MapAgent: A Hierarchical Agent for Geospatial Reasoning with Dynamic Map Tool Integration

Anonymous ACL submission

Agentic AI has significantly extended the capabilities of large language models (LLMs) by enabling complex reasoning and tool use. However, most existing frameworks are tailored to domains such as mathematics, coding, or web automation, and fall short on geospatial tasks that require spatial reasoning, multi-hop planning, and real-time map interaction. To address these challenges, we introduce **MapAgent**, a hierarchical multi-agent plug-and-play framework with customized toolsets and agentic scaffolds for map-integrated geospatial reasoning. Unlike existing flat agent-based approaches that treat tools uniformly—often overwhelming the LLM when handling similar but subtly different geospatial APIs—MapAgent decouples planning from execution. A high-level planner decomposes complex queries into subgoals, which are routed to specialized modules. For tool-heavy modules—such as map-based services—we then design a dedicated map-tool agent that efficiently orchestrates related APIs adaptively in parallel to effectively fetch geospatial data relevant for the query, while simpler modules (e.g., solution generation or answer extraction) operate without additional agent overhead. This hierarchical design reduces cognitive load, improves tool selection accuracy, and enables precise coordination across similar APIs. We evaluate MapAgent on four diverse geospatial benchmarks—MapEval-Textual, MapEval-API, MapEval-Visual, and MapQA—and demonstrate substantial gains over state-of-the-art tool-augmented and agentic baselines.

1 Introduction

Agentic large language models (LLMs) have significantly expanded the scope of AI systems by enabling complex reasoning, subgoal decomposition, and dynamic tool use (Yao et al., 2023; Kim et al., 2024; Du et al., 2024). Despite these advances, most existing frameworks predominantly focus on domains such as mathematics (Lu et al., 2023, 2025), software engineering (Jimenez et al., 2023), and web automation (Qin et al., 2023; Du et al., 2024), while their application to **geospatial reasoning**—a ubiquitous capability that enables the automation of everyday tasks through natural language instructions, particularly in scenarios requiring interaction with specialized tools like map services—remains limited. This includes tasks such as route planning, location-based decision-making, spatial comparison, and finding nearby points of interest (e.g., restaurants, gas stations, or EV charging points) that satisfy user constraints.

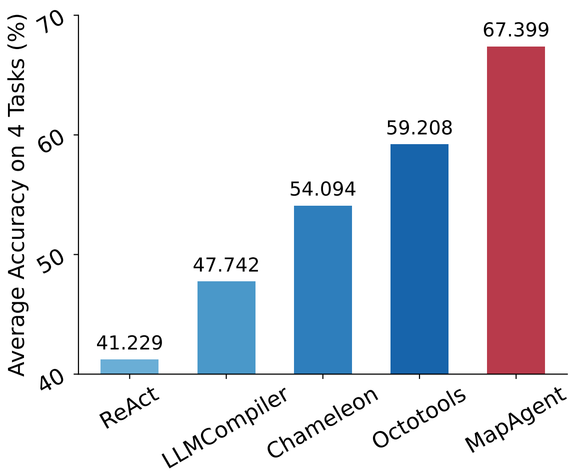


Figure 1: Performance comparison across 4 geo-spatial benchmarks. MapAgent significantly outperforms others and achieves state-of-the-art (SOTA) performance.

Geospatial tasks pose distinct challenges compared to other reasoning problems. Queries in this domain often require multi-hop planning, dynamic spatial grounding, and coordination across multiple external APIs with overlapping but subtly different capabilities. For instance, solving queries like *"Find the shortest route from home to the office with a stop at a highly rated coffee shop"* or *"Plan a three-day road trip from San Francisco to Yosemite with overnight stops at scenic towns and hiking*

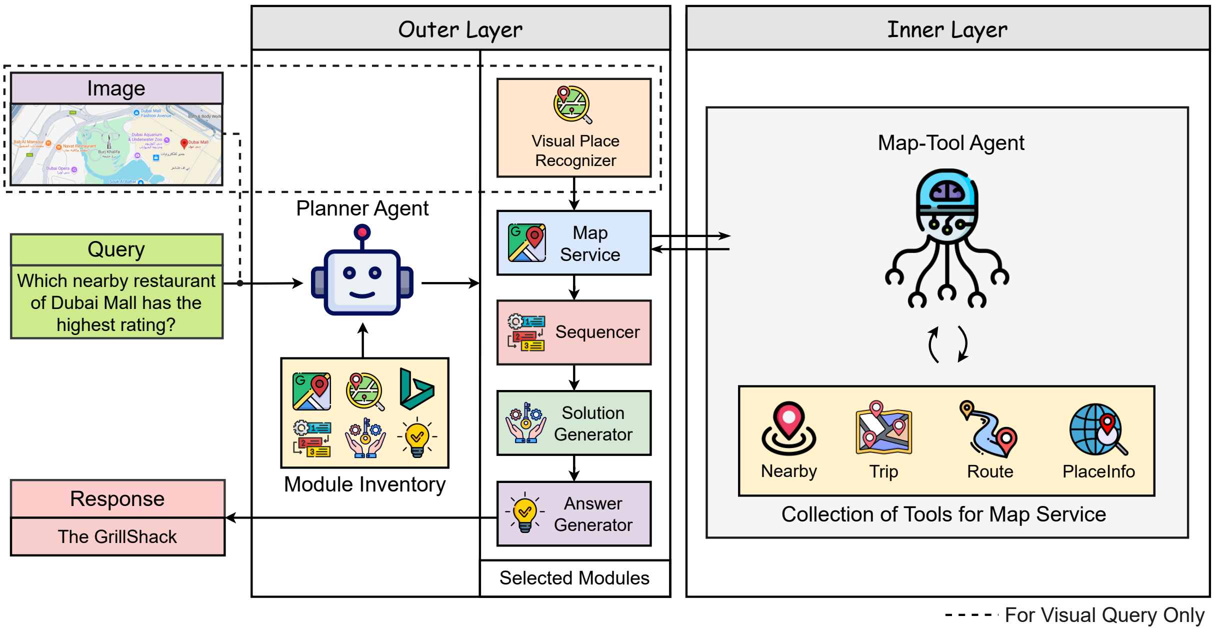


Figure 2: Overview of MapAgent. Given a user query (optionally with an image), the Planner Agent decomposes it into subtasks using the available Module Inventory and selects appropriate modules for each subtask. For tool-heavy modules, a dedicated agent (e.g., a Map-Tool Agent) manage interactions with the associated tools adaptively.

spots along the way" involves chaining services for nearby place search, distance estimation, POI (point-of-interest) retrieval, temporal scheduling, and real-time filtering. Despite addressing very related geospatial functions, the available Map APIs vary in their own schema, constraints, and precious functionalities, making it non-trivial to integrate them effectively. The agent must not only orchestrate multiple tools, but also combine and reason over the retrieved data to produce timely, coherent responses that preserve geospatial consistency.

However, current agentic systems are not designed for this setting. Most plug-and-play tool-using agents adopt flat execution architectures that treat tools as generic, interchangeable black boxes. This leads to two major limitations in the context of geospatial reasoning: *(i) Tool inflation:* Existing systems often bundle disparate tools—one of each kind—without accounting for the fine-grained functional variations present within map services. As geospatial APIs proliferate (e.g., distance, proximity, directions, routing, nearby search, place details, timelines), agents face a combinatorial burden in both planning and execution, increasing decision complexity and reducing overall effectiveness.

(ii) Tool incapability: Tools integrated into existing agents are typically primitive (single-API-based) and generic (e.g., image captioning, table lookup, code generation, or web search), and are not designed for the rich, mixed-mode (parallel and sequential) interactions required by real-world map services like Google Maps. While some API calls—such as computing distances and retrieving place details—can be performed in parallel, many tasks require tightly coordinated sequential steps, such as fetching detailed information only for places identified in a prior nearby search.

To address these challenges, we propose **MapAgent**, a hierarchical plug-and-play multi-agent framework designed for map-integrated geospatial reasoning. First, to overcome the issue of *tool incapability*, we design a set of four heterogeneous map tools, each composed of one or more primitive APIs tailored to perform key geospatial operations. These tools encapsulate common spatial functionalities—such as nearby search, route planning, and place detail retrieval—and are constructed to support both parallel and sequential execution flows. By abstracting low-level primitive API calls into higher-level tool interfaces, MapAgent enables robust handling of mixed-mode reasoning: for example, retrieving details of candidate places found via a nearby search in a sequential pipeline, while simultaneously computing alternative routes in parallel. This structured orchestration supports the complex, map-centric reasoning patterns required for real-world spatial tasks.

Second, to mitigate the problem of *tool inflation*, MapAgent adopts a hierarchical architecture that decouples high-level task planning from low-

Method	Parallel Tool Calling	Compositional Reasoning	Hierarchical	Plug-n-play or Map-tools
ReAct (Yao et al., 2023)	✗	✗	✗	✗
LLMCompiler (Kim et al., 2024)	✓	✗	✗	✗
Octotools (Lu et al., 2025)	✓	✗	✗	✓
Chameleon (Lu et al., 2023)	✗	✓	✗	✓
Anytool (Du et al., 2024)	✗	✗	✓	✗
MapAgent	✓	✓	✓	✓

Table 1: Feature comparison of different tool calling frameworks in context to map-integrated geospatial reasoning.

level tool execution. A top-level planner agent decomposes complex natural language queries into structured subgoals and routes them to appropriate tool workflows. For tool-intensive subgoals—such as those involving map services—we introduce a dedicated *map-tool agent* that adaptively manages interactions with multiple map tools, issuing parallel API calls where appropriate and coordinating sequential workflows when required. In contrast, lightweight tasks—such as answer formatting or preference synthesis—are handled directly without additional agent overhead.

We evaluate MapAgent across four diverse geospatial reasoning benchmarks—MapQA, MapEval-Textual, MapEval-Visual, and MapEval-API—each presenting distinct challenges, including textual context, multimodal inputs, and query-only settings. Built using both open-weight and closed-source backbone LLMs, MapAgent consistently outperforms prior state-of-the-art agentic and tool-augmented LLM frameworks across all benchmarks. Utilizing GPT-3.5-Turbo, it achieves a 10% improvement on both the MapEval-API and MapEval-Textual datasets, and an 11.22% improvement on the MapQA dataset over the strong baseline OctoTools (Lu et al., 2025). Furthermore, with GPT-4o, MapAgent achieves a 4.41% improvement on the MapEval-Visual dataset compared with the same baseline. Overall, MapAgent improves performance by 8.2% on average over the strongest baseline, OctoTools, in geospatial tasks that require multi-hop planning, spatial inference, and dynamic tool use. Our ablations, qualitative, and fine-grained analyses further provide more insights into the effectiveness of MapAgent .

