Project - High Level Design on Autonomous Logistics 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.	Reet Sood	EN22CS301796
2.	Roshni Rawat	EN22CS301826
3.	Parth Akhand	EN22CS301687
4.	Prakrati Rasaniya	EN22CS301722
5.	Riyanshi Jain	EN22CS301816
6.	Rishi Yadav	EN22CS301806

Group Name: Group 02D7

Project Number: AAI-25

Industry Mentor Name: Aashruti Shah

University Mentor Name: Ajeet Singh Rajput

Academic Year: 2022-2026

Table of Contents

1. Introduction

The Autonomous Logistics Researcher Agent is an LLM-powered multi-agent system designed to automate logistics research tasks.

The system uses:

- Large Language Models (LLMs)
- LangChain framework
- CrewAI for multi-agent orchestration

It autonomously retrieves logistics-related information from the web, synthesizes multi-source data, and generates structured research summaries stored in a knowledge repository.

1.1 Scope of the Document

This High-Level Design (HLD) document describes:

- Overall system architecture
- Major system components
- Workflow design
- Data design
- Integration points
- Key architectural decisions
- Non-functional requirements

1.2 Intended Audience

This document is intended for:

- Project evaluators
- Technical reviewers
- SMEs
- Development team members
- Academic assessment panel

1.3 System Overview

The system is a layered multi-agent architecture where:

- The user submits logistics-related queries
- Multiple agents collaborate to perform research
- Web search tools provide real-time information
- LLM synthesizes structured summaries
- Reports are stored for future reference

The system emphasizes modularity, scalability, and structured orchestration.

2. System Design

2.1 Application Design

The application follows a layered architecture:

1. Presentation Layer: Handles user input and displays results.

2. Orchestration Layer: CrewAI manages agent coordination and task execution.

3. Agent Layer

Includes:

- Research Agent
- Analysis Agent
- Summary Agent

4. Tool Layer

- Web Search API

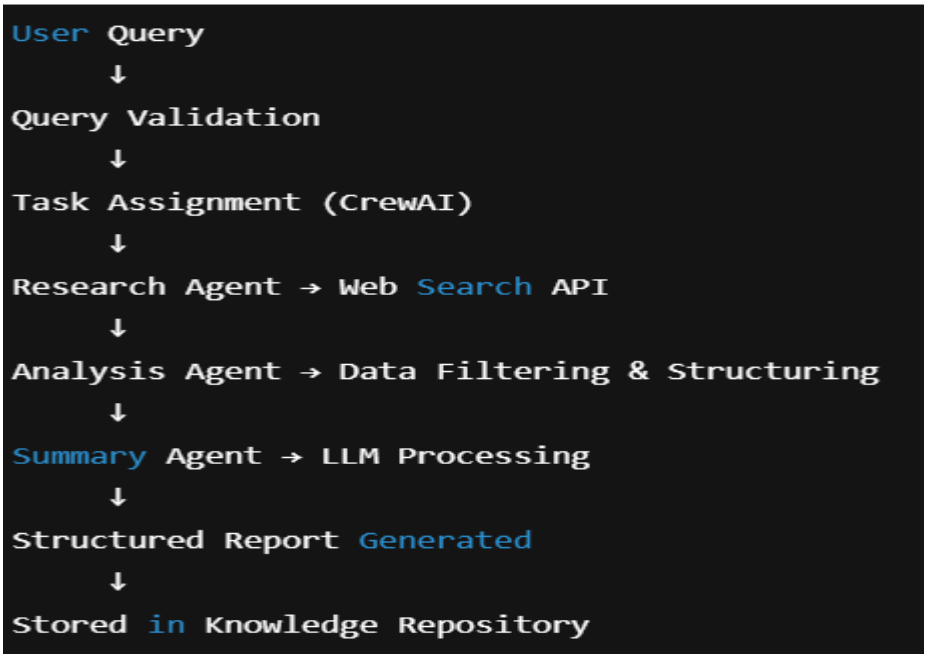
- LLM API

5. Storage Layer: Text-based knowledge repository.

This separation ensures clean architecture and maintainability.

