

Republic of Tunisia
Ministry of Higher Education
and Scientific Research
University of Sfax

National School of Electronics
and Telecommunications of Sfax



Diploma in :
Data Engineering
and Decision Making Systems
Code : PFA-IDSD-2026-25

Academic Year: 2025-2026

Final Year Project Report

Presented at
**National School of Electronics
and Telecommunications of Sfax**

**National Engineering Diploma in :
Data Engineering And Decision Making Systems**

by
**Rami AYOUNI
Emna KAANICHE**

Generative Engine Optimization

Supervisor: Mrs. Sonda AMMAR

Acknowledgments

We would like to express our sincere gratitude to all the individuals who, directly or indirectly, contributed to the success of this final year project and the writing of this report.

We extend our deepest thanks to our supervisor, **Madam Sonda AMMAR**, for her availability, insightful guidance, and continuous support throughout this work. Her trust, encouragement, and generous sharing of expertise have been invaluable assets in the realisation of this project. Her mentorship allowed us to progress with confidence, grow our technical skills, and fully flourish in our assigned missions.

We would also like to address a word of recognition to ourselves as a team for our commitment, perseverance, and collaborative spirit throughout this adventure.

This experience was an opportunity for us to explore enriching perspectives in the field of artificial intelligence applied to digital marketing and information retrieval, and we are enthusiastic about continuing on this promising path.

Table des matières

General Introduction	1
1 Project Context and State of the Art	2
Introduction	2
1.1 Context and Motivation	2
1.2 Problem Statement	3
1.3 Project Objectives and Scope	4
1.4 Literature Review and State of the Art	5
1.4.1 Search Engine Optimization	5
1.4.2 Generative AI in Information Retrieval and the Emergence of GEO . . .	5
1.4.3 Large Language Models and Entity Representation	6
1.4.4 Agentic AI Systems and Multi-Agent Architectures	7
1.4.5 Related Work and Research Gaps	9
Conclusion	11
2 System Architecture and Results	12
Introduction	12
2.1 Research Approach and Design Philosophy	12
2.2 System Overview	13
2.2.1 Comparative Analysis	13

2.2.2	Final System Overview	14
2.3	Dynamic Model Registry	15
2.4	Pipeline Agents	16
2.4.1	Agent 0: Intent Discovery and Prompt Generation	16
2.4.2	Agent 1: LLM Querying and Entity Extraction	17
2.4.3	Agent 2: Web Presence Research	20
2.4.4	Agent 3: Data Consolidation	23
2.5	Results	24
2.5.1	Generation and Extraction Results	25
2.5.2	Enrichment and Consolidation Results	25
2.5.3	Pipeline Summary Metrics	25
2.5.4	Visibility Distribution and Coverage Patterns	26
2.6	Limitations and Discussion	27
2.6.1	Current Pipeline Limitations	27
2.6.2	Toward a GEO-Based Recommendation System	28
	Conclusion	29
	General Conclusion	30

List of Figures

1.1	The ReAct framework [14]	8
1.2	Evolution of agentic system architectures [1].	8
1.3	System architectures without and with MCP [1].	9
2.1	Comparative output for <i>Dar El Jeld</i> : LangGraph static pipeline vs. agentic ReAct approach.	14
2.2	GEO system workflow	15
2.3	Agent 0 workflow	17
2.4	Agent 1 Workflow	20
2.5	Agent 2 Workflow	23

General Introduction

Digital marketing has long relied on search engine visibility as its primary lever for brand discovery. As Large Language Models (LLMs) increasingly supplant traditional search engines as the first interface between consumers and information, a new challenge emerges for marketers: brands must now be visible not only to ranking algorithms but to generative AI systems. This shift defines the domain in which our project is anchored *digital marketing* and gives rise to a new discipline known as ***Generative Engine Optimization*** (GEO). Unlike traditional Search Engine Optimization (SEO), which targets crawlers and ranking signals, GEO measures how often and how prominently a brand appears in the responses generated by LLMs when users ask domain-specific questions.

In this context, the present project proposes an **autonomous** multi-agent pipeline for GEO measurement and brand visibility analysis, applied as a pilot to the Tunisian restaurant sector. The system automatically collects LLM visibility signals, enriches them with real-world web presence data, and consolidates them into a structured dataset designed to support data-driven brand optimisation decisions in the era of generative AI.

This report is organised into two chapters. **Chapter 1** establishes the project context and theoretical foundations, covering the motivation, objectives, and a review of the relevant literature spanning SEO, generative AI in information retrieval, and agentic multi-agent architectures. **Chapter 2** presents the full system design, implementation, and evaluation, detailing the four-agent pipeline architecture and the results obtained on the Tunisian restaurant domain. A general conclusion closes the report with a summary of contributions and perspectives for future work.

Keywords: Generative Engine Optimization, Multi-Agent System, Large Language Models, Brand Visibility, Digital Marketing, Tunisian Restaurant Sector

Chapter 1

Project Context and State of the Art

Introduction

This chapter situates the project within the transformation of information access driven by generative AI. It traces the shift from traditional search to answer-based generation, motivates the growing importance of LLM visibility for local businesses, and formulates the core problem. It then defines the project objectives, reviews the relevant state of the art, and identifies the gaps the proposed system addresses.

1.1 | Context and Motivation

The way users access information has undergone a fundamental transformation. The rapid adoption of conversational AI systems ChatGPT, Gemini, Perplexity has shifted the dominant paradigm from *link-based retrieval*, where users select from a ranked list of sources, to *answer-based generation*, where the system synthesizes a direct response. A user asking “*Where should I eat tonight in Tunis?*” may now receive a curated set of recommendations presented as authoritative text, without consulting any external source.

This shift has direct commercial consequences. Visibility in an AI-generated response has become as commercially relevant as a first-page search ranking. Unlike traditional SEO signals backlinks, keywords, structured markup which are documented and actionable, LLM outputs are generated by probabilistic models whose entity selection mechanisms are not directly observ-

able. A business consistently mentioned benefits from significant exposure, one never mentioned effectively does not exist within this emerging information channel.

This situation motivates a new discipline: **Generative Engine Optimization (GEO)**, defined as the study of how entities are represented, measured, and potentially optimized within LLM-generated responses [2]. GEO is particularly relevant for local businesses in markets such as Tunisia, where limited digital footprints and low representation in web corpora create a structural disadvantage in model training data.

1.2 | Problem Statement

The Visibility Crisis in the Age of Generative AI

The widespread adoption of LLMs as everyday information tools is reshaping how businesses are discovered. ChatGPT surpassed **100 million** active users within two months of launch [2] an adoption pace unprecedented in consumer technology. As conversational AI becomes the primary interface for local discovery, a commercial reality emerges: *if a business does not appear in an LLM's response, it does not exist for that user.*

Unlike a search engine, which indexes content in near real-time, an LLM encodes knowledge statically during pre-training. Entities that were prominent and richly documented in the training corpus are reliably recalled, entities that were absent or sparse are structurally neglected, regardless of their actual quality. As LLM usage scales, this asymmetry compounds: every confident recommendation for one brand implicitly excludes equally deserving competitors the model has no knowledge of.

The consequences are disproportionately severe for local businesses. Large international chains benefit from accumulated web presence review aggregators, news coverage, Wikipedia articles, social media mentions all contributing to robust training data representation. A small Tunisian restaurant with no Wikipedia page, minimal reviews, and limited social reach remains invisible to the model, and by extension, to the growing share of users who discover businesses through AI-generated recommendations. Brands that start invisible risk remaining so, locked out of a feedback loop they cannot enter without first being visible.

