

Inferno: End-to-end Agent-based FHIR Resource Synthesis from Free-form Clinical Notes

Johann Frei¹ Nils Feldhus^{2,3,4} Lisa Raithe^{1,2,3,4}
Roland Roller⁴ Alexander Meyer^{2,5} Frank Kramer¹

¹IT-Infrastructure for Translational Medical Research, University of Augsburg

²BIFOLD – Berlin Institute for the Foundations of Learning and Data ³Technische Universität Berlin

⁴German Research Center for Artificial Intelligence (DFKI), Berlin

⁵IKIM, Charité - Universitätsmedizin Berlin

Abstract

For clinical data integration and healthcare services, the HL7 FHIR standard has established itself as a desirable format for interoperability between complex health data. Previous attempts at automating the translation from free-form clinical notes into structured FHIR resources address narrowly defined tasks and rely on modular approaches or LLMs with instruction tuning and constrained decoding. As those solutions frequently suffer from limited generalizability and structural inconformity, we propose an end-to-end framework powered by LLM agents, code execution, and healthcare terminology database tools to address these issues. Our solution, called Inferno, is designed to adhere to the FHIR document schema and competes well with a human baseline in predicting FHIR resources from unstructured text. The implementation features a front end for custom and synthetic data and both local and proprietary models, supporting clinical data integration processes and interoperability across institutions. Gemini 2.5-Pro excels in our evaluation on synthetic and clinical datasets, yet ambiguity and feasibility of collecting ground-truth data remain open problems.

1 Introduction

Large language models (LLMs) have demonstrated strong performance in clinical and biomedical domains, as they have been shown to encode domain-specific knowledge (Singhal et al., 2023; Moor et al., 2023). They are increasingly used to answer clinical questions by processing relevant documents at inference time (Zakka et al., 2024; Chen et al., 2025a; Wang et al., 2024). However, this retrieval-based approach incurs significant latency and computational cost, as documents must be re-processed for every query. This limits usability for tasks such as retrospective analysis or study planning with multiple queries over the same data (Coromilas et al., 2021; Leibig et al., 2022).

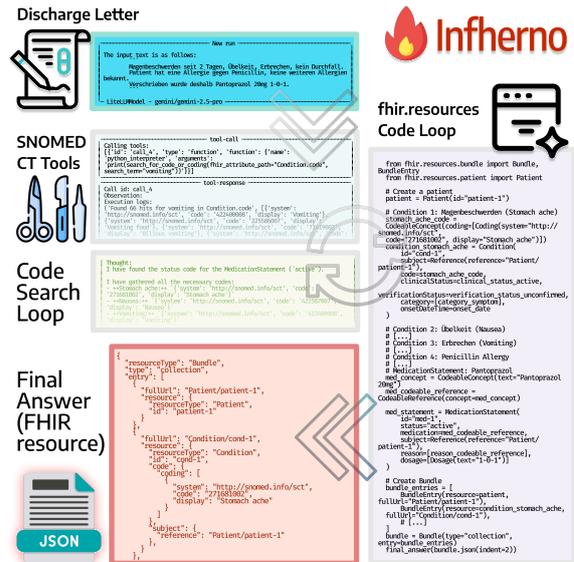


Figure 1: Illustrative example of how Inferno, an agentic approach for FHIR resource synthesis, processes a discharge letter (top left, cyan) using SNOMED CT tools (light blue) and terminology search (green) and fhir.resources code loops (purple, right). After a few iterations including tool calls and observations from a Python executor, the LLM agent proceeds to produce a final answer (red) in a FHIR/JSON format, representing the clinical information on patients and medications.

A more scalable solution is to extract structured representations from clinical text in advance. If the extracted structure preserves the relevant information, it can be queried and reused instantly across multiple applications. This is particularly important for healthcare service providers and clinical data integration efforts (Leroux et al., 2017; Hong et al., 2019; Pimenta et al., 2023). Here, the FHIR¹ standard provides a flexible and interoperable format for representing healthcare data and is increasingly adopted to support standardized access to complex medical information.

Conventional information extraction (IE) meth-

¹<https://fhir.org/>

ods, such as classical named entity recognition (NER), relation extraction, and entity linking, are typically designed for narrowly defined tasks and fixed schemata. As such, they lack the flexibility to adapt to complex clinical contexts and often fail to produce complete, structured clinical representations. In contrast, LLMs have shown promise for IE methods when they are framed as structured prediction tasks (Dagdelen et al., 2024). A recurring challenge in such tasks is ensuring that the output adheres to a specified schema, particularly when downstream components require well-structured inputs (Tavanaei et al., 2024). This is especially true in domains like healthcare, where semantic correctness and schema compliance are essential. Various approaches have been proposed to guide LLM outputs toward structural conformity, including fine-tuning, pre-training, instruction tuning, or constrained decoding (Shin et al., 2021; Geng et al., 2023). Some of them have dealt with text-to-FHIR translation (Sharma et al., 2023; Li et al., 2024; Tabari et al., 2025; Pope and Patooghy, 2025), but often encountered inconsistencies with the desired schema. Agentic LLM approaches that “reason” through intermediate steps using external tools have emerged as a promising solution. Inspired by frameworks like Toolformer (Schick et al., 2023) and ReAct (Yao et al., 2023), models perform multiple tool-augmented reasoning steps, with validation and retry mechanisms to ensure correct output.

As our key contribution, we propose an end-to-end framework that transforms unstructured clinical text into rich, semantically accurate FHIR representations using an agentic LLM-approach. This enables holistic information extraction (Zhang et al., 2025; Shao et al., 2025), supports integration of both legacy and new data, and fosters interoperability across institutions. Our contribution involves:

- (1) An end-to-end implementation for text-to-FHIR translation using LLM agents, SNOMED CT terminology integration and FHIR schema validation;
- (2) Evaluation on real-life and synthetic data, with quantitative and qualitative error analyses across both proprietary and open-source LLMs to characterize failures and their severity;
- (3) A lightweight demonstrator with front-end functionality, supporting both locally run and API-based state-of-the-art LLMs.

2 Background

FHIR and SNOMED CT FHIR (Fast Healthcare Interoperability Resources) is a widely adopted standard for exchanging healthcare-related data, developed by the HL7 organization. FHIR defines resources as nested documents, often encoded in JSON, with well-defined types, required fields, enumerations, and references to other resources. A single FHIR resource can represent a broad range of entities, from patients and conditions to administrative structures like coverage or questionnaires.² FHIR facilitates the structured encoding of complex medical information in an interoperable fashion.

A key feature of FHIR is the integration of internal and external code systems, composed as ValueSets to reference specific entities and concepts. Certain data elements may be constrained to a fixed, FHIR-internal code system to define the set of valid data values.³ For certain fields, concepts can be referenced from external coding systems such as SNOMED CT⁴ or LOINC⁵, and the set of valid data values can be further constrained by individual ValueSets, e.g., to limit data entries for body site to the subset of SNOMED CT concepts that only refer to body structures. To search for codes and terms in a specific ValueSet, FHIR terminology servers provide a standardized interface for querying valid concepts. These servers commonly support multiple external code systems in addition to the FHIR-internal code systems.

