FOM University of Applied Sciences for Economics and Management

university location Cologne

Bachelor Thesis

in the study course Business Informatics

to obtain the degree of

Bachelor of Science (B.Sc.)

on the subject

Implementation of an AI-Supported Centralized Procurement Interface to Enhance Procurement Processes in a Large-Scale Enterprise

by

Dominik Fey

Advisor:	Prof. Dr. Bernd Ulmann
Matriculation Number:	601373
Submission:	October 7, 2024

Contents

List of Figures

List of Tables

List of Abbreviations

API Application Programming Interface

IaC Infrastructure-as-Code

MVP Minimum Viable Product

NLU Natural Language Understanding

NLP Natural Language Processing

List of Symbols

Sperrvermerk

Die vorliegende Abschlussarbeit mit dem Titel ‘Implementation of an AI-Supported Centralized Procurement Interface to Enhance Procurement Processes in a Large-Scale Enterprise’ enthält unternehmensinterne Daten der Firma Deutsche Telekom AG. Daher ist sie nur zur Vorlage bei der FOM sowie den Begutachtern der Arbeit bestimmt. Für die Öffentlichkeit und dritte Personen darf sie nicht zugänglich sein.

Cologne, October 7, 2024

(Ort, Datum)

(Eigenhändige Unterschrift)

1 Einleitung

Dies soll eine LaTeX -Vorlage für den persönlichen Gebrauch werden. Sie hat weder einen Anspruch auf Richtigkeit, noch auf Vollständigkeit. Die Quellen liegen auf Github zur allgemeinen Verwendung. Verbesserungen sind jederzeit willkommen.

1.1 Objective

Kleiner Reminder für mich in Bezug auf die Dinge, die wir bei der Thesis beachten sollten und LaTeX -Vorlage für die Thesis.

1.2 Structure of the Paper

Kapitel 2 enthält die Inhalte des Thesis-Days und alles, was zum inhaltlichen erstellen der Thesis relevant sein könnte. In Kapitel 3 Methodology findet ihr wichtige Anmerkungen zu LaTeX , wobei die wirklich wichtigen Dinge im Quelltext dieses Dokumentes stehen (siehe auch die Verzeichnisstruktur in Abbildung.

1.3 Technical Stack Relevance

The selection of a robust and integrated technical stack is pivotal for the successful implementation of sophisticated software solutions, particularly when developing a chatbot designed to understand and fulfill complex customer needs. The project presented in this bachelor's thesis involves the creation of a chatbot that leverages Natural Language Processing (Natural Language Processing (NLP)) to interpret customer inquiries and match them with relevant items from a supplier catalog. To achieve this, the backend relies on Python and Haystack, while the frontend utilizes Vue, and the deployment is managed through containerization and orchestration technologies like Docker and Kubernetes.

1.3.1 Backend Technologies

Python and Haystack

Python serves as the backbone of the application due to its extensive ecosystem and its ability to seamlessly integrate various libraries and frameworks that facilitate rapid prototyping and the development of complex data-driven functionalities.[1,2] Its prominence in the fields of machine learning and data science makes it an ideal choice for implementing a chatbot that requires advanced Natural Language Understanding (Natural Language Understanding (NLU)) capabilities.[3,4] Specifically, Python's compatibility with NLP frameworks like Haystack and many libraries ensures that the chatbot can parse user inputs and perform context-aware semantic searches.[5] This capability is crucial for accurately interpreting customer needs and mapping them to appropriate products or services in the supplier catalog.

