Requirements Engineering

Stakeholders

In this section, the stakeholders will be presented together with a short description of their requirements.

ST01 - Students/Users

Students that are potential users of the application and that are interested in learning with lecture slides and understanding the content.

Requirements
Upload of lecture slides
Interactive chatbot that can be used to ask questions regarding uploaded lecture slides
Responses with "as small as possible delay"
Ingestion of industry knowledge for comprehensive responses
User-friendly web interface
Correctness and explainability of chat responses

ST02 - Chair of Software Systems

The Chair of Software Systems, that commissioned the development of the system and will evaluate the results.

Requirement
Selection of existing test lecture slides
Usage of API wrapper for GPT-3.5 Turbo
Economic efficiency - fixed budget of the GPT-3.5 Turbo API
Submission of prototypes, artifacts and the final product at fixed dates
System must run on Kubernetes for usage of chair infrastructure

ST03 - Creators of lecture slides / Lecturers

Lecturers that create slides and scripts for their lectures, which will potentially be uploaded to the chatbot.

Requirement
Preservation of the copyright of their uploaded lecture slides

ST04 - Development Team

Requirements
Maintainabiliy of system (e.g. modularized, exchangeable, parameterizable)
Collaboration in development process / usage of code repository
Vector store should be used for efficient selection of context

Scenarios

In this section, important scenarios for the usage of the system will be presented to understand how users will interact with the chatbot.

SC01 - Asking a question for an existing slide set

1. User calls web application (authentication may be necessary)
2. User is presented with a list of uploaded slide sets
3. User selects a set of slides
4. User is presented with a view combining the corresponding lecture slides and a chat window
5. User enters a question
6. User receives a matching answer to his question with content from the lecture and industry knowledge

SC02 - Uploading a new set of slides

1. User calls web application (authentication may be necessary)
2. User calls the function "upload new slide set"
3. User is presented with a view where he can upload the slide set
4. When the slide set has been uploaded and the integration of the data is in progress, the user should be signaled about the progress (staying on of the page is not necessary)
5. When the slide set is finished uploading and the user is still on the upload view, the user should be offered to ask questions
6. The user can now select the lecture slides equivalent to SC01 at any time

SC03 - Deleting a set of slides

1. User calls web application (authentication may be necessary)
2. User is presented with a list of uploaded slide sets
3. User selects slide set to be deleted
4. The slide set is no longer available to ask questions and the vectors introduced are removed

System Specification

By combining insights into the usage scenarios and the superfical description of stakeholder requirements, a detailled system specification will now be presented for functional + non-functional requirements and constraints.

Upload of lecture slides

Stakeholder	Specification	Type

Stakeholder	Specification	Type
Students	The system should offer a function for users to upload new lecture slides (PDF format), which then will be integrated for interaction with the chatbot. Meaning that for a new slide, the content must be integrated as needed to be retrieved as context for occuring user questions.	Functional

Interactive chatbot that can be used to ask questions regarding uploaded lecture slides

Stakeholder	Specification	Type
Students	The main functionality of the system consists of a chat window, where the user can ask questions regarding uploaded lecture slides. The answers generated by the LLM based on the extracted context are also shown inside the chat.	Functional

Responses with "as small as possible delay"

Stakeholder	Specification	Type
Students	To accomplish efficient usage of the chatbot, it must answer questions with as small as possible delay. This is not referring to the time it takes to guide new slides through the integration steps, where time is not that critical. But in an active dialog, it must be ensured that a maximum time of 10s is not exceeded. This number can be extracted from this paper https://ieeexplore.ieee.org/abstract/document/6263888 (10s is a limit to keep the user focussed on a conversation).	Non-Functional

Ingestion of industry knowledge for comprehensive responses

Stakeholder	Specification	Type
Students	Besides the integraton of the lecture slide content, additional content from technology blogs (e.g. Medium) should be scraped and integrated. Meaning, that for every new slide uploaded, a procedure must be executed, that is integrating text from slides and the additional content in the needed manner. As an outcome, the response should contain further details, that the chatbot would not be able to provide based only on the slides. This makes the answer more practical and also comprehensive.	Non-Functional

User-friendly web interface

Stakeholder	Specification	Type

Stakeholder	Specification	Type
Students	The application should be easily accessible and therefore a web-based user interface should be provided. It should be optimized both for large screens (desktop) and small screens (smartphones). Its simple design should not distract from the questions and answers themselves, as they are the core of the system. Components should be designed industry-typical (e.g. Material Design, Bootstrap), so that users get used to the application quickly.	Non-Functional

Correctness and explainability of chat responses

Stakeholder	Specification	Type
Students	The users, primarily being students that want to study for their lectures, might also get graded based on concepts they learned using the chatbot. Additionally, their performance might not only depend on the actual answers, but also on the creation and throught process of an answer. Therefore the responses generated by the chatbot should be both correct and fully explainable. It should not answer questions when the provided context is not sufficient. It should provide explanations for answers and conclusions, so that results can be reproduced.	Non-Functional

Selection of existing test lecture slides

Stakeholder	Specification	Type
Chair of Software Systems	To validate the target system and functionality of the chatbot, a basic set of provided slides needs to be uploaded and available for question answering initially.	Functional

Usage of API wrapper for GPT-3.5

Stakeholder	Specification	Type
Chair of Software Systems	To control LLM budget used by the project, an API wrapper was created to access the LLM Service. It is necessary to use this provided API wrapper for any requests to the LLM service, to use the provided budget. It is possible that there are small differences in comparison to the public API service.	Functional