2 Related Work

Compositional Reasoning with LLMs. Solving complex multi-step problems often involves decomposing tasks into manageable sub-problems (Perez et al., 2020; Khot et al., 2023). Prompt-based methods like Chain-of-Thought (Wei et al., 2022), Least-to-Most (Zhou et al., 2023), ReAct (Yao et al., 2023), Pearl (Sun et al., 2024), Forest-of-Thought (Bi et al., 2024), and rStar-Math (Guan et al., 2025) have advanced the ability of LLMs to reason sequentially. Other approaches synthesize structured programs from natural language (e.g., LLM Compiler (Kim et al., 2024)) or orchestrate modular tools (e.g., Chameleon (Lu et al., 2023), OctoTools (Lu et al., 2025)). While effective, these systems often assume flat or static module inventories and struggle with tool variants that differ subtly in schema or functionality. In contrast, MapAgent leverages hierarchical planning and dynamic module selection to support plug-and-play coordination over overlapping but heterogeneous APIs, particularly in the geospatial domain.

Geospatial Reasoning with Language Models. Geospatial QA has been studied through rule-based and template-driven systems that convert natural language into structured queries over static databases (e.g., PostGIS, DBpedia (Auer et al., 2007), YAGO (Suchanek et al., 2007), Freebase (Bollacker et al., 2007), OSM). Notable examples include GeoSPARQL-based systems (Car et al., 2022) and datasets such as GeoQuestions1089 (Kefalidis et al., 2023) and YAGO2geo (Karalis et al., 2019). These methods are precise but inflexible, limited by their reliance on static schemas and query grammars.

Recent work explores LLMs' intrinsic geographic knowledge (e.g., GPT4GEO (Roberts et al., 2023)) or their capacity to answer map-related queries directly (e.g., MapQA (Li et al., 2025), MapEval (Dihan et al., 2025)). While promising, these models lack mechanisms to integrate external tools or coordinate multi-API workflows. MapAgent builds on this line of work by introducing a multi-agent, tool-augmented framework that dynamically coordinates geospatial modules across modalities—including textual, visual, and API inputs—offering enhanced reasoning capabilities for real-world map-based tasks.

3 Method

We propose **MapAgent**, a hierarchical plug-and-play multi-agent framework for map-integrated geospatial reasoning. MapAgent addresses two core limitations in current agentic systems—*tool incapability* and *tool inflation*—through a structured two-layer scaffold that cleanly separates reasoning, planning, and tool execution.

3.1 Overview

We introduce **MapAgent**, a hierarchical multi-agent framework for solving complex geospatial reasoning tasks via real-world map services. Real-world queries often require chaining spatial operations, querying external APIs, and integrating geographic evidence—capabilities that single-shot prompting lacks. To address this, MapAgent enables structured planning, map-tool composition, and spatial reasoning through specialized agents orchestrated in a hierarchical coordination.

In MapAgent, a top-level *planner agent* $\mathcal{P}$ first decomposes the user query x into subgoals, enabling modular execution and interpretable reasoning. These are handled by specialized *functional modules* M, which delegate tasks such as filtering or aggregation to dedicated components. Figure 1 shows an overview of MapAgent.

Geospatial subgoals are routed to a *Map-Tool Agent* $\mathcal{M}_{\text{map}}$, which composes four core tools `Nearby`, `PlaceInfo`, `Route`, and `Trip`—each built over real-world Google Map APIs to support spatial search, detail retrieval, and routing. This modular design allows MapAgent to solve complex, multi-hop spatial queries by dynamically planning over tool-based workflows. We describe the architecture in Section 3.2, the planner agents and modules in Section 3.3, the map tools in Section 3.4.

3.2 Scaffolded Architecture (Hierocracy)

MapAgent comprises two conceptual layers:

Planner Layer (Top-level): A planner agent decomposes the input x into a sequence of semantically coherent subgoals $[g_1, \dots, g_n]$, and routes each g_i to a functional module or tool chain based on a structured inventory (discussed in Table 8).

Execution Layer (Bottom-level): This layer manages module-specific execution logic. It includes both lightweight modules (e.g., sequencing, formatting) and a specialized *Map-Service Module*, which integrates a dedicated **Map-Tool Agent** responsible for executing geospatial tasks using a curated set of map tools.

3.3 Module Inventory and Planner Agent

The planner agent $\mathcal{P}$ (based on an LLM or VLLM) receives the input x, an inventory M of available module to perform the subgoals, and outputs a structured execution plan $\pi = [(g_1, m_1), ..., (g_n, m_n)]$, where each g_i is a subgoal assigned to a module $m_i \in M$. Module Inventory M contains several task-specific modules:

Visual Place Recognizer: Given an image map-snapshot, using the corresponding VLLM, this module extracts the central geographic place (name) in the image and estimates the geographic scope (approximate radius) covered in the image.

Sequencer: This module is responsible for organizing and structuring the responses received from preceding modules. It arranges unstructured information into a logical sequence using LLMs.

Solution Generator: This module synthesizes the final answer to the user query using the corresponding LLM or VLM. It takes structured information and generates a comprehensive response.

Answer Generator: This module focuses on refining and verifying the answer generated by the Solution Generator. It extracts the predicted answer and checks its consistency.

Map-Service Module: A tool-heavy module managed by the Map-Tool Agent to handle core Map-tool interactions.

3.4 Map-Service Module: Map-Tool Agent and Tool Design

Map-Tool Agent $\mathcal{M}_{\textbf{map}}$ takes as input the query x, along with an optional place name and radius inferred by the Visual Place Recognizer for multimodal queries. It then adaptively interacts with a suite of geospatial tools designed to fetch the necessary spatio-temporal data from underlying map services (e.g., Google Maps).

We design four heterogeneous **map tools**, each composed of one or more primitive APIs from Google Map (e.g., directions, place info) to support essential geospatial operations:

Trip Tool: Retrieves information about locations and travel routes between them. Combines the Place Details and Directions APIs to fetch place metadata and step-by-step route instructions.

Route Tool: Fetches route information between

LLM	MapEval-API						
	Method	Overall	Place Info	Nearby	Routing	Trip	Unanswerable
GPT-3.5-Turbo	ReAct	27.33	39.06	22.89	33.33	19.40	15.00
	LLMCompiler	44.75	53.13	51.80	48.48	37.10	0.00
	Chameleon	50.45	54.54	54.10	52.00	43.07	5.00
	Octotools	56.27	76.56	62.65	56.06	40.30	5.00
	MapAgent	**66.31**	**88.24**	**77.05**	**68.18**	**49.23**	**20.00**
Qwen-2.5-72B	ReAct	41.56	55.44	38.25	42.40	32.75	0.00
	LLMCompiler	49.63	59.25	55.29	51.23	40.77	0.00
	Chameleon	52.70	61.30	60.29	55.77	46.55	10.00
	Octotools	59.40	80.95	71.08	59.09	42.42	0.00
	MapAgent	**70.94**	**89.40**	**82.85**	**69.76**	**54.54**	**25.00**

Table 2: Accuracy (%) of different methods and backbones on the MapEval-API Dataset.

LLM	MapEval-Textual						
	Method	Overall	Place Info	Nearby	Routing	Trip	Unanswerable
GPT-3.5-Turbo	Base Model	37.67	26.56	53.01	48.48	28.36	5.00
	ReAct	42.25	42.90	48.18	50.67	35.34	0.00
	LLMCompiler	49.23	58.44	56.98	53.33	40.81	0.00
	Chameleon	56.83	60.77	62.50	57.20	47.38	5.00
	Octotools	62.78	84.22	68.92	61.67	44.33	5.00
	MapAgent	**72.94**	**90.06**	**82.76**	**73.23**	**54.15**	**20.00**
Qwen-2.5-72B	Base Model	57.00	62.50	71.08	63.64	41.79	10.00
	ReAct	58.44	64.44	68.25	56.80	46.28	5.00
	LLMCompiler	62.58	74.40	75.38	64.22	48.32	5.00
	Chameleon	65.75	76.40	72.54	70.60	52.26	10.00
	Octotools	70.25	85.55	77.22	67.20	53.30	10.00
	MapAgent	**76.24**	**94.20**	**84.75**	**74.80**	**59.20**	**20.00**

Table 3: Accuracy (%) of different methods and backbones on the MapEval-Textual Dataset.

two points using the Directions API, including distance, estimated time, and navigation steps.
Nearby Tool: Identifies points of interest within a specified area using the Nearby Search API. Returns place names, ratings, and other metadata.
PlaceInfo Tool: Retrieves detailed information about a given place using the Place Details API, including address, hours, contact info, and reviews.

4 Experimental Setup

4.1 Datasets

We evaluated MapAgent on four benchmarks spanning two modalities as shown in Table 4. These benchmarks encompass a wide range of complex geospatial reasoning tasks, including long-context reasoning, API interactions, visual map analysis, numerical calculation, and multi-step reasoning. Three benchmarks are from MapEval (Dihan et al., 2025):

MapEval-Textual: Given a user query with a long textual context describing map locations, POIs, routes, navigation details, travel distances/times, and user-generated content such as ratings or reviews, the task is to answer the query based on the context.

Dataset	# Tasks	Modality
MapEval-Textual	300	Text
MapEval-API	300	Text
MapEval-Visual	400	Vision
MapQA	3154	Text

Table 4: Overview of datasets: statistics and modality

MapEval-API: Given a user query and access to map-based APIs, the task is to answer the query based on the structured data retrieved via the APIs.
MapEval-Visual: Given a user query and a digital map snapshot showing spatial layouts, routes, landmarks, OCR text (e.g., ratings), and symbolic elements (e.g., icons or signs), the task is to answer the query based on the visual context.