The Measurement and Optimization Gap

Compounding this structural problem is the absence of any reliable methodology to measure or act upon LLM-based visibility. The stochastic nature of LLM inference means that a business may appear in 60% of responses to a given query and be absent from the remaining 40%. Without repeated, systematic measurement, this variability is invisible. A single query gives no actionable signal.

As a result, a business owner in this environment faces three compounding deficits:

- **No visibility audit:** no tool reliably determines how often, in what contexts, and with what accuracy an LLM mentions their business.
- **No benchmarking:** without cross-entity comparison, a business cannot assess its relative position within its competitive landscape as perceived by generative models.
- **No optimization pathway:** without understanding which web presence signals correlate with LLM visibility, a business cannot take informed action.

This triple deficit limits both practical decision-making and the scientific understanding of how entity representation in generative systems relates to observable web presence. Addressing it requires autonomous multi-source data collection, statistically rigorous cross-model measurement, and structured analytical output the precise problem this project is designed to solve.

1.3 | Project Objectives and Scope

This project aims to design and implement an autonomous multi-agent pipeline for measuring brand visibility in LLM-generated responses. Three objectives guide its development:

- **O1.** Measure the consistency with which named entities appear in LLM-generated responses across varied prompts and models.
- **O2.** Acquire and structure multi-source web presence data associated with identified entities.
- **O3.** Construct an analytical dataset enabling the study of relationships between LLM visibility signals and web presence features.

Two research questions guide the investigation:

- **RQ1.** How consistently do LLMs mention local businesses across repeated queries?
- **RQ2.** What web presence features are associated with higher visibility in LLM-generated responses?

The pipeline is applied to the Tunisian restaurant sector using French-language prompts. Real-time monitoring and GEO optimization recommendations fall outside the scope of this project.

1.4 | Literature Review and State of the Art

1.4.1 Search Engine Optimization

Since the late 1990s, Search Engine Optimization has been the dominant framework for managing online visibility. The foundational mechanism is the PageRank algorithm introduced by Brin and Page in 1998 [4], which ranks pages by the number and quality of inbound links. Over the following decades, SEO evolved to encompass technical optimization (site speed, crawlability, structured markup), on-page content quality, and off-page authority signals [9].

However, SEO operates under a **fundamental assumption**: the user receives a ranked list of links and selects one. This assumption no longer holds when the system generates a synthesized answer directly. The signals that drove traditional SEO become insufficient as measures of visibility in AI-generated responses the central challenge addressed by this project.

1.4.2 Generative AI in Information Retrieval and the Emergence of GEO

The introduction of the Transformer architecture [13] and the public release of capable models ChatGPT, Gemini, LLaMA [12], Mistral [7] established a rapidly evolving landscape of generative systems available through commercial APIs and open-source releases.

The concept of **Generative Engine Optimization (GEO)**, formally introduced by Aggarwal et al. in 2023 [2], addresses what determines whether a given entity appears in an LLM-generated response. GEO differs from SEO in a fundamental way: SEO operates on a crawlable, indexed web with observable ranking signals. GEO targets systems that encode knowledge implicitly during pre-training, with no index to rank in and no direct feedback signal.

Visibility is an emergent property of how well an entity is represented across the web sources that formed the model’s training data. Table 1.1 summarizes the key distinctions.

	SEO (Search Engine Optimisation)	GEO (Generative Engine Optimisation)
Target Platforms	Traditional search engines: Google, Bing, Yahoo.	AI-driven engines: ChatGPT, Gemini, Perplexity, DeepSeek.
Ranking Method	Algorithm-based ranking: keywords, links, technical performance.	AI synthesis based on training data representation and content quality.
Key Metrics	Organic traffic, keyword rankings, CTR, backlink profile.	Inclusion rate in AI responses, citation frequency.
User Behaviour	Users click through to the website for answers.	Users consume information directly within the AI’s response.
Update Frequency	Influenced by search engine algorithm updates.	Influenced by AI model training cycles.

Table 1.1 – Comparison of SEO and GEO across key dimensions. Adapted from [2].

Early empirical work on GEO identified content-level signals associated with higher mention rates: authoritative citations, fluent prose, and domain-specific terminology [2]. However, this work focused on document-level optimization, leaving unanswered how *entity-level* web presence structured data on review platforms, social media, and knowledge bases correlates with visibility in generative responses.

1.4.3 Large Language Models and Entity Representation

LLMs acquire knowledge about entities implicitly during pre-training on large text corpora. Unlike a database, an LLM does not store facts explicitly, it encodes statistical associations across billions of training examples. The frequency and consistency with which an entity appears in training data directly influences how reliably the model recalls and mentions it [8].

This **frequency bias** has been empirically demonstrated by Mallen et al. (2023): LLMs are significantly more accurate when recalling popular, frequently referenced entities than rare or locally known ones [8]. Beyond frequency, two additional risks apply: **hallucination** the generation of plausible but factually incorrect entity information [6] and **representation inconsistency**, where the same entity is described differently across runs of the same prompt. Both are pronounced for entities with sparse training data coverage, making systematic multi-run measurement essential.

1.4.4 Agentic AI Systems and Multi-Agent Architectures

An AI agent is a system that perceives its environment, reasons about its current state, selects actions, and executes them in pursuit of a goal [11]. LLMs capable of multi-step reasoning and external tool use have enabled agents that autonomously decompose complex goals into executable steps.

The ReAct Framework : The **ReAct** framework [14] grounds LLM reasoning in external reality by interleaving reasoning traces with action steps calls to external tools or APIs. At each step the agent produces a *thought*, selects an *action* (a tool call with structured arguments), and receives an *observation* (the tool’s response). This cycle repeats until sufficient information has been gathered. Figure 1.1 illustrates the loop.

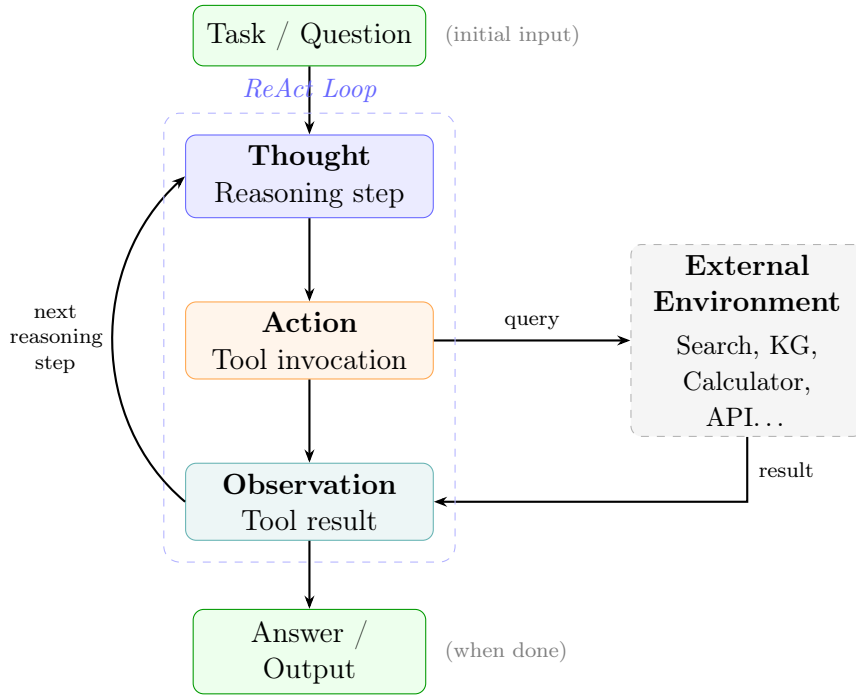


Figure 1.1 – The ReAct framework [14]

Multi-Agent Orchestration : Complex pipelines benefit from distributing responsibility across multiple specialized agents coordinated by an orchestrating layer. As shown in Figure 1.2, the field has progressed from single agents to fully orchestrated systems with a dedicated co-ordination layer. The key distinction is between **fixed pipelines** executing a predetermined sequence regardless of intermediate results and **autonomous orchestrators** which reason about the current state at each step and determine the next action dynamically.