The integration of Haystack within this project serves as a cornerstone for the development of an intelligent, search-driven chatbot, which is designed to address the intricate nature of customer inquiries. Haystack, a robust open-source Natural Language Processing (NLP) framework, employs a sophisticated Reader-Retriever architecture that harmonizes the capabilities of both information retrieval and deep semantic understanding. This dual approach capitalizes on advanced NLP methodologies to enhance the chatbot's performance in extracting pertinent information from extensive datasets.[6]

Notably, rather than utilizing the standard BERT model, this implementation leverages OpenAI's GPT-4 model within the Haystack framework. This allows the system to engage in nuanced contextual interpretation, thereby significantly improving the precision of semantic searches. The choice of GPT-4 is particularly advantageous in question-answering scenarios, as it allows the chatbot to comprehend the subtleties of customer queries and generate responses that are not only contextually relevant but also demonstrate a high degree of language understanding.[7]

Moreover, Haystack's modular architecture and extensible Application Programming Interfaces (Application Programming Interface (API)s) offer a high degree of flexibility, facilitating seamless integration within the chatbot's overall architecture. This ensures that the processes of searching and retrieving supplier catalog data are executed with optimal accuracy and efficiency. Consequently, Haystack's inclusion in the technical stack is not merely contributory to the current system's capabilities but also establishes a solid foundation for prospective advancements and refinements in the chatbot's functional repertoire.

PostgreSQL

1 cf. *Shrivastava*, *S.*, 2024, p. 12.
2 cf. *Christensen*, *S.* et al., 2022, pp. 240-241.
3 cf. *Lortie*, *C. J.*, 2022, p. 1.
4 cf. *Joshi*, *A.*, *Tiwari*, *H.*, 2024, p. 85.
5 cf. *Fareez*, *M. M. M.*, *Thangarajah*, *V.*, *Saabith*, *S.*, 2020, p. 21.
6 cf. *Krishnamoorthy*, *V.*, 2021, p. 236.
7 cf. *Syed*, *Z. H.* et al., 2021, pp. 943-944.

The backend system also incorporates PostgreSQL as its database solution. PostgreSQL's support for complex queries and its capability to handle structured data are essential for managing and accessing the supplier information and product details stored within the system.[8] The integration of PostgreSQL ensures that the chatbot can quickly and efficiently retrieve the necessary data, thereby reducing latency and enhancing the overall user experience.

Monitoring and Observability

For monitoring and observability, the project employs Langfuse and OpenTelemetry, which provide comprehensive tracing and metrics collection across the microservices architecture.[9] This is particularly relevant given the experimental nature of the prototype, where understanding system performance and identifying potential bottlenecks are crucial for iterative development and refinement. By utilizing these tools, the project gains valuable insights into the behavior of the chatbot, allowing for continuous improvement and optimization.

FastAPI and uv

FastAPI serves as the web framework for the backend, offering a high-performance environment that supports asynchronous programming.[10] This choice is particularly relevant for the chatbot, as it enables handling multiple concurrent requests with minimal overhead, ensuring that the application remains responsive even under heavy loads.

The project also leverages the `uv` package for dependency management and deployment configuration. `uv` simplifies the process of configuring Python dependencies and allows for a smoother deployment process by ensuring compatibility and consistency between various package versions.[11]

1.3.2 Frontend Technologie

On the frontend, Vue.js and Vuetify are utilized to create an intuitive and responsive user interface. The decision to use Vue.js stems from its reactive nature and modular architecture, which align with the need for a maintainable and easily extensible codebase.[12,13] It is

[8] cf. *Abbasi*, *M.* et al., 2024, pp. 23-24.
[9] cf. *Thakur*, *A.*, *Chandak*, *M. B.*, 2022, p. 15014.
[10] cf. *Chen*, *J.*, 2023, p. 9.
[11] cf. *Cano Rodríguez*, *J. L.*, 2024, URL last accessed on 2024-10-05.
[12] cf. *Kaluža*, *M.*, *Vukelic*, *B.*, 2018, p. 268.
[13] cf. *Li*, *N.*, *Zhang*, *B.*, 2021, pp. 1-2.

effective for developing a chatbot interface that needs to present complex data in an accessible manner, while also allowing for dynamic updates based on user interactions.[14]