VLM	MapEval-Visual						
	Method	Overall	Place Info	Nearby	Routing	Counting	Unanswerable
GPT-4o	Base Model	58.90	76.86	57.78	50.00	47.73	**40.00**
	ReAct	58.50	78.28	58.24	52.29	53.11	0.00
	LLMCompiler	61.88	81.29	60.80	54.42	55.22	0.00
	Chameleon	62.51	82.44	62.77	55.67	57.18	5.00
	Octotools	64.54	84.20	**64.53**	53.44	61.36	5.00
	MapAgent	**68.95**	**88.64**	63.54	**62.50**	**63.63**	25.00
Qwen-2.5-VL-72B	Base Model	60.35	76.86	54.44	43.04	52.33	**60.00**
	ReAct	59.18	74.54	54.40	56.40	55.40	10.00
	LLMCompiler	64.28	82.50	61.80	58.88	59.80	10.00
	Chameleon	66.40	83.46	64.32	59.30	62.35	15.00
	Octotools	69.50	88.78	**66.48**	63.22	64.28	20.00
	MapAgent	**72.30**	**89.45**	66.25	**67.50**	**66.50**	40.00

Table 5: Accuracy (%) of different methods and backbones on the MapEval-Textual Dataset.

MapQA (Li et al., 2025): A geospatial QA benchmark constructed from OpenStreetMap using SQL query templates. It includes QA pairs covering nine reasoning types (e.g., neighborhood inference, spatial proximity, type classification), grounded in real-world geo-entity geometries.

4.2 Baseline and Metric

MapAgent is a multi-agent compositional reasoning framework. We therefore compare its performance against agentic frameworks, such as ReAct (Yao et al., 2023) and LLMCompiler (Kim et al., 2024), and compositional reasoning frameworks, such as Chameleon (Lu et al., 2023) and OctoTool (Lu et al., 2025). For the MapEval-API , MapEval-Textual and MapQA datasets, we use GPT-3.5-Turbo (OpenAI, 2022) and Qwen-2.5-72b (Team, 2024). For the MapEval-Visual dataset, we use models served via vLLM, specifically GPT-4o (OpenAI, 2024) and Qwen-2.5-VL-72b (Bai et al., 2025). We evaluate model performance using accuracy, expressed as a percentage.

4.3 Results

The evaluation of the MapAgent framework across four distinct and challenging map-related datasets reveals its significant advancements in accuracy, consistently outperforming existing agentic approaches and demonstrating robust performance irrespective of the underlying language model backbone. These findings underscore MapAgent 's effectiveness as a robust solution for a broad spectrum of map-based query processing tasks.

Strong Gains Across Benchmarks Across all evaluated datasets, the MapAgent framework consistently achieves high accuracy. On the MapEval-API dataset in Table 2, MapAgent achieves overall accuracies of roughly 70% and 72% when powered by `GPT-3.5-Turbo` and `Qwen-2.5-72B`, respectively. Similarly, on the MapEval-Textual dataset in Table 3, MapAgent attains accuracies of 72.94% with `GPT-3.5-Turbo` and 76.24% with `Qwen-2.5-72B`. The MapEval-Visual results in Table 5 further reinforce this trend, with MapAgent achieving the highest overall accuracy–68.95% using `GPT-4o` and 72.30% with `Qwen-2.5-VL-72B`. Finally, on the MapQA benchmark in Table 6, MapAgent achieves the highest overall accuracy of 55.58% with `GPT-3.5-Turbo` and 55.93% with `Qwen-2.5-72B`.

Surpassing Prior Methods The performance of MapAgent consistently surpasses that of other prominent methods, including OctoTools, Chameleon, ReAct, and LLMCompiler—notably the second-best performing OctoTools—across all evaluated datasets and language models. In Figure 6, on the MapEval-API dataset, MapAgent shows accuracy gains of approximately 10% and 11.54% over OctoTools when using `GPT-3.5-Turbo` and `Qwen-2.5-72B`, respectively. For the MapQA benchmark, MapAgent outperforms OctoTools by approximately 11.22% with `GPT-3.5-Turbo` and 9% with `Qwen-2.5-72B`. Lastly, on the MapEval-Textual dataset, MapAgent's accuracy is nearly 10% and 6% higher than OctoTools with `GPT-3.5-Turbo` and `Qwen-2.5-72B`. In the MapEval-Visual evaluations, MapAgent outperforms OctoTools by about 4.41% with `GPT-4o` and 2.8% with `Qwen-2.5-VL-72B`.

LLM	MapQA										
	Method	Overall	Type 1	Type 2	Type 3	Type 4	Type 5	Type 6	Type 7	Type 8	Type 9
GPT-3.5-Turbo	Base Model	24.35	32.04	7.43	5.98	50.56	7.22	31.44	18.34	10.14	56.04
	ReAct	20.88	34.02	7.80	5.44	58.02	6.02	30.13	12.14	9.88	24.44
	LLMCompiler	23.99	36.07	8.09	6.29	67.02	7.28	31.32	14.76	14.24	30.88
	Chameleon	38.11	47.52	17.59	20.77	73.14	15.52	40.12	14.28	43.68	70.40
	Octotools	44.74	50.00	20.40	22.00	**90.67**	18.52	43.78	41.20	48.24	67.83
	MapAgent	**55.58**	**56.05**	**25.80**	**22.48**	83.36	**25.68**	**71.04**	**54.18**	**71.20**	**90.40**
Qwen-2.5-72B	Base Model	23.51	29.60	8.40	7.20	48.00	**7.60**	30.00	17.20	9.20	**54.40**
	ReAct	21.69	30.80	9.20	7.60	51.60	9.40	32.80	16.40	8.60	28.80
	LLMCompiler	25.60	38.40	9.20	8.00	62.40	11.60	33.20	17.60	18.40	31.60
	Chameleon	40.00	51.20	27.60	23.20	68.80	22.40	38.40	20.40	38.80	68.40
	Octotools	46.18	54.39	**30.91**	25.80	**85.67**	24.44	41.60	38.20	43.24	71.33
	MapAgent	**55.93**	**57.20**	29.60	**34.00**	81.20	34.56	73.60	**39.20**	**68.00**	86.00

Table 6: Accuracy (%) of different methods and backbones on the MapQA Dataset.

Backbone Agnostic A key strength of the MapAgent framework is its consistent high performance across different language model backbones. When evaluated on the MapEval-API and MapEval-Textual datasets, MapAgent demonstrates superior accuracy with both `GPT-3.5-Turbo` and `Qwen-2.5-72B` models. This backbone agnosticism is further evident in the MapEval-Visual dataset, where MapAgent achieves the highest accuracy with both `GPT-4o` and `Qwen-2.5-VL-72B` vision-language models. The consistent top-tier performance across these diverse LLMs and VLMs underscores the robustness and adaptability of the MapAgent framework, indicating that its effectiveness is not tied to a specific underlying model architecture.

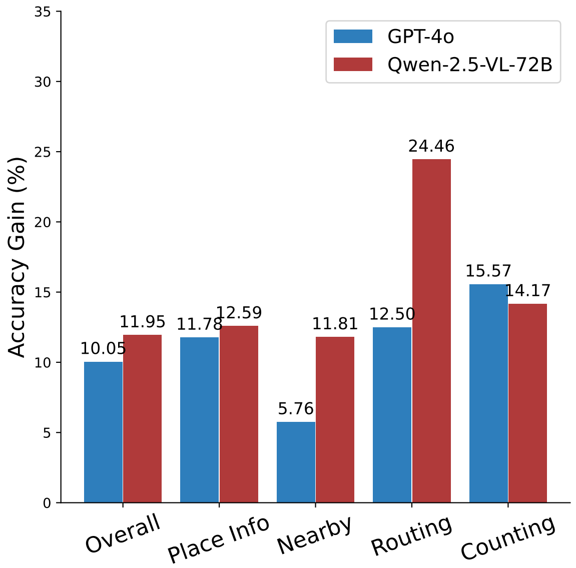


Figure 3: Performance gain with visual images using the API.

5 Ablation and Qualitative Analysis

5.1 Why does using visuals with an API perform better than simply passing visual questions to VLLM?

When MapAgent attempts to solve a visual query, it utilizes Visual Place Recognizer module to obtain information about the image's center and estimates its boundaries. Subsequently, the Google Maps module uses this information to fetch relevant data for resolving the query. This additional information enhances the accuracy of the VLLM's decision-making process. Consequently, evaluating visual queries with an API yields improved performance. Figure 3 supports this analysis by illustrating the performance gain achieved when using the API with a map image compared to relying solely on the map to solve the query with a VLLM.

Notably, in the routing category, Qwen-2.5-VL-72B exhibits a significant 24.46% accuracy gain (as shown in Fig. 3). This improvement is attributed to the difficulty in determining the optimal path by merely examining a map image. However, obtaining path information through route tools and calculating the optimal path based on time and distance is straightforward. This capability assists the VLLM in identifying the correct route. Listings 6 and 7 illustrate two scenarios where the response generated using MapAgent yields correct results, whereas the VLLM, without API assistance, fails to produce accurate outcomes.

5.2 Why does API-using MapAgent perform better on contextual text data?

Figure 4 illustrates the accuracy gain achieved by incorporating API calls with textual context. No-

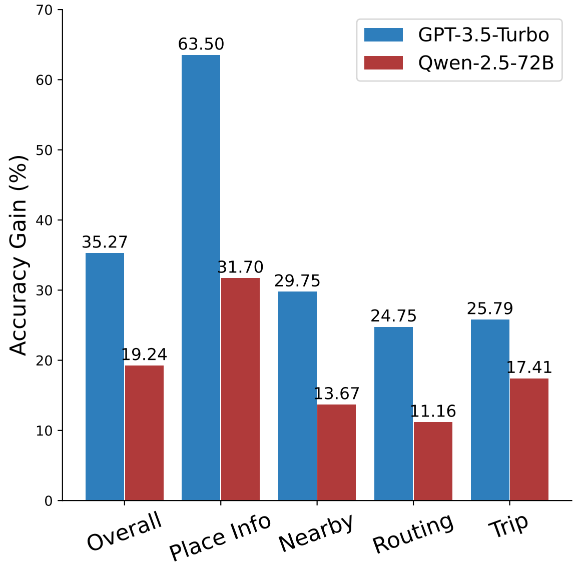


Figure 4: Performance gain of the model using both contextual text and API data compared to using only contextual text.