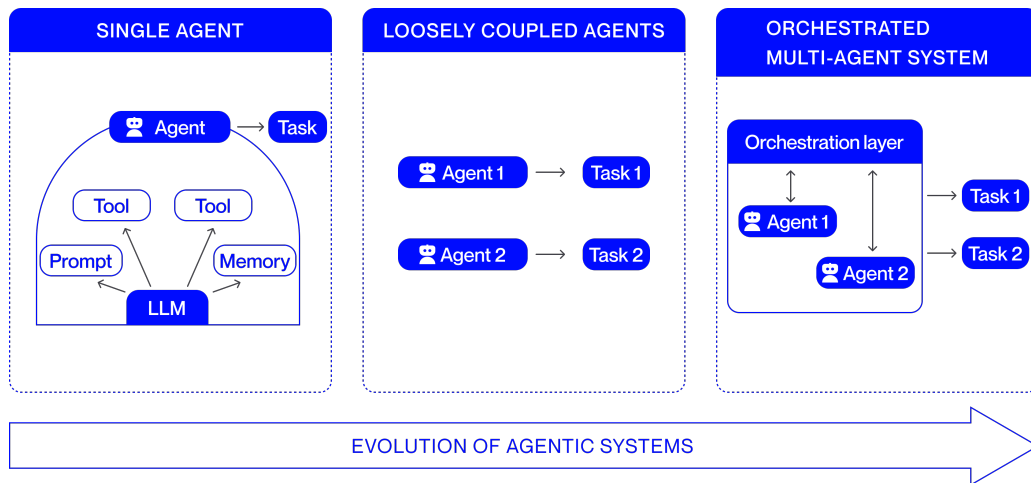


Figure 1.2 – Evolution of agentic system architectures [1].

Model Context Protocol : The **Model Context Protocol (MCP)**, introduced by Anthropic in 2024 [3], defines a standardized client-server protocol over a `stdio` interface: tool servers expose a uniform schema describing their capabilities, and agents discover and invoke tools dynamically at runtime. Each MCP server encapsulates its own tools as independent, swappable modules (Figure 1.3). In this project, MCP serves as the communication backbone between Agent 2’s ReAct loop and its web-research tool servers, enabling search, scraping, and encyclopedic lookup through a uniform interface without any tool-specific integration code in the agent itself.

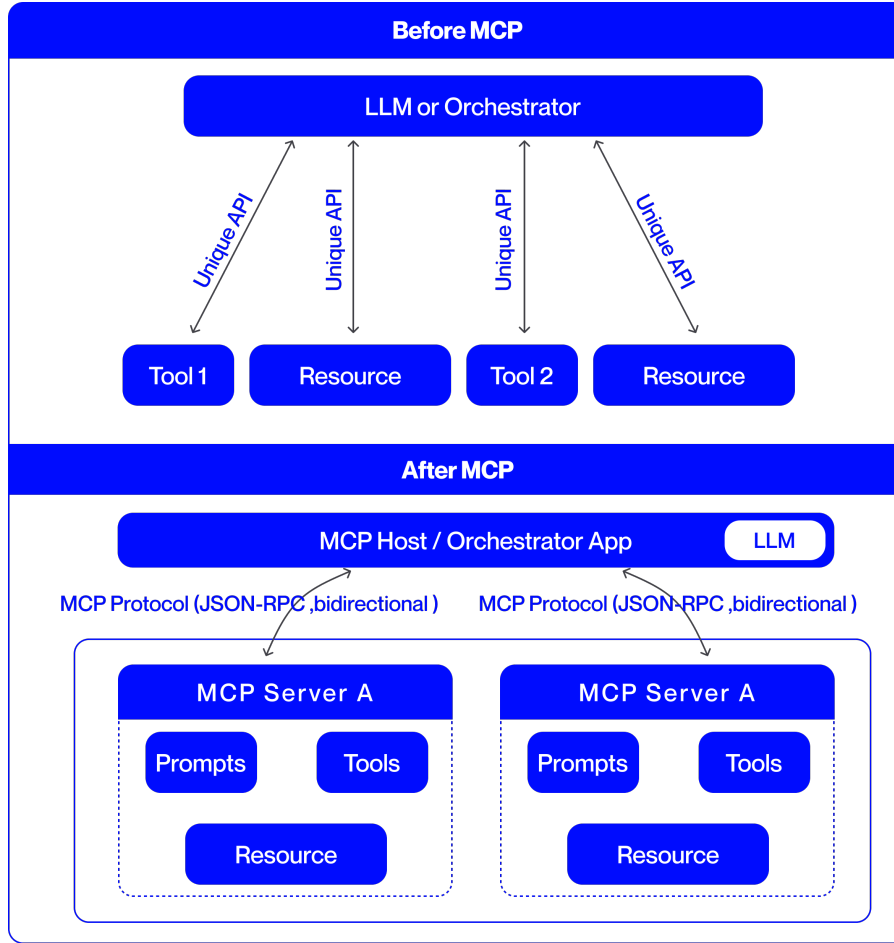


Figure 1.3 – System architectures without and with MCP [1].

1.4.5 Related Work and Research Gaps

Despite growing interest in LLM-based visibility, no existing framework provides an integrated, automated, and scientifically grounded solution for entity-level GEO measurement.

Visual workflow automation tools: Platforms such as n8n [10] and Zapier support basic prompt submission and response logging through predefined execution graphs, but offer no adaptive decision-making, dynamic model selection, or cross-run statistical aggregation making them unsuitable for research-grade measurement.

Commercial GEO and AI visibility platforms : Tools including BrandGPT, Otterly.ai, and Profound monitor brand mentions in LLM responses but with limited scientific validity: none discloses its measurement methodology, they query a small fixed model set, and they do not support multi-run evaluation essential for estimating mention stability [2].

Table 1.2 positions the contributions of this project against the key limitations of existing approaches.

Aspect	Existing Approaches	This Project
GEO measure- ment	Manual or single-run, no cross-model analysis	Automated, multi- prompt, multi-run, multi-model
Agent architec- ture	Fixed execution graphs or visual workflows	Autonomous ReAct or- chestrator with runtime decision-making
Web data ex- traction	Single source or manual lookup	Multi-source extraction via MCP-based tool servers
Target domain	Large, English-language brands	Local entities in emerging and non-English markets
Statistical rigor	Single-run, no stability analysis	Mention rate, stability score, prominence index

Table 1.2 – Limitations of existing approaches and contributions of the proposed system.