Economic efficiency - fixed budget of the GPT-3.5 Turbo API

Stakeholder	Specification	Type

Stakeholder	Specification	Type
Chair of Software Systems	The chatbot should work efficiently with LLM resources, meaning that the number of tokens sent to the LLM service should be as small as possible while still providing enough context. A budget of 5 Euros cannot be exceeded in the course of the development and system validation. Therefore it might be needed to operate certain system and AI components (such as topic modelling or vector embedding) locally and only use the LLM service for the generation of the final response.	Non-Functional

Submission of prototypes, artifacts and the final product at fixed dates

Stakeholder	Specification	Type
Chair of Software Systems	To assess and grade the chatbot system and development process, it is needed to deliver certain artifacts, prototypes and the final system at fixed dates. Prototype together with C4 diagram, requirements document and business plan must be delivered until 23.06.2023. The final delivery (software system running in the provisioned Kubernetes cluster) must be done until 25.08.2023.	Constraint

System must run on Kubernetes for usage of chair infrastructure

Stakeholder	Specification	Type
Chair of Software Systems	The Chair of Software Systems is only able to provision compute via their own Kubernetes cluster. Therefore it is needed, that the application is containerized and a concept must be developed on how the system can be deployed to the Kubernetes cluster. The final system must be running completely inside the cluster.	Constraint

Preservation of the copyright of their uploaded lecture slides

Stakeholder	Specification	Type
Lecturers	Lecturers and creators of lecture slides have copyright on their work, which needs to be respected by the students which upload the slides and also by the system itself. Uploaders should guarantee that they have permission to use the respective documents and the system should offer functionality for users to report answers, which they think contains copyrighted information.	Non-Functional

Maintainabiliy of system (e.g. modularized, exchangeable, parameterizable)

Stakeholder	Specification	Type

Stakeholder	Specification	Type
Development Team	The system should fulfill several requirements regarding maintainability to allow for easy enhancements and also reaction to changes. The different system components, such as creating embeddings, uploading embeddings to vector store and searching vector store should be implemented in a modular manner. Where possible and useful, parameters should be implemented to control behaviour of the system. Assumptions for parameters should not be hard-coded but rather shown as possible parameter configurations. This also applies to used models.	Non-Functional

Collaboration in development process / usage of code repository

Stakeholder	Specification	Type
Development Team	To support collaborative development and access to codebase a version control system should be used by every project participant. In particular, the Uni Leipzig Computer Science Department GitLab should be used, as every participant is already registered and a project is also already created there. The branch-merge approach should be used when working with the code repository. The main branch is called "main".	Constraint

Vector store should be used for efficient selection of context

Stakeholder	Specification	Type
Development Team	The recommendation of the Chair of Software Systems regarding database should be followed by using a vector database for information ingestion (slides, posts) and also information retrieval. The context can only be provided efficiently to the LLM (see response time) if it already exists in a semantically ordered manner, that can be queried rapdily. Vector stores offer this through leveraging vector embedding models and efficient vector storage. In particular, the Pinecone vector database service should be used for the implementation of this system.	Functional

AI-specific System Specification

In this section the AI-specific system specifications are presented.

Area	Specification
Context	The AI components are deployed inside a university owned Kubernetes Cluster, except the LLM, which is deployed and operated by OpenAI. The system relies on internet connectivity for user access and reachability of surrounding services, such as a LLM service, vector database service and web blogs for scraping of additional content. It depends on user interaction - primarily through supplying slides, asking questions and providing feedback.

Area	Specification
Data	The regular input consists of questions asked by the user, which may include questions targetted at content and concepts from lecture slides. The contextual data sources the system uses are uploaded lecture slides and web content (technology blog posts, industry best practices or how-to guides).
Model	Multiple AI components are involved for different purposes: topic modelling of slides, creation of embeddings from slides/additional content and finally answering of questions with context through LLM. The model parameters, specific requirements and constraints of the LLM are bound to the respective language model service, such as OpenAI. For the topic modelling, which is a statistical approach to get common keywords from text, the parameters are depending on the used software component (e.g. sklearn TfidfVectorizer with a ngram_range Parameter). Embeddings are created using a local model instance of a sentence transformer (e.g. all-MiniLM-L6-v2 from Huggingface) that has defined input sequence lengths and output dimensions.
Metrics	Metrics and Key Performance Indicators for measuring the success and accuracy of the deployed AI component can include: overall response time in the chat, upload and processing time of new slides, user satisfacton through feedback.
Human Factor	Users will interact with the chatbot plugin through an interactive web-based user interface. They are able to upload new slides (indirect interaction with AI components) and ask questions regarding uploaded slides (active interaction with AI components). To make the usage and question answering easier, uploaded slides will also be viewable inside the web application. The responses by the chatbot will be in natural language and highly dependant on the exact question and the context that could be provided to the LLM. Inside the chat, users are able to interact with the AI component by providing feedback (like/dislike or helpful/not helpful) and through error reporting.
Ethics	The language model should be designed trained to ensure fairness and avoid biases that may lead to discriminatory or unjust outcomes. It should not favor or disadvantage users based on factors such as gender, age, or other protected characteristics. Through the use of existing models and services their capabilities in recognizing unfair or harmful behaviour are inherited. An additional layer of protection should be created by the combination of context and user questions, which puts the focus of the model on the context and reduces the likelyhood of inappropriate answers. When inappropriate slides are uploaded and questions are asked with the given context, the language models should be capable to reject the answers.
Non-Functional/Quality	Reliability: The chatbot should demonstrate high reliability, ensuring consistent and accurate responses to user queries. Interpretability: The chatbot should provide explanations for its generated responses, especially when providing recommendations, suggestions, or code-related insights.

Area	Specification
Hardware	The LLM and vector database can be consumed directly from the web. The web application, together with the sentence transformer and topic modelling should be hosted inside university-owned compute (Kubernetes). The hosting on premises allows the usage of the chatbot with low resource consumption (browser) for the end user.