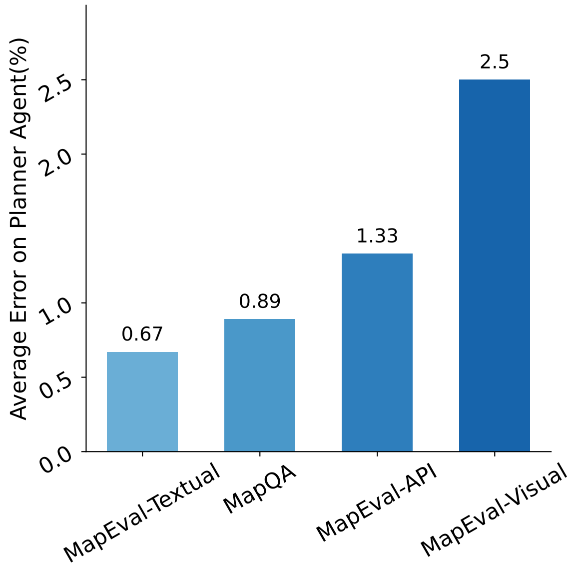


Figure 5: Average Error(%) in Planner Agent's Selection of Modules across 4 geo-spatial benchmarks.

tably, for the GPT-3.5-Turbo model, the overall gain is approximately 35.27%, while for the Qwen-2.5-72b model, it is 19.24%. The primary reason for this improvement is that the API can specifically retrieve information relevant to the question. In contrast, textual context may contain extraneous details, making it non-trivial to extract the precise information needed for an accurate answer. Furthermore, the context might lack essential information that can be obtained through the API. Therefore, to effectively address the query, the API plays a crucial role by providing supplementary relevant information. This additional input assists the LLM in predicting a more accurate response.

For qualitative examples illustrating agent behavior, please refer to Appendix D.

6 Error Analysis and Challenges

To identify the remaining challenges of MapAgent , we conducted a more granular examination to uncover specific types of errors occurring at different stages of the framework. First, we observed that in several instances, the Planner Agent fails to select the appropriate set of modules from the module inventory (see Fig. 5). The Map-Tool Agent also occasionally struggles to fetch the correct tool and, at times, invokes tools with incorrect parameters. Additionally, this agent faces challenges in accurately extracting the required parameters and passing them to the intended tools. We also noted several instances where the Answer Generator encounters difficulties in correctly parsing the final answer from the solution output. During the evaluation of MapEval-Visual, we identified that the Visual Place Recognizer introduces another potential source of error, with cases where it fails to correctly predict the image center or produces outputs in an unexpected format. These multifaceted challenges across various agents within MapAgent highlight key areas where targeted improvements can enhance the system's overall performance and robustness.

7 Conclusion

In this paper, we present MapAgent, a hierarchical multi-agent framework designed for effective geospatial reasoning. By decoupling planning from execution and introducing specialized modules with customized toolset–such as a dedicated map-tool agent–MapAgent addresses the limitations of flat agent-based architectures that struggle with tool overload and fine-grained API coordination. Through comprehensive evaluations on four challenging geospatial benchmarks—MapEval-Textual, MapEval-API, MapEval-Visual, and MapQA-MapAgent consistently outperforms existing tool-augmented and agentic baselines, including Chameleon and OctoTools. These results demonstrate MapAgent 's effectiveness and generalizability in real-world, map-integrated reasoning tasks. In future work, we plan to extend the framework to broader multimodal and spatial-temporal reasoning tasks.

Limitations

While our proposed hierarchical multi-agent framework, MapAgent, demonstrates substantial gains over state-of-the-art tool-augmented and agentic baselines for a variety of geospatial queries, it currently relies on a limited set of popular Google Maps APIs. Extending the framework to support a broader range of APIs and mapping services would further validate its generalizability and robustness. Additionally, although MapAgent features a plug-and-play architecture with the potential to be applied to other domains, such as web automation and software engineering, its effectiveness in these areas remains to be evaluated.

References

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *international semantic web conference*, pages 722–735. Springer.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, and 1 others. 2025. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*.

Zhenni Bi, Kai Han, Chuanjian Liu, Yehui Tang, and Yunhe Wang. 2024. Forest-of-thought: Scaling test-time compute for enhancing llm reasoning. *arXiv e-prints*, pages arXiv–2412.

Kurt Bollacker, Robert Cook, and Patrick Tufts. 2007. Freebase: A shared database of structured general human knowledge. In *AAAI*, volume 7, pages 1962–1963.

Nicholas J Car, Timo Homburg, Matthew Perry, John Herring, Frans Knibbe, Simon JD Cox, Joseph Abhayaratna, and Mathias Bonduel. 2022. Ogc geosparql-a geographic query language for rdf data. *OGC Implementation Standard., Open Geospatial Consortium*.

Mahir Labib Dihan, Md Tanvir Hassan, Md Tanvir Parvez, Md Hasebul Hasan, Md Almash Alam, Muhammad Aamir Cheema, Mohammed Eunus Ali, and Md Rizwan Parvez. 2025. Mapeval: A map-based evaluation of geo-spatial reasoning in foundation models. In *Forty-second International Conference on Machine Learning*.

Yu Du, Fangyun Wei, and Hongyang Zhang. 2024. Anytool: Self-reflective, hierarchical agents for large-scale api calls. In *International Conference on Machine Learning*, pages 11812–11829. PMLR.

Xinyu Guan, Li Lyna Zhang, Yifei Liu, Ning Shang, Youran Sun, Yi Zhu, Fan Yang, and Mao Yang. 2025. rstar-math: Small llms can master math reasoning with self-evolved deep thinking. *arXiv preprint arXiv:2501.04519*.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. 2023. Swe-bench: Can language models resolve real-world github issues? *arXiv preprint arXiv:2310.06770*.

Nikolaos Karalis, Georgios Mandilaras, and Manolis Koubarakis. 2019. Extending the yago2 knowledge graph with precise geospatial knowledge. In *The Semantic Web–ISWC 2019: 18th International Semantic Web Conference, Auckland, New Zealand, October 26–30, 2019, Proceedings, Part II 18*, pages 181–197. Springer.

Sergios-Anestis Kefalidis, Dharmen Punjani, Eleni Tsalapati, Konstantinos Plas, Mariangela Pollali, Michail Mitsios, Myrto Tsokanaridou, Manolis Koubarakis, and Pierre Maret. 2023. Benchmarking geospatial question answering engines using the dataset geoquestions1089. In *International semantic web conference*, pages 266–284. Springer.

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2023. Decomposed prompting: A modular approach for solving complex tasks. In *The Eleventh International Conference on Learning Representations*.

Sehoon Kim, Suhong Moon, Ryan Tabrizi, Nicholas Lee, Michael W Mahoney, Kurt Keutzer, and Amir Gholami. 2024. An llm compiler for parallel function calling. In *Forty-first International Conference on Machine Learning*.

Zekun Li, Malcolm Grossman, Mihir Kulkarni, Muhao Chen, Yao-Yi Chiang, and 1 others. 2025. Mapqa: Open-domain geospatial question answering on map data. *arXiv preprint arXiv:2503.07871*.

Pan Lu, Bowen Chen, Sheng Liu, Rahul Thapa, Joseph Boen, and James Zou. 2025. Octotools: An agentic framework with extensible tools for complex reasoning. In *ICLR 2025 Workshop on Foundation Models in the Wild*.

Pan Lu, Baolin Peng, Hao Cheng, Michel Galley, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, and Jianfeng Gao. 2023. Chameleon: Plug-and-play compositional reasoning with large language models. *Advances in Neural Information Processing Systems*, 36:43447–43478.

OpenAI. 2022. Chatgpt.

OpenAI. 2024. https://openai.com/index/hello-gpt-4o/.

Ethan Perez, Patrick Lewis, Wen-tau Yih, Kyunghyun Cho, and Douwe Kiela. 2020. Unsupervised question decomposition for question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8864–8880.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, and 1 others. 2023. Toolllm: Facilitating large language models to master 16000+ real-world apis. *arXiv preprint arXiv:2307.16789*.

Jonathan Roberts, Timo Lüddecke, Sowmen Das, Kai Han, and Samuel Albanie. 2023. GPT4GEO: How a language model sees the world's geography. In *NeurIPS 2023 Foundation Models for Decision Making Workshop*.

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706.

Simeng Sun, Yang Liu, Shuohang Wang, Dan Iter, Chenguang Zhu, and Mohit Iyyer. 2024. Pearl: Prompting large language models to plan and execute actions over long documents. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 469–486.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V Le, and Ed H. Chi. 2023. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations*.

A How MapAgent Outperforms Chameleon and OctoTools

Appendices A.1, A.2, and A.3 illustrate an example where OctoTools and Chameleon failed to solve a complex geospatial query, but MapAgent could. The main reason behind this failure is that both OctoTools and Chameleon have all their module or tool cards in one layer. Consequently, they have trouble guessing the exact tool/module for fetching information. The example illustrates that solving this query requires the distance and travel time from one location to another for each location in the query, which a trip tool can deliver correctly. However, both Chameleon and OctoTools struggle here and select a route tool to fetch the information, subsequently lacking information about all the routes between all the places. Additionally, OctoTools encounters another limitation. As shown in Table 7, where the intermediate reasoning steps of OctoTools are displayed, we found that this framework hallucinates and triggers a tool with parameters that have no relation to this question, which yields wrong information. These limitations trigger a failure to provide all the required information to the subsequent steps. However, MapAgent can effectively solve this by introducing an inner layer in the Google Map module. This module, communicating with a map-tool agent, appropriately triggers the trips tool and retrieves the relevant information that is sufficient for the subsequent steps to determine the solutions.

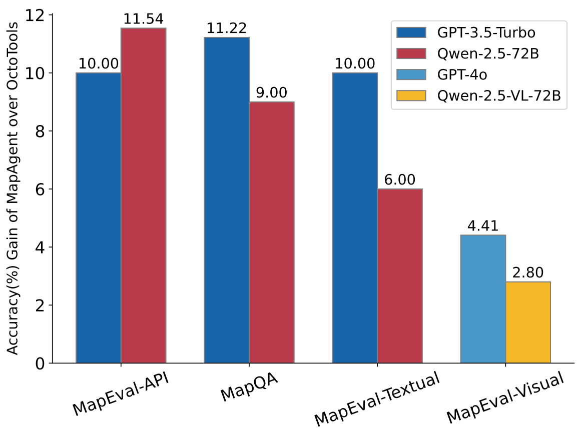


Figure 6: Accuracy improvements of MapAgent over OctoTools on various benchmarks.