Three gaps are of particular significance: no existing tool measures entity visibility with statistical rigor across multiple prompts, runs, and models simultaneously, no open framework automates the full pipeline from prompt generation to web data extraction within a single autonomous system, and the relationship between structured web presence signals and LLM visibility remains empirically uninvestigated for local businesses in non-English markets. This project addresses all three within a single integrated pipeline.

Conclusion

This chapter has traced the progression from link-based visibility governed by documented algorithmic signals to representation-based visibility, where presence in LLM-generated responses is an emergent property of training data coverage. As LLM adoption scales globally, the asymmetry between well-represented global brands and structurally neglected local businesses intensifies, making rigorous measurement tools increasingly urgent. Existing approaches visual automation platforms, commercial monitoring dashboards, and single-model GEO tools address this problem only partially, each falling short in adaptability, rigor, or scope. These limitations directly motivate the design of the system presented in the following chapter.

Chapter 2

System Architecture and Results

Introduction

This chapter presents the design, architecture, and results of the GEO measurement pipeline. It establishes the research approach and core design decisions, then describes each component in detail: the dynamic model registry and the four specialised agents with their internal workflows. The chapter concludes with evaluation results and a discussion of observed limitations.

2.1 | Research Approach and Design Philosophy

This project follows a **design science research** methodology [5]: the primary contribution is a working software artifact evaluated against the problem it was designed to solve. The approach involves iterative construction and empirical evaluation each component is built, tested under real API conditions, measured for output quality, and refined before the next is designed. Three principles guided all architectural decisions.

Autonomy over prescription: the system makes its own decisions about which model to use, which data source to query, and how to recover from failures without human intervention at runtime. This ruled out fixed execution graphs in favor of an orchestrator that reasons about the current state and selects the next action dynamically.

Resilience over speed: the pipeline operates under constrained conditions free-tier API quotas, variable rate limits, and tool servers that may time out. The system must checkpoint progress and resume without data loss, motivating the per-model state tracking in the model registry.

Reproducibility over convenience: all outputs are saved in structured CSV files with fixed schemas. Intermediate outputs from each agent are preserved separately so any stage can be re-run without repeating earlier stages.

2.2 | System Overview

2.2.1 Comparative Analysis

The first implementation of Agent 2 was built as a **LangGraph** state graph: a fixed sequence of tool calls Google Maps, TripAdvisor, Wikipedia, Instagram, Facebook executed unconditionally for every entity, regardless of what each step returned. This approach guaranteed structural completeness but exposed a fundamental limitation: the agent had no mechanism to reason about intermediate results. If Google Maps returned no website, the agent still invoked `extract_website_socials` on a null URL. If no Instagram handle was found from any source, the agent still called `scrape_instagram` on an empty string. Each failure consumed an API call and contributed noise to the output, while the agent remained unable to adapt its search strategy.

The **ReAct** (Reason + Act) paradigm was adopted to address this structural limitation. Rather than following a hardcoded call sequence, the ReAct agent produces an explicit reasoning step before each action assessing what has been collected, what is still missing, and which tool is most likely to fill the gap then incorporates the tool result as an observation before deciding the next step. This Thought-Action-Observation cycle continues until the agent determines that sufficient data has been gathered, making the execution path *entity-specific* rather than uniform.

Figure 2.1 illustrates the practical difference on *Dar El Jeld*: the static LangGraph pipeline issued redundant tool calls and produced several null fields that a correctly sequenced agent would have populated, whereas the ReAct agent collected a more complete profile with fewer inference steps by conditioning each action on the previous observation.

This comparison directly motivates the architectural choice:

canonical_entity	mention_count	mention_rate	average_ranking_position	rank_variance	top1_rate	frequency_across_runs	stability_score
dar eljeld (ReAct Agentic)	159	0.4066	2.013	1.3338	0.4591	0.8239	5
dar eljeld (LangGraph pipeline)	8	0.3478	1	0	0.875	2	0.3478

Figure 2.1 – Comparative output for *Dar El Jeld*: LangGraph static pipeline vs. agentic ReAct approach.

The ReAct approach was therefore adopted as the execution model for Agent 2, and its conditional 8-step research protocol described in the following subsection replaces the uniform call sequence of the initial design.

2.2.2 Final System Overview

The pipeline is organized into four sequential stages, each handled by a dedicated agent, coordinated by an orchestration layer, and supported by a dynamic model registry shared across all agents. Figure 2.2 presents the full system architecture.

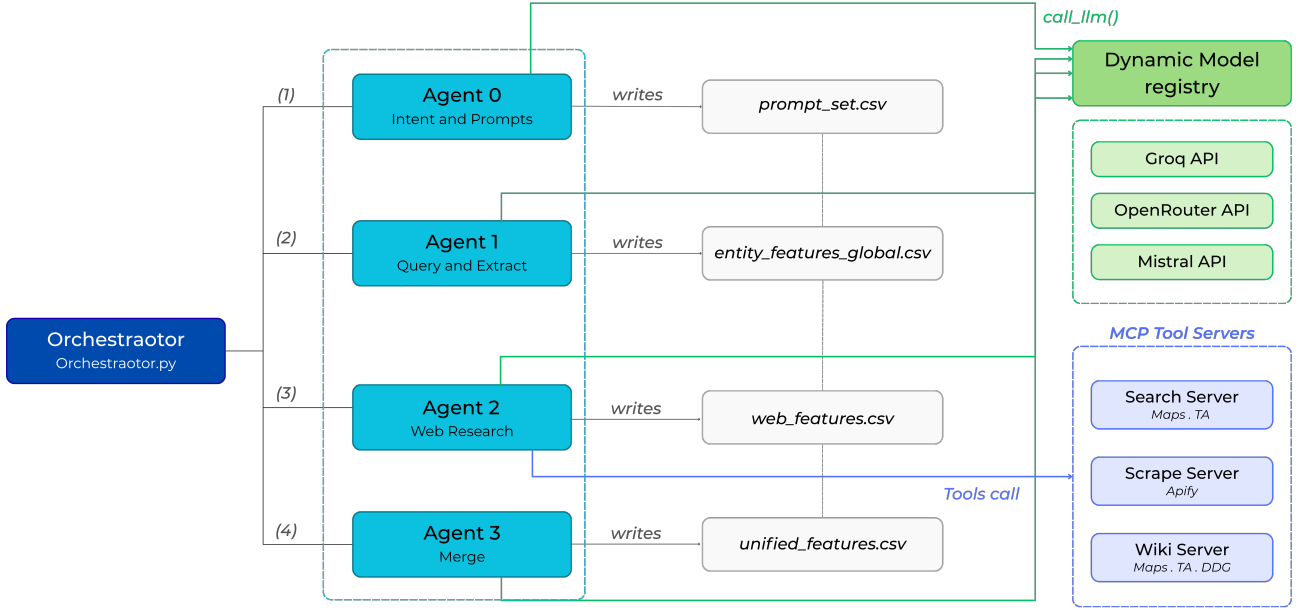


Figure 2.2 – GEO system workflow

The **orchestrator** (`orchestrator.py`) drives the four agents (1)–(4). Each agent reads its input from the upstream CSV checkpoint and writes its output to the next. Agent 3 merges both Agent 1 and Agent 2 outputs using **fuzzy** entity matching. Agents query the dynamic model registry (**DMR**) for the best available (model, provider, key) triple at each inference call. Agent 2 performs all web data collection exclusively through the three MCP tool servers.

2.3 | Dynamic Model Registry

Operating across multiple free-tier LLM providers requires managing heterogeneous and frequently binding quota constraints. The **dynamic model registry** maintains a continuously updated state for every available model slot and selects the best available option autonomously at each inference call eliminating the need for hardcoded fallback chains.

