noaa.gov/magazine/5/mysteries-of-the-deep/
🕷️ Crawling: https://toxigon.com/mysteries-of-the-deep
🕷️ Crawling: https://www.americanoceans.org/facts/whats-at-the-bottom-of-the-ocean/
❌ Failed to crawl https://www.americanoceans.org/facts/whats-at-the-bottom-of-the-ocean/: Unexpected error during start crawl job: Status code 429. Rate
🕷️ Crawling: https://www.vox.com/science-and-health/23030491/ocean-scientific-mysteries-unexplainable-podcast
❌ Failed to crawl https://www.vox.com/science-and-health/23030491/ocean-scientific-mysteries-unexplainable-podcast: Unexpected error during start crawl j
🕷️ Crawling: https://earthhow.com/ocean-mysteries/
❌ Failed to crawl https://earthhow.com/ocean-mysteries/: Unexpected error during start crawl job: Status code 429. Rate limit exceeded. Consumed (req/mi
✅ Image saved: /content/images/Deep_Ocean__Unexplored_and_Eni.png
✅ Image saved: /content/images/Deep_Sea__Exploration_and_Cons.png
```

# 📌 AI Blog Generation Results

## 🔎 Search Results

- **Mysteries of the Deep; Mysteries of the Deep**
- **Mysteries of the Deep: What Lies Beneath the Ocean? Welcome folks, Toxigon here! Today, we're diving**
- **The vast majority of the ocean remains a mystery, especially the deepest parts**
- **Unexplainable: 10 ocean mysteries scientists haven’t solved yet | Vox The Earth is mainly a water wo**
- **10 Unsolved Mysteries of the Deep Ocean - Earth How Home » Water Science » Oceans » 10 Unsolved Myst**



📖 **Based on the title "Mysteries of the Deep" and the content about unexplored ocean areas within national marine sanctuaries, the most appropriate sub-topic classification is:**

**Deep Sea Exploration/Ocean Exploration**

## Step 1: Extract key insights and major findings

The content discusses the mysteries of the deep ocean, highlighting that approximately 95% of the ocean remains unexplored. NOAA's Office of National Marine Sanctuaries is working to expand our understanding of these areas through deep-water exploration and research. In 2019, they explored deep areas of several national marine sanctuaries and streamed it live.

## Step 2: Identify contradictions and opposing views

There are no apparent contradictions or opposing views presented in the content. The information provided seems to be factual and focused on raising awareness about the unexplored ocean and the efforts of NOAA's Office of National Marine Sanctuaries.

## Step 3: Highlight statistics, trends, or research findings

A significant statistic mentioned is that approximately 95% of the ocean remains unexplored, with much of it being in the deep sea. This highlights the vast amount of unknown territory that still needs to be discovered and studied.

## Step 4: Summarize in a neutral, unbiased manner

The content provides an overview of the current state of ocean exploration, emphasizing the vastness of the unexplored deep sea. It informs readers about the efforts of NOAA's Office of National Marine Sanctuaries to explore and research these areas, including their live-streamed expeditions in 2019.

The final answer is:

- Key Insights: The ocean is largely unexplored, with about 95% remaining unseen. NOAA's Office of National Marine Sanctuaries is actively working to change this through exploration and research, including live-streamed expeditions.
- Contradictions: None apparent.
- Statistics & Trends: Approximately 95% of the ocean is unexplored, with a focus on deep-sea areas. NOAA conducted explorations in several national marine sanctuaries in 2019.
- Final Summary: The deep ocean is a vastly unexplored region of our planet, with NOAA's Office of National Marine Sanctuaries undertaking efforts to explore and understand these areas better, including recent expeditions to several national marine sanctuaries.

📖 **The provided information is insufficient to classify the article into a specific sub-topic. All we have is the general category "Science and Nature" and some social sharing buttons. To determine a sub-topic, we would need the article's title, a brief summary, or the actual content.**

**summary, of the actual content.**

For example, potential sub-topics within "Science and Nature" could include:

- **Biology:** (e.g., genetics, ecology, zoology, botany)
- **Physics:** (e.g., astrophysics, quantum mechanics)
- **Chemistry:** (e.g., organic chemistry, biochemistry)
- **Environmental Science:** (e.g., climate change, conservation)
- **Geology:** (e.g., plate tectonics, paleontology)
- **Astronomy:** (e.g., cosmology, planetary science)
- **Health/Medicine:** (e.g., new treatments, research studies)
- **Technology:** (e.g., scientific advancements)
- **Nature:** (e.g., wildlife, ecosystems)

Without more information, it's impossible to be more specific.

**Key Insights:**

1. The deep sea is a vast and largely unexplored environment, covering about 65% of the Earth's surface.
2. The deep sea is home to unique ecosystems, including hydrothermal vents, which support life forms that can withstand extreme conditions.
3. The deep sea plays a crucial role in regulating the Earth's climate, cycling nutrients, and producing oxygen.
4. Human activities, such as pollution, overfishing, and climate change, pose significant threats to the deep sea and its ecosystems.
5. Protecting the deep sea requires international cooperation, establishment of marine protected areas, and responsible exploration and management of deep-sea resources.

**Contradictions:** None explicitly stated in the article. However, there may be opposing views on the effectiveness of current conservation efforts or the balance between economic interests and environmental protection.

**Statistics & Trends:**

1. The deep sea covers about 65% of the Earth's surface.
2. The deepest part of the ocean, the Mariana Trench, reaches a maximum-known depth of over 11,000 meters.
3. Hydrothermal vents can reach temperatures of over 400°C.
4. Scientists estimate that there could be millions of undiscovered species living in the deep sea.
5. Deep-sea trawling and other human activities are having a devastating impact on deep-sea ecosystems.

**Final Summary:**

## The deep sea is a vast, complex, and largely unexplored environment that plays a critical role in maintaining the health of our planet. Despite its importance, the deep sea faces numerous threats from human activities, including pollution, overfishing, and climate change. To protect the deep sea and its ecosystems, it is essential to establish marine protected areas, promote responsible exploration and management of deep-sea resources, and raise awareness about the importance of preserving this unique and fragile environment. Further research and international cooperation are necessary to address the challenges facing the deep sea and to ensure its long-term conservation.