From Natural Language to Structured Queries

An LLM-Powered Chatbot for a Smart Tourism Observatory

Rodrigo Bravo Simões[1]
Instituto Universitário de Lisboa (Iscte-IUL), Portugal

Adriano Lopes
Instituto Universitário de Lisboa (Iscte-IUL), Portugal

Fernando Brito e Abreu
Instituto Universitário de Lisboa (Iscte-IUL), Portugal

Abstract

The accessibility of Smart Tourism Initiatives and Tools (STITs) is a cornerstone of digital transformation in the tourism sector. The EUROSTIT project addresses this through a European Observatory that catalogs these technologies. However, sifting through hundreds of results remains a significant challenge for users. This paper presents a specialized conversational agent that facilitates discovery through a filter-based natural language interface. Departing from computationally intensive fine-tuning, our architecture uses prompt engineering – specifically, one-shot Chain-of-Thought (CoT) reasoning – to generate boolean search queries (BSQs). This approach enables the system to distinguish conversational fillers and other irrelevant terms from domain-specific smart tourism entities, autonomously translating free-text into structured database queries. We describe a dynamic web application with a synchronized user experience where conversational input acts as cumulative filters. The performance of different LLMs is compared through various metrics. We conducted quantitative evaluations to gain insight into the feasibility of using readily available commercial LLMs, instead of resource-heavy model fine-tuning for domain-specific boolean search query generation.

2012 ACM Subject Classification Computing methodologies → Natural language processing; Human-centered computing → Natural language interfaces; Information systems → Information retrieval query processing; Information systems → Search interfaces

Keywords and phrases Large Language Models (LLMs), Prompt Engineering, Chain-of-Thought (CoT), Boolean Search Query (BSQ) Generation, Natural Language Interfaces (NLI), Smart Tourism, Chatbots

Digital Object Identifier 10.4230/OASIcs.SLATE.2026.17

Supplementary Material
Software (Source Code): `https://github.com/EUROSTIT/Intelligent-Interaction`
Dataset: `https://zenodo.org/records/19742395`

Funding This work was partly supported by project 2024.07579.IACDC/2024, funded by FCT – Fundação para a Ciência e a Tecnologia, I.P., under PRR – Plano de Recuperação e Resiliência funding, investment RE-C05-i08 – Ciência Mais Digital and, partially supported by national funds through FCT, ISTAR-Iscte Pluriannual Projects UID/04466/2025.

1 Introduction

The adoption of digital technologies in the tourism sector depends heavily on the accessibility of Smart Tourism Tools (STTs) and the lessons learned from their implementation in Smart Tourism Initiatives (STIs). Making that information publicly available was the motivation

[1] Corresponding author

underlying EUROSTIT project (European Observatory of Smart Tourism Initiatives and Tools), which successfully developed a digital infrastructure to catalog and classify European smart tourism technologies, available online at `https://www.eurostit.eu/`.

The EUROSTIT content is extracted from several sources, namely PDF catalogs created by tourism-related organizations [10, 15]. Such catalogs are not prone to online search, and their content is not structured or easily indexable. As the number of collected solutions grows, the value of the observatory increasingly depends on the ability to: (i) organize solutions consistently according to an adequate taxonomy [11] and (ii) allow tourism stakeholders to perform searches that retrieve STITs (Smart Tourism Initiatives and Tools) aligning with their declared aim.

The first concern is addressed in [9], and the second concern is addressed in the current paper, where we describe how we improved user interaction and facilitated the discovery of STITs within the observatory by developing an intelligent conversational agent supporting natural language interaction. Each step of the interaction introduces cumulative filters that reduce the universe of relevant STITs. This paper details the requirements, architectural design, implementation, and evaluation of this filter-based conversational agent.

The remainder of this paper is organized as follows. Section 2 reviews related work in the context of conversational agents in tourism. Section 3 details the system requirements and architecture, user interface, and prompt engineering methodology. Section 4 presents the evaluation of the implemented system. Section 5 provides a discussion of methodological choices and research implications, and Section 6 concludes the article with directions for future work.