1.3.3 Deployment and Infrastructure Management

Deployment is managed through a combination of Docker, Kubernetes, and Terraform. Docker's role in containerizing the application ensures that the entire software stack can be encapsulated and deployed consistently across various environments.[15] This is essential for a project like this, where different iterations of the prototype may need to be tested in different setups. Kubernetes, in turn, provides the orchestration needed to manage these containers, allowing for automated scaling and high availability.[16] The use of Terraform as an infrastructure-as-code (Infrastructure-as-Code (IaC)) tool ensures that cloud resources can be provisioned and managed efficiently, providing a stable and reproducible deployment environment.[17]

In summary, each component of the technical stack has been carefully selected to meet the unique requirements of the chatbot project. The combination of Python and Haystack provides robust NLP capabilities for understanding and processing user inputs, while FastAPI support real-time interactions. PostgreSQL ensures efficient data management, and Langfuse and OpenTelemetry offer the necessary monitoring tools. On the frontend, Vue.js and Vuetify deliver a responsive and interactive user interface, and the deployment stack, comprised of Docker, Kubernetes, and Terraform, guarantees scalability and reliability. This cohesive selection of technologies forms a solid foundation for the development of a chatbot that not only meets the functional requirements but also adheres to best practices in software engineering.

[14] cf. *Mokoginta*, *D.*, *Putri*, *D. E.*, *Wattimena*, *F. Y.*, 2024, p. 493.
[15] cf. *Openja*, *M.* et al., 2022, p. 191.
[16] cf. *Carrión*, *C.*, 2022, pp. 2, 7–8.
[17] cf. *N.*, *N.*, *Pub*, *I.*, 2023, p. 24.

2 Fundamentals

3 Methodology

3.1 Prototyping

3.1.1 Introduction to the Prototyping Methodology According to Floyd

The prototyping approach developed by Christiane Floyd represents a structured methodology used primarily in software development to improve communication between developers and users, reduce misunderstandings, and ultimately enhance the quality of the final product.[18] This methodology provides an alternative to the traditional linear, phase-oriented development process by introducing a dynamic element of iteration and feedback.[19] As a result, it facilitates a more interactive and user-centered development process.[20]

Floyd's prototyping process is structured as a cyclical sequence of four distinct steps: functional selection, construction, evaluation, and further use.
The first step, Functional Selection, involves identifying the specific functions that the prototype should demonstrate. The selected functionalities are derived from the relevant work tasks to ensure meaningful demonstrations, while acknowledging that the prototype does not need to represent the final product comprehensively. This allows for a degree of flexibility in the selection and prioritization of features to be included in the prototype.[21]

The second step, Construction, entails building the prototype using techniques and tools that enable rapid development and easy adjustments. At this stage, the focus is not on developing a fully functional system, but rather on demonstrating and assessing specific aspects of the final product. This approach enables the prototype to act as a tool for exploring different solutions and gathering feedback.[22]

Evaluation, the third step in the process, serves as the cornerstone of the prototyping methodology. In this step, feedback from all relevant stakeholders—including potential users—is collected and analyzed to refine and guide subsequent development stages. This iterative process ensures that the prototype evolves in alignment with user expectations and needs.[23]

[18] cf. *Floyd*, *C.*, 1984, p.2-3.
[19] cf. *Floyd*, *C.*, 1984, p.3.
[20] cf. *Floyd*, *C.*, 1984, p.3-4.
[21] cf. *Floyd*, *C.*, 1984, p.4.
[22] cf. *Floyd*, *C.*, 1984, p.4.
[23] cf. *Floyd*, *C.*, 1984, p.4-5.

