Agentic AI Sales Agent

Prepared by: Saksham Agrawal

Organization: Greyamp Consulting

Internship Period: 1st Oct 2025 to 30th Nov 2025

Executive Summary

In this Agentic AI Sales Agent internship project, a multi-agent AI pipeline is designed and built to automate B2B lead generation for Greyamp Consulting. The end-to-end system autonomously generates Ideal Customer Profiles (ICPs), scouts real-time news for relevant AI-related signals, and enriches and filters leads using LLMs and APIs. Initially implemented in Python, the validated pipeline has been ported into the Langflow visual framework for modularity. Key achievements include automated ICP creation, news-based lead discovery, lead enrichment via the Tavily API, and generation of a high-quality, enriched prospect list. The report details each project phase including Strategist, ICP Generator, Scout, Analyst, Outreach, technical design, results, challenges faced and solutions, and future directions. Overall, the project showcases how agentic AI can transform sales outreach into a data-driven, scalable process.

Introduction and Business Problem

In the B2B sector, identifying potential customers often relies on manual research and static lead lists, which are time-consuming, unscalable, and unable to capture dynamic "buying signals" such as new AI initiatives or key technology hires. These real-time signals are strong indicators of a company's readiness to adopt new solutions, yet traditional methods struggle to track them effectively. As markets evolve rapidly, relying on outdated lists limits the ability to discover emerging opportunities.

Greyamp Consulting, with its focus on AI and data-driven services, needed an automated solution that continually scans public sources for relevant insights and filters companies aligning with its target profile. The key problem, therefore, is to leverage AI to automate lead discovery, improve accuracy, and deliver timely, high-quality prospects to the sales team. A system that eliminates manual research, reduces human error, and systematically surfaces the right companies at the right moment would significantly accelerate Greyamp's sales pipeline and outreach effectiveness.

Objective

The objective of this internship project was to design and implement an end-to-end multi-agent AI pipeline, called the Agentic AI Sales Agent, to automate Greyamp's sales lead-generation process. The system aims to replace manual prospecting with an intelligence-driven workflow capable of identifying, qualifying, and preparing high-potential leads. The pipeline focuses on automatically generating Ideal Customer Profiles (ICPs), monitoring public news sources for real-time AI-related signals, and enriching discovered leads with essential information such as company location, leadership details, industry alignment, and contextual signals. It further lays the foundation for personalized outreach by ensuring that each lead carries the necessary business context for tailored communication. Built using a modular, multi-agent architecture and supported by LangFlow for visual orchestration, the solution demonstrates how an AI-driven approach can consistently produce structured, enriched, and relevant leads aligned with Greyamp's target market. Ultimately, the project validates the effectiveness of an agentic system in transforming traditional B2B prospecting into a scalable, automated process.

Technical Architecture

The Agentic AI Sales Agent is built on a modular multi-agent architecture designed for scalability, clear data flow, and flexible orchestration. Each agent—Strategist, ICP Generator, Scout, Analyst, and the upcoming Outreach module—functions as an independent component exchanging structured JSON between stages. This separation enhances maintainability and makes it easy to debug, extend, or upgrade individual modules without impacting the entire pipeline.

The initial implementation was done in Python to maximize control and flexibility. It uses OpenAI models (GPT-4 and GPT-4o-mini) for strategic reasoning, ICP generation, article parsing, and entity extraction. A tiered LLM strategy ensures cost-efficiency: lighter models handle fast filtering tasks, while more powerful models perform deeper validation. The system is supported by external data sources such as GNews and NewsAPI keys for real-time signals, and the Tavily API for company enrichment (CEO, location, industry).

As the project matures, the pipeline is being migrated into LangFlow to enable visual orchestration, type-safe data passing, and reusable building blocks accessible to non-developers. LangFlow recreates the Python logic through components like File, Prompt Template, Language Model, Type Convert, Loop, API Request, and Save File. The data progresses through a clear sequence—ICPs → raw articles → enriched leads → outreach-ready summaries—with intermediate data stored temporarily. Future enhancements include SQLite-based persistence, scheduled automation, and easy integration of new agents or additional data sources.