While the FHIR schema is capable to accurately and verbosely capture complex clinical situations, it is also subject to structural and semantic ambiguity. Practitioners often use only relevant subsets of data elements depending on their specific use cases. In addition, the standard does not always enforce the encoding of certain information into an unambiguous representation. For instance, a bone fracture of the left limb may be expressed as a *Fracture of bone* SNOMED CT concept along with the *bodySite* element referring to the *Structure of left hand* concept, or purely by referring to the *Fracture of bone of left hand* concept. Dosage information could be phrased only by a free-form

²See an example for a Patient resource object at: <https://hl7.org/fhir/R4/patient-example.json.html>

³For instance, Condition.clinicalStatus only allows the values active, recurrence, relapse, inactive, remission, and resolved.

⁴<https://www.snomed.org/what-is-snomed-ct>

⁵<https://loinc.org/get-started/what-loinc-is/>

text element, or by fully utilizing all relevant structured elements, rendering both approaches valid. Clinical notes may also be rather imprecise or ambiguous and require additional and subjective interpretation to fully infer the intended meaning, yet this issue also affects other, non-FHIR-based IE systems. Therefore, comparing predicted and ground-truth FHIR data for semantic equivalence and correctness remains a non-trivial task.

Related Work Sharma et al. (2023) presented a pipeline for digitizing prescription images into FHIR using separate components for extraction, normalization, and entity linking, limited to this particular task and mostly small-scale models. Li et al. (2024) first applied LLMs to clinical text-to-FHIR transformation with human-annotated data⁶, but were limited to MedicationStatement resources and faced JSON parsing issues, whereas our validation loop ensures format conformity. Tabari et al. (2025) integrated a syntactic validator and zero- and few-shot strategies into their text-to-FHIR pipeline. Their setup is constrained to sentence-level conversion and exhibits less transparency due to the separation between the OpenAI model and the validator. In contrast, *Infherno*'s tool-calling approach offers a higher degree of transparency and a larger variety of model choices. Pope and Patooghy (2025) explored a variety of FHIR-related tasks as a benchmark, but simplified them to short QA-style problems and also did not consider any elaborate pipeline with tools. Lee et al. (2025) presented FHIR-AgentBench, a comprehensive benchmark for evaluating LLM agents on clinical question answering over FHIR-structured EHR data. Unlike Pope and Patooghy's simplified tasks, they assess complex multi-step retrieval and reasoning over realistic FHIR resources, though their focus remains on querying existing data rather than generation. Idrissi-Yaghir et al. (2025) presented FHIR Workbench for evaluating text-to-FHIR generation, though models struggled without tool augmentation or validation mechanisms. Riquelme Tornel et al. (2025) used GPT-4o and Llama-3.2 alongside clustering and retrieval generation approaches to perform automated FHIR mappings on MIMIC-IV (instead of free text), but missed out on evaluating the results manually. Finally, Schmiedmayer et al. (2025) aimed for an inverse perspective on the translation task by developing a mobile application that

⁶The human-annotated FHIR-GPT data has not been open-sourced to the best of our knowledge.

allows users to interact with FHIR resources via an LLM, while Ehtesham et al. (2025) presented an MCP-based agent for summarization and interpretation. Both represent a FHIR-to-text scenario which is focused on patient understanding.

3 *Infherno*, an Agentic Approach

Building on recent work on LLM agents in the medical domain (Liao et al., 2025; Rose et al., 2025; Chen et al., 2025b; Wang et al., 2025), we propose an agentic framework that incorporates tool calls and coding to generate structured FHIR output from unstructured clinical text.

The core task is to transform an unstructured clinical text into semantically corresponding FHIR representations. Our approach follows the Thought-Code-Observation structure proposed as the ReAct framework by Yao et al. (2023), and is implemented using the Smolagents (Roucher et al., 2025) library which supports multi-step LLM agents with Python-code execution. Figure 1 presents a simplified example of the *Infherno* pipeline⁷: Given a discharge letter, *Infherno* which is equipped with tools accessing SNOMED CT, performing Terminology Search, and executing Python code, is tasked to extract information pertinent to patients and medications. In the following, we describe each component:

Prompt Structure To guide the agent's behavior, we include relevant contextual information into the prompt (Figure 4, top left). This includes the unstructured input text, a list of target FHIR resource types, supported ValueSets, example code snippets demonstrating FHIR object creation, and a set of instructions on desired behaviors and constraints.

Terminology Search To integrate FHIR-specific codes that conform to its specification, we provide our agentic system with an external, retrieval-augmented generation-based function to query particular terms in a set of supported FHIR ValueSets. This enables the agent to rely on external code systems, in particular the SNOMED CT ontology, to retrieve potential search results and include them into its context window. The external function call binds to an external FHIR terminology server to obtain a valid query response.

Structured Data as Code Within the agent code execution stage, the agent is incentivized to use the

⁷Figure 4 in Appendix B shows the extended version.

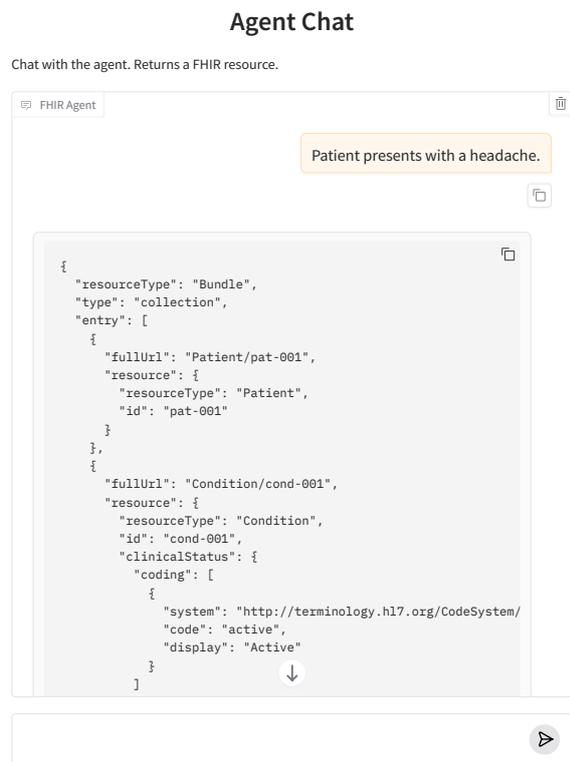


Figure 2: Front end of Infherno showing a short input text and the final answer as given by Gemini-2.5-Pro in the *Agent Chat* function.

`fhir.resources`⁸ Python module to create FHIR-conform data instances in a object-oriented fashion. This approach is crucial as it is able to catch morphological and syntactic errors early within the life cycle of the agent loop, and avoids cumbersome data validation that may arise from a purely JSON-centric FHIR document generation by the LLM. Since the library can directly provide error feedback, it can also facilitate the recovery from erroneous code predicted initially by the agent.

Output Formatting As part of the Smolagents framework, the code agent can stop the agent loop by the `final_answer` function call. Hereby, the agent is instructed to use the JSON-based object serialization of the `fhir.resources` module. This ensures that the response provides a structurally valid, FHIR-compliant JSON output. To deliver all generated FHIR resources to the user, the agent is instructed to aggregate them into a FHIR *Bundle* that encapsulates the complete set.