The final step, Further Use, determines the prototype's role after the evaluation phase. Depending on its effectiveness and the degree to which it meets requirements, the prototype can be discarded, modified for continued use, or serve as a foundation for the final product. This flexibility is critical in accommodating evolving requirements and objectives, making the prototyping approach particularly valuable in contexts where specifications are expected to change frequently.[24]

Floyd's methodology also categorizes prototyping into three primary approaches, each based on the goals and context of development: exploratory prototyping, experimental prototyping, and evolutionary prototyping.[25]
Exploratory Prototyping is primarily used to clarify requirements and foster creative cooperation between developers and users during the early stages of development. It is particularly useful when there is a lack of clarity on the system's final objectives, as it allows for broad experimentation and refinement of ideas before committing to a specific solution.[26]

In contrast, Experimental Prototyping focuses on testing proposed solutions to validate specific hypotheses, such as user interface design, system performance, or algorithm feasibility. This approach might involve techniques such as full functional simulation or human interface simulation to verify that the proposed design meets the intended objectives.[27]

Finally, Evolutionary Prototyping treats the prototype as a system that continuously evolves to adapt to changing requirements over time. Each version of the prototype serves as a basis for the next iteration, incorporating new insights and user feedback. This approach is especially beneficial in scenarios where requirements are expected to change frequently, rendering a static set of requirements impractical.[28]

To support these prototyping processes, various techniques and tools can be utilized.[29]
Modular Design, for instance, encourages the use of small, independent modules that can be replaced or refined as needed, thereby facilitating iterative development and easing integration into the final product.[30]
Additionally, Dialogue Design plays a crucial role in ensuring that the user interface is adaptable and transparent, which enables effective evaluation and modification of the user

[24] cf. *Floyd*, *C.*, 1984, p.5.
[25] cf. *Floyd*, *C.*, 1984, p.6.
[26] cf. *Floyd*, *C.*, 1984, p.6-7.
[27] cf. *Floyd*, *C.*, 1984, p.8-10.
[28] cf. *Floyd*, *C.*, 1984, p.10-12.
[29] cf. *Floyd*, *C.*, 1984, p.12.
[30] cf. *Floyd*, *C.*, 1984, p.12.

experience.[31]
Furthermore, Simulation techniques allow for simulating aspects of the final system that are not yet fully implemented, enabling the assessment of system performance and user interaction without the need for a complete implementation.[32]

3.1.2 Selection of Prototyping Approach

Given the context and objectives of this project, the decision was made to adopt the experimental prototyping approach. This choice is rooted in the fact that the project requirements have already been well-defined through a comprehensive requirements engineering process, thus eliminating the need for exploratory prototyping. The clear specification of functionalities and user expectations ensures that the focus can shift from understanding requirements to validating and testing specific design choices.

Furthermore, the experimental approach is particularly well-suited for scenarios where a prototype is intended to serve as a preliminary proof of concept rather than a foundation for incremental development. This aligns perfectly with the anticipated lifecycle of the prototype in this project, which is expected to be discarded once the Minimum Viable Product (Minimum Viable Product (MVP)) phase begins. As the project moves towards the MVP stage, a fresh start will be made, incorporating only validated concepts and findings from the experimental prototype. Therefore, evolutionary prototyping is not applicable, as it is primarily designed for projects that involve iterative refinement and continuous evolution of the same prototype.

The experimental prototyping approach allows for focused experimentation with various design elements, interface interactions, and technical implementations, all within a controlled environment that does not necessitate long-term integration into the final product.

3.1.3 Application of Experimental Prototyping in the Project

The experimental prototyping approach will be implemented in this project with a focus on validating user interface designs, interactions, and core functionalities against predefined requirements. The development of the prototype will leverage a set of carefully selected techniques that facilitate rapid iteration and feedback.

A key technique employed is simulation, which is used to mimic certain system behaviors without integrating the prototype into live production environments. This decision is

[31] cf. *Floyd, C.*, 1984, p.12.
[32] cf. *Floyd, C.*, 1984, p.13.

motivated by the limited scope of the prototype and the intention to avoid disruptions to existing systems. By relying on test data instead of actual production data, the prototype can simulate real-world scenarios and provide valuable insights into its performance and user experience.

