Project-High Level Design

on

Autonomous Energy Researcher Agent

Course Name:Agentic AI

Institution Name: Medicaps University – Datagami Skill Based Course

Student Name(s) & Enrolment Number(s):

Sr no	Student Name	Enrolment Number
1.	NAINA MANGHANI	EN22CS301626
2.	MAHAK SHRIMAL	EN22CS301566
3.	MAHAK TAOSE	EN22CS301567
4.	MOHIT PATEL	EN22CS301610
5.	PAAVAN PATNI	EN22CS301671

Group Name:Group 02D6

Project Number:AAI-13

Industry Mentor Name:

University Mentor Name:Prof.Nishant Shrivastava

Academic Year:

TABLE OF CONTENTS

1. INTRODUCTION

The Autonomous Energy Researcher Agent is an advanced Agentic AI-based research automation system developed for the Energy domain. The global energy sector is experiencing rapid transformation due to renewable energy adoption, climate policies, technological advancements, fluctuating markets, and sustainability initiatives. Every day, large volumes of information are published in the form of research papers, government regulations, industry news, sustainability reports, and market analyses. Extracting meaningful insights from this continuously growing data is time-consuming and requires structured research workflows.Traditional research methods involve manual browsing, reading multiple sources, comparing viewpoints, summarizing findings, and organizing insights for future reference. This process is not only inefficient but also prone to human bias and inconsistency. Additionally, conventional AI systems such as chatbots or prompt-based LLM applications lack autonomous planning, tool usage capability, and structured workflow management.To overcome these limitations, this project proposes a multi-agent autonomous research system built using CrewAI, FastAPI, Streamlit, Tavily Search API, and Groq LLM services. The system implements Agentic AI principles such as reasoning, planning, task delegation, tool integration, and persistent memory. Instead of relying on a single AI response, the system assigns distinct responsibilities to specialized agents (Research Agent, Analyst Agent, Writer Agent), enabling structured collaboration similar to real-world organizational research teams.

1.1 SCOPE OF THE DOCUMENT

This High-Level Design document provides a comprehensive overview of the system architecture and major technical components of the Autonomous Energy Researcher Agent.

The scope of this document includes:

1. Description of the overall system architecture and layered design.
2. Identification of major system components and their responsibilities.
3. Explanation of application design and process workflow.
4. Overview of information flow between agents and external services.
5. High-level description of API endpoints and integration mechanisms.
6. Outline of data storage, persistence, and access strategies.
7. Definition of non-functional requirements such as performance and security.
8. Identification of key design principles and architectural decisions.

The document does not include:

1. Detailed source code implementation.
2. Low-level class definitions and method logic.

3. Internal algorithmic specifications.
4. Deployment configuration scripts.
5. Unit-level testing documentation.

1.2 INTENDED AUDIENCE

The Autonomous Energy Researcher Agent is designed for individuals and organizations involved in energy research, analysis, policy-making, and strategic planning. The system is intended to support users who require structured, reliable, and up-to-date research outputs without manually browsing multiple sources.

The primary users of the system include:

1. **Energy Analysts**

A. Professionals analyzing renewable energy trends, fossil fuel markets, and sustainability initiatives.
B. Use the system to generate quick comparative research reports.
C. Benefit from automated multi-source synthesis.

2. **Policy Makers and Government Officials**

A. Require summarized insights on energy policies and climate regulations.
B. Use the system for policy comparison and impact analysis.
C. Benefit from structured, unbiased research summaries.

3. **Researchers and Academicians**

A. Conduct academic research in renewable energy, smart grids, and sustainability.
B. Use the system to gather literature summaries.
C. Benefit from automated information aggregation.

4. **Business Strategists and Energy Companies**

A. Analyze market trends, investments, and regulatory updates.
B. Use the system for competitive analysis and forecasting support.
C. Benefit from real-time web-integrated research outputs.

5. **Students and Project Developers**

A. Use the system for academic projects and case studies.
B. Benefit from structured report generation and research history storage.