A.1 Octotools Output Green: Correct Answer Red Wrong Answer

- **PID:** 19
- **Query:** I am at Rydges Wellington. Starting from my current location, I will first go to Kelburn Park, followed by a trip to Victoria University of Wellington, and finally return to my current location. All the travel will be done using my own car. How much time will the entire journey take?
- **Options:** A.11 mins B.10 mins C.12 mins D.9 mins
- **Correct Answer:** B
- **Reasoning Steps:** Table 7 shows the reasoning steps of OctoTools and the tools it executes in each step. Within the table, red highlights indicate hallucinations, and orange highlights denote incorrect information that OctoTools attempts to fetch after execution. We observe that the initial call to the Route Tool, using the same origin and destination, leads to inaccurate information retrieval. Furthermore, the subsequent nine calls, shown in table rows 2 through 5, demonstrate hallucination and the use of parameters unrelated to the question, revealing a lack of correct tool calling. Although a trip tool capable of providing all relevant information in a single call exists, it is not executed correctly.
- **Octotools Final Output:** To determine the total travel time for the round trip starting and ending at Rydges Wellington, with stops at Kelburn Park and Victoria University of Wellington, the following steps were taken:

  Step 1: Analyze the Query The query specifies a round trip: 1. Start at **Rydges Wellington**. 2. Visit **Kelburn Park**. 3. Proceed to **Victoria University of Wellington**. 4. Return to **Rydges Wellington**. The mode of transport is driving.

  Step 2: Use the Route Tool The **Route Tool** was used to calculate the total driving time for the specified route. The tool provided the following result: - **Total Route Distance**: 1.9 km. - **Total Driving Time**: 9 minutes.

  Step 3: Verify the Route Details The route details confirm the driving time includes all segments of the trip: - Rydges Wellington → Kelburn Park. - Kelburn Park → Victoria University of Wellington. - Victoria University of Wellington → Rydges Wellington.

Conclusion: The total driving time for the entire journey is **9 minutes**.

Final Answer: D. 9 mins

A.2 Chameleon Output Green: Correct Answer Red Wrong Answer

- **PID:** 19
- **Query:** I am at Rydges Wellington. Starting from my current location, I will first go to Kelburn Park, followed by a trip to Victoria University of Wellington, and finally return to my current location. All the travel will be done using my own car. How much time will the entire journey take?
- **Options:** A.11 mins B.10 mins C.12 mins D.9 mins
- **Correct Answer:** B
- **Selected Module by the Planner:** Route, solution_generator, answer_generator
- **Information gathered by Route Module:** There are total 1 routes from Rydges Wellington, Wellington 6011, New Zealand to Victoria University of Wellington, Wellington 6012, New Zealand. The route information is provided below:
  Route 1:(VIA) Parkvale Rd (7.6 km, 20 mins).
  Details steps are provided below:
  Head southwest on Featherston St toward Whitmore St (35 m, 1 min)
  Turn right at the 1st cross street onto Whitmore St (0.2 km, 1 min)
  Continue onto Bowen St (0.8 km, 2 mins)
  Turn left onto Tinakori Rd (signs for Karori/Botanic Garden) (0.2 km, 1 min)
  Continue onto Glenmore St (1.6 km, 3 mins)
  At the roundabout, take the 1st exit and stay on Glenmore St (0.2 km, 1 min)
  Continue onto Karori Tunnel (0.1 km, 1 min)
  Continue onto Chaytor St (0.9 km, 2 mins)
  Continue onto Karori Rd (1.2 km, 2 mins)
  Turn right onto Parkvale Rd (1.1 km, 2 mins)
  Turn right to stay on Parkvale Rd (0.3 km, 1 min)
  Turn right to stay on Parkvale Rd (1.1 km, 5 mins)
- **Solution Generated by Solution Generator:** It seems there's a mismatch in the information provided. The detailed route you've given is for travel between Rydges Wellington and Victoria University of Wellington, taking approximately 20 minutes. You haven't provided a detailed route or travel time for the leg between Rydges Wellington and Kelburn Park, or for the return trip from Victoria University of Wellington back to Rydges Wellington via Kelburn Park. Based solely on the 20-minute travel time you just provided for Rydges to Victoria University (Route 1), we cannot accurately answer the question about the entire journey involving Kelburn Park without knowing the travel times for the other segments. However, if we were to ignore the previous travel times you mentioned and assume the question implies a direct trip from Rydges to Victoria University and back, using the 20-minute route in each direction, the total would be 20 minutes + 20 minutes = 40 minutes. This doesn't align with the provided options. So, the answer should be E.
- **Answer Generator Module Output:** Not able to answer the question

A.3 MapAgent Output Green: Correct Answer Red Wrong Answer

- **PID:** 19
- **Query:** I am at Rydges Wellington. Starting from my current location, I will first go to Kelburn Park, followed by a trip to Victoria University of Wellington, and finally return to my current location. All the travel will be done using my own car. How much time will the entire journey take?
- **Options:** A.11 mins B.10 mins C.12 mins D.9 mins
- **Correct Answer:** B
- **Selected Module by the Planner Agent:** map_service, solution_generator, answer_generator
- **Information gathered by Google_Maps Module:**
  All Location Info:
  Rydges Wellington
  Name: Rydges Wellington
  Address: 75 Featherston Street, Pipitea, Wellington 6011, New Zealand

Rating: 4.2
Types: lodging, point_of_interest, establishment
Is Open Now: True
Weekday Opening Hours:
- Monday: Open 24 hours
- Tuesday: Open 24 hours
- Wednesday: Open 24 hours
- Thursday: Open 24 hours
- Friday: Open 24 hours
- Saturday: Open 24 hours
- Sunday: Open 24 hours
Kelburn Park
Name: Kelburn Park
Address: Salamanca Road, Kelburn, Wellington 6012, New Zealand
Rating: 4.4
Types: park, tourist_attraction, point_of_interest, establishment
Is Open Now: True
Weekday Opening Hours:
- Monday: Open 24 hours
- Tuesday: Open 24 hours
- Wednesday: Open 24 hours
- Thursday: Open 24 hours
- Friday: Open 24 hours
- Saturday: Open 24 hours
- Sunday: Open 24 hours
Victoria University of Wellington
Name: Victoria University of Wellington
Address: Kelburn Parade, Kelburn, Wellington 6012, New Zealand
Rating: 4.3
Types: university, point_of_interest, establishment
Is Open Now: N/A
Weekday Opening Hours:
- Unknown
The travel time(distance) from Rydges Wellington to Kelburn Park is 5 mins (1.6 km)
The travel time(distance) from Rydges Wellington to Victoria University of Wellington is 1 min (1 m)
The travel time(distance) from Kelburn Park to Rydges Wellington is 4 mins (1.4 km)
The travel time(distance) from Kelburn Park to Victoria University of Wellington is 4 mins (1.4 km)
The travel time(distance) from Victoria University of Wellington to Rydges Wellington is 1 min (1 m)
The travel time(distance) from Victoria University of Wellington to Kelburn Park is 5 mins (1.6 km)

- **Solution Generated by Solution Generator Module:** To calculate the total time for the entire journey, we need to consider the travel time from Rydges Wellington to Kelburn Park, then from Kelburn Park to Victoria University of Wellington, and finally the return trip from Victoria University of Wellington to Rydges Wellington.

  The travel time from Rydges Wellington to Kelburn Park is 5 minutes. The travel time from Kelburn Park to Victoria University of Wellington is 4 minutes. The return trip from Victoria University of Wellington to Rydges Wellington is 1 minutes.

  Adding these times together, the total time for the entire journey is 5 + 4 + 1 = 10 minutes. So, the answer is 10 minutes. The most possible option is "the answer is B

- **Answer Generated by Answer Generator Module:** B. 10 mins

B Configurations in MapAgent

B.1 Map-Tool Agent

Prompt for Map-Tool Agent

You are an agent designed to understand user questions and retrieve relevant information using specific tools. When a user asks a question, your task is to identify the appropriate tool from the available list (Trip Tool, Route Tool, Nearby, and PlaceInfo) that can best answer the query. You will then use that tool to fetch the information and provide the retrieved data to the user. You are not expected to answer the question directly; your role is solely to retrieve the necessary information using the designated tools. For instance, if you think you need to use more than one tool, you can use them in parallel.

B.2 Solution Generator (MapEval-API)

Prompt for Solution Generator

Given the question (and the context), select the answer from the options ["A", "B", "C", "D"]. You should give concise and step-by-step solutions. Finally, conclude the answer in the format of "the answer is [ANSWER]", where [ANSWER] is one from the options ["A", "B", "C", "D"]. For example, "the answer is A", "the answer is B", "the answer is C", or "the answer is D". If the answer is not in the options, declare it as Unanswerable.
 # Example 1
Question: What is the address of Multiplan Center?
Options: (A) 69, 71 New Elephant Rd, Dhaka 1205, Bangladesh (B) 38/1/C BC DAS Street Lalbagh (C) Polashi,BUET (D) Central Road, USA
Metadata: {"skill":"Fetch context from corresponding google map api and based on the context answer the question"}
Google Maps response:
Name: Multiplan Center
Address: 69, 71 New Elephant Rd, Dhaka 1205, Bangladesh
Rating: 4.4
Types: point_of_interest, establishment
Is Open Now: False
Weekday Opening Hours:
- Monday: 10:00AM–8:00PM
- Tuesday: Closed
- Wednesday: 10:00AM–8:00PM
- Thursday: 10:00AM–8:00PM
- Friday: 10:00AM–8:00PM
- Saturday: 10:00AM–8:00PM
- Sunday: 10:00AM–8:00PM

Solution: If you look at the context and search, then after reaching The address of Multiplan Center is 69, 71 New Elephant Rd, Dhaka 1205, Bangladesh. Therefore, the answer is B.

.... <4 more examples> ....

