Project - High Level Design
On
Autonomous Retail Researcher Agent

Agentic AI

Institution Name: Medicaps University – Datagami Skill Based Course

BY

Sr no	Student Name	Enrolment Number
1	Shivani Vishwakarma	EN22CS301913
2	Shivanshu Verma	EN22CS301917
3	Shobha Goswami	EN22CS301922
4	Shubh Kapadia	EN22CS301942
5	Shubham Kumawat	EN22CS301943

Group Name: Group 06D8

Project Number: AAI-41

Industry Mentor Name: Mr Suraj Nayak

University Mentor Name: Prof. Pradeep Baniya

Academic Year:2026

Table of Contents

1.Introduction

1.1. Scope of Document

This document describes the high-level design of the Autonomous Retail Researcher Agent, a multi-agent intelligent system developed to automate retail market analysis and business insight generation.

The purpose of this document is to present the architecture, components, workflows, and interactions of the system without going into low-level implementation details. It provides a structured understanding of how the system collects data, processes it using artificial intelligence, and generates meaningful business insights for users.

This document is intended to help stakeholders understand how the system operates conceptually and how different modules interact to perform autonomous retail research tasks.

1.2. Intended Audience

This document is intended for stakeholders involved in the design, development, evaluation, and usage of the Autonomous Retail Researcher Agent system. It provides a high-level understanding of the architecture, workflow, and functional behaviour of the system.

The primary audience includes:

- Academic Evaluators and Faculty
- Developers and Technical Team
- Project Mentors and Reviewers
- Future Maintainers and Contributors
- End Users

1.3. System Overview

The Autonomous Retail Researcher Agent is an intelligent multi-agent system designed to automatically analyze retail market information and generate business insights based on user queries. The system simulates a team of human analysts by using multiple specialized AI agents that collaborate to perform research, analysis, and reporting tasks.

The system accepts a retail-related query from the user through a web interface. Once the query is submitted, it is processed by a coordinated workflow of autonomous agents. Each agent performs a specific responsibility, such as collecting market data, analyzing competitors, evaluating pricing strategies, and preparing a final report.

The system combines real-time internet data and stored historical retail data using Retrieval Augmented Generation (RAG). A large language model acts as the reasoning engine that interprets information and produces meaningful business insights.

Working Principle

- The user enters a retail analysis question in the interface.
- The Researcher Agent gathers relevant market information using web search and knowledge base memory.
- The Analyst Agent evaluates competitors and market positioning.
- The Pricing Agent analyzes pricing strategies and patterns.
- The Writer Agent generates a final structured report.
- The final insights are displayed to the user.

\

2. System Design

2.1. Application Design

The Autonomous Retail Researcher Agent is designed as a modular, layered application that separates user interaction, intelligent processing, and data management.
The application follows a multi-agent collaborative architecture where each component performs a specialized function and communicates through a centralized orchestration layer.
The design ensures scalability, maintainability, and clear separation of responsibilities.
Application Layers

a. Presentation Layer (User Interface)
This layer provides interaction between the user and the system through a web-based dashboard.
Responsibilities

- Accept retail-related queries from the user
- Display generated business insights and reports
- Trigger analysis workflow

b. Orchestration Layer (Agent Controller)
This layer coordinates the workflow of multiple AI agents and manages task execution order.
Responsibilities

- Receive user query
- Assign tasks to appropriate agents
- Maintain context flow between agents
- Control sequential decision-making process

c. Intelligent Processing Layer (AI Agents)

This is the core reasoning layer where specialized agents perform analysis collaboratively.

- Researcher Agent
  Collects relevant market data from external sources and knowledge base.
- Market Analyst Agent
  Evaluates competitor strategies and market positioning.
- Pricing Analyst Agent
  Analyzes pricing trends and discount strategies.
- Report Writer Agent
  Combines outputs and generates the final structured report.

d. Tool Layer (External Capabilities)

Provides real-world data access and memory retrieval functionality.

e. Data Layer (Knowledge Storage)

Stores structured retail data used for contextual reasoning and comparison.

2.2. Process Flow

The Autonomous Retail Researcher Agent follows a sequential multi-stage processing workflow where multiple agents collaborate to transform a user query into actionable business intelligence.