1.3 SYSTEM OVERVIEW

1. User Interaction Stage

A. The user enters an energy-related research topic via the Streamlit-based frontend.
B. The system provides a simple and interactive interface for query submission.
C. The user can view the generated report and access previous research history.

2. Backend Processing Stage

A. The Streamlit interface sends the request to the FastAPI backend.
B. FastAPI validates the input for correctness and completeness.
C. The Research Controller initializes the research workflow.

3. Multi-Agent Orchestration Stage-The system uses CrewAI to implement structured agent collaboration.

A. Research Agent-Performs autonomous web searches.Retrieves relevant information using Tavily API.
B. Analyst Agent-Filters irrelevant content.Extracts key insights.
C. Writer Agent-Organizes insights into structured report format.
D. Crew Manager-Assigns tasks to agents.Manages sequential and parallel execution.

4. External Tool Integration Stage-The system integrates external services to enhance intelligence:

A. Tavily API for real-time web search.
B. Groq LLM API for reasoning, summarization, and report generation.
C. .env configuration for secure API key management.

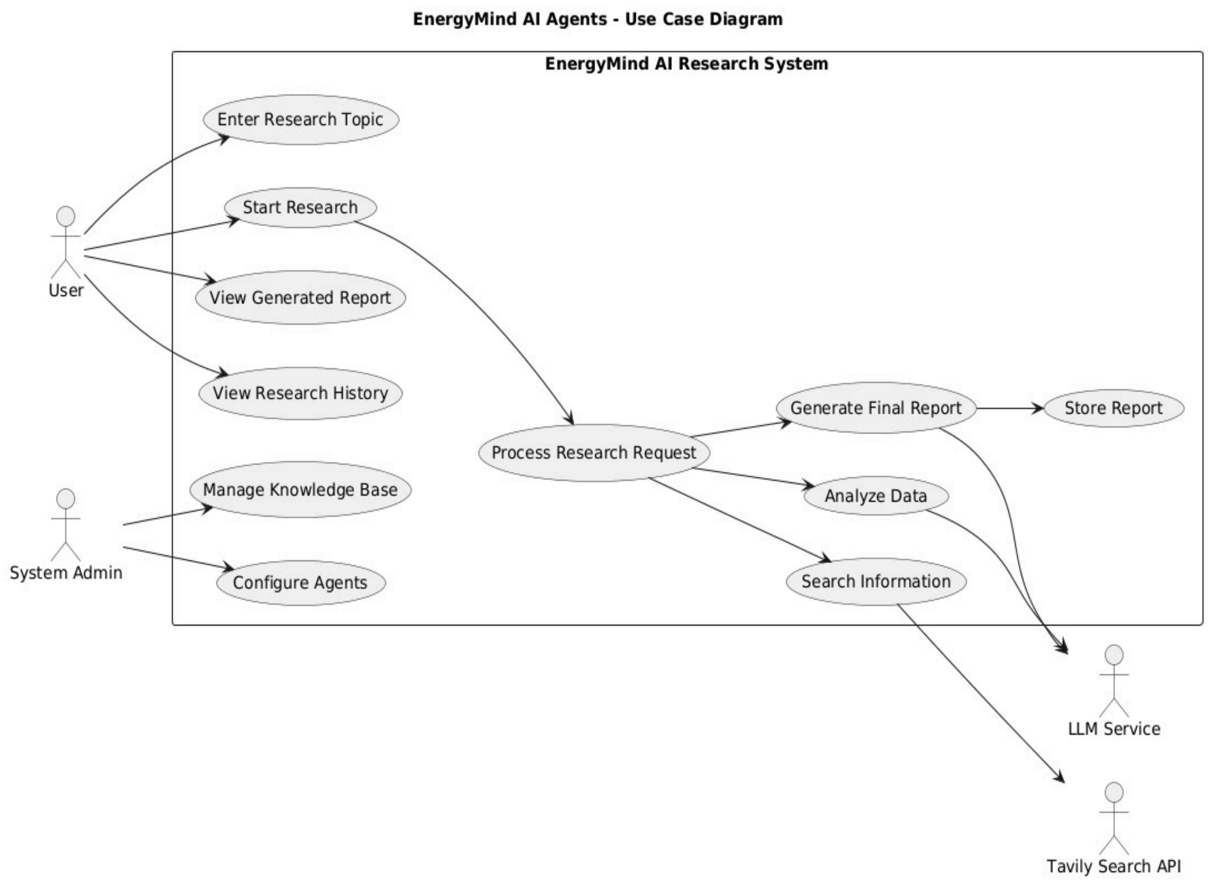
5. Knowledge Persistence Stage-The generated research report is saved in a text-based knowledge repository.Reports are stored with structured naming conventions.