2 Related Work

The integration of conversational artificial intelligence (AI) and chatbots into the tourism and hospitality sectors has experienced significant growth over the past decade. This surge is driven by the industry's need for personalized, 24/7 customer service and the optimization of digital ecosystems within smart tourism destinations.

Several systematic literature reviews have recently mapped the landscape of conversational agents in this domain. Benaddi et al. [4, 5] provided a comprehensive analysis of chatbot evolution, classifying their development from simple rule-based decision trees to advanced AI-driven systems. Their review highlights how chatbots impact the tourism industry within the "6A framework" [6] (Attractions, Amenities, Accessibility, Ancillary services, Available packages, and Activities), demonstrating their utility in helping users in all phases of the travel journey [4].

The behavioral and psychological dimensions of user interaction are explored by Alharbi et al. [2], who proposed an integrated conceptual framework that details the mechanisms, modifiers, and implications of AI-enabled conversational agents (AICAs). Their systematic review emphasizes the critical importance of emotional, ethical, and trust-related factors in AICA adoption models that influence user engagement [2].

In terms of practical applications, much of the existing research was centered on end-consumer applications, particularly those related to cultural heritage and tourist routing. For example, HeriBot [7] uses latent Dirichlet allocation (LDA) for text analysis and was later extended with ontologies that represent knowledge about destinations [8]. Future work on HeriBot will incorporate deep learning, which has already been employed in another chatbot to support tourist trips using Gated Recurrent Unit (GRU) cells[2] [16]. These chatbots can

[2] Basic computational units of a GRU neural network, a type of recurrent neural network (RNN) designed to process sequential data such as text, time series, speech, or sensor streams.

perform tasks such as recommending points of interest and building customized itineraries. Mobile chatbots have also been successfully deployed to help tourists with local navigation and recommendations [3].

More recently, the focus has shifted toward the disruptive potential of Generative AI in smart tourism. Kontogianni and Alepis [13] examined how LLMs (Large Language Models) and, in particular, Generative Pre-trained Transformers (GPTs) are improving tourist experiences and streamlining operations. Saleh [14] investigated AI integration patterns and prompt optimization techniques, highlighting the operational impacts of using models such as ChatGPT, Claude, and Gemini in hospitality settings.

Although the literature extensively covers B2C (Business-to-Consumer) chatbots designed for tourist itinerary planning or customer service, there is a noticeable gap with respect to tools designed to assist stakeholders in navigating complex technological ecosystems, such as repositories of Smart Tourism Initiatives and Tools. Our work departs from these paradigms: instead of directing tourists to physical attractions, the EUROSTIT chatbot is designed to direct tourism operators and researchers to appropriate digital technologies.

Another aspect worth considering is related to the use of LLMs for generating boolean search queries (BSQs)[3]. Research suggests that fine-tuning LLMs to generate BSQs yields better results than prompt engineering [1]. However, fine-tuning is not available for proprietary models and is computationally expensive. A 2023 study [17] evaluated prompt engineering with ChatGPT for BSQ generation and concluded that this approach had potential, but noted that its results had a lower recall than state-of-the-art alternatives at the time. Since commercial LLMs have improved since then, it is useful to revisit this problem using better models. A 2025 study [12] compared different commercially available LLMs from OpenAI to evaluate the quality of the generated BSQs. However, the methodology rewards models that generate more complex queries with many syntactical elements without analyzing their search results using relevance metrics such as precision and recall. Thus, we identify the comparison of newer LLMs for BSQ generation without fine-tuning as a research gap that we aim to address.

3 Methodology and Architecture

The proposed chatbot architecture follows the classic three-tier model, comprising a presentation tier, a logic tier, and a data tier. The chatbot itself is implemented as a dynamic web application that can be integrated into virtually any content management system (CMS), although some integration effort may be required depending on the target platform. The following subsections describe how the chatbot's components map to the three tiers and discuss the associated implementation challenges.