Multi-Agent Approach

The solution was structured as a multi-agent pipeline, where each agent performs a specific role. This approach makes the system modular and allows each component to be developed and tested independently. The main agents and their roles are:

Strategist Agent

Python approach:

- Built `phase1_strategist.py` to read `greyamp_context.txt` and call OpenAI (gpt-4o-mini) for a structured company summary.
- Implemented prompt templates with JSON examples to force machine-readable outputs and saved `greyamp_summary.json`.
- Validated outputs with unit checks and manual review to ensure consistent fields for downstream use.
- Automated reruns via a simple script wrapper so the strategy summary could be regenerated after updates.

Langflow approach:

- Designed `strategist_flow` using File → Prompt Template → Language Model → Save File nodes to produce the same JSON summary.
- Configured Prompt Template with explicit JSON schema examples and used Structured Output or Advanced Parser to enforce format.
- Added Type Convert where necessary to reconcile Message/Data types and ensure the Save File node receives a clean JSON payload.
- Enabled rapid iteration in Playground to tweak prompts visually and hand off the flow to non-developers.

ICP Generator Agent

Python approach:

- Implemented `phase2_icp_generator.py` that consumes strategist summary and produces `icp_profiles.json` with fields (`icp_id`, `industry`, `location_cities`, `ai_buying_signals`, `search_query_suggestion`).
- Engineered robust parsing logic and JSON schema validation (e.g., `jsonschema`) to catch malformed LLM outputs.
- Added small heuristics (city normalization, keyword deduplication) to standardize ICP entries.
- Versioned ICP outputs so changes to prompts or context could be rolled back and compared.

Langflow approach:

- Built `icp_generator_flow` with Prompt Template → Structured Output → Save File to produce validated ICP JSON directly from LLM output.
- Enforced schema in Structured Output and provided example responses in the prompt to reduce JSON formatting errors.
- Connected Playground preview + Save File so outputs could be inspected and downloaded immediately.
- Added an optional Python Interpreter node to perform final sanitization (city normalization, list flattening) if complex rules were needed.

Scout Agent

Python approach:

- Built `phase3_scout.py` to loop `icp_profiles.json`, construct encoded queries (e.g., `"{industry} AND AI"`) and call GNews + NewsAPI.
- Implemented robust response parsing and a URL-based deduplication system to produce `raw_leads.json`.
- Handled rate limits with retry/backoff, pagination, and basic error handling to ensure reliable harvesting.
- Saved normalized article objects (title, url, source, publishedAt, snippet, matched_icp) for downstream enrichment.

Langflow approach 1:

- Designed `scout_flow` using Batch Run (per-ICP) → Prompt Template (query builder) → Type Convert → API Request → Type Convert → Structured Output → Save File.
- Resolved type mismatches using Type Convert nodes and binding Batch Run outputs to API Request URL fields (globe binding).
- Used Structured Output schemas to normalize heterogeneous API JSON into the common article format.
- Added logging nodes and small Python snippets in Python Interpreter nodes for deduplication and retry logic not easily expressed as pure nodes.

Langflow approach 2:

- Built a multi-step flow using File → Prompt Template → Python Interpreter → Loop → API Request → Save File to replicate the Python scouting pipeline visually.
- Used Prompt Template + Python Interpreter to transform ICP JSON into a row-wise DataFrame, ensuring each ICP is processed individually in the Loop node.
- Integrated multiple Type Convert nodes to manage Message/Data/DataFrame conversions, allowing ICP fields to flow cleanly into the second Prompt Template and API Request parameters.

- Configured API Request with dynamic query parameters produced per ICP, enabling automated article retrieval and saving consolidated raw leads via Save File.