6.Architectural Characteristics-The system exhibits the following architectural properties:

A. **Layered Architecture** – Separation of presentation, API, orchestration, tools, and storage layers.
B. **Modularity** – Each component operates independently.
C. **Scalability** – New agents and tools can be added.
D. **Extensibility** – Can integrate databases or dashboards in future.
E. **Autonomous Workflow Execution** – Agents plan and execute tasks independently.
F. **Tool-Augmented Intelligence** – LLM combined with external search capabilities.

7. Overall System Behavior Summary-The complete system behavior can be summarized as:

A. Accept Query
B. Validate Input
C. Create Research Tasks
D. Execute Multi-Agent Workflow
E. Retrieve and Analyze Web Data
F. Generate Structured Report
G. Store Report in Knowledge Base
H. Display Final Output to User



2. SYSTEM DESIGN

The System Design section describes the architectural structure, workflow organization, component interaction, and major technical decisions of the Autonomous Energy Researcher Agent. The design follows a layered and modular architecture to ensure scalability, maintainability, and extensibility. Each layer performs a distinct responsibility while interacting with other layers in a structured manner.The system is built around Agentic AI principles where intelligent agents collaborate, use external tools, and execute structured workflows autonomously.

2.1 APPLICATION DESIGN

The application follows a multi-layered architectural design, separating concerns into independent layers. This design improves clarity, scalability, and fault isolation.

The major layers include:

1. Presentation Layer (User Interface)

- **A.** Implemented using Streamlit.
- **B.** Allows users to enter energy research topics.
- **C.** Displays structured research outputs.
- **D.** Provides access to research history.

2. API Management Layer (Backend Layer)

- **A.** Implemented using FastAPI.
- **B.** Receives HTTP requests from the frontend.
- **C.** Validates user input.
- **D.** Handles request routing.

3. Agent Orchestration Layer (Core Intelligence Layer)

- **A.** Implemented using CrewAI framework.
- **B.** Manages multiple specialized agents.
- **C.** Assigns tasks based on query.
- **D.** Controls sequential and parallel execution.

4. Tool Integration Layer

- **A.** Tavily Search API for real-time web search.
- **B.** Groq LLM API for reasoning and report generation.
- **C.** .env file for secure API key management.

D. Supports future tool integration.

5. Persistence Layer (Knowledge Storage)

A. Stores generated research reports.
B. Maintains structured file-based repository.
C. Enables report retrieval.
D. Provides long-term memory functionality.

2.2 PROCESS FLOW

The system follows a structured multi-step execution process.Step-by-Step Process Flow:

1. User submits research topic.
2. Streamlit sends requests to the FastAPI endpoint.
3. FastAPI validates and forwards requests.
4. Research Controller generates structured tasks.
5. Crew Manager distributes tasks to agents.
6. Research Agent performs web search via Tavily API.
7. Analyst Agent analyzes collected data using LLM.
8. Writer Agent generates formatted research reports.
9. The report is saved in the Knowledge Base.
10. Final output is returned to the user interface.

This flow ensures controlled and traceable execution of research tasks.

2.3 INFORMATION FLOW

The Information Flow describes how data moves between components and external services.