At pipeline startup, the registry queries the model listing APIs of Groq, OpenRouter, and Mistral, constructing a pool of slots identified by `provider:keyIndex/model_id`. Each slot is assigned a capability tier (Table 2.1) and maintains three runtime states: **TPD exhaustion** (daily quota hit permanent, persisted to disk across restarts), **TPM block** (per-minute rate limit temporary, cleared when the retry window expires), and **learned token capacity** (set

on 413 errors used to pre-filter the model for oversized prompts, eliminating repeated failures without manual configuration).

Tier	Quality Level	Identifying Keywords
0	Highest	70b, 72b, 120b, llama-4, qwen3-32b, claude-3.5, deepseek-r1
1	Medium	32b, 27b, 14b, gemini-flash, mistral-small, claude-3-haiku
2	Lightweight	8b, 7b, 3b, instant, mini, nano

Table 2.1 – Model capability tiers used for selection ranking.

The `select(preferred, estimated_tokens)` method returns the best available slot: (1) if a preferred model is available on any key and fits the estimated prompt size, return it immediately (2) otherwise filter all available, non-blocked slots that fit the token count (3) sort by tier, provider priority (Groq > Mistral > OpenRouter), key index, and model ID for stability (4) return the top candidate. If no slots are available, a `RuntimeError` is raised listing all exhausted and blocked models.

2.4 Pipeline Agents

2.4.1 Agent 0: Intent Discovery and Prompt Generation

A single query formulation such as “best restaurants in Tunis” would produce results sensitive to the experimenter’s specific phrasing. Agent 0 addresses this by autonomously generating a diverse prompt set covering multiple user intents and linguistic variants, ensuring that downstream visibility metrics reflect robust entity representation across a broad query space.

The generation pipeline consists of the following steps:

- **Step 1: Intent discovery.** Given a domain description and target language, an LLM identifies **`n` distinct query intents** representing real user interactions. The number of discovered intents is controlled by the `n_intents` parameter. The resulting taxonomy is validated and persisted as structured JSON within the pipeline state.
- **Step 2: Prompt generation.** For each discovered intent, the system generates `n_variants`

alternative query formulations. Variations differ in vocabulary, sentence structure, level of detail, and phrasing style in order to simulate realistic user diversity.

- **Step 3: Self-reflection and validation.** The LLM evaluates prompts against intent *diversity*, *linguistic naturalness*, and a GEO-specific constraint requiring at least 60% of prompts to elicit named establishment references. Upon failure, generation restarts with augmented *n_intents*, up to *max_reflection_loops* iterations, and the final set is saved to `prompt_set.csv`.

Figure 2.3 illustrates the internal **workflow** of Agent 0:

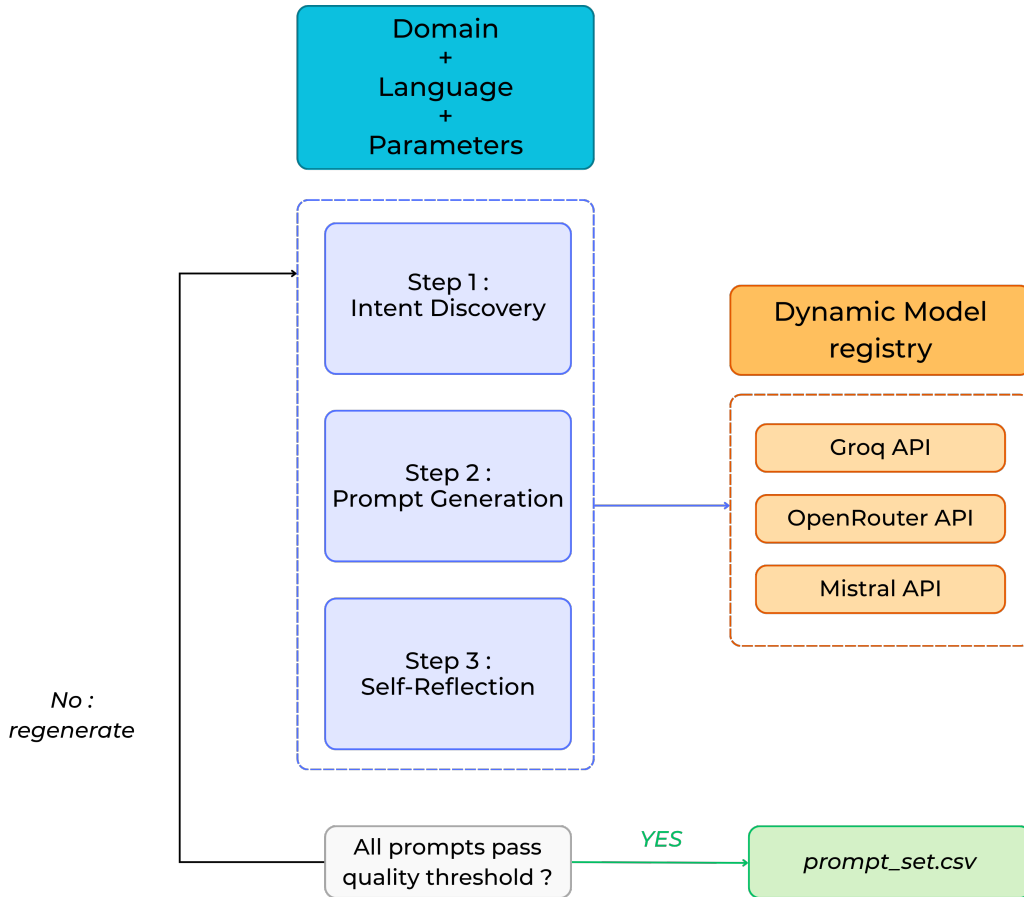


Figure 2.3 – Agent 0 workflow

2.4.2 Agent 1: LLM Querying and Entity Extraction

Agent 1 is responsible for the core GEO measurement task: submitting generated prompts to multiple LLMs, extracting named entities from responses, and computing visibility metrics that quantify how consistently each entity appears across prompts, models, and runs. *Figure 2.4 represents the overall workflow of Agent 1, from multi-model querying to GEO metric*

computation.

Step 1: Multi-model querying

- Each prompt is submitted to every model slot defined in `QUERY_MODELS`.
- Queries are repeated `n_runs` times per model slot.
- This cross-model, multi-run setup ensures that visibility is measured across diverse LLM behaviors, not a single response.
- All responses are incrementally stored in `raw_responses.csv` with resume support.