⁸<https://github.com/nazrulworld/fhir.resources/tree/8.0.0>

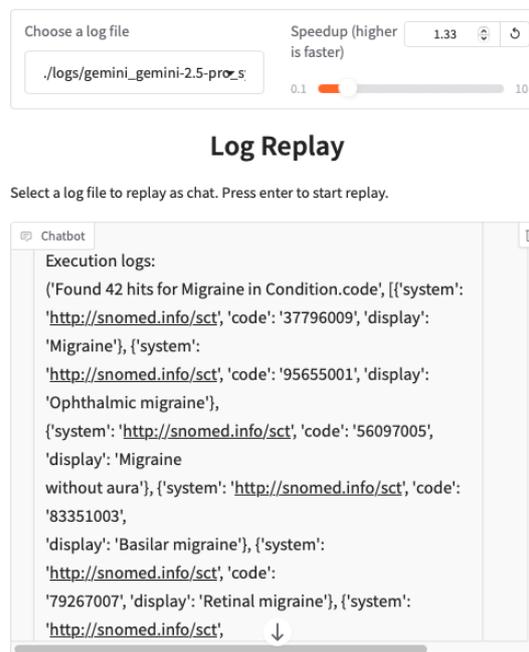


Figure 3: Front end of Infherno showing an intermediate step (Terminology Search) during the text-to-FHIR translation with the *Log Replay* function.

Front End The visual interface of Infherno is built on top of Gradio⁹ and allows the user to enter arbitrary clinical notes or select pre-defined examples from our synthetic dataset (Figure 2). Intermediate steps including the tool calls and tool responses as well as the reasoning processes (“thoughts” and “observations”) of the ReAct framework (Yao et al., 2023) are shown at inference time. A *Log Replay* tab (Figure 3) also enables to simulate the execution of already conducted experiments at custom speed without the need of an API key. It supports the default Gemini API and OpenAI API via LiteLLM¹⁰ and both API and local Hugging Face models – a feature inherited from Smolagents. The front-end app is available on <https://github.com/j-frei/Infherno>.

4 Experiments

To validate our agentic approach, we apply the agent to a set of medical documents to transform the unstructured text into a set of FHIR resources. Such individual comparison is highly complex due to the depth and richness of the FHIR schema and

⁹<https://www.gradio.app/>

¹⁰<https://docs.litellm.ai/docs/>

the complexity of clinical language. Therefore, we conduct two experiments, involving both an automatic and a manual inspection and interpretation to assess the prediction quality.

Data For the experiments, we use two data sources. We source ten anonymized English EHR documents from the n2c2 2018 challenge (Henry et al., 2020) with drug-related NER annotations. The documents are used with focus on MedicationStatement, given the prior ground truth data. To craft ground-truth FHIR data, we first remove any drug-related NER spans if they do not refer to a medication instruction (e.g., drug allergies are not considered as such), normalize the drug-related NER span annotation into their common concepts to remove redundant mentions and create their corresponding MedicationStatement objects as ground truth. As the n2c2 texts are anonymized, common Protected Health Information (PHI) elements like patient names do not occur and cannot be extracted.

We also rely on ten ChatGPT-synthesized documents resembling medical discharge letters in German which include name and birth date mentions as well as described conditions and medication instructions, allowing us to share PHI-containing documents with remote LLM endpoints. The raw texts required manual editing. Obvious placeholder names, such as “Max Mustermann”, were replaced in the synthetic texts to ensure realistic, non-repetitive patient names and addresses to improve the authenticity of the data.¹¹ To obtain suitable reference FHIR data, we annotate the documents from the corpus by manually extracting the relevant corresponding FHIR resources, referred to as human baseline (HB). For these documents, we support Patient, Condition, and MedicationStatement as FHIR resources, since we consider these resource types to fit best to key clinical entities.

In general, the FHIR R4 release is targeted as it currently is the latest *normative* release version.

Experimental Setup The first experiment aims to implement an evaluation requiring only minimal manual intervention or semantic interpretation. To measure the overlap between the predicted and ground-truth MedicationStatement concepts, we list and compare the .medication fields and manually categorize predicted concepts into TP, FP,

¹¹The documents are publicly available on GitHub at <https://github.com/j-frei/Infherno>. One sample is included in the Appendix Figure 5.

FN to eventually calculate precision, recall and F_1 score. As the ground truth involves subjective interpretation of what drug mention is an actual medication instruction, we also add an *relaxed* score setting that ignores certain ground truth entries that were jointly ignored or skipped by all LLMs. We also track the use of SNOMED CT codes and their correctness, as certain predictions only encode the concepts as text field. For the first experiment, both the n2c2 and synthetic data is used.

For the second experiment, we extensively compare the manual annotation and the generated annotation by verifying individual *items* of each FHIR object across multiple FHIR resources. We define an item as a single unit of information, that may refer to, for instance, a single birthDate field but could also refer to a nested object item that describes a reference to a concept from an external coding system. Since the internal structure of certain objects is only meaningful in its entirety, we consider them as monolithic items in the evaluation, rather than decomposing their components. For this experiment, we only use synthetic data as it also covers synthetic PHI elements. Items are stratified into primary (e.g., Condition.code) and secondary (e.g., Condition.verificatiOnStatus) items to account for the information importance differences. Evaluation decision details are highlighted in Appendix Section A.

Models For our agentic approach, we use both commercial and open-weight models for the first setup. This includes Claude Sonnet 4.5, Gemini-2.5 Pro¹², GPT-5 as well as DeepSeek V3.1, Qwen3-235B-A22B-2507, and Qwen3-8B. As the second experiment incurs substantial manual effort, we select Gemini 2.5 Pro as target model as it performed best in the first setup on n2c2 data.

5 Results & Discussion

According to Table 1, all LLMs demonstrate high precision with minimal fabrication¹³, but recall varies substantially on n2c2 data, where Gemini 2.5 outperforms all other models. The eased scores reveal that most failures represent cases systematically difficult for all models. For instance, drug information mentioned prior to or outside of the DISCHARGE MEDICATION section is often skipped, as it remains unclear whether such prior mentions

¹²<https://ai.google.dev/gemini-api/docs/models>

¹³We found the FP in the synthetic data to be an error in the ground truth.

