

Title: Generating AI/GenAI Use Cases for Industry-Specific Applications

Objective

The aim is to create a **Multi-Agent System** that generates AI and Generative AI (GenAI) use cases for a specific company or industry. This system conducts market research, identifies relevant industries, and suggests resource assets for AI/ML solutions with a focus on operations, supply chain, and customer experience.

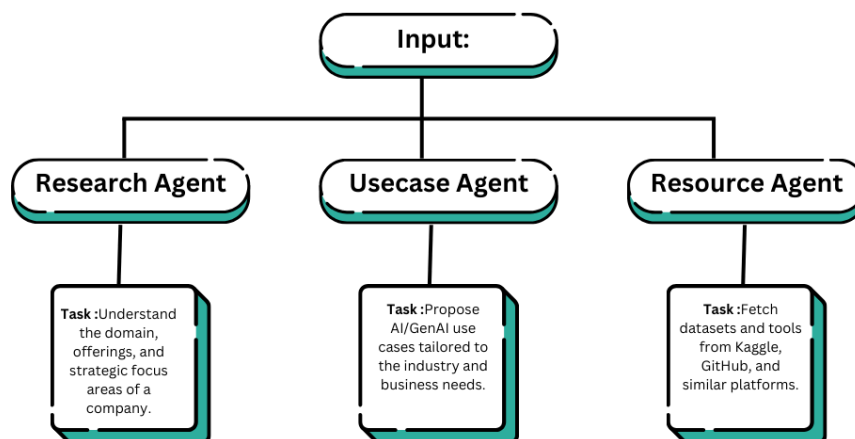
System Design

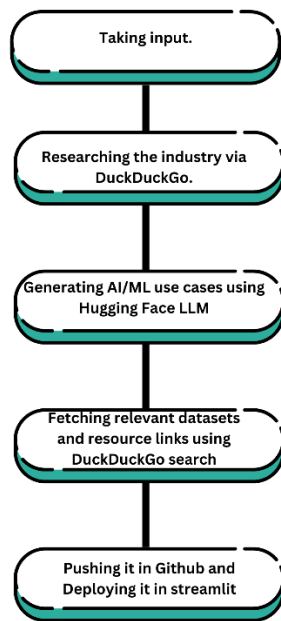
Overview

The system uses a Multi-Agent architecture with three key agents:

1. **Research Agent:**
Conducts market research to infer the industry and gather insights.
Tool: DuckDuckGo search.
2. **Use Case Agent:**
Generates AI/ML use cases based on the inferred industry.
Tool: Hugging Face LLM API.
3. **Resource Agent:**
Searches for datasets and resource links relevant to the company and industry.
Tool: DuckDuckGo search with targeted queries.

Flow Charts





Methodology

Input:

- **Company Name** (string).

Workflow:

1. Research Agent:

- Uses the company name to query its industry using DuckDuckGo.
- Extracts industry-specific keywords from search results.

2. Use Case Agent:

- Sends the inferred industry to the Hugging Face LLM.
- Generates actionable AI/ML use cases focusing on operations, supply chain, and customer experience.

3. Resource Agent:

- Constructs queries for relevant datasets or resources on Kaggle and GitHub.
- Formats the extracted links and resources.

4. Output:

- Industry Insights
- AI/ML Use Cases
- Dataset and Resource Links

Code implementation :

```
import streamlit as st
```

```
from langchain_community.tools import DuckDuckGoSearchRun
```

```
from huggingface_hub import InferenceClient
```

Initialize DuckDuckGoSearchRun and Hugging Face client

```
search = DuckDuckGoSearchRun()
```

```
client = InferenceClient(api_key="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")
```

Streamlit App

```
st.title("AI Use Case Generator ")
```

Input field for company name

```
company = st.text_input("Enter the Company Name:")
```

Function to infer industry based on company name

```
def infer_industry(company):
```

```
    query = f"{company} industry"
```

```
    results = search.invoke(query)
```

```
    return results
```

Function to generate AI/ML use cases based on the industry

```
def generate_use_cases_with_hf(industry):
```

```
    messages = [
```

```
        {
```

```
            "role": "user",
```

```
            "content": f"Suggest relevant AI and GenAI use cases for the {industry} industry, focusing on operations, supply chain, and customer experience. Highlight actionable insights and potential challenges."
```

```
        }
```

```
    ]
```

```

response = ""
with st.spinner("Generating use cases with Hugging Face LLM..."):
    stream = client.chat.completions.create(
        model="mistralai/Mistral-7B-Instruct-v0.3",
        messages=messages,
        max_tokens=500,
        stream=True
    )
    for chunk in stream:
        delta = chunk.choices[0].delta.content
        response += delta

return response.strip() # Clean up any leading/trailing whitespace

```

Function to fetch relevant links (datasets or resources)

```

def fetch_relevant_links(company, industry):
    query = f'{company} {industry} datasets site:kaggle.com OR site:github.com'
    results = search.invoke(query)

```

Check if results are valid and format them

```

if results:
    github_links = [result for result in results if 'github.com' in result]
    kaggle_links = [result for result in results if 'kaggle.com' in result]

    formatted_results = []
    if github_links:
        formatted_results.append(f"***GitHub Links***")
        formatted_results.extend(github_links)

    if kaggle_links:
        formatted_results.append(f"***Kaggle Links***")
        formatted_results.extend(kaggle_links)