The system operates as an automated analysis pipeline, with each stage refining the output of the previous stage.

Step-by-Step Execution Flow

a. User Query Submission

The user enters a retail-related question through the web interface.The query may involve market trends, competitor analysis, pricing strategy, or demand prediction.

The system validates the input and forwards it to the orchestration module.

b. Task Initialization

The orchestration layer creates a sequence of tasks and assigns them to specialized agents. Context linking is established so that each agent can use the previous agent's output.

c. Data Collection (Researcher Agent)

The Researcher Agent gathers relevant information from:

- Web search sources (real-time market data)
- Knowledge base (historical retail dataset)

The agent filters and extracts only relevant information related to the query.

Output: Raw research findings

d. Market Analysis (Analyst Agent)

The Analyst Agent interprets research findings and performs:

- Competitor comparison
- Market positioning analysis
- Strength and weakness identification

e. Pricing Evaluation (Price Analyst Agent)

The Pricing Agent evaluates pricing patterns using analysed data:

- Discount strategies
- Premium vs budget segmentation
- Customer pricing behaviour

Output: Pricing intelligence insights

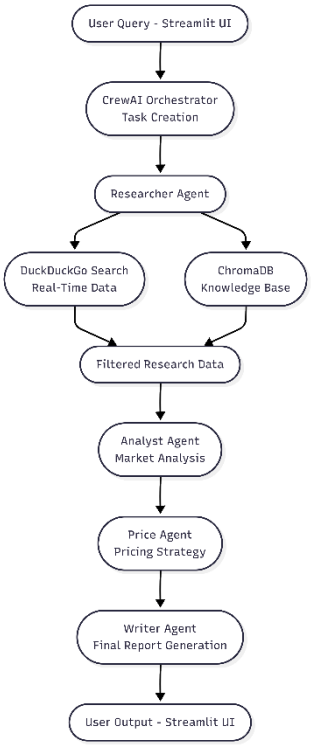
f. Report Generation (Writer Agent)

The Writer Agent consolidates all previous outputs and produces a final report:

- Summarized findings
- Business recommendations
- Decision-ready insights

g. Response Delivery

The final report is returned to the user interface and displayed in readable format.



Process Flow Diagram

2.3. Information Flow

The Information Flow describes how data moves between system components during execution . Unlike the process flow, which focuses on execution steps, the information flow focuses on the transformation and transfer of data from input to output.
The system follows a structured data transformation pipeline where each agent refines the information received from the previous stage.
Data Movement Description
a. User Input
The system receives a natural language retail query from the user through the interface.
This input is forwarded to the orchestration layer.

b. Task Context Generation
The orchestration module converts the query into a structured task context.
This context is passed to the Researcher Agent as the initial working information.

c. Data Acquisition (Researcher Agent)
The Researcher Agent gathers information from:

- Real-time web sources
- Knowledge base memory

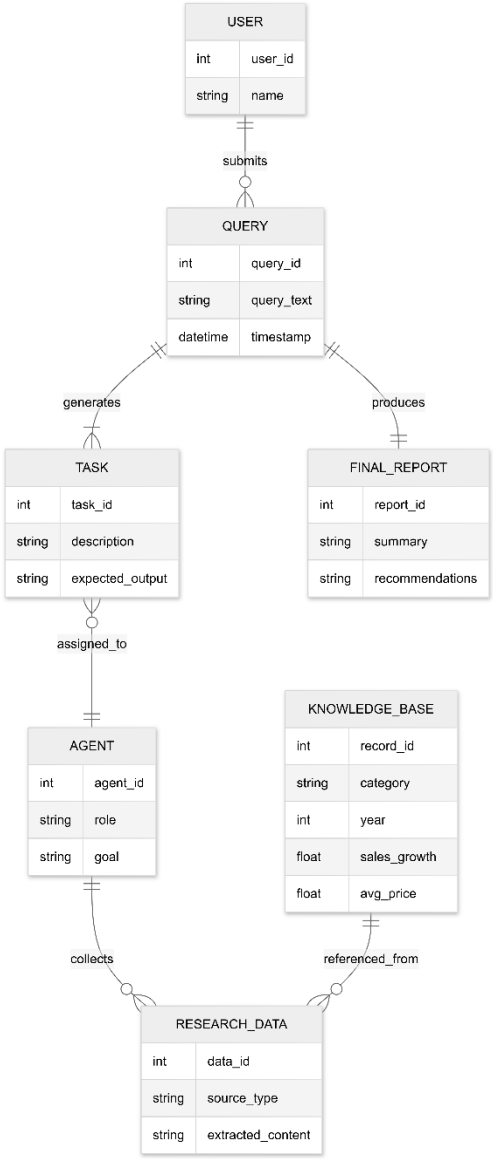
It filters irrelevant information and produces a focused dataset.