Internal Flow:

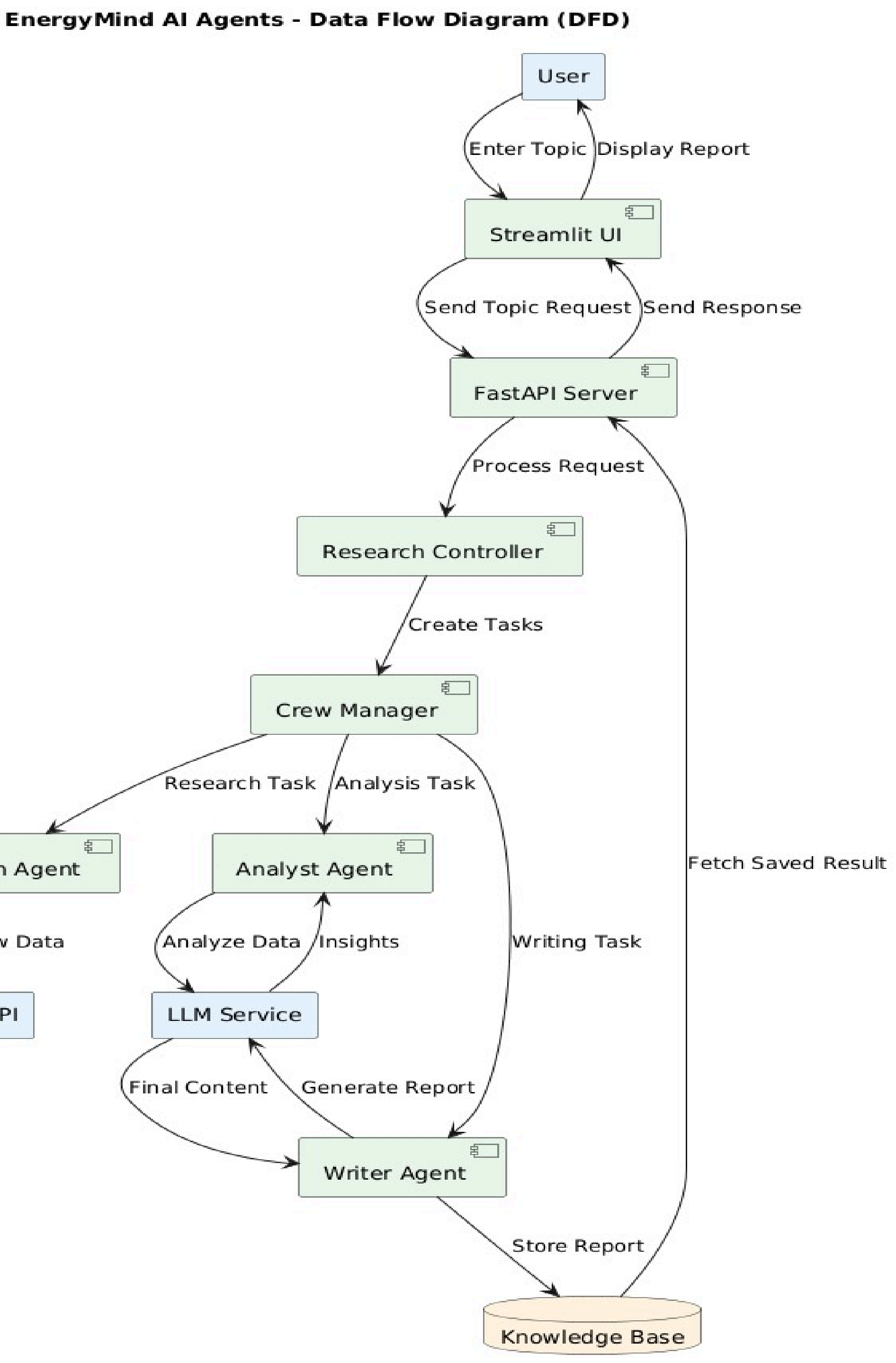
1. User-Streamlit UI
2. Streamlit-FastAPI
3. FastAPI-Research Controller
4. Research Controller-Crew Manager
5. Crew Manager-Agents