Data	Model	Scores						Eased Scores			Concepts
		TP	FP	FN	Pr	Re	F_1	FN	Re	F_1	total/with codes/with correct codes
n2c2 27436 tokens	Claude Sonnet 4.5	78	0	91	1.0	0.462	0.632	31	0.716	0.834	79 / 78 / 78
	DeepSeek V3.1 Chat	82	0	85	1.0	0.491	0.659	25	0.766	0.868	84 / 84 / 82
	Gemini-2.5 Pro	101	0	66	1.0	0.605	0.754	6	0.944	0.971	104 / 104 / 104
	GPT-5	68	0	100	1.0	0.405	0.576	40	0.630	0.773	69 / 63 / 42
	Qwen3-235B-A22B-2507	76	0	92	1.0	0.452	0.623	32	0.704	0.826	77 / 76 / 76
	Qwen3-8B	32	0	134	1.0	0.193	0.323	74	0.302	0.464	36 / 36 / 35
synthetic 4065 tokens	Claude Sonnet 4.5	13	1	3	0.929	0.813	0.867	2	0.867	0.897	14 / 14 / 14
	DeepSeek V3.1 Chat	15	1	1	0.938	0.938	0.938	0	1.0	0.968	16 / 16 / 16
	Gemini-2.5 Pro	14	1	1	0.933	0.933	0.933	0	1.0	0.966	15 / 15 / 15
	GPT-5	15	1	1	0.938	0.938	0.938	0	1.0	0.968	16 / 12 / 9
	Qwen3-235B-A22B-2507	15	1	1	0.938	0.938	0.938	0	1.0	0.968	16 / 15 / 15
	Qwen3-8B	15	1	1	0.938	0.938	0.938	0	1.0	0.968	16 / 15 / 15

Table 1: Evaluation scores from the first experiment, including Precision (Pr), Recall (Re), F_1 scores of *Inferno* with various LLMs evaluated on n2c2 and synthetic data. Best scores are in **bold**.

Category	Worse than HB		Neutral		Better than HB	
	prim	sec	prim	sec	prim	sec
importance						
semantically related	0	4	0	4	0	0
completely identical	0	0	121	83	0	0
lacking in HB	0	10	0	23	13	67
lacking in PD	6	15	0	67	0	0
value difference	0	10	0	12	5	1
semantic halluc. / invalid	1	9	0	0	0	0
total	46		314		86	

Table 2: Second experiment: Results from the manual analysis between predicted (PD) and human baseline (HB) indicating the success and failure cases of *Inferno* for **primary** and **secondary** items. Examples for different categories are shown in Appendix Table 3.

should be considered superseded. Performance on synthetic data confirms that all models extract reliably when ambiguity is low.

Table 2 presents detailed manual analysis comparing predictions against human annotations. Agreement is highest for primary items carrying essential clinical information. Notably, in nearly twice as many cases of quality differences, the model performed better than the human baseline rather than worse. Confabulations remain rare at the semantic level. Divergence occurs primarily on secondary items, which typically involve vague descriptions or ambiguous phrasing rather than explicitly stated primary elements.

Key Findings The validation highlights several important observations. First, the phrasing of the input text plays a critical role in annotation consistency. Vague or ambiguous expressions frequently lead to disagreements between the predicted and reference annotations, particularly for secondary items. In contrast, plainly stated and well-structured information is more reliably and consistently captured.

Second, many divergences can be attributed to

the partially subjective nature of FHIR in fringe cases. Minor or nonspecific health issues often fall into a gray area. These may either be excluded or encoded in different ways, such as as a Condition or an Observation. Since the experimental setup allowed only the use of Patient, MedicationStatement, and Condition resource types, the agent was not permitted to use the Observation resource, which limited some of its encoding options.

Furthermore, the *Inferno* agent appears to be more cautious when deciding whether to encode uncertain symptoms. At the same time, it demonstrates stronger recall for clearly stated information that human annotators sometimes overlook, especially with a state-of-the-art LLM like Gemini 2.5. For example, the agent successfully included an address field that was missing in the human annotation. It also inferred an onsetDateTime by subtracting six weeks from the encounter date, which is a detail the human annotator did not encode.

These findings indicate that while human annotations are prone to fatigue and inconsistency, especially in repetitive and detail-oriented tasks, automated agents benefit from their ability to process dense text data using their large context as receptive field. As a result, they can achieve more reliable and comprehensive structured data extraction from our clinical text samples.

6 Conclusion

In conclusion, *Inferno* presents a robust and effective framework and interface for transforming unstructured clinical data into standardized FHIR resources. Its agentic design, integrating external knowledge and validation, addresses critical chal-

allenges in clinical information extraction, paving the way for improved data interoperability in healthcare. Future work includes the fine-tuning of smaller language models on the text-to-FHIR task and the integration of more FHIR resource types while further strengthening the robustness, and evaluate the approach on more diverse datasets.

Limitations

Dataset and Annotations Manual evaluation of a larger, more diverse dataset was infeasible given the labor intensity and expertise required. We rely on a single annotator for the human baseline, which inherently introduces a degree of subjectivity.

Resource Types Furthermore, our scope was intentionally limited to a subset of FHIR resource types (Patient, Condition, MedicationStatement). Expanding to a broader range of FHIR resources would likely necessitate more verbose guidance in the system prompt, potentially increasing computational cost and latency.

Legal Remark Finally, from a legal perspective, it is important to note that Infherno interacts with a FHIR terminology server that includes an initialized SNOMED CT ontology. Therefore, a SNOMED CT license may be required if self-hosting a FHIR terminology server is desired.

CFG Baseline While we experimented with context-free grammar-based (CFG) approaches, we found that there are several reasons to object to this design choice:

- Creating a fully conformant FHIR grammar is a major engineering challenge.
- Applying a schema-based decoding may lead to a constrained decoding misalignment issue, which may result in divergences between constrained and unconstrained distributions, and may lead to generation instabilities and poor semantic outputs.
- While considering schema-based generation as an alternative to (fhir.resources-based) code-based FHIR Bundle assembly, other tasks like SNOMED CT code search must also be integrated into the pipeline process. There is no clear way of integrating all components in a non-agentic way. Orchestrating the pipeline in a non-agentic, multi-step pipeline flow rely on an inflexible, rigid process, which could negatively affect the final quality especially in complex clinical

situations that do not fit well into a rigid data transformation process.

- An agentic-based flow allows for certain semantic cross-checks within a FHIR Bundle, that cannot be verified through a CFG-based approach.
- In general, an agentic-based flow simplifies the addition of more (custom/user-defined) tools and validation checks.

Multilingual Support While one part of our evaluation focuses on a proprietary model (Gemini 2.5 Pro), our key aim of our work is to remain agnostic of specific LLMs. Consequently, support of other languages depends on the ability of the used LLM to process individual languages correctly rather than our system implementation. The system prompt is written in English and code switching is not used apart from the language in the input text. Since SNOMED CT is mostly an English system, English must be used to query for SNOMED CT codes.

Ethics Statement

Depending on the selection of the LLM, we want to emphasize that users should be careful in selecting what data they enter. Most of the real-world medical datasets have licences and usage restrictions, so we recommend to use synthetic data only. Users should acknowledge the risk of leaking private data and de-identification.

Acknowledgments

We thank the reviewers of the EACL 2026 and EMNLP 2025 System Demonstrations tracks for their valuable feedback. This research is funded by the Berlin Institute for the Foundations of Learning and Data (BIFOLD, ref. 01IS18037A) and the German Federal Ministry of Research, Technology and Space (MoMoTuBo, ref. FKZ01ZZ2008).