Now Answer the question following.
 # Question: `{question}`

B.3 Visual Place Recognizer

Prompt for Visual Place Recognizer

I will provide you with an image. You must determine the precise center location within the image and a prediction of the boundary of the image by predicting a radius from the center. Your return format should be the center location name followed by a space, then the radius. Based on that, provide only one complete address in a single, consistent format followed by a space and a radius. Do not include any extraneous text before or after the address.

C Detailed Tool Parameters

The following section outlines the tools utilized by the Map Service module, as retrieved by the map-tool agent. Each tool is defined along with its input parameters.

Trip Tool

`current_location` `(str)`: The starting location of the trip.
`visiting_places` `(list)`: A list of locations to visit.
`travel_mode` `(str)`: The mode of travel, defaults to driving.

Route Tool

`origin` `(str)`: The starting location for the directions.
`destination` `(str)`: The destination for the directions.
`travel_mode` `(str)`: The mode of travel, defaults to driving.
`mode` `(str)`: The mode of transportation to use for the directions, such as "driving"
`alternatives` `(bool)`: Whether to return multiple possible routes.

Nearby Tool

`query` (str): The search term to look for geospatial places.
`location` (str): The name of the current location (e.g., Ibn Sina Hospital, Dhaka).
`type` (str): The type of place to search for, such as "restaurant", "cafe", or "hospital".
`radius` (float): The radius of the circular area for filtering, defaults to 20 Kilometer

PlaceInfo Tool

`location_address` (str): The address or name of the location to search for.

D Qualitative Examples

D.1 How MapAgent Answer a Query

Listing 1 illustrates how a MapAgent addresses a typical geospatial query. In this example, the query is taken from the trip category: "*I am currently at Planetshakers Church Melbourne City Campus. I want to drive to Carrot Tattoo Studio and then to Melbourne Lifestyle Apartments - Docklands Accommodation. How long will this take?*"

Initially, the query is passed to the Planner Agent, which analyzes it and selects three modules to solve it: google_maps, solution_generator, and answer_generator. Subsequently, the google_maps module is executed and attempts to fetch information from Google Maps by engaging its map-tool agent and relevant tools. The execution is then handed over to the map-tool agent. Based on the query, the map-tool agent selects the trip tool to retrieve the necessary information for this specific question. The trip tool utilizes the Directions and Place Details APIs to fetch information for solving this particular query. The map-tool agent then returns this information to the google_maps module. The Google Maps module then passes this information to the solution_generator. Based on the information fetched by google_maps and the query itself, the solution_generator attempts to formulate the answer with a proper explanation and hands this over to the answer_generator. Finally, after verifying the answer, the answer_generator extracts the final answer from the explanation provided by the solution_generator and presents the final output.

D.2 How MapAgent Answer a Query with Visual Context

Listings 2, 3, 4, 5 illustrate examples of how MapAgent handles geospatial queries that include a visual context, such as a snapshot of a location. To address this type of problem, the question and image are processed by the Planner Agent. The Planner Agent typically selects the Visual Place Recognizer Module, the `google_maps` module, the `solution_generator` module, and the `answer_generator` module. The Image Search module identifies the central location within the image and estimates its boundaries, often providing a radius from the central location. This location information, along with the original query, is then passed to the `google_maps` module. Google Maps, through its map-tool agent and tools that invoke the Google Maps API, fetches further relevant information from its inner layer. Finally, similar to the previous example, the solution is generated and refined.

Table 7: Detailed Reasoning Steps of Octotools

Step	Tool	Sub-goal	Command	Result Summary
Action Step 1	Route Tool	Calculate the total driving time for the round trip starting at Rydges Wellington, visiting Kelburn Park and Victoria University of Wellington, and returning to Rydges Wellington.	`execution=tool.execute(origin="RydgesWellington",destination="RydgesWellington",waypoints=["KelburnPark","VictoriaUniversityofWellington"],mode="driving",alternatives=False)`	Total 1 route. VIA Dixon St (1.9 km, 9 mins). Details include step-by-step directions.
Action Step 2	Route Tool	Calculate the travel time from the Trevi Fountain to the Vatican Museum using public transport.	`execution=tool.execute(origin="TreviFountain",destination="VaticanMuseum",mode="transit")`	Total 6 routes from Trevi Fountain to Vatican Museum via different public transport options with times ranging from 30 to 39 mins.
Action Step 3	Route Tool	Calculate the travel time for the transit leg from Kockelscheuer Ice Rink to Kallektuffquell using the Route_Tool.	`execution=tool.execute(origin="KockelscheuerIceRink",destination="Kallektuffquell",mode="transit")`	Total 6 routes from Kockelscheuer Ice Rink to Kallektuffquell via public transport with times ranging from 1 hour 50 mins to 2 hours 10 mins.
Action Step 4	Route Tool	Calculate the total driving time for the route from Planetshakers Church Melbourne City Campus to Melbourne Lifestyle Apartments - Docklands Accommodation with a stop at Carrot Tattoo Studio	`execution=tool.execute(origin="PlanetshakersChurchMelbourneCityCampus",destination="MelbourneLifestyleApartments-DocklandsAccommodation",waypoints=["CarrotTattooStudio"],mode="driving",alternatives=False)`	Total 1 route from Planetshakers Church to Melbourne Lifestyle Apartments (via Carrot Tattoo Studio) (2.0 km, 9 mins) with step-by-step directions.
Action Steps 5-10	Route Tool	Calculate the total driving time for the route from Hotel Saltel to Nil Ratan Sircar Medical College and Hospital with a stop at Science City - Kolkata.	`execution=tool.execute(origin="HotelSaltel",destination="NilRatanSircarMedicalCollegeandHospital",waypoints=["ScienceCity-Kolkata"],mode="driving",alternatives=False)`	Each step shows 1 route from Hotel Saltel to Nil Ratan Sircar Medical College (via Science City - Kolkata) (2.8 km, 7 mins) with step-by-step directions.

Module	Description
Map Service	Fetches geospatial data using a Coordinator Agent called Map-Tool Agent. Retrieves place details, routes, nearby locations, and other map-related information.
Visual Place Recognizer	Uses VLLM to analyze geospatial imagery. Identifies the central geographic place and estimates the geographic scope (approximate radius).
Sequencer	Organizes and structures the responses from preceding modules into a logical sequence.
Solution Generator	Uses a LLM or VLLM to synthesize the final answer based on structured Input.
Answer Generator	Refines and verifies the solution produced by the Solution Generator. Extracts and checks the predicted answer for consistency.

Table 8: Modules in the Module Inventory essential for solving geospatial queries

Tool	Description
Trip Tool	Retrieves information about locations and travel routes between them. Combines the Place Details and Directions APIs to fetch place metadata and step-by-step route instructions.
Route Tool	Fetches route information between two points using the Directions API, including distance, estimated time, and navigation steps.
Nearby Tool	Identifies points of interest within a specified area using the Nearby Search API. Returns place names, ratings, and other metadata.
PlaceInfo Tool	Retrieves detailed information about a given place using the Place Details API, including address, hours, contact info, and reviews.

Table 9: Map tools used by the MapAgent framework, each built over a specific Google Maps API.

Listing 1: Example how MapAgent answer a query Green : Correct Answer. Red : Wrong Answer.

```
Question:
I am currently at Planetshakers Church Melbourne City Campus. I want to drive to Carrot Tattoo Studio and
then to Melbourne Lifestyle Apartments - Docklands Accommodation. How long will this take?
Options: A. 18 mins B. 19 mins C. 20 mins D. 21 mins
______
Modules selected by Planner agent:
1.Map Service
2.Solution Generator
3.Answer Generator
Information gathered by Map Service module via Map-Tool Agent (invoking Trip tool):
All Location Info:
Planetshakers Church Melbourne City Campus
Name: Planetshakers Church Melbourne City Campus
Address: 400 City Rd, Southbank VIC 3006, Australia
Rating: 4.8
Types: church, tourist_attraction, place_of_worship, point_of_interest, establishment
Is Open Now: False
Weekday Opening Hours:
- Monday: Closed
....
....
- Saturday: Closed
- Sunday: 9:00AM - 8:00PM
Carrot Tattoo Studio
Name: Carrot Tattoo Studio
Address: Basement/63 Elizabeth St, Melbourne VIC 3000, Australia
Rating: 5
Types: store, point_of_interest, establishment
Is Open Now: False
Weekday Opening Hours:
- Monday: 10:00 AM - 8:00 PM
- Tuesday: 10:00AM - 8:00PM
....
....
- Sunday: 10:00AM - 8:00PM
Melbourne Lifestyle Apartments - Docklands Accommodation
Name: Melbourne Lifestyle Apartments - Best Views on Collins
Address: 893A Collins St, Docklands VIC 3008, Australia
Rating: 4.8
Types: lodging, point_of_interest, establishment
Is Open Now: True
Weekday Opening Hours:
- Monday: Open 24 hours
- Tuesday: Open 24 hours
....
....
The travel time(distance) from Planetshakers Church Melbourne City Campus to Carrot Tattoo Studio is 9 mins
(2.0 km)
The travel time(distance) from Planetshakers Church Melbourne City Campus to Melbourne Lifestyle Apartments
- Docklands Accommodation is 8 mins (2.5 km)
The travel time(distance) from Carrot Tattoo Studio to Planetshakers Church Melbourne City Campus is 9 mins
(1.9 km)
....
....
The travel time(distance) from Melbourne Lifestyle Apartments - Docklands Accommodation to Carrot Tattoo
Studio is 11 mins (2.7 km)

Solution generated by solution_generator module:
To determine the total travel time from Planetshakers Church Melbourne City Campus to Carrot Tattoo Studio
and then to Melbourne Lifestyle Apartments - Docklands Accommodation, we need to add the travel times for
each segment of the journey.

1. **From Planetshakers Church Melbourne City Campus to Carrot Tattoo Studio:**
   - Travel time: 9 minutes

2. **From Carrot Tattoo Studio to Melbourne Lifestyle Apartments - Docklands Accommodation:**
   - Travel time: 10 minutes

Now, we add these travel times together:
9 minutes + 10  minutes = 19 minutes

Therefore, the total travel time is 19 minutes.

Correct option extracted by answer_generator module:  B.19 minutes
```