2.2 Process Flow

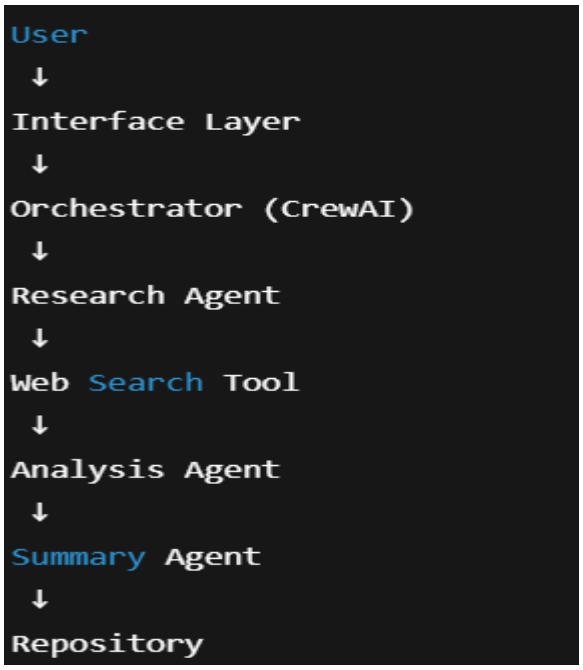
1. User submits a logistics-related query
2. Query is validated
3. CrewAI orchestrator assigns research task
4. Research Agent performs web search
5. Analysis Agent processes collected data
6. Summary Agent generates structured output
7. Output is saved in repository



2.3 Information Flow

User Query
→ Interface Layer
→ Orchestrator
→ Research Agent
→ Web Search Tool
→ Analysis Agent
→ Summary Agent
→ Knowledge Repository

The information flows sequentially to maintain dependency control.



2.4 Components Design

1. User Interface

- Input handling
- Output display

2. Orchestrator (CrewAI)

- Role assignment
- Task sequencing
- Dependency management

3. Research Agent

- Data collection
- Multi-source retrieval

4. Analysis Agent

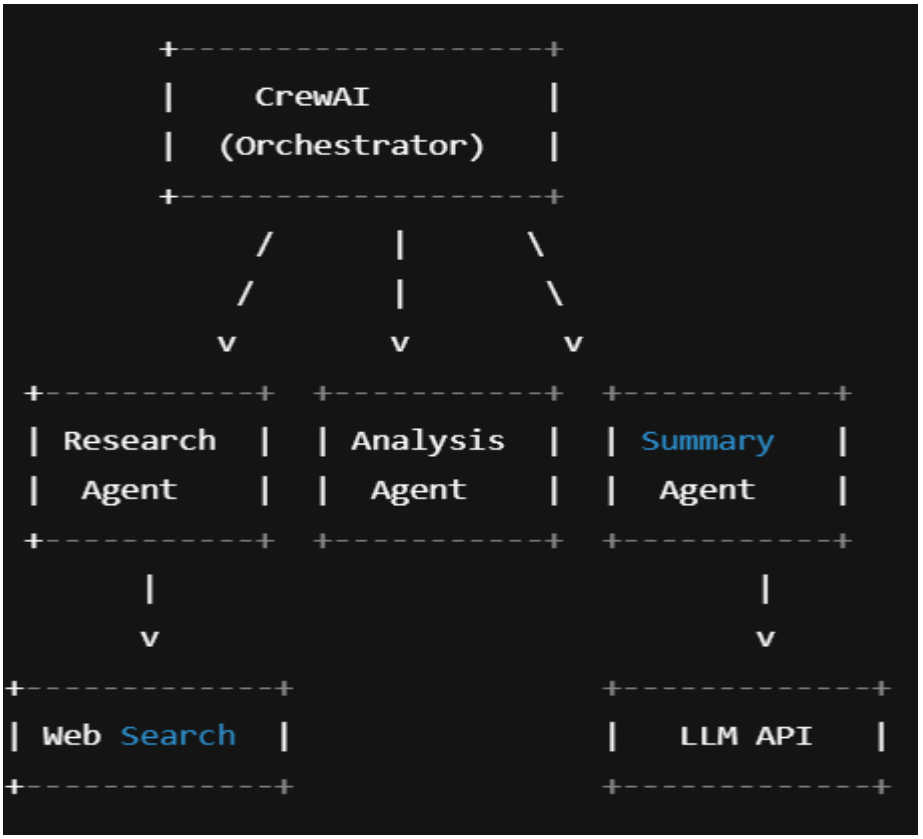
- Filtering
- Deduplication
- Structuring