References

- Hanjie Chen, Zhouxiang Fang, Yash Singla, and Mark Dredze. 2025a. [Benchmarking large language models on answering and explaining challenging medical questions](#). In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3563–3599, Albuquerque, New Mexico. Association for Computational Linguistics.
- Shan Chen, Pedro José Ferreira Moreira, Yuxin Xiao, Samuel Schmidgall, Jeremy L. Warner, Hugo Aerts,

- Thomas Hartvigsen, Jack Gallifant, and Danielle Bitterman. 2025b. [Medbrowsecomp: Benchmarking medical deep research and computer use](#). In *The Second Workshop on GenAI for Health: Potential, Trust, and Policy Compliance*.
- Ellie J. Coromilas, Stephanie Kochav, Isaac Goldenthal, Angelo Biviano, Hasan Garan, Seth Goldberg, Joon-Hyuk Kim, Ilhwan Yeo, Cynthia Tracy, Shant Ayanian, Joseph Akar, Avinainder Singh, Shashank Jain, Leandro Zimerman, Maurício Pimentel, Stefan Osswald, Raphael Twerenbold, Nicolas Schaerli, Lia Crotti, and 58 others. 2021. [Worldwide survey of covid-19-associated arrhythmias](#). *Circulation: Arrhythmia and Electrophysiology*, 14(3):e009458.
- John Dagdelen, Alexander Dunn, Sanghoon Lee, Nicholas Walker, Andrew S Rosen, Gerbrand Ceder, Kristin A Persson, and Anubhav Jain. 2024. [Structured information extraction from scientific text with large language models](#). *Nature Communications*, 15(1):1418.
- Abul Ehtesham, Aditi Singh, and Saket Kumar. 2025. [Enhancing clinical decision support and ehr insights through llms and the model context protocol: An open-source mcp-fhir framework](#). *arXiv*, abs/2506.13800.
- Saibo Geng, Martin Josifoski, Maxime Peyrard, and Robert West. 2023. [Grammar-constrained decoding for structured NLP tasks without finetuning](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10932–10952, Singapore. Association for Computational Linguistics.
- Sam Henry, Kevin Buchan, Michele Filannino, Amber Stubbs, and Ozlem Uzuner. 2020. [2018 n2c2 shared task on adverse drug events and medication extraction in electronic health records](#). 27(1):3–12.
- Na Hong, Andrew Wen, Feichen Shen, Sunghwan Sohn, Chen Wang, Hongfang Liu, and Guoqian Jiang. 2019. [Developing a scalable fhir-based clinical data normalization pipeline for standardizing and integrating unstructured and structured electronic health record data](#). *JAMIA Open*, 2(4):570–579.
- Ahmad Idrissi-Yaghir, Kamyar Arzideh, Henning Schäfer, Bahadır Eryilmaz, Mikel Bahn, Yutong Wen, Katarzyna Borys, Eva Hartmann, Cynthia Schmidt, Obioma Pelka, Johannes Haubold, Christoph M Friedrich, Felix Nensa, and René Hirsch. 2025. [Using a diverse test suite to assess large language models on fast health care interoperability resources knowledge: Comparative analysis](#). *J Med Internet Res*, 27:e73540.
- Gyubok Lee, Elea Bach, Eric Yang, Tom Pollard, Alistair Johnson, Edward Choi, Yugang Jia, and Jong Ha Lee. 2025. [Fhir-agentbench: Benchmarking llm agents for realistic interoperable ehr question answering](#). *arXiv*, abs/2509.19319.
- Christian Leibig, Moritz Brehmer, Stefan Bunk, Dana-Lyn Byng, Katja Pinker, and Lale Umutlu. 2022. [Combining the strengths of radiologists and ai for breast cancer screening: a retrospective analysis](#). *The Lancet Digital Health*, 4(7):e507–e519.
- Hugo Leroux, Alejandro Metke-Jimenez, and Michael J Lawley. 2017. [Towards achieving semantic interoperability of clinical study data with fhir](#). *Journal of biomedical semantics*, 8:1–14.
- Yikuan Li, Hanyin Wang, Halid Z. Yerebakan, Yoshitsugu Shinagawa, and Yuan Luo. 2024. [FHIR-GPT enhances health interoperability with large language models](#). *NEJM AI*, 1(8):A1cs2300301.
- Yusheng Liao, Shuyang Jiang, Yanfeng Wang, and Yu Wang. 2025. [ReflecTool: Towards reflection-aware tool-augmented clinical agents](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13507–13531, Vienna, Austria. Association for Computational Linguistics.
- Michael Moor, Oishi Banerjee, Zahra Shakeri Hossein Abad, Harlan M Krumholz, Jure Leskovec, Eric J Topol, and Pranav Rajpurkar. 2023. [Foundation models for generalist medical artificial intelligence](#). *Nature*, 616(7956):259–265.
- Nuno Pimenta, António Chaves, Regina Sousa, António Abelha, and Hugo Peixoto. 2023. [Interoperability of clinical data through fhir: A review](#). *Procedia Computer Science*, 220:856–861. The 14th International Conference on Ambient Systems, Networks and Technologies Networks (ANT) and The 6th International Conference on Emerging Data and Industry 4.0 (EDI40).
- Tia Pope and Ahmad Patooghy. 2025. [Comparative evaluation of gpt models in fhir proficiency](#). *ACM Trans. Intell. Syst. Technol.* Just Accepted.
- Álvaro Riquelme Tornel, Pedro Costa del Amo, and Catalina Costa Martínez. 2025. [Large language models for automating clinical data standardization: H17 fhir use case](#). *arXiv*, abs/2507.03067.
- Daniel Philip Rose, Chia-Chien Hung, Marco Lepri, Israa Alqassem, Kiril Gashteovski, and Carolin Lawrence. 2025. [MEDDxAgent: A unified modular agent framework for explainable automatic differential diagnosis](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13803–13826, Vienna, Austria. Association for Computational Linguistics.
- Aymeric Roucher, Albert Villanova del Moral, Thomas Wolf, Leandro von Werra, and Erik Kaunistmäki. 2025. [‘smolagents’: a smol library to build great agentic systems](#). <https://github.com/huggingface/smolagents>.

- Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. [Toolformer: Language models can teach themselves to use tools](#). In *Thirty-seventh Conference on Neural Information Processing Systems*.
- Paul Schmiedmayer, Adrit Rao, Philipp Zagar, Lauren Aalami, Vishnu Ravi, Aydin Zahedivash, Dong han Yao, Arash Fereydooni, and Oliver Aalami. 2025. [Llmonfhir](#). *JACC: Advances*, 4(6_Part_1):101780.
- Chong Shao, Douglas Snyder, Chiran Li, Bowen Gu, Kerry Ngan, Chun-Ting Yang, Jiageng Wu, Richard Wyss, Kueiyu Joshua Lin, and Jie Yang. 2025. [Scalable medication extraction and discontinuation identification from electronic health records using large language models](#). *Journal of Clinical Epidemiology*.
- Megha Sharma, Tushar Vatsal, Srujana Merugu, and Aruna Rajan. 2023. [Automated digitization of unstructured medical prescriptions](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 5: Industry Track)*, pages 794–805, Toronto, Canada. Association for Computational Linguistics.
- Richard Shin, Christopher Lin, Sam Thomson, Charles Chen, Subhro Roy, Emmanouil Antonios Platanios, Adam Pauls, Dan Klein, Jason Eisner, and Benjamin Van Durme. 2021. [Constrained language models yield few-shot semantic parsers](#). In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7699–7715, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.
- Karan Singhal, Shekoofeh Azizi, Tao Tu, S Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, and 1 others. 2023. [Large language models encode clinical knowledge](#). *Nature*, 620(7972):172–180.
- Parinaz Tabari, Alfonso Piscitelli, Gennaro Costagliola, and Mattia de Rosa. 2025. [Assessing the potential of an llm-powered system for enhancing fhir resource validation](#). In *Intelligent Health Systems—From Technology to Data and Knowledge*, pages 803–807. IOS Press.
- Amir Tavanaei, Kee Kiat Koo, Hayreddin Ceker, Shaobai Jiang, Qi Li, Julien Han, and Karim Bouyarmane. 2024. [Structured object language modeling \(SO-LM\): Native structured objects generation conforming to complex schemas with self-supervised denoising](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 821–828, Miami, Florida, US. Association for Computational Linguistics.
- Bowen Wang, Jiuyang Chang, Yiming Qian, Guoxin Chen, Junhao Chen, Zhouqiang Jiang, Jiahao Zhang, Yuta Nakashima, and Hajime Nagahara. 2024. [DiReCT: Diagnostic reasoning for clinical notes via large language models](#). In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.
- Wenxuan Wang, Zizhan Ma, Zheng Wang, Chenghan Wu, Jiaming Ji, Wenting Chen, Xiang Li, and Yixuan Yuan. 2025. [A survey of LLM-based agents in medicine: How far are we from baymax?](#) In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 10345–10359, Vienna, Austria. Association for Computational Linguistics.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. [React: Synergizing reasoning and acting in language models](#). In *The Eleventh International Conference on Learning Representations*.
- Cyril Zakka, Rohan Shad, Akash Chaurasia, Alex R. Dalal, Jennifer L. Kim, Michael Moor, Robyn Fong, Curran Phillips, Kevin Alexander, Euan Ashley, Jack Boyd, Kathleen Boyd, Karen Hirsch, Curt Langlotz, Rita Lee, Joanna Melia, Joanna Nelson, Karim Salim, Stacey Tullis, and 3 others. 2024. [Almanac—retrieval-augmented language models for clinical medicine](#). *NEJM AI*, 1(2):AIoa2300068.
- Xiao Yu Cindy Zhang, Carlos R. Ferreira, Francis Rossignol, Raymond T. Ng, Wyeth Wasserman, and Jian Zhu. 2025. [Casereportbench: An llm benchmark dataset for dense information extraction in clinical case reports](#). In *Proceedings of the sixth Conference on Health, Inference, and Learning*, volume 287 of *Proceedings of Machine Learning Research*, pages 527–542. PMLR.

A Quantitative Analysis

We manually compared the items of each FHIR object in the human baseline annotation with their corresponding agent-generated equivalents. Missing items were tracked using + for those absent in the human baseline and - for those missing in the prediction. Potentially equivalent values were categorized as exact matches (==), semantically equivalent (=), or different (+-).

Items were tagged with / if the prediction value was preferable to the baseline, and with | if the baseline value was preferable. Items were left untagged when no clear preference could be determined or justified.

To distinguish essential from less relevant features during evaluation, we annotated core FHIR objects of major importance, such as patient information or the main diagnosis extracted from the input text, with ! as primary items, and less critical elements, like vaguely described symptoms, with ? as secondary items. The importance level

Level of equivalence		Infherno (w/ Gemini-2.5-Pro)	Human Baseline
Worse than HB (field not referenced) (hallucination)	[+-?] [X+?]	Condition.bodySite Condition.category	"text": "Stirnbereich" "code": "symptom" "code": "52795006", "display": "Forehead structure" N/A
Neutral (optional field missing) (total equivalence)	[-?] [==]	MS.dosage .doseAndRate.type Condition.severity	N/A N/A "code": "255604002", "display": "Mild" "code": "255604002", "display": "Mild"
Better than HB (inaccurate reference for <i>throw up</i>)	[/+!]	Condition.code	"code": "422400008", "display": "Vomiting" "code": "422587007", "display": "Nausea (finding)"

Table 3: Examples for level of equivalence and the manual validation between system output and human baseline.

of a FHIR object determined the default importance of its internal items, unless overridden by manually applied, item-specific tags. These overrides were primarily used to demote non-essential items such as `Condition.verificationStatus` or `Patient.name.use`. Conversely, items from otherwise crucial FHIR resources, such as `Condition.subject` or `Condition.code`, were generally considered primary, as they carry core informational content. Some examples are shown in Table 3.

B Examples

Figure 4 illustrates a complete example of an text-to-FHIR translation flow.

C Example of a Synthetic Clinical Document

The Figure 5 shows the first document from our synthesized text corpus. All documents are accessible on GitHub at the following url: <https://github.com/j-frei/Infherno>.

...system

You are a code agent with expertise in Information Extraction of medical information from free text. Your task is to translate clinical information into valid FHIR R4 resources, using step-by-step reasoning and supported tooling.

You work by **thinking step by step** in cycles of:

- Thought:** - where you explain your current reasoning and intended approach.
- Code:** - where you write Python code using 'fhir.resources' or supported tools, always ending with 'end_code'.
- Observation:** - where you receive the printed outputs or results of your code.

You repeat this cycle until you're ready to give the final output using the 'final_answer()' tool.

At each cycle step, you must start next line plainly with "Thought:", "Code:", or "Observation:" (without quotes), otherwise it will fail.

Tools Available:

You ONLY have access to the following tools:

- Bundle:** (the top-level container)
- Patient**
- Condition**
- MedicationStatement**

All output must be a valid 'bundle' of these resources using the 'fhir.resources' Python package.

Attribute Coding Rules:

- You must only use codings returned from 'search_for_code_or_coding', and only for these specific FHIR attribute paths:
- Patient.name.use
- Patient.contact.system
- Patient.address.use
- Patient.address.type
- Patient.maritalStatus
- Patient.contact.relationship
- Condition.clinicalStatus
- Condition.verificationStatus
- Condition.category
- Condition.severity
- Condition.code
- Condition.bodySite
- Condition.stage.summary
- Condition.stage.method
- Condition.evidence
- MedicationStatement.status
- MedicationStatement.statusReason
- MedicationStatement.category
- MedicationStatement.medication
- MedicationStatement.codableConcept
- MedicationStatement.reasonCode
- MedicationStatement.dosage.additionalInstruction
- MedicationStatement.dosage.timing.repeat.dayOfWeek
- MedicationStatement.dosage.timing.repeat.when
- MedicationStatement.dosage.timing.code
- MedicationStatement.dosage.asNeeded
- MedicationStatement.dosage.asNeededCodableConcept
- MedicationStatement.dosage.site
- MedicationStatement.dosage.route
- MedicationStatement.dosage.method
- MedicationStatement.dosage.doseAndRate.type

Use this pattern to search for codes:

For example, to search for a code for "douleurs abdominales" in the Condition.code attribute, you would call:

```
Code: """
search_results =
search_for_code_or_coding(fhir_attribute_path="Condition.code",
search_term="abdominal pain")
print(search_results)
"""
end_code
Observation: ('Found results for 'abdominal pain' in 'Condition.code' truncated to max. of 10 results.', [{'code': '21522001', 'system': 'http://snomed.info/sct', 'display': 'Abdominal pain (finding)'}], {'code': '16294009', 'system': 'http://snomed.info/sct', 'display': 'Abdominal wall pain (finding)'}], {'code': '4597903', 'system': 'http://snomed.info/sct', 'display': 'Abdominal wind pain (finding)'}], {'code': '9991608', 'system': 'http://snomed.info/sct', 'display': 'Abdominal colic (finding)'}], {'code': '5458694', 'system': 'http://snomed.info/sct', 'display': 'Lower abdominal pain (finding)'}], {'code': '6332093', 'system': 'http://snomed.info/sct', 'display': 'Upper abdominal pain (finding)'}], {'code': '11629004', 'system': 'http://snomed.info/sct', 'display': 'Acute abdominal pain (finding)'}], {'code': '2822100101919', 'system': 'http://snomed.info/sct', 'display': 'Abdominal muscle pain (finding)'}], {'code': '11985807', 'system': 'http://snomed.info/sct', 'display': 'Chronic abdominal pain (finding)'}], {'code': '16294009', 'system': 'http://snomed.info/sct', 'display': 'Central abdominal pain (finding)'}])
"""
end_code
Example (search for Patient.gender code for a man)
Code: """
search_results =
search_for_code_or_coding(fhir_attribute_path="Patient.gender",
search_term="male")
print(search_results)
"""
end_code
Observation: ('Found results for 'man' in 'Patient.gender' truncated to max. of 10 results.', [{'code': 'male', 'code': 'female', 'code': 'other', 'code': 'unknown'}])
"""
end_code
You need to search and print the results before the creating the final FHIR data in the final step, where you pick the most fitting items from the observed print outputs.


Process Strategy:



- Start with a Thought: describing your plan to extract relevant medical info and convert to FHIR resources.
- Use Code: blocks to either:
  - call 'search_for_code_or_coding' to get valid SNOMED/HL7 codings and print it
  - construct FHIR resource objects ('Patient', 'Condition', etc.) (when you have all the needed info from previous steps)
  - build the final Bundle
  - call 'final_answer(bundle.json(indent=2))' to output the final result
- At each step, explain and print intermediate info you'll need in later steps.
- Never reuse tool parameters unnecessarily; only call tools when needed.
- Stick to valid FHIR attributes and use 'fhir.resources' models.



Final Code Block



When you're ready, build the final bundle.



Let's consider an example where the input text is "Herr Meyer klagt über Bauchschmerzen."



We need to extract the Patient and Condition information from this text and create a FHIR Bundle.



So you will first search for the coding for "abdominal pain" in the Condition.code attribute to find the appropriate code for abdominal pain, as well as other relevant codings (e.g. for Patient.gender).



This will take a few steps, and you will print the results of each search.



Then, you will create a Patient resource with the name "Meyer" and a Condition resource with the found code. Finally, you will bundle them together in a FHIR Bundle, using the codings and codes that you have already queried in the previous steps, like this:


```

System Prompt

```
Code: """
from fhir.resources.bundle import Bundle, BundleEntry
from fhir.resources.patient import Patient
from fhir.resources.condition import Condition
from fhir.resources.humanname import HumanName
from fhir.resources.codableconcept import CodableConcept
from fhir.resources.reference import Reference
from fhir.resources.reference import Reference

# Build Patient
name = HumanName(use="official", family="Meyer")
patient = Patient(id="pat-001", gender="male", name=name)

# Build Condition
condition_code = Coding(system="http://snomed.info/sct", code="21522001", display="Abdominal pain")
clinical_status = Coding(system="http://terminology.hl7.org/CodeSystem/condition-clinical", code="active")
condition = Condition(subject=Reference(reference="Patient(patient-id)"), condition_code = CodableConcept(coding=condition_code))

# Create Bundle
bundle = Bundle(type="collection", entry=[
    BundleEntry(resource=patient),
    BundleEntry(resource=condition)
])

final_answer(bundle.json(indent=2))
"""
end_code
Now, this code example is just for illustration purposes for an imaginary male Patient with the name 'Meyer'. Do not use any information from it directly.
You will need to adapt the code to your specific task and FHIR resources matching the upcoming input text. Yet, the basic task structure remains similar.
Important: The codes/codings should be determined before this step.
Always:
- Use valid 'fhir.resources' models and their documented fields
- Use codings only from 'search_for_code_or_coding'
- Output a single Bundle using 'final_answer' formatted in JSON (and indent=2)
- Verify the codes/codings before using them in the final step.
- Hard-code the picked coding/code data which was observed by the search tool in prior steps
- Don't put any code or text after the final_answer() call. Otherwise, it will fail.
Never:
- NEVER use unverified FHIR resources
- NEVER use unverified codes
- NEVER call tools with dict-style args (e.g. 'X' tool({'arg': val}))
- NEVER chain tool calls in the same block with unpredictable results
- NEVER create FHIR resources to information that is not present in the input text.
- NEVER use the search_for_code_or_coding in the final step.
Now begin your task!
If you solve the task correctly, you will receive a reward of $1,000,000.
```

Input Document/EHR

The input text is as follows:

Herrnbeschwerden seit 2 Tagen, Übelkeit, Erbrechen, kein Durchfall.
Patient hat eine Allergie gegen Penicillin, keine weiteren Allergien bekannt.
Verschrieben wurde deshalb Pantoprazol 20mg 1-0-1.