d. Market Interpretation (Analyst Agent)
The Analyst Agent interprets the research findings and identifies meaningful market patterns.

e. Pricing Intelligence (Price Agent)
The Price Agent evaluates pricing patterns and purchasing behavior.
Input: Market analysis insights
Output: Pricing intelligence data

f. Report Consolidation (Writer Agent)
The Writer Agent combines all collected knowledge into a final response.
Input: Pricing insights + analysis
Output: Final structured report

g. Response Delivery
The final report is sent back to the presentation layer and displayed to the user.
Data Type: Human-readable business intelligence output



ER Diagram

2.4.Components Design

The Autonomous Retail Researcher Agent is composed of multiple interconnected components that collectively perform autonomous retail market analysis. Each component has a clearly defined responsibility and communicates with other components through structured data exchange.

The system follows a modular architecture, allowing independent development, maintenance, and scalability of components.

a. User Interface Component

Purpose: Handles interaction between the user and the system.

Responsibilities

- Accept retail-related queries from the user

- Display generated insights and reports

b. Orchestration Component (Agent Controller)
Purpose: Manages workflow execution and agent coordination.
Responsibilities

- Convert user query into tasks
- Assign tasks to specialized agents
- Maintain context passing between agents
- Control execution order

This component acts as the central coordinator of the system.

c. Researcher Agent Component
Purpose: Collect relevant market information.
Responsibilities

- Retrieve real-time data from web search tools
- Access historical knowledge base
- Filter and extract relevant information

d. Analyst Agent Component
Purpose: Perform competitor and market analysis.
Responsibilities

- Interpret research data
- Identify market positioning
- Compare competitor strategies

e. Pricing Agent Component
Purpose: Evaluate pricing strategies.
Responsibilities

- Analyze price trends
- Identify discount patterns
- Determine affordability segments

f. Report Writer Component
Purpose: Generate final business report.
Responsibilities

- Combine outputs from all agents
- Produce clear structured insights
- Answer user query directly

e. Tool Component

Purpose: Provide external capabilities to agents.

Subcomponents

- Web Search Tool – Retrieves real-time retail information
- Knowledge Base Tool – Retrieves stored historical data

h. Knowledge Base Component

Purpose: Store and retrieve historical retail data.

Responsibilities

- Maintain retail dataset
- Provide semantic search capability
- Support contextual reasoning

2.5. Key Design Considerations

The design of the Autonomous Retail Researcher Agent focuses on reliability, scalability, and intelligent decision-making. Several architectural and functional factors were considered to ensure the system produces accurate and relevant retail insights while remaining maintainable and extensible.

a. Modular Architecture

The system is divided into independent components such as agents, tools, and data storage modules. This allows easy maintenance, debugging, and future expansion without affecting the entire system. New agents or tools can be added without redesigning the application.

b. Multi-Agent Collaboration

The system uses specialized agents instead of a single monolithic model. Complex business analysis requires different types of reasoning.

c. Context Preservation

Each agent receives the output of the previous agent as input. Ensures decisions are based on prior analysis rather than isolated responses. Produces coherent and logically connected results.

d. Retrieval Augmented Generation (RAG)

The system integrates a knowledge base with real-time search. Language models alone may generate outdated or generic answers. Improves factual accuracy and reduces hallucinations.

2.6. API Catalogue

The Autonomous Retail Researcher Agent integrates multiple internal and external APIs to perform autonomous retail market analysis. These APIs enable communication between the user interface, multi-agent orchestration system, external data sources, and the knowledge base.