3.1 User Interface Design

In this presentation tier, the chatbot user interface (UI) is inherently simple, as the primary focus is to facilitate sequential and easy-to-follow conversations. However, the proposed chatbot deviates from traditional free-text conversational interfaces. After discussion with various stakeholders, the functional requirements drawn for the chatbot highlight the need for users to interact primarily via keyboard and to view a continuously updated list of relevant STITs based on applied filters, and also to allow filtering by STT category according to

[3] Search expressions that use Boolean operators to combine keywords and control how a search engine, database, or digital library retrieves results.

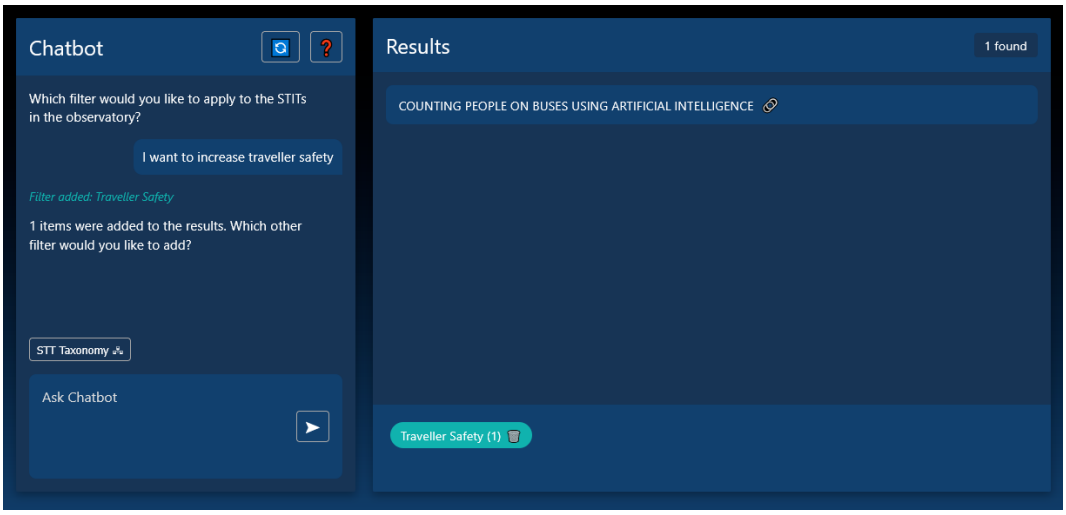

Figure 1 Chatbot UI, featuring the conversational pane, applied filters, and results list.

Figure 2 The Taxonomy Pane.

the taxonomy presented by Galvão et al. [11][4] As for the non-functional requirements, they dictate that the system must be a responsive web application, capable of operating across multiple screen sizes.

The initial user interface mockups explored a three-pane desktop layout comprising a chatbot pane, a filters pane, and a results pane. Following stakeholder discussions, this was optimized into a two-pane layout to conserve screen real estate, where the list of applied filters is shown at the bottom of the results pane. Figure 1 presents the UI.

In the chatbot pane above the input area, there is a button that provides access to a taxonomy pane (see Figure 2). This pane allows users to manually select or deselect categories from the STT taxonomy, effectively adding or removing filters. Additional features include a reset button to clear the conversation and a help button to guide new users, both at the top of the chatbot pane.

[4] A summary of this taxonomy is available in the observatory's Taxonomy page.

3.2 Application Intelligence and Data Integration

At the core of the application's intelligence is the ability to transform a user's natural language input into a boolean search query via an LLM. This query can contain the boolean operators AND and OR, as well as grouping to obtain higher precedence. For instance, given the input *"I am searching for sustainable transport"*, a possible resulting query could be `((Sustainability OR Sustainable) AND (Transport OR Mobility))`. The query is then run against the titles and descriptions of STITs in the CMS's database.

To perform this transformation, the LLM receives a set of instructions that indicate which terms should be ignored and which should be retained as keywords, as shown in Table 1. However, the examples in table 1 are incomplete and insufficient because unexpected types of terms may be present in the user's input. To handle these unforeseen categories, the LLM is instructed to categorize the terms in the user prompt as relevant or irrelevant. This is an instance of Chain-of-Thought prompting since the model is instructed to generate an intermediate reasoning step.