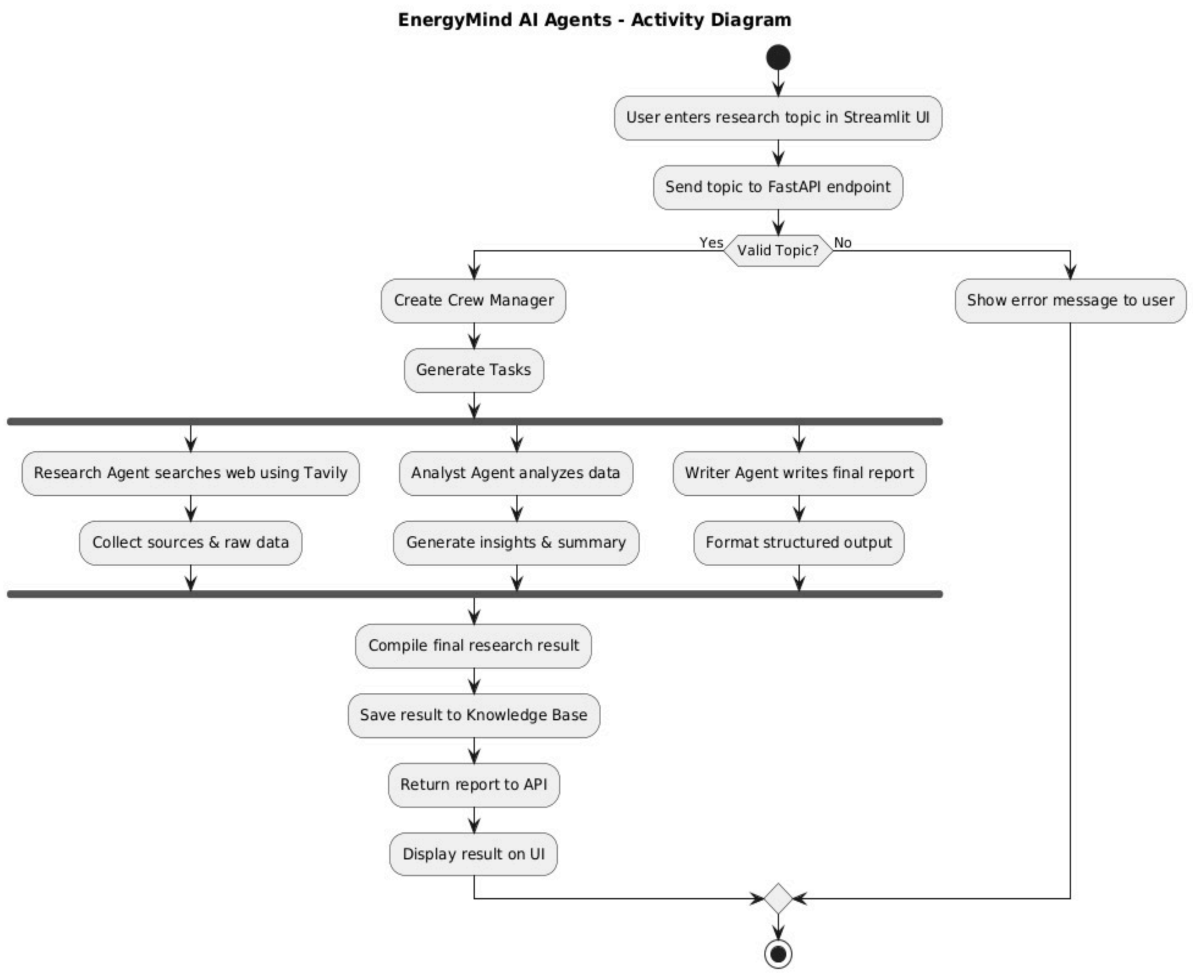
External Flow:

6. Research Agent-Tavily API
7. Tavily API-Research Agent
8. Analyst Agent-Grow LLM API
9. Groq LLM API-Analyst Agent

Final Output Flow:

10. Writer Agent-Knowledge Base
11. FastAPI-Streamlit
12. Streamlit-User




EnergyMind AI Agents - Activity Diagram
User enters research topic in Streamlit UI
Send topic to FastAPI endpoint
Valid Topic?
Yes
No
Create Crew Manager
Show error message to user
Generate Tasks
Research Agent searches web using Tavily
Analyst Agent analyzes data
Writer Agent writes final report
Collect sources & raw data
Generate insights & summary
Format structured output
Compile final research result
Save result to Knowledge Base
Return report to API
Display result on UI

2.4 COMPONENTS DESIGN

The system consists of logically separated components with defined responsibilities.