The system primarily uses the following APIs and service interfaces:

a. Groq LLM API

The Groq API provides access to the Large Language Model used by all agents for:

- Market interpretation
- Competitor analysis
- Pricing evaluation
- Report generation

b. CrewAI Orchestration API

CrewAI manages:

- Task creation
- Context passing between agents
- Sequential execution of agent roles
- Task(description, expected output, context)

c. DuckDuckGo Search API

Provides up-to-date market trends, competitor information, and industry insights.

- Search query string
- List of relevant search results

d. ChromaDB Vector Database API

Stores retail dataset embeddings and performs similarity search using vector comparison.

- Query embedding
- Relevant historical retail records

e. Streamlit Interface API

User interaction and result display

- User input collection
- Triggering analysis workflow
- Displaying final output

3. Data Design

3.1. Data Model

The Autonomous Retail Researcher Agent uses a hybrid runtime data model combining:

- Natural language inputs
- Agent task objects (CrewAI)
- Tool outputs (Search + Vector DB)
- Embedding vectors
- Context-passing objects
- Final generated report

The system does NOT use a traditional relational database schema.
Instead, it operates on structured Python objects and vector-based storage.

a. User Query Model
This represents the raw input received from the Streamlit interface.
Structure
- query_id (auto-generated runtime ID)
- query_text (string)
- timestamp

b. Task Object Model (CrewAI)
Each agent receives a Task object.
Attributes
- description (instruction text)
- expected_output (format constraint)
- agent_reference
- context (previous task outputs)

This object is created and managed by CrewAI.

c. Agent Output Model
Each agent produces structured text output.
Researcher Output
- web_data (real-time results)
- kb_data (retrieved historical data)
- filtered_insights

Analyst Output

- competitor_analysis
- market_positioning

Pricing Agent Output

- pricing_patterns
- discount_strategy

Writer Output

- final_summary
- recommendations

These outputs are passed sequentially via task context.

d. Web Search Data Model
Data retrieved using DuckDuckGo search API.
Structure
- search_query
- search_results[]
  - title
  - snippet
  - source_link

This data is temporary .

e. Knowledge Base Data Model (ChromaDB)
Stored in ChromaDB
Retail Record Schema (CSV-based)
- category
- year
- sales_growth
- avg_price
- embedding_vector

f. LLM Processing Model
All reasoning is handled through Groq API.
LLM Input Structure
- Prompt
- Context from previous agent
- Tool output (if any)
  LLM Output
- Generated structured analysis text

g. Final Report Model
Final response returned to Streamlit UI.
Structure

- executive_summary
- key_findings

- pricing_insights
- business_recommendation

3.2 . Data Access Mechanism

The Autonomous Retail Researcher Agent retrieves and processes data from multiple sources using a controlled access mechanism. The system does not rely on direct database querying alone; instead, it combines tool-based retrieval, semantic search, and contextual transfer between agents. The goal of the data access mechanism is to ensure that each agent receives only relevant information while maintaining accuracy and efficiency.

Data Sources

The system accesses data from three primary sources:

- User Input – Natural language query provided by the user
- External Web Sources – Real-time retail information
- Knowledge Base – Stored historical retail dataset

Data Retrieval Method

a. Tool-Based Retrieval

The Researcher Agent accesses real-time information through a web search tool.

Process

- Accept search query
- Fetch relevant retail information
- Filter unnecessary content

b. Semantic Retrieval (Vector Search)

The system uses vector embeddings to retrieve relevant historical data from the knowledge base.

Process

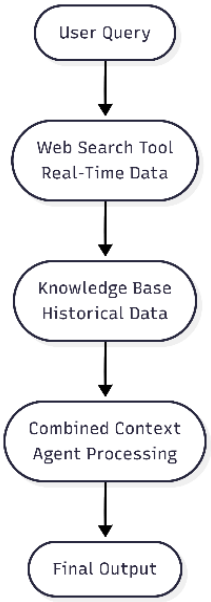
- Convert query into embedding
- Compare similarity with stored vectors
- Retrieve closest matching records

c. Context Transfer Between Agents

Instead of storing intermediate results in a database, the system passes information directly between agents.