Step 2: Zero-shot entity extraction

- A dedicated Mistral-based extractor (fixed model, `temperature = 0`) processes each response.
- A structured system prompt enforces:
 - Domain scope: Tunisian establishments only
 - Normalization: lowercase, removal of generic prefixes
 - Output format: flat JSON string array
- The extractor outputs:
 - Entity names only (no confidence scores)
 - Associated descriptive text blocks for downstream use

Step 3: Enrichment

- For each entity, two structural features are computed:
 - **Ranking position:** order of first appearance in the response, extracted via a secondary LLM call
 - **Description length:** estimated in tokens using the character ratio of the entity description to the full response

Step 4: Five-phase cleaning pipeline

- **(A) Hard pre-filtering**
 - Removes empty, very short (< 3 characters), numeric, and singleton entities (mention count < 2)
- **(B) Fuzzy clustering**
 - Uses RapidFuzz (threshold = 88) to group highly similar entity names
 - The most frequent variant becomes the canonical form
- **(C) LLM arbitration**

- Resolves ambiguous pairs (similarity 75–87)
- Filters out invalid entities (generic terms, wrong geography, missing Tunisian context)
- **(D) Self-reflection**
 - Reviews previous cleaning decisions
 - Corrects erroneous merges or invalidations flagged as suspicious
- **(E) Mapping application**
 - Outputs final canonical entities
 - Adds quality flags and null-rank penalties
 - Stores results in `clean_entities.csv`

Step 5: GEO metric computation

- Computes three visibility metrics per canonical entity:
 - **Mention rate:** fraction of valid responses containing the entity
 - **Stability score:**

$$\text{stability_score} = \text{mention_rate} \times \frac{1}{1 + \text{rank_variance}}$$

- **Consistency label:**
 - *stable* (≥ 0.75)
 - *variable* (≥ 0.40)
 - *unstable* (otherwise)
- Entities with fewer than 2 mentions are already excluded in Phase A.
- Final eligibility thresholds (e.g., for Agent 2) are applied by the orchestrator.

Figure 2.4 illustrates the internal **workflow** of Agent 1:

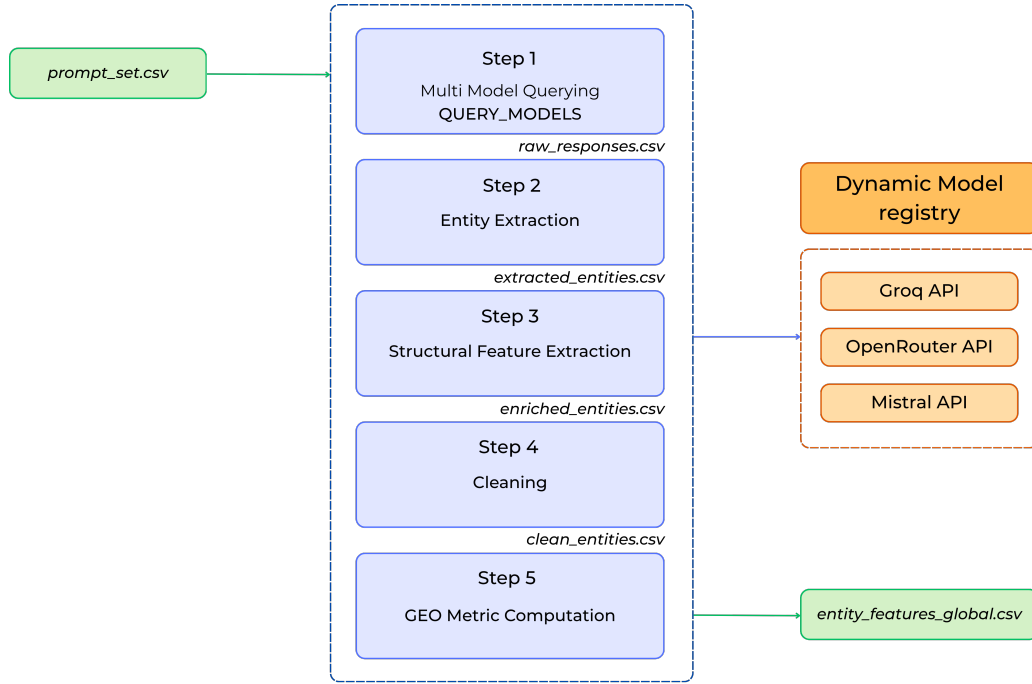


Figure 2.4 – Agent 1 Workflow

2.4.3 Agent 2: Web Presence Research

Agent 2 is the most technically complex component. For each entity that meets the visibility threshold, it operates as a fully autonomous **ReAct agent** over multiple MCP tool servers.

At each iteration, the agent follows a reasoning loop composed of:

- a *thought*: reasoning about the information gathered so far,
- an *action*: selection and invocation of a tool,
- an *observation*: integration of the tool’s response.

Figure 2.5 represents the overall execution flow of Agent 2, illustrating the interaction between the ReAct reasoning loop and the external MCP tools.

Research Protocol : The agent follows a **prescribed 8-step research protocol** defined in its system prompt:

— **Core steps :**

1. google_maps_search
2. wikipedia_lookup
3. tripadvisor_search
4. wikidata_lookup

5. `extract_website_socials`
6. `scrape_instagram`
7. `scrape_facebook`
8. `ddg_search`

Data Sources and Constraints : The MCP servers expose the tools described in Table 2.2. The agent collects information across six categories: Google Maps, TripAdvisor, Wikipedia, Wikidata, Instagram, Facebook.

To ensure relevance, the protocol enforces **geographic anchoring**:

- Every query must include “Tunisia” or a Tunisian city name.
- Any result referring to a non-Tunisian context is discarded.
- Queries are retried with more specific terms when necessary.

Server	Tool	Description
search_server	<code>google_maps_search</code>	Rating, review count, address, phone, website. Rotates through a pool of SerpAPI keys.
	<code>tripadvisor_search</code>	Two-step process: a web search locates the TripAdvisor page URL, then a scraper extracts rating and ranking.
	<code>ddg_search</code>	General DuckDuckGo search returning titles, URLs, and snippets.
scrape_server	<code>scrape_instagram</code>	Extracts follower count, post count, and biography via Apify. Rotates through multiple API tokens.
	<code>scrape_facebook</code>	Extracts page likes and average post engagement via Apify.
	<code>extract_website_socials</code>	Extracts Instagram and Facebook handles from a website’s HTML.
wiki_server	<code>wikipedia_lookup</code>	Queries the Wikipedia API, returns presence and URL.
enrichment_server	<code>wikidata_lookup</code>	Queries the Wikidata API returns entity type, country, founding date, and official website.

Table 2.2 – MCP tool servers and their exposed tools.

Supervisor Agent

API errors are handled by the **Supervisor Agent**, a rule-based error classifier with an optional LLM-based escalation mechanism.

Error Handling Workflow : For each error, the supervisor returns one of the following actions:

- **switch_model**: triggered by billing or quota exhaustion
- **wait_retry**: triggered by rate limiting (uses retry-after duration)
- **retry**: triggered by transient 5xx errors
- **skip**: triggered after four unsuccessful attempts

Decision Mechanism : The supervisor operates in two stages:

- **Primary mode**: classification using a lightweight LLM
- **Fallback mode**: deterministic **if/elif** rule-based logic

Both modes produce identical decisions for standard structured API errors. The LLM path mainly improves robustness for ambiguous or previously unseen errors.