Table 1 Categorization of User Input Terms for Query Rewriting.

Category	Examples
Irrelevant (Conversational Filler)	"I want", "show me"
Irrelevant (Goal Verbs/Subjective)	"improve", "boost", "best", "top"
Irrelevant (Generic Domain Terms)	"solution", "technology"
Relevant (Physical Objects/Concepts)	"bus", "museum", "safety", "sustainability"
Relevant (Specific Technologies)	"5G", "app"

Additional instructions were included in the system prompt to handle certain edge cases, which we will describe below.

The models are instructed to avoid abbreviations such as "AR" (Augmented Reality) because the database query returns items containing these characters in any position (for example, "AR" would match the word "car"). Pluralization is another issue. The keyword "sensors" would not match text containing "sensor" in the singular, so the singular form is preferred for generated keywords. However, if the plural form changes the spelling of the base word (e.g., "city" vs. "cities"), both the singular and plural forms should be included in a boolean OR group.

The LLM is instructed to respond with JSON containing a recursive representation of the structured query, a short title for the filter, and the relevance classification of the original terms. The application handles cases where the LLM returns an invalid JSON.

We use a one-shot prompting technique. Therefore, the system prompt concludes with an example of user input and a possible response that adheres to the given instructions.

The system uses OpenRouter, a unified interface for LLMs that offloads computation of inference and facilitates model comparison and the implementation of fallback mechanisms. In our chatbot implementation, a configuration file contains an ordered list of models to try. If a model is unavailable or removed from the platform, fallback models can be used. This is also useful if the OpenRouter API key expires or a credit limit is reached, as free models can instead be tried.

Integrating the STT taxonomy into the application involves a simple database lookup based on the user's selection in the taxonomy pane. This is because the STTs in the observatory were previously annotated with properties that indicate their taxonomic categories..

3.3 Technology Stack

During the technology exploration phase, several existing chatbot libraries and templates were evaluated, including BotUI, react-chatbot-kit, Streamlit, and Gradio. However, these solutions were deemed too limited to support the envisioned filter-based paradigm without substantial modifications. Even promising templates, such as Next.js Chatbot available on Vercel's website, presented limitations regarding the integration of arbitrary LLMs due to their reliance on specific SDKs. Consequently, a custom frontend development approach was adopted.

The application was built from scratch using the Next.js framework, which combines React for the frontend and Node.js for the backend. The chatbot is decoupled from the observatory's primary Content Management System, Omeka S, ensuring that it can be more easily integrated into another CMS in the future, if necessary. TypeScript was selected as the programming language to leverage static typing and better avoid runtime errors. Furthermore, to foster transparency, reproducibility and future research within the smart tourism domain, the code produced for this project is open-source at `https://github.com/EUROSTIT`.

As an example of some work required to integrate the chatbot and the CMS, we developed an Omeka S module (extension) to create a table containing the title and description of each item, with a full-text index. This table is automatically synchronized when an item is created, updated, or deleted. The LLM generated boolean search JSON is converted into MySQL syntax for boolean full-text search expressions. This allows for efficient retrieval of matching items while protecting against SQL injection through the use of a prepared statement in the code.

4 Evaluation

4.1 Evaluation Methodology

We evaluated the performance of various large language models (LLMs) on the task of converting a natural language input prompt into a structured query to search the observatory's database. To achieve this, we conducted quantitative evaluations based on a manually curated set of 18 user prompts of varying complexity.

In addition to this, we analysed the structured queries generated by the model to check whether they matched or contradicted the instructions provided in the system prompt.

For the quantitative evaluation, we collected the set of relevant STITs that should appear in the chatbot's results for each of the 18 prompts. By comparing this set with the results obtained by running the generated query against the database, we could calculate precision and recall metrics. We also collected response times for model inference, enabling us to analyse the trade-offs between speed, quality and cost.