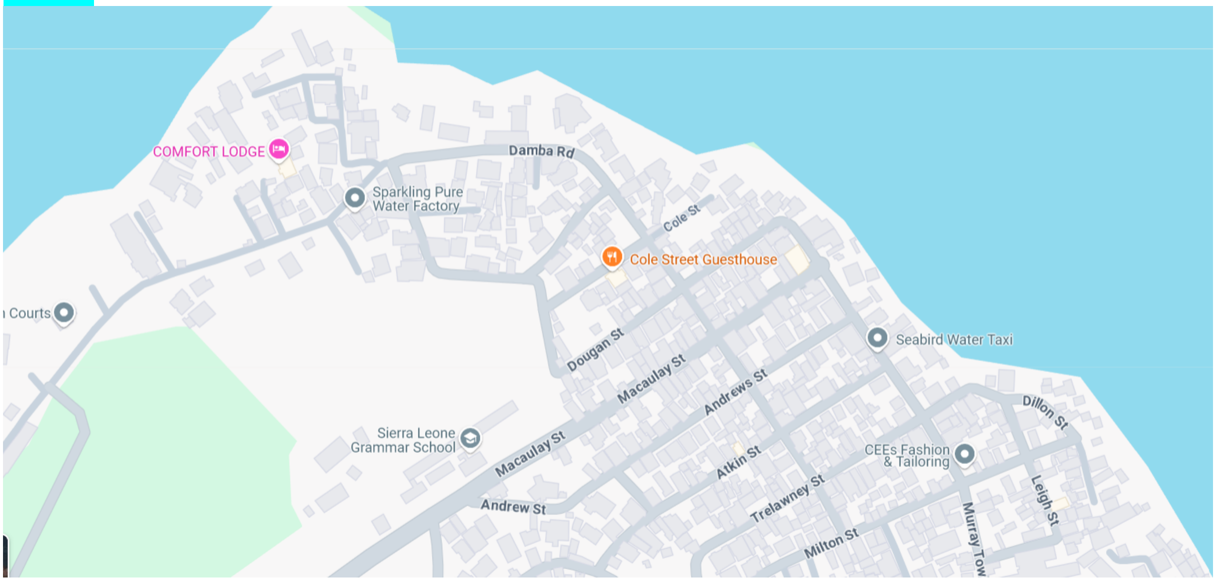
Listing 2: Example how MapAgent answer MapEval-Visual counting query Green: Correct Answer. Red: Wrong Answer.

```
Context:
```


```
Question:
How many churces are visible in the lower half of the map?
Options:  A.Five B.Two C.Four D.Three

Modules selected by Planner agent:
Module Selection from the Module Catalog and their execution order:
1.Visual Place Recognizer module
2.map_service module
3.solution_generator module
4.answer_generator module

Visual Place Recognizer module Output:
This image center represents the location of Radmansgatan 46, 113 57 Stockholm, Sweden.

Map Service module Output:
When executing the map_service module, it connects to the map-tool agent in the Inner Layer.

Map-Tool Agent Output:
Nearby tool is selected to fetch information.

Nearby Tool's Output:
There are some churches within 0.8 kilometers of the location Radmansgatan 46, 113 57 Stockholm, Sweden, as
shown below:
City Church ( distance: 0.46526450497253263 kilometers, rating:4.6, total reviewer:94)
St. Peter's Church, Stockholm ( distance: 0.584886996790922 kilometers, rating:4.5, total reviewer:42)
Adolf Fredriks Forsamling ( distance: 0.36249979086127565 kilometers, rating:4.5, total reviewer:341)

solution generator module's Output:
Based on the provided map and Google Maps context, the churches visible in the lower half are:
St. Peter's Church, Stockholm
Adolf Fredriks Forsamling
City Church

Thus, the correct answer is B. Three.

answer generator module's Output:
Based on the response from the solution generator, the answer should be 3, which seems logical and correct.
The answer is B. Three.

Comments:
The Final answer B. 3 is a correct Answer
```

Listing 3: Example how MapAgent answer MapEval-Visual Route query Green : Correct Answer. Red : Wrong Answer.

Context:



Question:

How to get to Sierra Leone Grammer School from Seabird Water Taxi after having lunch at Cole Street Guesthouse?

Options:

A. Seabird Water Taxi -> Head northwest on High Broad St toward Andrews St -> Turn left onto Andrews St -> Continue onto Andrew St -> Turn right onto Macaulay St -> Sierra Leone Grammar School -> Head northeast on Macaulay St -> Turn left at the 1st cross street toward Cole St -> Turn left at the 2nd cross street onto Cole St -> Cole Street Guesthouse

B. Seabird Water Taxi -> Head northwest on High Broad St toward Andrews St -> Turn left onto Andrews St -> Continue onto Andrew St -> Turn right onto Macaulay St -> Sierra Leone Grammar School

C. Seabird Water Taxi -> Head northwest on High Broad St toward Andrews St -> Turn left onto Macaulay St -> Sierra Leone Grammar School -> Head northeast on Macaulay St -> Turn left at the 1st cross street toward Cole St -> Turn left at the 2nd cross street onto Cole St -> Cole Street Guesthouse

D. Seabird Water Taxi -> Head northwest on High Broad St toward Andrews St -> Turn right toward Cole St -> Turn left at the 1st cross street onto Cole St -> Cole Street Guesthouse -> Head northeast on Cole St toward Damba Rd -> Turn right at the 1st cross street toward Macaulay St -> Turn right at the 2nd cross street onto Macaulay St -> Sierra Leone Grammar School

---

Modules selected by **Planner agent**:

Module Selection from the Module Catalog and their execution order:
1. Visual Place Recognizer module
2. map_service module
3. solution_generator module
4. answer_generator module

Image Search module Output:

This image center represents the location of Cole Street Guesthouse, 5 Cole St, Freetown, Sierra Leone.

Map Service module Output:

When executing the map_service module, it connects to the map-tool agent in the Inner Layer.

Map-Tool Agent Output:

Route tool is selected to fetch information.

Route Tool's Output:

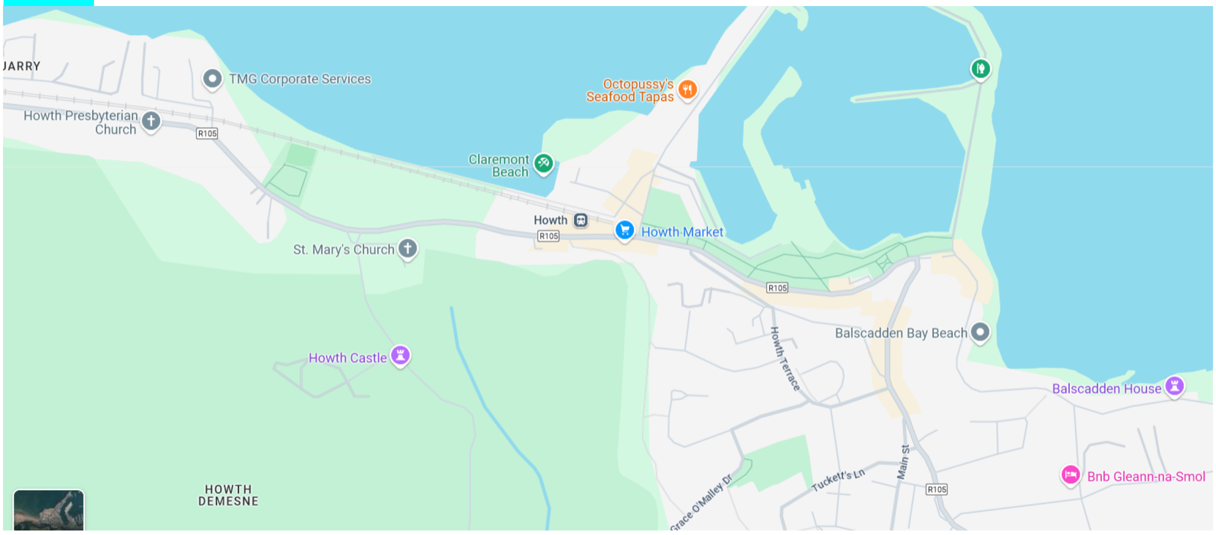
There is a total of one route from Seabird Water Taxi, Freetown, Sierra Leone, to Sierra Leone Grammar School, Freetown, Sierra Leone, with a waypoint at Cole Street Guesthouse. The route information is provided below:
Route 1:(VIA) High Broad St (0.4 km, 5 mins)
Details steps are provided below:
Head northwest on High Broad St toward Andrews St (0.3 km, 4 mins)
Turn right toward Cole St (56 m, 1 min)
Turn left onto Cole StDestination will be on the left (46 m, 1 min)
Head northeast on cole st toward Damba Rd (46m, 1 min)
Turn right at the 1st cross street toward Macaulay St (110 m, 1.5 min)
Turn right at the 2nd cross street onto Macaulay St (260 m, 2 min)
Sierra Leone Grammar School

Listing 4: Continuation of Example how MapAgent answer MapEval-Visual Route query Green : Correct Answer. Red : Wrong Answer.

```
solution generator module's Output:
Based on the provided map and Google Maps context, the best way point is:
Head northwest on High Broad St toward Andrews St
Turn right toward Cole St
Turn left onto Cole StDestination will be on the left
Head northeast on cole st toward Damba Rd
Turn right at the 1st cross street toward Macaulay St
Turn right at the 2nd cross street onto Macaulay St
Sierra Leone Grammar School

Therefor the correct answer is:
**(D) Seabird Water Taxi
Head northwest on High Broad St toward Andrews St
Turn right toward Cole St
Turn left at the 1st cross street onto Cole St
Cole Street Guesthouse
Head northeast on Cole St toward Damba Rd
Turn right at the 1st cross street toward Macaulay St
Turn right at the 2nd cross street onto Macaulay St
Sierra Leone Grammar School**

____

answer generator module's Output:
Based on the response from the solution generator, the answer should be D, which seems logical and correct.
The answer is D . Seabird Water Taxi -> Head northwest on High Broad St toward Andrews St -> Turn right
toward Cole St -> Turn left at the 1st cross street onto Cole St -> Cole Street Guesthouse -> Head northeast
 on Cole St toward Damba Rd -> Turn right at the 1st cross street toward Macaulay St -> Turn right at the 2
nd cross street onto Macaulay St -> Sierra Leone Grammar School

____
Comments:
The Final answer D is a correct Answer
```

Listing 5: Example how MapAgent answer MapEval-Visual Nearby query Green : Correct Answer. Red : Wrong Answer.