5. Summary Agent

- Synthesis
- Professional formatting

6. Repository

- Persistent text storage
- Organized report management



2.5 Key Design Considerations

- Modular architecture
- Clear separation of responsibilities
- Sequential dependency handling
- Extensibility for future enhancements
- Lightweight storage mechanism

2.6 API Catalogue

1. LLM API

Purpose:

- Natural language understanding
- Reasoning
- Summarization

2. Web Search API

Purpose:

- Real-time information retrieval
- Multi-source research

Both APIs are integrated using LangChain.

3. Data Design

3.1 Data Model

The system primarily handles unstructured text data.

Data entities include:

- User Query
- Retrieved Web Data
- Processed Insights
- Final Research Report

Reports are stored as structured text files.

3.2 Data Access Mechanism

- File-based storage system
- Read and write operations handled via Python

reports

- Organized folder structure for

3.3 Data Retention Policies

The system follows simple data retention guidelines:

- Conversation history is stored for a limited duration.
- Sensitive data is not permanently stored.
- Logs may be anonymized for debugging or improvement.

If deployed in production:

- User data must comply with data protection standards.
- Logs older than a defined period should be deleted automatically.

3.4 Data Migration

Currently, the system does not require complex data migration since it handles conversational workflows.

However, if database upgrades or schema changes are needed:

- Version control for schema will be maintained.
- Backup will be taken before migration.
- Migration scripts will be tested in staging before production use.

4. Interfaces

User Interface

- Accepts textual input
- Displays structured output

External Interfaces

- LLM API
- Web Search API
- All integrations occur via LangChain.

5. State and Session Management

- The system follows a stateless execution model.
- Each query is processed independently.
- No persistent user session tracking is implemented.

Future scope includes conversational memory integration.

6. Caching

Currently:

- No caching mechanism implemented.
- Each query triggers fresh web search and LLM processing.

Future enhancement:

- Implement response caching to reduce repeated API calls.

7. Non-Functional Requirements

7.1 Security Aspects

- API keys stored securely
- No exposure of sensitive credentials
- Limited external integration
- Controlled file storage

7.2 Performance Aspects

Performance is critical in AI-based systems.

Important performance factors:

- Response time under 5–10 seconds
- Efficient orchestration logic

- Controlled token usage
- Optimized tool execution
- Proper memory management

8. References

- OpenAI. (2023). *GPT-4 Technical Report.*
  Explains the architecture and capabilities of large language models used for reasoning and generation.
- Yao, S., Zhao, J., Yu, D., et al. (2023). *ReAct: Synergizing Reasoning and Acting in Language Models.*
- This paper explains how LLMs can reason and use tools — which directly relates to agent orchestration.
- Wu, B., et al. (2023). *AutoGPT: Autonomous GPT-4 Experiment.*
  A reference for autonomous agent systems that perform multi-step tasks.
- LangChain Documentation. (2024).
  Official documentation explaining tool calling, memory, chains, and agent workflows.
- LangGraph Documentation. (2024).
  Used for orchestration and graph-based agent execution.