At the time of writing, the observatory contains more than a thousand STITs. Screening them all manually to collect those relevant to a specific keyword would be too time-consuming. Therefore, we first sort the STITs by relevance to a keyword using embeddings. Specifically, we sorted them according to the distance between the keyword's embedding and the embedding of a concatenation of the STIT's title and description. We used a model available on OpenRouter (perplexity/pplx-embed-v1-4b) to determine those embeddings.

For prompts containing multiple relevant keywords, we created short natural language phrases that connected the keywords. Initially, we tried another approach where two keywords, conceptually joined by a boolean AND, were individually turned into embeddings. The list of items was then sorted by the highest distance between the various keyword embeddings

and the item's embeddings. After testing both approaches on a few prompts, we found that using a single embedding to represent a more complex concept was more appropriate. Once such a list is generated, manual screening can be performed more efficiently because, at a certain point, the items become less relevant. During the screening, if 10 consecutive items are marked as irrelevant, the process ends and all unmarked items are considered irrelevant.

In our evaluation methodology, we compared high- and low-priced models from the four providers with the highest market share in OpenRouter as of April 5, 2026: Qwen, Google, OpenAI, and Anthropic. To ensure a fair comparison, the models were selected so that the higher-priced options shared a similar price point across all providers, with the same principle applied to the lower-priced tier. Models available free of charge were excluded from this evaluation because one provider did not have free models available in OpenRouter, and the free models of two other providers were rate-limited whenever we attempted to conduct our evaluation.

The dataset used for this evaluation is available at `https://zenodo.org/records/19742395`. It includes prompts, results of manual screening, boolean search queries generated by the LLMs, and items matched by full-text search.

4.2 Test Set Characterization

The test set contains manually created sets of relevant items for each of the 18 prompts. On average, there are 31 relevant items per prompt, with a standard deviation of 21, ranging from a minimum of 7 to a maximum of 87 items.

The test set contains six simple prompts, six medium-complexity prompts, and six complex prompts. The level of complexity is related to the number of relevant concepts for query construction: simple prompts have one concept, medium-complexity prompts have two, and complex prompts have more. The suitability of these prompts was assessed during manual screening. Several prompts were discarded because they targeted a very specific niche with little representation in the observatory and had very few relevant items. For simple and medium-complexity prompts, we discarded those with fewer than 10 corresponding STITs. However, it was challenging to find many relevant items for complex prompts, so we accepted those with fewer items. Some of the prompts in the test set are transcribed in the following subsections. The complete list of prompts can be found in the supplementary material.

4.3 Evaluation of Instruction-Following

We will now present some interesting findings regarding boolean queries generated from user prompts in the test set. All of the model responses were syntactically valid JSON and generally adhered to the instructions in the system prompt. The main difference was in the synonyms generated in the OR groups. In the following, we discuss some prompts, referring to them by the position they appear in the published test set.

Prompt #1 is *"Usage of chatbots for tourism"*. When considering that all items in the observatory should be related to tourism, this prompt is simple because it contains only one keyword that is relevant for searching: "chatbot". As previously mentioned, the Chain-of-Thought step was added to the system prompt to better equip the model to consider terms such as "tourism" as filler. However, only one model has done so (qwen/qwen-max). All other models marked two terms for keeping ("chatbot" and "tourism"), while all models correctly ignored all the other terms in the user prompt.

Prompt #2 is *"I am searching for sustainable transport."*. Some models identified the concept "sustainable transport" as a single unit, while others separated it into two concepts. These two different approaches to the analysis step lead to two different query structures.

Table 2 Metrics of the evaluation for each model.

Model	Precision	Recall	F1 Score	Latency (ms)
qwen/qwen3-235b-a22b-2507	0.53	0.28	0.37	6305
qwen/qwen-max	0.59	0.28	0.38	2003
google/gemini-2.5-flash	0.52	0.36	0.43	1397
google/gemini-2.5-pro	0.71	0.42	0.53	7965
openai/gpt-4.1-nano	0.57	0.21	0.30	1994
openai/gpt-5.2	0.49	0.20	0.29	6221
anthropic/claude-3-haiku	0.46	0.27	0.34	1800
anthropic/claude-haiku-4.5	0.48	0.33	0.39	1844

In the first case, a single OR grouped synonyms for "sustainable transport", while in the second case, an AND joined two OR groups, one for each term. These different approaches to grouping terms into keywords can be seen in the results of other prompts in the test set, with similar consequences. The system prompt instructs the models to avoid grouping terms unless they form a well-known phrase. However, it does not prescribe a methodology for deciding when to group terms. Therefore, both approaches are correct.