1. StreamlitApp-Collects user input.Displays research output.Handles UI interaction.

2. FastAPIApp-Exposes REST endpoints.Validates input.Triggers research execution.

3. Research Controller-Creates research tasks.Initializes Crew Manager.Manages execution lifecycle.

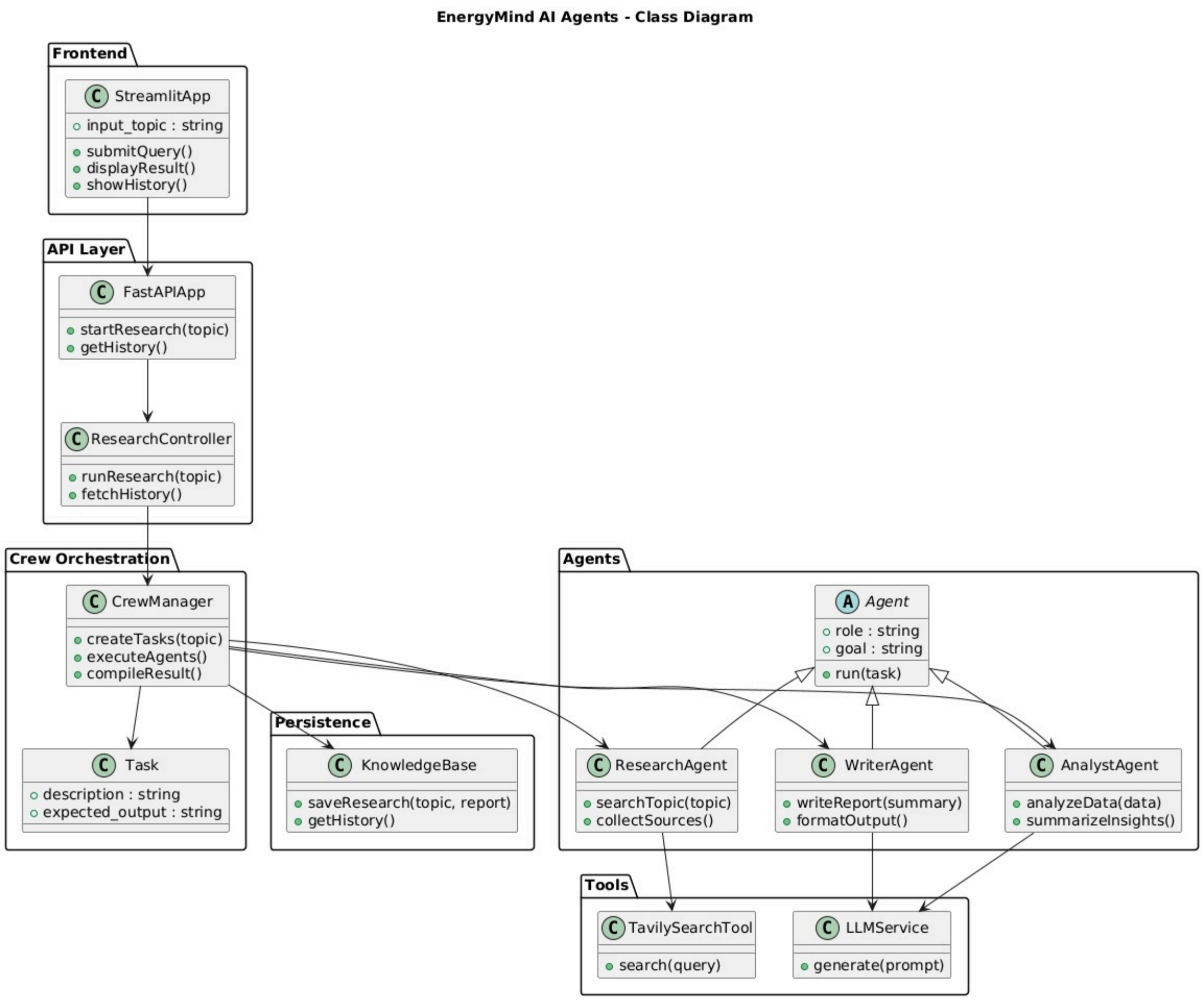
4.Crew Manager-Coordinates multiple agents.Assigns roles and tasks.Ensures task completion.

