InstaResz Business Services Pvt.Ltd

(AI Assignment)

Introduction

The goal of this project was to design and develop a **fully automated system** that can intelligently analyze a company's internal documents (such as strategic reports or vision statements), research the company and its broader industry environment, identify current market trends, and **generate customized AI/ML/GenAI use cases**.

Furthermore, the system connects each proposed use case to **real-world datasets** to enable immediate experimentation. The entire workflow is presented to users via a **modern, chat-based web interface** built with Streamlit, offering an intuitive and engaging user experience.

Methodology

The solution follows a **modularized, agent-based architecture** to maximize scalability, reusability, and clarity:

a. Input Phase

- **User Interaction**: The user uploads a **PDF document** (e.g., a strategy plan) through the Streamlit web application.
- **Optional Fields**: Users can optionally input the **company name** or provide **manual text input** in case the file is unavailable.

b. Processing Pipeline (Orchestration Layer)

The main automation happens through **auto_pipeline.py**, which orchestrates the following steps:

1. **Text Extraction**
    - Extract full textual content from the uploaded PDF using efficient parsing libraries.
2. **Summarization**

    - Summarize extracted text using **HuggingFace** transformer models (e.g., BART or T5) to create a concise version suitable for analysis.

3. **Company Research**
    - Perform a **web search** via **SerpAPI** to find:
        - Key product offerings
        - Strategic focus areas
        - Industry classification
        - Company mission and vision
4. **Market Trend Analysis**
    - Analyze the latest **AI/ML/GenAI trends** in the identified industry using **OpenAI's GPT models**, generating brief, actionable insights.
5. **Use Case Generation**
    - Based on the researched industry and market trends, automatically generate tailored **AI/ML/GenAI use cases**.
6. **Dataset Search**
    - Search for datasets related to each generated use case across:
        - **Kaggle**
        - **HuggingFace Datasets Hub**
        - **GitHub repositories** (datasets tagged)
7. **Bundling Results**
    - Collect all outputs (summary, research, trends, use cases, datasets) into a unified response structure.

Knowledge Base Agent (knowledge_base_agent.py)
Provides a simpler, standalone service to **just read and summarize PDFs**, useful for building a local knowledge repository.

c. Frontend User Interface (Streamlit)

- **Chat-Style Interaction**: Users interact with an **assistant-like conversational interface**.

- **Visual Output Organization**:
  - Summarized Document View
  - Company Research Findings
  - Industry Trends
  - AI/ML/GenAI Use Case Ideas
  - Dataset Resources (Expandable Sections)
- **Technical Details**:
  - Maintains chat history via st.session_state.messages.
  - Uses Markdown and expanders for clear sectioned display.

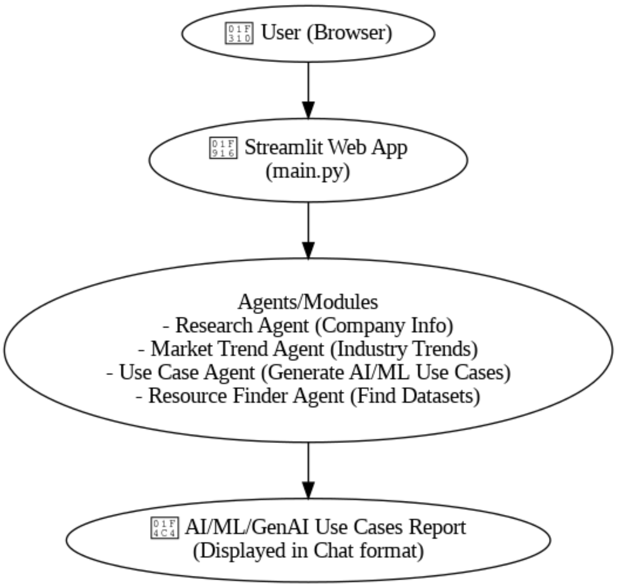
d. Optional Backend (Flask API)

A lightweight **Flask server** exposes a /predict endpoint for:

- Text summarization or classification tasks (Optional enhancement).
- Offers flexibility to plug into different systems beyond Streamlit.

Architecture Flowchart

The system architecture is summarized below:


User (Browser)
Streamlit Web App (main.py)
Agents/Modules
- Research Agent (Company Info)
- Market Trend Agent (Industry Trends)
- Use Case Agent (Generate AI/ML Use Cases)
- Resource Finder Agent (Find Datasets)
AI/ML/GenAI Use Cases Report (Displayed in Chat format)

Results

- Successfully deployed an **end-to-end AI/ML discovery pipeline**.
- Achieved **high modularity** by dividing logic into reusable agents.
- Integrated **state-of-the-art APIs**:
  - **OpenAI** for GPT-based trend analysis
  - **SerpAPI** for real-time web research
  - **Kaggle**, **HuggingFace**, **GitHub** for dataset discovery
- Delivered an intuitive **chat-based user experience**.
- Ensured **robust fallback strategies**:
  - Defaults to manual user input if company research fails.
  - Provides clear messaging if no datasets are found.
- Implemented **error resilience**:
  - Graceful handling of unknown industries.
  - Session state preserved throughout conversation.
- Built-in **future extensibility** via modular agent design.

Key Observations & Improvements

Issue	Observation	Action Taken / Future Suggestion
API Input Mismatch	"input_text" vs "text" mismatch between Flask and Streamlit	Standardized key names across all components.
Duplicated Headers	Dataset resources header appeared twice	Modularized resource display via helper functions.
Text Slicing Issue	Trends were cut mid-sentence if under 200 characters	Improved logic to dynamically check length before truncating.
Code Reusability	Repeated code patterns in resource displays	Suggested writing centralized utility functions for resource handling.
Unknown Industry Cases	Minimal default suggestions for unknown industries	Expanded with more generalized AI/ML/GenAI use case ideas.

Conclusions

This project demonstrates the **practical application** of **agent-based pipelines** combined with **LLMs** and **open APIs** to automate traditionally manual, research-heavy tasks in business strategy and data science.

By automatically:

- Understanding company vision and offerings,
- Researching market trends,
- Generating customized AI/ML use case ideas,
- Linking real datasets for experimentation,

the system empowers business leaders, data scientists, and consultants to **accelerate innovation cycles** and **initiate AI projects** with minimal effort.

Future Extensions

- 📈 **Financial Analysis Agent**: Automate financial data collection and insights generation.
- 🌱 **ESG Trend Agent**: Analyze environmental, social, and governance trends.
- 📝 **Patent Research Agent**: Explore innovation trends via patent databases.

Key Takeaway:

Agent-based modular pipelines + LLMs + Open APIs = Powerful automation frameworks for real-world domain-specific insights.