Analyst Agent

Python approach:

- Implemented `phase4_analyst.py` to parse `raw_leads.json`, extract company mentions using gpt-4o-mini, and list candidate companies.
- Integrated Tavily API calls to enrich company records (CEO, HQ, website, India presence) with retries and fallback queries.
- Applied strict LLM filtering (GPT-4/GPT-4o) and rule-based checks to produce `qualified_leads.json` with justification fields (why qualified).
- Logged & Audited decisions for sample sets to iterate thresholds and prompt wording, achieving high-quality lead generation.

Langflow approach (planned):

- Design an Analyst flow: Structured Output (article parsing) → Batch Run (per-article extraction) → API Request / Python node (Tavily enrichment) → Structured Output (final schema) → Save File.
- Use Python Interpreter node for iterative enrichment loops (multiple Tavily queries per company) and to implement robust retry/backoff inside Langflow orchestration.
- Wire a higher-capacity Language Model node for final qualification prompts and feed structured evidence fields into the prompt to ensure deterministic filtering.
- Add monitoring + sample auditing nodes (Save File of intermediate states) so human reviewers can validate and tune prompts before fully trusting automated outputs.

Outreach Agent

Langflow approach (planned):

- Design an Outreach flow: Qualified Leads → RAG Retriever (vector DB of Greyamp case studies) → Prompt Template (email writer) → Language Model → Save File / UI.
- Implement RAG by embedding Greyamp marketing assets into a vector store (e.g., FAISS) and wiring a Retriever node to feed contextual snippets into the prompt.
- Add options in the flow to select tone/template and to surface drafts for human-in-the-loop editing before sending; include Save File + export-to-CSV nodes for CRM ingestion.
- Instrument the flow with logging and a feedback loop to capture reply/outcome metrics so prompts and scoring can be iteratively improved.

Challenges and Solutions

API Request Errors: Early news-API calls failed or returned empty results due to overly narrow queries. Broadening search terms (industry + “AI”) and refining prompt logic improved reliability and ensured the Scout consistently retrieved relevant articles.

Zero Initial Leads: Initial pipeline runs produced almost no leads. Expanding the Scout’s query scope and shifting strict filtering to the Analyst allowed more raw articles to pass through, giving the system enough data for meaningful qualification.

Duplicate Articles: Multiple APIs often returned identical articles. A URL-based deduplication step in the Scout removed repeated items, ensuring each signal was processed once and preventing redundant enrichment work by the Analyst.

Missing Lead Information: Many articles lacked CEO names, locations, or company websites. The Analyst used Tavily search to fill missing fields, enriching each lead with reliable metadata and producing complete, outreach-ready lead profiles.

LangFlow Migration Issues: Migrating Python logic into LangFlow introduced issues with node configuration, input/output types, and API handling. These were resolved through iterative debugging, restructuring flows, and adapting components to LangFlow’s data-passing conventions.

Achievements

A complete multi-agent lead-generation pipeline was designed and implemented in Python, covering Strategist, ICP Generator, Scout, and Analyst agents. The Strategist produced structured Ideal Customer Profiles aligned with Greyamp’s AI-focused market, while the ICP Generator standardized these into a reliable JSON format for downstream use.

The Scout Agent successfully integrated GNews and NewsAPI to gather AI-related market signals, applying keyword-based queries and deduplication logic to ensure clean, unique inputs. The Analyst Agent extracted company names from articles using LLMs and enriched each lead using Tavily, retrieving details such as industry, CEO, and India presence. This resulted in an initial set of high-quality, AI-focused leads demonstrating full end-to-end pipeline functionality.

Core components were later migrated into LangFlow, creating modular, visual flows that simplify maintenance and future enhancements. Along the way, strong proficiency was developed in prompt engineering, LLM orchestration, API integration, and modular pipeline design. Comprehensive documentation was prepared to support future handoff and expansion.