Checkpointing : Each successfully processed entity is immediately stored in: `web_features.csv`
Figure 2.5 illustrates the internal **workflow** of Agent 2:

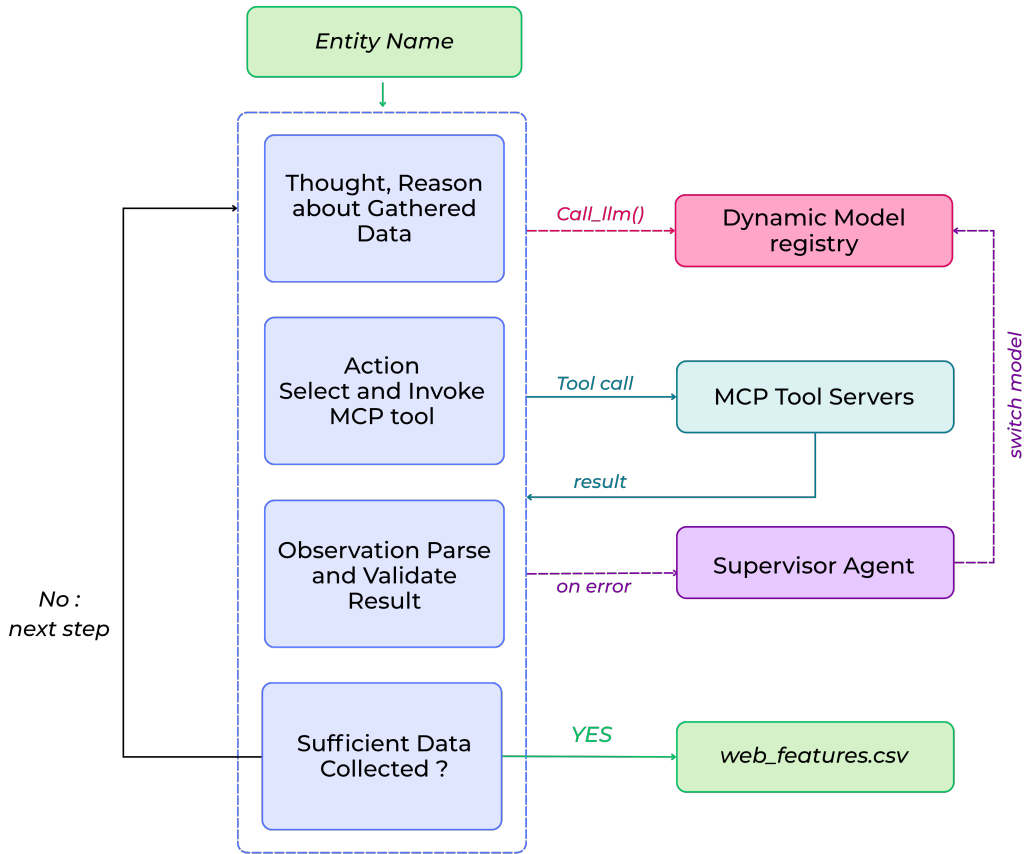


Figure 2.5 – Agent 2 Workflow

2.4.4 Agent 3: Data Consolidation

Agent 3 is the final stage of the pipeline. It joins the LLM visibility metrics from Agent 1 with the web presence features from Agent 2 using a three-stage fuzzy matching strategy (exact normalized match, substring containment, then word overlap), with Agent 1 as the source of truth so that every extracted entity is retained.

Two LLM-based reasoning steps then clean the merged dataset.

- First, **deduplication**: entities are grouped by shared content words (stop words such as *le*, *la*, *de*, *el*, *dar* removed) and each candidate group is submitted to the LLM to decide whether the rows refer to the same place. Confirmed duplicates are merged mention counts summed, the richer web-data row kept.
- Second, **triage**: entities for which Agent 2 found no web evidence are classified by the LLM as *retry* (real restaurant, missed by Agent 2), *hallucination*, *non_restaurant*, or *generic*. Non-viable entities are removed and saved to `rejected_entities.csv`; *retry* entities are written to the merge report so the orchestrator can re-invoke Agent 2 on

them.

- Finally, Agent 3 computes a **data-completeness score** (fraction of ten key fields that are non-null) and assigns a **unified quality** label (*complete*, *partial*, *sparse*, or *agent2_failed*) to each row.

The result is saved to `unified_features.csv`. Table 2.3 lists the key output fields.

Field	Source	Description
<code>canonical_entity</code>	Agent 1	Normalized entity name
<code>mention_rate</code>	Agent 1	Fraction of valid responses with a mention
<code>stability_score</code>	Agent 1	$\text{mention_rate} \times \frac{1}{1+\text{rank_variance}}$
<code>consistency_label</code>	Agent 1	stable / variable / unstable
<code>gm_rating</code>	Agent 2	Google Maps rating (0–5)
<code>gm_review_count</code>	Agent 2	Google Maps number of reviews
<code>ta_rating</code>	Agent 2	TripAdvisor rating
<code>ta_ranking</code>	Agent 2	TripAdvisor ranking position
<code>has_wikipedia</code>	Agent 2	Boolean Wikipedia presence
<code>ig_followers</code>	Agent 2	Instagram follower count
<code>overall_confidence</code>	Agent 2	Agent-assigned data quality score (0–1)
<code>data_completeness</code>	Agent 3	Fraction of key fields non-null (0–1)
<code>unified_quality</code>	Agent 3	complete / partial / sparse / agent2_failed
<code>triage_decision</code>	Agent 3	retry / hallucination / non_restaurant / generic (failed entities only)
<code>triage_reason</code>	Agent 3	LLM justification for the triage decision

Table 2.3 – Key fields in the final output dataset (`unified_features.csv`).

2.5 | Results

The pipeline was evaluated on the Tunisian restaurant domain.

2.5.1 Generation and Extraction Results

Agent 0 generated **30 prompts** covering **6 distinct user intents** (top recommendation, tourist reviews, comparative, authentic cuisine, special occasion, and location-based queries), with 5 variants per intent in French.

Agent 1 submitted each prompt to **3 model slots**, repeated 5 times per slot, resulting in **450 total LLM responses**. From these responses, the extraction pipeline produced **2,020 raw entity records**, corresponding to **831 unique surface forms**.

After the five-phase cleaning process, these records were consolidated into **234 canonical entities**.

2.5.2 Enrichment and Consolidation Results

Among the 234 canonical entities, **213 met the minimum mention-rate threshold** and were forwarded to Agent 2 for web enrichment.

Agent 3 then performed:

- LLM-based deduplication of spelling variants
- LLM-based triage of unresolved entities
- Completeness scoring and quality labeling

As a result:

- 4 entities were rejected:
 - 1 hallucination
 - 1 non-restaurant venue
 - 2 overly generic names
- The final dataset contains **182 enriched entities**

2.5.3 Pipeline Summary Metrics

Table 2.4 summarizes the main output statistics.

Metric	Value
Prompt Generation and Extraction	
Prompts generated (Agent 0)	30
LLM responses collected (Agent 1)	450
Raw entity extraction records	2,020
Unique surface forms before cleaning	831
Canonical entities after cleaning	234
Enrichment and Final Dataset	
Entities submitted to Agent 2	213
Final entities after dedup + triage (Agent 3)	182
Complete (≥ 3 sources)	75
Partial (1–2 sources)	84
Failed (no sources recovered)	22
Entities rejected by LLM triage	4
Web Coverage Metrics	
Google Maps coverage	85.2%
TripAdvisor coverage	38.5%
Wikipedia presence	0%
Instagram presence	31.9%
Facebook presence	28.0%
Quality Metrics	
Mean data-completeness score	0.584
Mean overall confidence score	0.541

Table 2.4 – Pipeline results on the Tunisian restaurant domain.

2.5.4 Visibility Distribution and Coverage Patterns

The distribution of LLM visibility is highly concentrated. The top 15% of entities (27 out of 182) account for 62% of all 1,351 mention instances, while the remaining 85% share only 38%.

The most visible restaurant, *Dar El Jeld*, received 159 mentions across all prompts and model slots, more than twice the number of the second-ranked entity.

All entities received an *unstable* consistency label. This is expected given the relatively small prompt set (30 prompts) and the high variance of mention rates.