Process

- Previous agent output becomes next agent input
- Maintains workflow continuity



3.3 . Data Retention Policies

The Autonomous Retail Researcher Agent follows a minimal data retention strategy. The system primarily processes data in real time and stores only essential information required for knowledge base functionality. Most data exists only during execution and is automatically discarded after processing.

The retention approach ensures privacy, efficient storage usage, and controlled persistence of analytical resources.

a. Session-Based Data Retention

The following data is retained only during system execution and automatically deleted after the session ends:

- User Queries submitted through the interface
- Intermediate Agent Outputs generated during multi-agent processing
- Web Search Results retrieved from external sources

These data elements:

- Exist only in temporary system memory
- Are not permanently stored in a database
- Are cleared once the task workflow completes

This ensures user privacy and prevents unnecessary storage of sensitive business questions.

b. Persistent Data Retention

Certain data components are stored persistently to support analytical functionality:

Retail Knowledge Base Dataset

- Stored in CSV format
- Retained until manually updated or replaced
- Used for historical market comparison

Vector Embeddings (ChromaDB)

- Generated using sentence transformer models
- Stored in vector database
- Retained as long as the dataset remains active
- Deleted when the knowledge base is refreshed or reset

Persistent storage is limited strictly to analytical reference data.

c. External API Data Handling

Data processed through external services such as:

- Groq LLM API
- DuckDuckGo Search API

is used only during inference and is not stored by the system beyond execution.

d. Security and Privacy Compliance

- No personally identifiable information (PII) is intentionally collected
- User queries are not logged permanently
- No user credentials are stored
- Knowledge base access is restricted to authorized administrators

3.4. Data Migration

Data migration in the Autonomous Retail Researcher Agent refers to the process of transferring retail datasets, vector embeddings, and knowledge base records from one storage environment to another when upgrading, modifying, or deploying the system. Since the system primarily uses a CSV-based retail dataset and a vector database (ChromaDB), migration is focused on structured dataset updates and embedding regeneration.

a. Retail Dataset Migration

Migration Process

- Backup existing dataset.
- Replace or append updated retail records.
- Validate data format and schema consistency.
- Reload dataset into system.
- Regenerate embeddings if necessary.

b. Vector Database Migration (ChromaDB)

Migration Steps

- Backup existing vector store directory.
- Transfer storage folder to new environment.
- Reinitialize ChromaDB client.
- Validate embedding consistency.

4.Interfaces

The Autonomous Retail Researcher Agent interacts with users, external services, and internal components through well-defined interfaces. These interfaces enable communication between the presentation layer, orchestration layer, AI agents, tools, and the knowledge base.

The system primarily uses programmatic interfaces and API-based integrations rather than traditional REST endpoints.

a. User Interface (UI)

Type-Graphical Web Interface

Technology-Streamlit (Python-based web framework)

Description

The user interface allows users to:

- Enter retail-related queries
- Trigger the multi-agent analysis workflow
- View structured business intelligence reports

b. Agent Orchestration Interface

Type-Internal Programmatic Interface

Technology-CrewAI Framework

Description

This interface manages:

- Task creation
- Agent execution order
- Context passing between agents

It connects the user query with agent workflow execution.

c. Large Language Model (LLM) Interface

Type-External API Interface

Technology-Groq LLM API

Description- Provides natural language reasoning capability for:

- Market analysis
- Pricing evaluation
- Report generation

d. Web Search Interface

Type-External Search API

Technology-DuckDuckGo Search API

Description-Allows the Researcher Agent to fetch real-time retail data from online sources.

e. Knowledge Base Interface

Type-Vector Database Interface

Technology-ChromaDB

Description-Enables semantic search within historical retail dataset using vector embeddings.

f. Embedding Interface

Type-Internal Machine Learning Interface

Technology-Sentence Transformers

Description-Converts text data (queries and dataset entries) into numerical embeddings for semantic search.

5. Session and State Management

The Autonomous Retail Researcher Agent follows a session-based execution model. The system does not maintain long-term conversational memory by default. Instead, it processes each user query independently within a single execution lifecycle.

State is maintained temporarily during workflow execution and is cleared once the analysis is complete.

a. Session Management

A session begins when a user submits a retail query through the Streamlit interface and ends when the final report is generated and displayed.