Business Impact

This project directly addresses Greyamp's need for scalable and intelligent B2B prospecting. By automating lead discovery, enrichment, and qualification, the system significantly accelerates sales research and captures real-time market signals that traditional manual processes often overlook. The pipeline continuously monitors news sources and updates leads dynamically, enabling the company to act quickly when potential clients announce AI initiatives, funding, or strategic moves. Automated ICP generation further ensures that the search space remains aligned with Greyamp's target segments as the market evolves.

Beyond efficiency gains, the enriched lead data directly strengthens outreach quality. Instead of relying on generic or static lists, sales teams receive high-context, AI-filtered leads with company details, signals, and relevance already validated. This shifts effort from data gathering to meaningful engagement, allowing outreach to be more personalized and timely—ultimately improving response and conversion rates. In effect, the 'AI Sales Agent' amplifies Greyamp's sales capacity, enabling the team to engage a larger, more accurate, and more opportunity-ready pipeline with minimal manual input.

Learnings

Multi-Agent Architecture: Breaking down the problem into specialized agents proved highly effective. Each agent could be optimized for its task (e.g. one for brainstorming ICPs, another for data retrieval), and errors could be isolated. This modular approach is recommended for complex AI workflows.

Prompt Engineering: Crafting the LLM prompts for each agent required careful attention. Providing clear instructions, examples, and output format constraints greatly improved reliability. For instance, specifying JSON schemas in prompts reduced parsing errors.

Tiered Model Strategy: Using a smaller LLM (GPT-4o-mini) for initial quick tasks (like name extraction) and the full GPT-4 for final analysis saved time and cost. This balance showed that not every step needs the most powerful model.

Data Quality Matters: The quality of leads depends on the quality of input signals. Learning to formulate queries that are neither too broad nor too narrow was a process. Multi-source aggregation (GNews + NewsAPI) enriched coverage, reinforcing that diverse inputs yield better outcomes.

Tooling with LangFlow: Adopting LangFlow as the orchestrator taught me the benefits and trade-offs of low-code AI pipelines. I gained skills in constructing visual flows, using custom nodes, and deploying an agentic system without writing all code from scratch. It also highlighted the importance of planning data flows clearly.

Handling Unstructured Information: Extracting company data from free-form articles and then augmenting it via search was non-trivial. I learned techniques for using LLMs as a form of information extraction, combined with traditional API lookups for verification.

Business Context Awareness: Understanding the company's target market was essential for effective ICPs. The project showed that technical solutions must align with business strategy, and close mentor collaboration ensured the work stayed relevant.

Potential Next Steps

Outreach Agent: Implement the final module in LangFlow to auto-generate personalized outreach emails using LLMs, incorporating each lead's signal context.

RAG Implementation: Integrate a retrieval system to pull relevant Greyamp case studies or whitepapers, enabling more evidence-based, tailored outreach messages.

Database Integration: Store all leads and their statuses in a lightweight database (e.g., SQLite) to avoid duplicate processing and maintain historical context.

Scheduling & Monitoring: Automating the scheduled pipeline execution and add logging and error alerts to ensure reliability and consistent lead refresh.

Expand Signal Sources: Extend the pipeline to include additional inputs (LinkedIn signals, press releases, tech blogs) for wider and more real-time coverage.

Refine Criteria and Scoring: Develop a scoring mechanism to rank leads by priority (e.g. company size, recency of signal, estimated deal size). This would help the sales team focus on the highest-potential contacts.

Conclusion

The Agentic AI Sales Agent project successfully demonstrated how AI agents can automate end-to-end B2B lead generation. A multi-agent system is designed and built that autonomously generates ICPs, scours the web for opportunities, and filters high-quality leads. Key outcomes include a functioning Langflow pipeline and an enriched leads database, showing tangible progress toward Greyamp's goal of scalable, signal-driven prospecting. The challenges encountered, from API issues to data gaps, provided valuable lessons in resilience and iterative development. Overall, this workflow lays a strong foundation for future expansion (analyst automation, outreach, etc.) and represents a significant step toward modernizing company's sales process with AI.