```

```
    return "\n".join(formatted_results) if formatted_results else "No relevant datasets or resources found."
```

```
else:
```

```
    return "No relevant datasets or resources found."
```

Streamlit Workflow

```
if st.button("Generate") and company:
```

```
    with st.spinner("Processing..."):
```

```
        try:
```

Step 1: Industry Insights

```
            st.subheader("Industry Insights")
```

```
            industry_info = infer_industry(company)
```

```
            st.write(industry_info)
```

Step 2: AI Use Cases

```
            st.subheader("AI/ML Use Cases")
```

```
            use_cases = generate_use_cases_with_hf(industry_info)
```

```
            st.write(use_cases)
```

Step 3: Dataset and Resource Links

```
            st.subheader("Relevant Datasets and Resources")
```

```
            links = fetch_relevant_links(company, industry_info)
```

```
            st.write(links)
```

```
            st.success("Data Generated Successfully!")
```

```
        except Exception as e:
```

```
            st.error(f"An error occurred: {e}")
```

Requirements.txt

```
streamlit
```

```
langchain
```

```
llama-index
```

```
huggingface-hub
```

accelerate

langchain_community

duckduckgo-search

Outputs

Industry Insights: Provides a summary of the company's industry and strategic areas.

AI/ML Use Cases: Suggests actionable solutions tailored to the company's operations and goals.

Resource Links: Lists datasets and tools from Kaggle, GitHub, and other platforms.

Conclusion

The system effectively combines market research, AI use case generation, and resource collection into a streamlined application. Its modular and agent-based design ensures scalability and adaptability across industries.

System Requirements

Python 3.8+

Hugging Face API Key

DuckDuckGoSearchRun Integration

References

McKinsey AI Insights

Deloitte Digital Transformation Reports

Kaggle and GitHub resources for datasets

Future Work

Expand search capabilities with APIs like Serper for more refined insights.

Integrate deployment tools like Gradio for broader accessibility.

Enhance visualization of generated insights using charts and graphs.

Photos and links:

Streamlit : <https://research-agents-eetunste6v8e8flzwcda6.streamlit.app/>

Share ☆ ↗ 🔍 ⋮

AI Use Case Generator

Enter the Company Name:

tesla

Generate

Industry Insights

Tesla is the leading producer of plug-in electric vehicles globally. Its Model 3 has become the world's best-selling all-electric vehicle model. ... Statista R identifies and awards industry ... Tesla, Inc. is an American manufacturer of electric vehicles, solar panels, and automobile batteries. It was founded in 2003 by American entrepreneurs Martin Eberhard and Marc Tarpenning and was named after Serbian American inventor Nikola Tesla. Elon Musk, an early investor in the company, became CEO in 2008. Tesla sees 20%-30% growth in vehicle sales next year. Tesla's Q3 profit beats estimates, shares surge 12%. Costs of making a car fell to record low, boosting profit margins. This month's robotaxi ... by Florian Zandt, Jan 25, 2024. Tesla ended Q4 2023 with a net income of 7.9billionandthe full yearwith15 billion in profits. While Elon Musk's company more than doubled its earnings ... A Tesla electric car recharges at a Tesla dealership on January 16, 2024 in Burbank, California. ... "The industry has seen a surge of new entrants as well as legacy automakers competing for a ...

AI/ML Use Cases

1. **Autonomous Driving and AI:** With the development of the much-anticipated Tesla robotaxi, AI will

< Manage app

Share ☆ ↗ 🔍 ⋮

AI/ML Use Cases

1. **Autonomous Driving and AI:** With the development of the much-anticipated Tesla robotaxi, AI will play a crucial role. The AI system will be responsible for navigating the vehicle, finding parking spots, and following traffic rules. This could lead to improved safety, efficiency, and convenience for users. However, challenges include ensuring the AI system's ability to handle various driving conditions and ensuring passenger safety.

2. **Predictive Maintenance:** AI can be used to predict when a vehicle might require maintenance. By analyzing data from sensors and cameras installed in the vehicles, AI algorithms can identify patterns that indicate potential issues before they become significant problems. This can help reduce downtime, enhance customer satisfaction, and improve operating costs.

3. **Smart Grid and Energy Management:** Tesla's production of solar panels and energy storage systems are complementary to its electric vehicles. AI can be used to optimize energy production, storage, and consumption. For example, AI can help Tesla's Powerwalls to manage household energy consumption more efficiently, reducing costs and promoting sustainable living.

4. **Customer Experience:** AI can be used to personalize the user experience. This could include tailoring features in the car to individual preferences, providing proactive customer support, and offering personalized recommendations for energy usage and charging times.

5. **Supply Chain Optimization:** AI can help Tesla manage its supply chain more efficiently. By predicting demand, optimizing production schedules, and managing inventory levels, AI can help reduce unnecessary costs and ensure that Tesla can meet demand in a timely manner.

6. **Autonomous Fleet Management:** As Tesla expands its robotaxi fleet, AI will be essential for managing

< Manage app