Regarding web enrichment coverage:

- Google Maps achieved the highest coverage rate (85.2%)
- TripAdvisor coverage remained moderate (38.5%)
- No entity had a dedicated Wikipedia article
- Instagram and Facebook coverage remained limited (31.9% and 28.0%)

These results reflect the local and partially digitalized nature of the Tunisian restaurant ecosystem.

2.6 | Limitations and Discussion

Despite the improvements introduced by the agentic design, several limitations remain.

2.6.1 Current Pipeline Limitations

- **Partial web enrichment.** Of the 182 final entities, 22 (12%) remain in a *failed* state even after one enrichment pass and a second targeted retry pass. These are typically restaurants with very common names, inconsistent spellings, or no indexed web presence. Improving recall for this subset would require more precise search strategies, such as city-level geocoding or combining additional web sources.
- **Geographic disambiguation.** Some extracted entities corresponded to well-known establishments outside Tunisia, such as *Le Grand Véfour* in Paris or *La Mamounia* in Marrakech. Because such entities are highly represented in LLM training data, they can overshadow Tunisian homonyms. Agent 3 successfully identified and rejected 4 such non-viable cases.
- **Social media data sparsity.** Only 31.9% of the final entities had an identified Instagram presence, and only 28.0% had a Facebook page. This reflects the relatively low digital formalization of small hospitality businesses in Tunisia and reduces the contribution of social features to the GEO score.
- **Retry convergence.** The current two-pass retry design (Agent 2 → Agent 3 triage → Agent 2 retry) still leaves 21 unresolved entities in the retry queue. A more robust implementation would require an explicit convergence criterion, such as stopping after two retries with no measurable improvement.

- **API quota and rate-limit constraints.** The pipeline depends on several external services, each with its own limitations:
 - LLM inference providers such as Groq, OpenRouter, and Mistral impose token quotas and request limits on free tiers.
 - SerpAPI provides only 100 free queries per key, which is insufficient for large-scale enrichment across hundreds of entities.
 - Wikidata and Instagram require progressive throttling to avoid HTTP 429 errors.

These constraints forced execution to be split across multiple sessions and days, introducing some temporal inconsistency in the collected data. A production deployment would require paid APIs or self-hosted inference models.

2.6.2 Toward a GEO-Based Recommendation System

Beyond simple measurement, the long-term objective of this work is to build a **recommendation system for generative visibility optimization**.

The current pipeline acts as a diagnostic layer:

- It measures how often a restaurant is cited by LLMs
- It evaluates the completeness of its digital footprint
- It identifies missing signals that may reduce visibility

This dataset can then be transformed into actionable recommendations.

For example:

- A restaurant with high Google Maps credibility but low LLM mention rate may require stronger textual signals, such as review coverage, editorial mentions, or linked-data presence.
- A restaurant with strong LLM visibility but weak TripAdvisor coverage may benefit from reinforcing its consistency across review platforms.
- Restaurants with weak social media signals may improve their visibility by consolidating Instagram and Facebook presence.

The GEO score provides a unified prioritization signal that can help rank these actions according to their expected impact on generative visibility.

This approach positions the system not only as a measurement tool, but also as a practical decision-support platform for restaurant owners and marketers seeking to compete in an ecosystem increasingly shaped by AI-generated recommendations.

Conclusion

This chapter has presented the full design, architecture, and results of the GEO measurement pipeline. The four agents each with a well-defined input, processing workflow, and structured output collectively realize the three design principles: autonomy through supervisor-based error recovery, registry-driven model selection, and LLM-based deduplication and triage resilience through progressive checkpointing, fuzzy entity matching, and a two-pass retry loop and reproducibility through fixed CSV schemas and decoupled outputs. Applied to the Tunisian restaurant domain, the pipeline produced a final enriched dataset of **182 entities** with 85.2% Google Maps coverage and a mean data-completeness of 0.584, confirming both a strong concentration of LLM visibility (top 15% of entities attract 62% of mentions) and the practical challenges of web data acquisition for local businesses. These findings provide the analytical dataset from which the research questions posed in Chapter 1 are systematically addressed in the general conclusion.

General Conclusion

This project set out to address a concrete gap in the digital marketing landscape: the absence of objective, automated tooling for measuring brand visibility in generative AI systems. As LLMs become the dominant interface through which consumers discover products and services, the ability to quantify and act on that visibility is no longer optional for brand owners it is a strategic necessity.

To that end, we designed and implemented a fully autonomous four-agent pipeline that operationalizes GEO measurement end to end. Starting from raw user intents, the system generates multilingual prompts, queries multiple LLM providers, extracts and normalizes brand mentions, enriches each entity with structured web presence data, and consolidates the result into a clean, analysis-ready dataset.

The results revealed a strong concentration of LLM visibility: the top 15% of entities attract 62% of all mentions, while the vast majority of restaurants remain virtually invisible to AI-generated responses. This asymmetry is the central finding of this work. It confirms that GEO inequality exists in local markets, and that it is measurable which is the prerequisite for any optimization intervention.

Looking ahead, this work opens the path to its natural continuation: a **GEO-based recommendation system** that translates the collected visibility and web presence data into prioritized, actionable guidance for brand owners. Extending the pipeline to other domains and markets, improving retry convergence for hard-to-enrich entities, and integrating richer content signals structured data markup, citation patterns, editorial mentions represent the key directions for future development. The foundation is in place the next step is to make it prescriptive.

Bibliography

- [1] A Survey on Multi-Agent Systems, 2025. URL <https://arxiv.org/abs/2601.13671>.
- [2] Pranjal Aggarwal, Vishvak Murahari, Tanmay Rajpurohit, Ashwin Kalyan, Karthik Narasimhan, and Ameet Talwalkar. GEO: Generative engine optimization, 2023. URL <https://arxiv.org/abs/2311.09735>.
- [3] Anthropic. Model context protocol specification, 2024. URL <https://modelcontextprotocol.io>.
- [4] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. Computer Networks and ISDN Systems, 30:107–117, 1998. URL <https://snap.stanford.edu/class/cs224w-readings/Brin98Anatomy.pdf>.
- [5] Alan R. Hevner, Salvatore T. March, Jinsoo Park, and Sudha Ram. Design science in information systems research. MIS Quarterly, 28(1):75–105, 2004. URL <https://misq.umn.edu/design-science-in-information-systems-research.html>.
- [6] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation, 2023. URL <https://arxiv.org/abs/2202.03629>.
- [7] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, et al. Mistral 7b, 2023. URL <https://arxiv.org/abs/2310.06825>.
- [8] Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics, 2023. URL <https://arxiv.org/abs/2212.10511>.
- [9] Moz. The beginner’s guide to SEO, 2023. URL <https://moz.com/beginners-guide-to-seo>.

- [10] n8n GmbH. n8n workflow automation, 2024. URL <https://n8n.io>.
- [11] Stuart Russell and Peter Norvig. Artificial Intelligence: A Modern Approach. Pearson, 4th edition, 2020. URL <https://aima.cs.berkeley.edu/>.
- [12] Hugo Touvron, Thibaut Lavril, Gautier Izacard, et al. LLaMA: Open and efficient foundation language models, 2023. URL <https://arxiv.org/abs/2302.13971>.
- [13] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Advances in Neural Information Processing Systems, volume 30, 2017. URL <https://arxiv.org/abs/1706.03762>.
- [14] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In International Conference on Learning Representations, 2023. URL <https://arxiv.org/abs/2210.03629>.