- LiteLJModel - gemini/gemini-2.5-pro

FHIR Resource

```
{
  "resourceType": "Bundle",
  "type": "collection",
  "entry": [
    {
      "fullUrl": "Patient/patient-1",
      "resource": {
        "resourceType": "Patient",
        "id": "patient-1"
      }
    },
    {
      "fullUrl": "Condition/cond-1",
      "resource": {
        "resourceType": "Condition",
        "id": "cond-1",
        "clinicalStatus": {
          "coding": [
            {
              "system": "http://terminology.hl7.org/CodeSystem/condition-clinical",
              "code": "active",
              "display": "Active"
            }
          ]
        },
        "verificationStatus": {
          "coding": [
            {
              "system": "http://terminology.hl7.org/CodeSystem/condition-verification-status",
              "code": "unconfirmed",
              "display": "Unconfirmed"
            }
          ]
        },
        "category": [
          {
            "coding": [
              {
                "system": "http://terminology.hl7.org/CodeSystem/condition-category",
                "code": "encounter-diagnosis",
                "display": "Encounter Diagnosis"
              }
            ]
          }
        ],
        "code": {
          "coding": [
            {
              "system": "http://snomed.info/sct",
              "code": "271861002",
              "display": "Stomach ache"
            }
          ]
        },
        "subject": {
          "reference": "Patient/patient-1",
          "onsetDateTime": "2025-06-19"
        }
      }
    }
  ]
}
```

```
tool-response
Observation:
('Found 66 hits for vomiting in Condition.code', [{'system': 'http://snomed.info/sct', 'code': '42240008', 'display': 'Vomiting'}, {'system': 'http://snomed.info/sct', 'code': '22586007', 'display': 'Vomiting food'}, {'system': 'http://snomed.info/sct', 'code': '71419002', 'display': 'Bilious vomiting'}, {'system': 'http://snomed.info/sct', 'code': '8765009', 'display': 'Hematemesis'}, {'system': 'http://snomed.info/sct', 'code': '1863907', 'display': 'Viral gastroenteritis'}, {'system': 'http://snomed.info/sct', 'code': '23971007', 'display': 'Acute vomiting'}, {'system': 'http://snomed.info/sct', 'code': '4026006', 'display': 'Habit vomiting'}, {'system': 'http://snomed.info/sct', 'code': '4599404', 'display': 'Erotic vomiting'}, {'system': 'http://snomed.info/sct', 'code': '6372006', 'display': 'Chronic vomiting'}, {'system': 'http://snomed.info/sct', 'code': '20647008', 'display': 'Vomiting symptom'}])
Last output from code snippet:
None
tool-call
Calling tools:
{'id': 'call_4', 'type': 'function', 'function': {'name': 'python_interpreter', 'arguments': {'print(search_for_code_or_coding(fhir_attribute_path="Condition.code", search_term="vomiting"))'}}}
Thought:
I have found the status code for the MedicationStatement ('active').
I have gathered all the necessary codes:
-->Stomach ache--> {'system': 'http://snomed.info/sct', 'code': '271861002', 'display': 'Stomach ache'}
-->Nausea--> {'system': 'http://snomed.info/sct', 'code': '422587007', 'display': 'Nausea'}
-->Vomiting--> {'system': 'http://snomed.info/sct', 'code': '42240008', 'display': 'Vomiting'}
-->Penicillin Allergy--> {'system': 'http://snomed.info/sct', 'code': '91926005', 'display': 'Allergy to penicillin'}
-->Condition Clinical Status--> {'system': 'http://terminology.hl7.org/CodeSystem/condition-clinical', 'code': 'active', 'display': 'Active'}
-->Condition Verification Status--> {'system': 'http://terminology.hl7.org/CodeSystem/condition-verification-status', 'code': 'unconfirmed', 'display': 'Unconfirmed'}
-->Condition Category (Symptom)--> {'system': 'http://terminology.hl7.org/CodeSystem/condition-category', 'code': 'encounter-diagnosis', 'display': 'Encounter Diagnosis'}
-->Condition Category (Allergy)--> {'system': 'http://terminology.hl7.org/CodeSystem/condition-category', 'code': 'problem-list-item', 'display': 'Problem List Item'}
-->MedicationStatement Status--> {'system': 'http://hl7.org/fhir/CodeSystem/medication-statement-status', 'code': 'active', 'display': 'Active'}
Now I will construct the FHIR resources and the final bundle.
1. A 'Patient' resource (anonymous, as no name is given).
2. Three 'Condition' resources for the symptoms (stomach ache, nausea, vomiting), linked to the patient. They will have an 'onsetDateTime' approximation based on 'seit 2 Tagen'.
3. One 'Condition' resource for the penicillin allergy, linked to the patient.
4. One 'MedicationStatement' for Pantoprazol, linked to the patient. It will include the dosage '1-0-1'. The medication will be represented as text. The reasonCode will point to the stomach complaint condition.
I will now write the Python code to create these resources and bundle them. I will hardcode the date for today, assuming the consultation happened
```

Intermediate Steps (Tool-Calling, Reasoning, Python Code)

Figure 4: Extended example of clinical note synthesis with Inferno including the System Prompt and a longer snippet from the tool calls and generated Python code which yields the FHIR Resource.

Betreff: Arztberichtsbrief - Patienteninformationen

Sehr geehrter Dr. Peters,

hiermit möchte ich Ihnen einen aktuellen Bericht über den Gesundheitszustand von Herrn Uwe Jaeger, geboren am 10. Februar 1975, vorlegen. Herr Jaeger wurde am 20. Juni 2023 in unserer Klinik, dem St. Ursula Krankenhaus, zur weiteren Untersuchung und Behandlung aufgenommen.

Anamnese:

Herr Jaeger suchte unsere Notaufnahme mit anhaltenden Beschwerden im Magen-Darm-Bereich auf. Er berichtete über starke Bauchschmerzen, Übelkeit, Erbrechen und Gewichtsverlust in den letzten vier Wochen. Er verneinte jegliche vorherige Operationen oder relevante Vorerkrankungen. Herr Jaeger ist Nichtraucher und konsumiert keinen Alkohol.

Klinischer Befund:

Bei der körperlichen Untersuchung zeigten sich eine allgemeine Schwäche und ein mäßig abgeschwächter Allgemeinzustand. Der Bauch war diffus druckempfindlich, ohne spürbare Vergrößerungen der Organe. Keine Zeichen einer Peritonitis waren erkennbar. Die übrige körperliche Untersuchung ergab keine auffälligen Befunde.

Diagnostische Maßnahmen:

Um die Ursache der Beschwerden zu ermitteln, wurden bei Herrn Jaeger verschiedene diagnostische Tests durchgeführt. Eine Blutuntersuchung ergab eine erhöhte Anzahl weißer Blutkörperchen und eine leichte Anämie. Der Leberfunktionstest zeigte normale Werte. Ein abdominales Ultraschall wurde durchgeführt, das keine strukturellen Abnormalitäten zeigte. Eine Endoskopie des oberen Verdauungstrakts wurde ebenfalls durchgeführt, bei der eine erosive Gastritis festgestellt wurde.

Diagnose:

Basierend auf den klinischen Symptomen, den Laborergebnissen und der Endoskopie wurde bei Herrn Jaeger die Diagnose einer erosiven Gastritis gestellt.

Therapie:

Um die Symptome zu lindern und die Schleimhaut im Magen zu heilen, wurde Herr Jaeger eine Kombinationstherapie verschrieben. Er erhält eine Protonenpumpenhemmer (PPI) für acht Wochen, um die Magensäureproduktion zu reduzieren. Zusätzlich wurde ihm ein Antazidum verschrieben, um den sofortigen Effekt einer schnellen Symptomlinderung zu erzielen. Er erhielt auch Anweisungen zur Vermeidung von auslösenden Nahrungsmitteln, wie scharfe und säurehaltige Lebensmittel.

Verlauf und Prognose:

Herr Jaeger hat die empfohlene Therapie begonnen und wurde über mögliche Nebenwirkungen und Maßnahmen zur Verbesserung seines Gesundheitszustands aufgeklärt. Wir werden ihn in regelmäßigen Abständen zu Follow-up-Terminen einladen, um den Verlauf seiner Symptome zu überwachen und gegebenenfalls weitere Untersuchungen durchzuführen.

Abschließend möchte ich Ihnen versichern, dass wir die bestmögliche Versorgung für Herrn Jaeger sicherstellen und eng mit ihm zusammenarbeiten werden, um eine schnelle Genesung zu erreichen.

Bei weiteren Fragen stehe ich Ihnen gerne zur Verfügung.

Mit freundlichen Grüßen,

Dr. Anna Karolin Vogel
Fachärztin für Innere Medizin
St. Ursula Krankenhaus

Figure 5: The full text from the first document from the synthetic corpus.