Moreover, the prototype will not attempt to implement every function in its final depth and breadth. Instead, certain features will be simulated to convey the look and feel of the system, providing a realistic representation of how the final product would function. This is where Modular Design plays a crucial role. By leveraging modular design principles , the prototype can separate the user interface from the underlying logic and backend functionalities. This allows specific modules to be developed exclusively for the UI, simulating the presence of certain features without the need for fully developed backend logic. For instance, UI elements such as buttons, forms, and interactive components can be displayed and interacted with as if they were functional, even though the backend processing is either simulated or entirely absent. This approach enables the developer to receive early feedback on key aspects of the design and functionality without committing extensive resources to full-scale implementation.

By focusing on these aspects, the prototype will serve as a learning tool, guiding the refinement of requirements and design choices before moving into the more resource-intensive MVP phase.

4 Fazit

Wünsche Euch allen viel Erfolg für das 7. Semester und bei der Erstellung der Thesis. Über Anregungen und Verbesserung an dieser Vorlage würde ich mich sehr freuen.

Appendix

Appendix 1: Beispielanhang

Dieser Abschnitt dient nur dazu zu demonstrieren, wie ein Anhang aufgebaut seien kann.

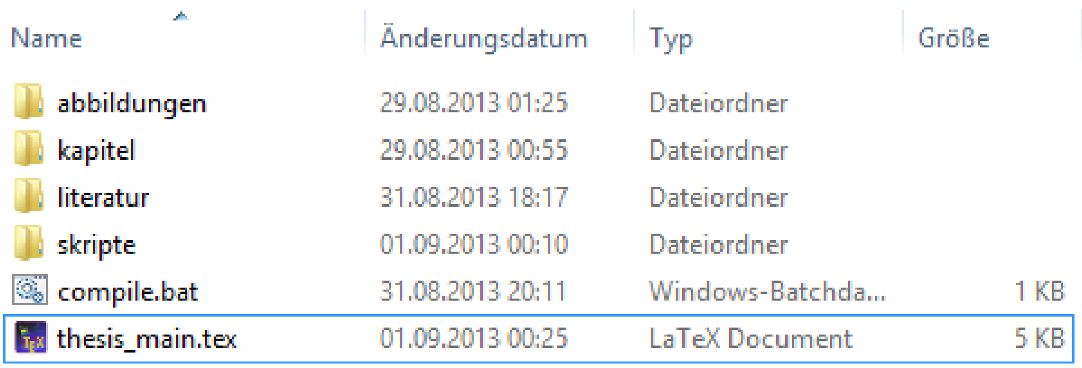
Appendix 1.1: Weitere Gliederungsebene

Auch eine zweite Gliederungsebene ist möglich.

Appendix 2: Bilder

Auch mit Bildern. Diese tauchen nicht im Abbildungsverzeichnis auf.

Figure 1: Beispielbild



Bibliography

Abbasi, *Maryam*, *Bernardo*, *Marco V.*, *Váz*, *Paulo*, *Silva*, *José*, *Martins*, *Pedro* (2024): Adaptive and Scalable Database Management with Machine Learning Integration: A PostgreSQL Case Study, in: Information, 15 (2024), Nr. 9, p. 574

Carrión, *Carmen* (2022): Kubernetes Scheduling: Taxonomy, Ongoing Issues and Challenges, in: ACM Comput. Surv. 55 (2022), Nr. 7, 138:1–138:37

Chen, *Junqiao* (2023): Model Algorithm Research Based on Python Fast API, in: Frontiers in Science and Engineering, 3 (2023), Nr. 9, pp. 7–10

Christensen, *Scott*, *Brown*, *Marvin*, *Haehnel*, *Robert*, *Church*, *Joshua*, *Catlett*, *Amanda*, *Schofield*, *Dallon*, *Brannon*, *Quyen*, *Smith*, *Stacy* (2022): A Python Pipeline for Rapid Application Development (RAD), in: s.l., 2022-01-01, pp. 240–243

Fareez, *MMM*, *Thangarajah*, *Vinothraj*, *Saabith*, *Sayeth* (2020): POPULAR PYTHON LIBRARIES AND THEIR APPLICATION DOMAINS, in (2020)