Session Characteristics

- Each query is treated as an independent session.
- No cross-session memory is maintained by default.
- Session data exists only in application runtime memory.

b. Application State Management

During a session, the system maintains temporary state information required for multi-agent collaboration.

State Components

- User Query-Stored temporarily to initialize tasks.
- Task Context-Maintained within CrewAI to pass outputs between agents.
- Agent Outputs-Sequential outputs are stored in memory and forwarded to the next agent.
- Final Report-Generated and returned to the UI before session termination.

This data remains constant across sessions and is not modified during normal query execution.

d. Session Handling in Streamlit

Streamlit manages UI state temporarily using session variables, which:

- Hold current query
- Store generated response
- Reset upon application refresh

6.Caching

The Autonomous Retail Researcher Agent uses selective caching to improve performance and reduce redundant computation. Since the system performs real-time analysis

using external APIs and vector search, caching is implemented only where appropriate to balance performance, freshness of data, and accuracy.

a. Dataset Caching

The retail dataset (CSV file) is loaded once during application startup.

- Loaded into memory at runtime.
- Not reloaded for every user query.
- Refreshed only when dataset is manually updated.

b. Vector Database Caching

ChromaDB vector store remains initialized during application runtime.

- Embeddings are generated once during dataset indexing.
- Vector collection is reused across multiple queries.
- No re-indexing unless dataset changes.

c. Streamlit Session Caching

Streamlit session state temporarily stores:

- Current query
- Generated response

7.Non Functional Requirements

7.1. Security Aspects

Security is a critical non-functional requirement of the AI-Based Autonomous Retail Researcher Agent. The system is designed to ensure secure handling of user input, controlled access to external APIs, and protection of stored retail data.
The security model focuses on confidentiality, integrity, controlled access, and safe API usage.

a. Data Confidentiality

- User queries are processed in real time and not permanently stored.
- Sensitive business queries are not logged in persistent storage.
- API keys (e.g., Groq API key) are stored securely using environment variables (.env file).

b. Secure API Communication

- External API calls (Groq LLM, DuckDuckGo Search) are made using secure HTTPS protocols.
- API keys are excluded from version control repositories.

c. Access Control

- No public database exposure.
- Application runs locally or within a controlled deployment environment.

7.1. Performance Aspects

Performance is a critical non-functional requirement of the Autonomous Retail Researcher Agent. The system must generate accurate retail insights within acceptable response times while efficiently managing computational and API resources.
The performance design focuses on response time, scalability, resource utilization, and optimization of AI processing.

a. Response Time

The system should generate a complete retail analysis report within an acceptable time range (typically 5–15 seconds depending on API latency).
Factors Affecting Response Time

- LLM inference time (Groq API)
- Web search latency
- Vector database retrieval time
- Network connectivity

b. Throughput

The system should handle multiple independent user queries sequentially without degradation in performance.
Design Consideration

- Stateless architecture
- No heavy database transactions
- Lightweight UI framework (Streamlit)

c. Resource Utilization

CPU and Memory

- Embedding model loaded once at startup
- Vector database reused during runtime
- Temporary agent outputs cleared after execution

API Usage

- Web search invoked only by Researcher Agent
- LLM calls controlled using limited iterations

d. Caching Optimization

- Retail dataset loaded once per application runtime

- Vector embeddings generated once
- No repeated dataset indexing
- No unnecessary API calls

8.References

[1] CrewAI, "CrewAI Documentation," [Online]. Available: https://docs.crewai.com. [Accessed: 19-Feb-2026].

[2] Groq, "Groq LLM API Documentation," [Online]. Available: https://console.groq.com/docs. [Accessed: 19-Feb-2026].

[3] DuckDuckGo, "duckduckgo-search Python Package," [Online]. Available: https://pypi.org/project/duckduckgo-search/. [Accessed: 19-Feb-2026].

[4] Chroma, "ChromaDB Documentation," [Online]. Available: https://docs.trychroma.com. [Accessed: 19-Feb-2026].

[5] Streamlit, "Streamlit Documentation," [Online]. Available: https://docs.streamlit.io. [Accessed: 19-Feb-2026].

[6] Python Software Foundation, "Python 3 Documentation," [Online]. Available: https://docs.python.org/3/. [Accessed: 19-Feb-2026].