5. Research Agent-Performs autonomous web search.Collects raw research data.

6. Analyst Agent-Filters irrelevant information.Extracts insights.Performs reasoning using LLM.

7. Writer Agent-Structures insights into report format.Generates readable output.

8. Knowledge Base-Stores reports.Supports retrieval.Maintains persistent memory.



2.5 KEY DESIGN CONSIDERATIONS

The following principles guided system design:

1. **Separation of Concerns-**Each layer performs a specific responsibility.

2. **Modularity-**Components are independent and replaceable.

3. **Scalability-**Additional agents can be added.

4. **Extensibility-**Database or dashboard integration possible.

5. **Security-**API keys stored securely.

6. **Maintainability-**Clear structure simplifies updates.

2.6 API CATALOGUE

The system exposes the following REST APIs:

1. POST /research

A. **Input-**JSON object containing research topic.
B. **Processing-**Initiates multi-agent workflow.
C. **Output-**Structured research report.

2. GET /history

A. Returns list of stored reports.
B. Enables research history retrieval.

3. GET /health

A. Checks system availability.
B. Returns server status.

3. DATA DESIGN

The Data Design section defines how information is structured, stored, accessed, and managed within the Autonomous Energy Researcher Agent system. Since the system performs autonomous research and generates structured reports, proper data organization and persistence are essential for maintaining reliability and long-term usability.The system uses a file-based knowledge repository to store research outputs. Although the current implementation uses text-based storage, the architecture is designed to allow future integration with relational or NoSQL databases.

3.1 DATA MODEL

The Data Model defines the structure and organization of data handled by the system. Even though a formal database schema is not implemented, the system logically structures research information into defined fields.

1. Research Topic-The original query submitted by the user.Acts as the primary identifier for the report.Used in file naming and indexing.

2. Timestamp-Date and time of report generation.Ensures chronological tracking.Useful for research history management.

3.Raw Search Data-Data retrieved from Tavily API.May include URLs, article titles, summaries.Used internally by Analyst Agent.

4.Processed Insights-Extracted and filtered information.Contains summarized findings.Removes redundancy and irrelevant content.

5. Structured Research Report-Final output generated by Writer Agent.Includes:

A. Introduction
B. Key Trends
C. Comparative Analysis
D. Conclusions

3.2 DATA ACCESS MECHANISM

The Data Access Mechanism defines how the system reads and writes research data.Currently, the system uses a file-based storage mechanism implemented in Python.The data access process includes:

1. Write Operation

A. Writer Agent generates final report.
B. File name is generated using slugified research topic.
C. The report is saved in the knowledge_base directory.
D. Timestamp metadata is stored within file content.

2. Read Operation

A. GET /history endpoint fetches stored file list.
B. Files are read using Python file handling.
C. Content is returned via FastAPI to the frontend.

3. Directory Organization

A. Reports are stored in a structured folder hierarchy.
B. Prevents overwriting of files.
C. Enables organized knowledge accumulation.

3.3 DATA RETENTION POLICIES

The Data Retention Policy defines how long data is stored and how it is managed.Currently, the system follows a persistent retention model.

Retention Characteristics:

1. Research reports are stored permanently.
2. No automatic deletion mechanism.
3. No expiration policy implemented.
4. Reports remain available unless manually removed.

Data Integrity Measures:

1. Unique file naming prevents overwriting.
2. Slug-based naming ensures readability.
3. Files stored locally in controlled environment.