Prompt #5 was designed to have a lot of filler: *"I was thinking that some type of solutions for improving traveller health could be interesting for my business"*. As with most prompts in the test set, most of the evaluated models correctly ignore the filler. An exception for this prompt is anthropic/claude-3-haiku, which does not exclude "solution" and in fact includes the generic domain terms "solution", "initiative", and "tool", as well as the subjective goal terms "improve", "enhance", and "boost". Some of these synonyms were included in the system prompt as examples of what not to include.

4.4 Quantitative Evaluation

The macro results of the quantitative evaluation are shown in Table 2 and Figure 3. Precision, recall, and latency were calculated as the average of those metrics for the prompts in the test set. The F1 score was calculated based on the averages of precision and recall, and it is the most relevant metric as it combines both the relevance of the items matched by the query and the coverage of the relevant items in the observatory.

The model google/gemini-2.5-pro dominates all others in terms of precision and recall. However, it is the slowest, with an average latency of almost eight seconds. If latency is a concern, google/gemini-2.5-flash would be a better alternative, as can be seen in Figure 3. It is both the fastest model and the second best in terms of F1 score. It could be expected that higher-priced models outperform lower-priced models from the same provider, which is true for three of these providers; however, openai/gpt-4.1-nano has a negligibly higher F1 score than openai/gpt-5.2 (0.30 versus 0.29).

Regarding the previoulsly mentioned prompt #1, in which only Qwen's qwen/qwen-max correctly identified "tourism" as a term to exclude from the search query, it is interesting to note that the lower-priced model from the same provider (qwen/qwen3-235b-a22b-2507) has a higher F1 score for that prompt (0.53 versus 0.38). An analysis of the generated queries reveals that the higher-priced model did not generate synonyms of the keyword "chatbot" (such as "virtual assistant"), while the lower-priced model did. This explains the difference in F1 scores despite the error identified in the previous subsection.

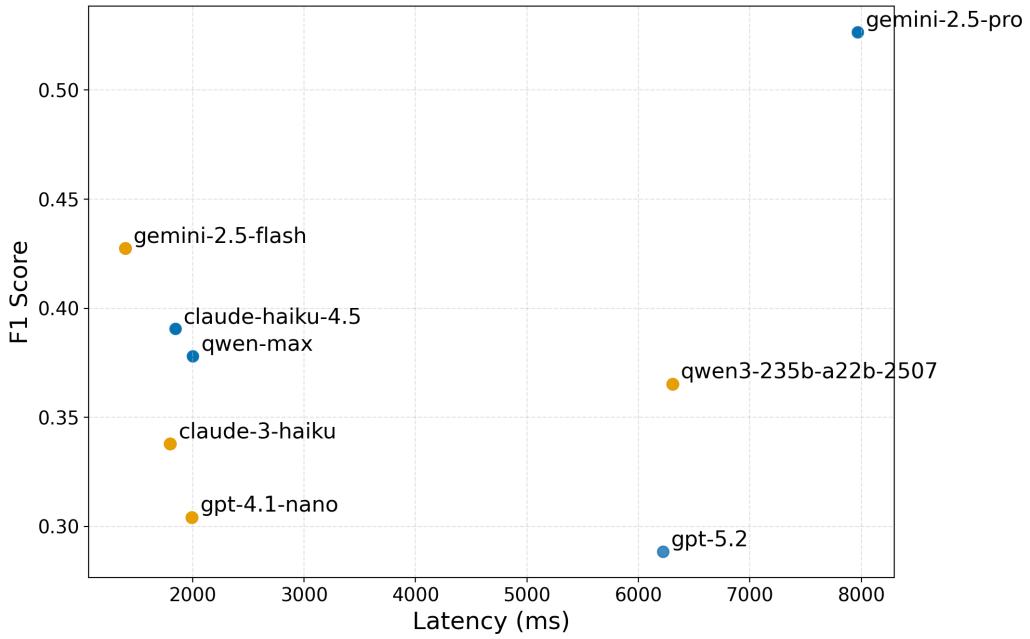

Figure 3 Plot of models' F1 Score versus Latency. Higher-priced models are marked dark blue, while lower-priced models are light orange.