Floyd, *Christiane* (1984): A Systematic Look at Prototyping, in: *Budde*, *Reinhard*, *Kuhlenkamp*, *Karin*, *Mathiassen*, *Lars*, *Züllighoven*, *Heinz* (eds.), Approaches to Prototyping, Berlin, Heidelberg: Springer, 1984, pp. 1–18

Joshi, *Ankush*, *Tiwari*, *Haripriya* (2024): An Overview of Python Libraries for Data Science | Journal of Engineering Technology and Applied Physics, in ()

Kaluža, *Marin*, *Vukelic*, *Bernard* (2018): Comparison of Front-End Frameworks for Web Applications Development, in: Zbornik Veleučilišta u Rijeci, 6 (2018), pp. 261–282

Krishnamoorthy, *Venkatesh* (2021): Evolution of Reading Comprehension and Question Answering Systems, in: Procedia Computer Science, Big Data, IoT, and AI for a Smarter Future, 185 (2021), pp. 231–238

Li, *Nian*, *Zhang*, *Bo* (2021): The Research on Single Page Application Front-end Development Based on Vue, in: Journal of Physics: Conference Series, 1883 (2021), Nr. 1, p. 012030

Lortie, *Christopher J.* (2022): Python and R for the Modern Data Scientist, in: Journal of Statistical Software, 103 (2022), pp. 1–4

Mokoginta, *Deyidi*, *Putri*, *Desfita Eka*, *Wattimena*, *Fegie Yoanti* (2024): Developing Modern JavaScript Frameworks for Building Interactive Single-Page Applications, in: International Journal Software Engineering and Computer Science (IJSECS), 4 (2024), Nr. 2, pp. 484–496

N., *Nivedhaa*, *Pub*, *Iaeme* (2023): Evaluating Devops Tools and Technologies for Effective Cloud Management, in: 1 (2023), pp. 20–32

Openja, *Moses*, *Majidi*, *Forough*, *Khomh*, *Foutse*, *Chembakottu*, *Bhagya*, *Li*, *Heng* (2022): Studying the Practices of Deploying Machine Learning Projects on Docker, in: Proceedings of the 26th International Conference on Evaluation and Assessment in Software Engineering, EASE '22, New York, NY, USA: Association for Computing Machinery, 2022-06-13, pp. 190–200

Shrivastava, *Samiksha* (2024): Design and Implementation of Chatbot Using Python, in: IJFMR - International Journal For Multidisciplinary Research, 5 (), Nr. 6

Syed, *Zeeshan Haque*, *Trabelsi*, *Asma*, *Helbert*, *Emmanuel*, *Bailleau*, *Vincent*, *Muths*, *Christian* (2021): Question Answering Chatbot for Troubleshooting Queries Based on Transfer Learning, in: Procedia Computer Science, 192 (2021), pp. 941–950

Thakur, *Aadi*, *Chandak*, *M. B.* (2022): A Review on Opentelemetry and HTTP Implementation, in: International journal of health sciences, 6 (2022), Nr. S2, pp. 15013–15023

Internet sources

Cano Rodríguez, *Juan Luis* (2024): Python Packaging Is Great Now: ‘uv‘ Is All You Need, <https://dev.to/astrojuanlu/python-packaging-is-great-now-uv-is-all-you-need-4i2d> (2024-08-10) [Access: 2024-10-05]

Declaration in lieu of oath

I hereby declare that I produced the submitted paper with no assistance from any other party and without the use of any unauthorized aids and, in particular, that I have marked as quotations all passages which are reproduced verbatim or near-verbatim from publications. Also, I declare that the submitted print version of this thesis is identical with its digital version. Further, I declare that this thesis has never been submitted before to any examination board in either its present form or in any other similar version. I herewith disagree that this thesis may be published. I herewith consent that this thesis may be uploaded to the server of external contractors for the purpose of submitting it to the contractors' plagiarism detection systems. Uploading this thesis for the purpose of submitting it to plagiarism detection systems is not a form of publication.

Cologne, 7.10.2024	
(Location, Date)	(handwritten signature)