Context:




```
Question:
Which castle is closest to Howth rail station?
Options:   A.Howth Castle   B.Mary Castle  C.Claremont Castle  D.Balscadden House

_____
Modules selected by Planner agent:
Module Selection from the Module Catalog and their execution order:
1.visual_place_recognizer module
2.map_service module
3.solution_generator module
4.answer_generator module
_____
Visual Place Recognizer module Output:
The center of the image appears to be Howth, Ireland.
Map Service module Output:
When executing the map_service module, it connects to the map-tool agent in the Inner Layer.
Map-Tool Agent Output:
Nearby tool is selected to fetch information.
Nearby Tool's Output:
There are some castle distance from the current location Howth Market, 3 Harbour Rd, Howth, Dublin, D13 T8Y7
, Ireland within 15 Kilometers in below:
Howth Castle Estate ( distance: 0.4770576879845442 kilometers, rating:4.1, total reviewer:714)
Malahide Castle & Gardens ( distance: 8.742973877916125 kilometers, rating:4.6, total reviewer:15235)
Dublin Castle ( distance: 13.875697894024961 kilometers, rating:4.3, total reviewer:33876)
Swords Castle ( distance: 12.548387008403804 kilometers, rating:4.5, total reviewer:1764)
Dalkey Castle & Heritage Centre ( distance: 12.505523354933828 kilometers, rating:4.6, total reviewer:911)
Monkstown Castle ( distance: 12.235452667377206 kilometers, rating:4.4, total reviewer:104)
Clontarf Castle Hotel ( distance: 9.26724243435518 kilometers, rating:4.5, total reviewer:5073)
Sandycove Castle ( distance: 11.6217343658077 kilometers, rating:4.5, total reviewer:24)
Castle ( distance: 6.943123269967969 kilometers, rating:4, total reviewer:2)
Wooden Castle ( distance: 8.662582766435369 kilometers, rating:4.6, total reviewer:11)
Lambay Castle ( distance: 11.901973302290772 kilometers, rating:4.3, total reviewer:27)
Sutton Castle ( distance: 2.3206824179509207 kilometers, rating:4.2, total reviewer:9)
Archbold's Castle ( distance: 12.507363732093697 kilometers, rating:4.3, total reviewer:8)
_____

solution generator module's Output:
Based on the provided map and Google Maps context, The castle closest to Howth rail station is  Howth Castle
 which is only 0.477 km away from Howth rail station.

Thus, the correct answer is A. Howth Castle

_____

answer generator module's Output:
Based on the response from the solution generator, the answer should be Howth Castle, which seems logical
and correct. The answer is  A. Howth Castle .

_____
Comments:
The Final answer A. Howth Castle is a correct Answer
```

Listing 6: Example 1 how VLLM used API to guess correct answer Green: Correct Answer. Red: Wrong Answer.

Context:

Question:

```
How many churces are visible in the lower half of the map?
Options: A.Five B.Two C.Four D.Three
```

Response with using API(MapAgent):

Google Map Response:

```
There are some churches within 0.8 kilometers of the location Radmansgatan 46, 113 57 Stockholm, Sweden, as
shown below:
City Church ( distance: 0.46526450497253263 kilometers, rating:4.6, total reviewer:94)
St. Peter's Church, Stockholm ( distance: 0.584886996790922 kilometers, rating:4.5, total reviewer:42)
Adolf Fredriks Forsamling ( distance: 0.36249979086127565 kilometers, rating:4.5, total reviewer:341)
```

---

final answer from solution generator:

```
"Based on the map provided and the context from the google map module, the churches visible in the lower
half of the map are:"
"1. St. Peter's Church, Stockholm "
"2. Adolf Fredriks Frsamling "
"3. City Church  "
"Thus, the correct answer is:  (D) Three** "
```

Response without using API(Base Model):

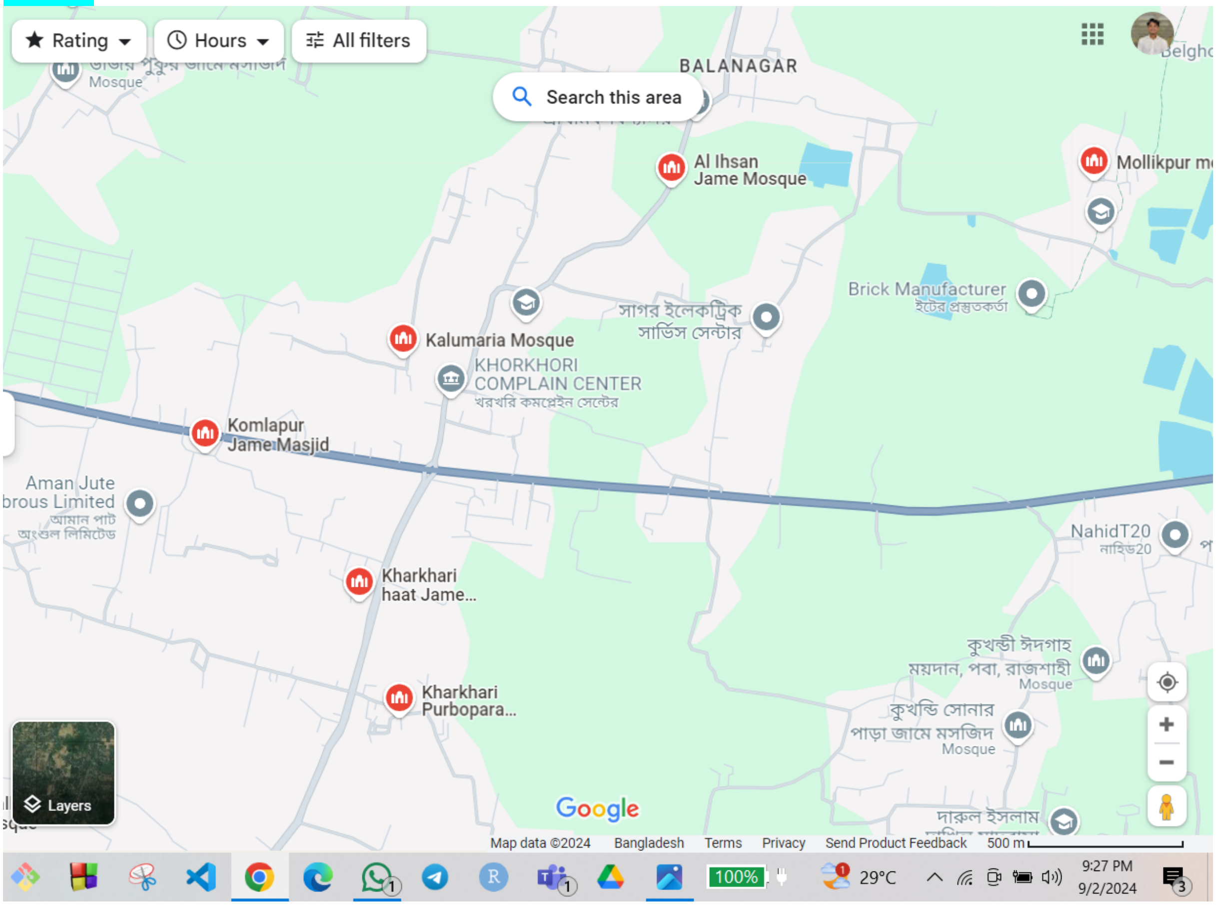
```
Based on the provided map, there are four visible churches in the map. Thus, the answer is: (C) Four .
```

Observation:

```
The correct answer should be D. Three which api version can predict but the base model which do not have api
 not able to answer.
```

Listing 7: Example 2 how VLLM used API to guess correct answer Green: Correct Answer. Red: Wrong Answer.

```
Context:
```




```
Question:
I am at Sagor Electronix service center. What's the nearest mosque to me?
Options: A.Kalumaria Mosque B.Al Ihsan Jame Mosque C.Komlapur Jame Masjid D.Mollikpur mosque

Response with using API(MapAgent):

Google Map Response:
There are some mosque distance from the current location  Sagor Electronix service center in below:
Al Ihsan Jame Mosque ( distance: 0.5823095053737362 kilometers, rating:5, total reviewer:1)
Mollikpur mosque ( distance: 1.1680710806133061 kilometers, rating:4.7, total reviewer:3)
Kalumaria Mosque ( distance: 1.1879085293895582 kilometers, rating:4.3, total reviewer:4)
Lolithar Bazar Central Masjid ( distance: 1.1566211819806247 kilometers, rating:0, total reviewer:0)
Komlapur Jame Masjid ( distance: 1.8654544610349613 kilometers, rating:4.4, total reviewer:25)
Rajshahi University Central Mosque ( distance: 4.459173025195116 kilometers, rating:4.9, total reviewer:221)
Maria Jama Masjid ( distance: 1.8367913382567742 kilometers, rating:4, total reviewer:9)
Maherchandi East Jame Mosque ( distance: 2.3601717780592995 kilometers, rating:5, total reviewer:5)
Vodra Central Jame Mosque ( distance: 5.131809940553015 kilometers, rating:4.6, total reviewer:145)
Madina Nagar Jame Mosque ( distance: 4.21188964565389 kilometers, rating:4.4, total reviewer:57)
Purbo Para Jame Mosque ( distance: 3.614167437327442 kilometers, rating:4.2, total reviewer:26)
Meherchandi Purbapara Central Jame Mosque ( distance: 3.4298745224188134 kilometers, rating:4.5, total
reviewer:69)
Katakhali Bazar Jame Masjid ( distance: 4.28092651862087 kilometers, rating:4.5, total reviewer:193)
Belghoria Jama Masjid ( distance: 5.697832533309013 kilometers, rating:4.6, total reviewer:19)
Hazrapukur Jame Mosque ( distance: 3.260713598861443 kilometers, rating:4.6, total reviewer:30)

final answer from solution generator:
The nearest mosque to your location, is **Al Ihsan Jame Mosque** with a distance of 0.582 kilometers.\n\
nAnswer: **(B) Al Ihsan Jame Mosque**

Response without using API(Base Model):

Okay, I see your location on the map! The nearest mosque to  Sagor Electronix service center appears to be
(C) Kalumaria Mosque.

Observation:
The correct answer should be B. Three which api version can predict but the base model which do not have api
 not able to answer.
```