5 Discussion

A primary methodological decision was to forgo model fine-tuning in favor of one-shot Chain-of-Thought prompting. Although fine-tuning provides domain adaptation, it frequently involves substantial additional data labeling and computing resources. However, the findings of this study indicate that the reliance on prompt engineering of readily available LLMs remains constrained. Our quantitative evaluation indicates that even the model with the highest precision and recall values exhibits a precision of only 0.70, signifying that almost one-third of the items it returns are irrelevant. Furthermore, the recall value is 0.41, indicating that it does not return more than half of the relevant items in the observatory.

The construction of the test set used in this study also merits discussion. A single human annotator was assigned to label items in the observatory as relevant or irrelevant for a given user prompt. The efficacy of this process was improved using embeddings to sort items according to their relevance, as detailed in Subsection 4.1. However, the execution of this labeling task requires expertise that the annotator may not possess. For example, the annotator may not discern that an item mentioning "barrier-free tourism" is pertinent to a prompt inquiry about accessibility in tourism. Another issue that has been identified is the ambiguity that may be present in the user prompt. One of the inquiries posed in the test set pertains to "booking systems," thereby prompting the following question: do ticketing systems constitute booking systems? The question arises as to whether only systems utilized for the reservation of a particular, time-bound resource should be designated as booking systems. It is important to note that different annotators may interpret this differently, just as different models can generate boolean queries that imply different interpretations.

6 Conclusion and Future Work

This paper presents the design and implementation of an LLM-powered, filter-based conversational agent for the EUROSTIT Observatory. Through prompt engineering, we developed a system that translates natural language into structured boolean search queries, making it easier to find Smart Tourism Initiatives and Tools. Our approach prioritizes simplicity and adaptability, so our application can adapt to different content management systems and domains without costly fine-tuning.

Future work will involve the implementation of more sophisticated AI engineering methodologies to enhance the efficacy of the filter-based paradigm. The goal is to generate more relevant matches and decrease the number of irrelevant ones. For instance, we could incorporate keyword embeddings, similar to those employed in our evaluation, into the application's intelligence to enhance its capacity to more effectively handle synonyms. Furthermore, the system's effectiveness could be enhanced by replacing the current full-text search based on MySQL with dedicated search solutions, such as Apache Solr.

References

1 Gaelen P Adam, Jay DeYoung, Alice Paul, Ian J Saldanha, Ethan M Balk, Thomas A Trikalinos, and Byron C Wallace. Literature search sandbox: a large language model that generates search queries for systematic reviews. *JAMIA open*, 7(3):ooae098, 2024. doi:10.1093/jamiaopen/ooae098.

2 A. Alharbi, A. Pandit, V. Wilk, P.J. Rosenberger III, and S. Miah. Artificial intelligence-enabled conversational agents in tourism & hospitality: a systematic literature review & future research directions. *Asia Pacific Journal of Tourism Research*, 31(1):21–52, 2026. doi:10.1080/10941665.2025.2555204.

3 R. Alotaibi, A. Ali, H. Alharthi, and R. Almehamdi. AI chatbot for tourist recommendations: a case study in the city of Jeddah, Saudi Arabia. *International Journal of Interactive Mobile Technologies*, 14(19):18–30, 2020. doi:10.3991/ijim.v14i19.17201.

4 L. Benaddi, C. Ouaddi, A. Jakimi, and B. Ouchao. A systematic review of chatbots: Classification, development, and their impact on tourism. *IEEE Access*, 12:78799–78810, 2024. doi:10.1109/ACCESS.2024.3408108.