3.4 DATA MIGRATION

At present, no formal database migration is required since the system uses file-based storage. However, the architecture allows future migration to advanced storage solutions.Potential Migration Strategies:

1. **Migration to SQL Database**

A. Structured schema.
B. Indexed search.
C. Improved performance.

2. **Migration to NoSQL Database**

A. Flexible document storage.
B. Better scalability.
C. Suitable for unstructured research data.

3. **Cloud Storage Integration**

A. Centralized storage.
B. Multi-user support.
C. Remote accessibility.

4. **Search Engine Integration (ElasticSearch)**

A. Advanced indexing.
B. Fast retrieval.
C. Keyword-based search capability.

4. INTERFACES

1. **User Interface(Presentation Interface)-**The user interface is implemented using Streamlit and acts as the primary interaction point between the user and the system.

2. **REST API Interface (Backend Interface)-**The backend communication is handled through FastAPI REST endpoints. This interface allows the frontend to communicate with the agent orchestration system.
3. **Agent-Orchestration Interface-**This interface represents the internal interaction between:

A. Research Controller
B. Crew Manager
C. Specialized Agents (Research, Analyst, Writer)

4. External Tool Interfaces-The system integrates with external services to enhance intelligence and functionality.

5.Knowledge Base Interface(Storage Interface)-The storage interface manages data persistence.

5. STATE AND SESSION MANAGEMENT

1. **Stateless API Architecture**

A. Each request is processed independently.
B. No session token is required.
C. Backend does not maintain user-specific runtime memory.
D. All necessary data is passed within each request.

2. Persistent Knowledge Memory-Although the API is stateless, the system maintains memory through:

A. File-based knowledge repository.
B. Storage of research outputs.

C. Retrieval through API endpoints.
D. Accumulation of research history.

3.Session Handling-Currently:

A. No user authentication implemented.
B. No login-based session tracking.
C. Single-user academic deployment model.
D. No user-specific research segregation.

6. CACHING

1. Current Behavior

A. Every research query triggers a fresh Tavily search.
B. LLM API is called for each request.
C. No intermediate results are stored in the memory cache.
D. Ensures real-time and updated research output.

2. Advantages of Current Model

A. Always retrieves the latest information.
B. No stale data.
C. Simpler architecture.
D. Suitable for prototype deployment.

3. Future Caching Strategy-To improve scalability:

A. Implement Redis caching.
B. Cache frequently requested topics.
C. Store temporary Tavily results.
D. Reduce redundant LLM API calls.

7. NON-FUNCTIONAL REQUIREMENTS

7.1 Security Aspects

Security is critical because the system integrates external APIs and processes user-generated queries.The following security measures are implemented:

1. Secure API Key Management

- **A.** All API keys (Tavily, Groq LLM) are stored in a .env file.
- **B.** No sensitive credentials are hardcoded.

2. Input Validation

- **A.** FastAPI validates user input.
- **B.** Prevents malformed or empty queries.

7.2 Performance Aspects

Performance refers to system responsiveness and execution efficiency.

1. Asynchronous Request Handling

- **A.** FastAPI supports asynchronous operations.
- **B.** Reduces blocking during API calls.

2. Multi-Agent Parallelism

- **A.** Agents can execute tasks sequentially or in parallel.
- **B.** Improves structured workflow execution.

8. REFERENCES

[1] CrewAI Inc., "CrewAI Documentation," 2024. [Online]. Available

[2] S. P. Corporation, "FastAPI Documentation," 2024. [Online]. Available

[3] Streamlit Inc., "Streamlit Documentation," 2024. [Online]. Available

[4] Tavily AI, "Tavily Search API Documentation," 2024. [Online]. Available

[5] Groq Inc., "Groq LLM API Documentation," 2024. [Online]. Available

[6] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 4th ed. Hoboken, NJ, USA: Pearson, 2021.

[7] I. Sommerville, *Software Engineering*, 10th ed. Boston, MA, USA: Pearson, 2016.

[8] Medicaps University – Datagami Skill Based Course, "Autonomous Energy Researcher Agent – Project Report," 2025.