5 L. Benaddi, C. Ouaddi, A. Souha, A. Jakimi, and B. Ouchao. Chatbots in tourism sector: Classification, evolution, and functionalities. In *Proceedings of the 12th International Symposium on Signal, Image, Video and Communications (ISIVC)*, pages 1–6. IEEE, 2024. doi:10.1109/ISIVC61350.2024.10577889.

6 Dimitrios Buhalis. Marketing the competitive destination of the future. *Tourism management*, 21(1):97–116, 2000. doi:10.1016/S0261-5177(99)00095-3.

7 Mario Casillo, Federica Clarizia, Giuseppe D'Aniello, Massimo De Santo, Marco Lombardi, and Domenico Santaniello. CHAT-Bot: A cultural heritage aware teller-bot for supporting touristic experiences. *Pattern Recognition Letters*, 131:234–243, 2020. doi:10.1016/j.patrec.2020.01.003.

8 Mario Casillo, Massimo De Santo, R. Mosca, et al. An ontology-based chatbot to enhance experiential learning in a cultural heritage scenario. *Frontiers in Artificial Intelligence*, 5, 2022. doi:10.3389/frai.2022.808281.

9 Diogo Cosme, Fernando Batista, António Galvão, and Fernando Brito e Abreu. Automating the Classification of Smart Tourism Tools with LLMs: A Taxonomy-Driven Approach. In *Proceedings of the 15th Symposium on Languages, Applications and Technologies (SLATE)*, Open Access Series in Informatics (OASIcs), Lisbon, Portugal, June 2026. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. To appear. doi:10.4230/OASIcs.SLATE.2026.18.

10 European Commission. Leading Examples of Smart Tourism Practices in Europe from the 2025 European Capital of Smart Tourism Competition, 2025. URL: https://smart-tourism-capital.ec.europa.eu/best-practices/european-capital-smart-tourism-best-practices_en.

11 António Galvão, Fernando Brito e Abreu, and João Joanaz de Melo. Toward a Consensual Definition for Smart Tourism and Smart Tourism Tools. In *Smart Life and Smart Life Engineering: Current State and Future Vision*, pages 153–183. Springer, 2025. doi:10.1007/978-3-031-75887-4_8.

12 Joanna Gotlib-Małkowska, Ilona Cieślak, Mariusz Jaworski, and Mariusz Panczyk. Evidence-based advanced prompt engineering in nursing research: quality analysis of ChatGPT-generated Boolean search query. *Nursing in the 21st Century*, 24(1 (90)):9–19, 2025. doi:10.12923/pielxxiw-2025-0002.

13 A. Kontogianni and E. Alepis. AI in smart tourism: LLMs & GPTs leading the way. In *Proceedings of the 5th International Conference on Information, Intelligence, Systems and Applications (IISA)*, pages 1–8. IEEE, 2024. doi:10.1109/IISA62523.2024.10786696.

14 M.I. Saleh. Generative artificial intelligence in hospitality and tourism: future capabilities, AI prompts and real-world applications. *Journal of Hospitality Marketing & Management*, 34(4):467–498, 2025. doi:10.1080/19368623.2025.2458603.

15 SEGITTUR. Catalogue of Technological Solutions for Smart Tourist Destinations – Third Edition. Smart Tourist Destination, 2024. URL: https://www.destinosinteligentes.es/wp-content/uploads/2024/02/Segitur_guia_fichas_2024@14703546813.pdf.

16 Giancarlo Sperlí. A cultural heritage framework using a Deep Learning based Chatbot for supporting tourist journey. *Expert Systems with Applications*, 183:115277, 2021. doi:10.1016/j.eswa.2021.115277.

17 Shuai Wang, Harrisen Scells, Bevan Koopman, and Guido Zuccon. Can ChatGPT Write a Good Boolean Query for Systematic Review Literature Search? In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'23)*, pages 1426–1436. ACM, 2023. doi:10.1145/3539618.3591703.