unibz

Freie Universität Bozen
Libera Università di Bolzano
Università Liedia de Bulsan

Fakultät für Informatik
Facoltà di Scienze e Tecnologie informatiche
Faculty of Computer Science

Master in Software Engineering for Information Systems

Research Master Thesis

Applying Prompt Engineering to turn LLM into a Digital Assistant for Startups

Candidate: Seyedmoein Mohsenimofidi

Supervisor: Prof. Xiaofeng Wang

March 2024

Abstract

Motivation: Many sectors, such as startups, face with challenges due to limited resources and a lack of expertise in pivotal domains. Relying on modern digital assistance such as generative AI tools and especially Large Language Models(LLMs) can be a reliable solution to overcome these challenges. Furthermore, by using such tools like chatbots, startups have more opportunities to develop their novel ideas.

Problem statement: In using LLMs, prompts are crucial, and prompt engineering and its techniques can be used to reinforce prompts for more accurate and efficient responses. In order to maximize the potential of LLMs, prompt engineering skills must be mastered. These skills require dedicated learning efforts. The challenge is how startups can benefit from LLMs without investing substantial time and resources.

Approach: Design Science Research (DSR) will be applied in this study to address the aforementioned problem by identifying the intent of the user question and refining the proposed prompt templates based on the appropriate prompt patterns for each intent. These components are the main pillars of the prompt book which is the foundation for creating a unique chatbot for startups that leverages prompt engineering techniques to generate responses for the users.

Results: Based on an extensive literature review, iterations of design and implementation, and empirical experiments with startup teams, this research proposes a practical approach to identifying the user's question intent as well as its corresponding prompt patterns so that prompt engineering techniques can be used effectively. As a result by introducing the comprehensive prompt book, we successfully implemented the desired chatbot which we call Digital Assistant for Startups (DAS).

Conclusions: This master's thesis, by transforming ChatGPT into a specialized chatbot for startups capable of delivering comprehensive, precise, dependable, and relevant responses to user inquiries, empowers startups to address their challenges with essential resources and support. It facilitates brainstorming sessions and offers guidance and information-seeking assistance to startups.

acknowledgement

I would like to express my sincere gratitude to my advisor, Professor Xiofeng Wang, for her invaluable guidance, unwavering encouragement, and understanding throughout my Master's studies. Her exceptional support has been instrumental in shaping my academic journey.

I would like to sincerely thank Akshy Sripad Raghavendra and Aida Zahid, for their invaluable contribution to this research work. Additionally, My heartfelt thanks go to the participants who generously dedicated their time to shaping the findings of this study.

Additionally, I would like to extend my appreciation for the inspiring teaching and critical feedback provided by the faculty members at the Free University of Bozen-Bolzano.

I would like to extend my heartfelt appreciation to my family for their unwavering love, support, continuous encouragement, and profound understanding throughout every stage of my academic journey.

Last but not least, I would like to express my deep appreciation to my girlfriend for her endless patience, understanding, and encouragement. Her support and belief in me have been a constant source of motivation for doing the research through my master's studies.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Large Language Models (LLMs) are extensive pre-trained language models that have undergone training on billions of tokens. This training enables these models to understand the patterns and structure of natural language, allowing them to generate contextually relevant and coherent responses to user prompts. LLMs can be effectively utilized in a wide array of natural language processing (NLP) tasks.[Lun+23; Cla+23]

Users can enhance and optimize the results of LLM by integrating prompts, which are textual instructions and examples of the desired interactions [SYG23]. This optimization process can lead to various challenges, including increasing ambiguity, reinforcing bias, over-fitting ethical concerns, unexpected consequences, and unreasonable reliance on the model's constraints.[Gir23]

In order to overcome the aforementioned challenges, utilizing Prompt engineering can be a reliable solution. Prompt engineering is the systematic design and implementation of a prompt that enables the customization of the desired output from an AI language model, such as ChatGPT. Prompt engineering refers to the practice of gaining the skills to interact with AI in a way that achieves the intended outcome by instructing it to perform the necessary tasks.[Kor+23].

Knowing the theory and tactics behind the prompts engineering in the field of applications is essential for efficiently utilizing large language models (LLMs) and generating the best potential answers from them. Research in this area has shown that It is more effective to train a model to create relevant prompts using the prompt engineering framework rather than telling it what to do.[Whi+23; Eki23; Lin+23], also Several common threads emerge from a review of the research on AI prompt engineering that shed light on its potential uses and consequences in many fields like entrepreneurship, healthcare, and art[SS23; OLS23; PM23].

1.1 Motivation

Numerous industries, such as healthcare, education, and various others, encounter difficulties and setbacks throughout their lifespan [BH07; DG02; CJH08]. There are many factors that contribute to these challenges and difficulties, such as technological limitations, lack of access to expert knowledge, and evolving market demands. The reliance on cutting-edge solutions, such as mentorship programs and digital tools, has increased as a result of these challenges [All+04; HB23]. Using these tools not only offers instructions and support but also provides an opportunity to continue learning and adapting.

Concerning the significant rise in the number of startups and their high failure rate, only a restricted proportion ultimately acquires profitability, and the majority necessitate several years of careful planning and trial before achieving it[Ric]. The suboptimal technological choices and methods employed by these organizations contribute significantly to the elevated failure rate of technology startups[Mis22].In the context of early-stage startups, the main challenges include the unpredictability of emerging markets, lacking expertise in pivotal domains, limited resources, and poor distribution, as well as various other concerns. In addition, many companies have a restricted pool of individuals within the organization whom entrepreneurs can turn to for technical support[CK15; Gia+15].

To overcome these challenges and reduce the factors that lead to startup failure, many forms of assistance have been offered to entrepreneurs, such as mentorship, incubation, and digital tools[Gia+15]. Generative artificial intelligence (AI) is expanding, which provides new opportunities for startups to gain support and develop their innovative ideas. However, considering both the improvements in LLMs and the use of prompt engineering techniques that enhance the precision and efficiency of responses, additional investigation is necessary to examine their potential uses in supporting startups.

As the aim of this thesis, A unique digital assistant chatbot as a reliable assistant is proposed to help startups take advantage of prompt engineering during their challenging journey. We leveraged prompt engineering to provide more accurate and efficient responses by optimizing the performance of ChatGPT as the large language model(LLM) for the different needs of startups.

1.2 Objective

This thesis aims to turn ChatGPT as a Large Language Model (LLM) into a digital assistant specifically for startups to maximize the use of the effectiveness and potential advantages of prompt engineering techniques when interacting with Large Language Models (LLMs).In this study, we aim to make the generated prompts of the LLM more accurate and tailored by utilizing the appropriate prompt patterns and evaluate the effectiveness of this method in enhancing their ability to provide valuable insight and recommendations for optimizing their assistance in the startup ecosystem, which facilitates brainstorming, problem-solving, decision making, and knowledge acquisition processes for startups and entrepreneurs.

The ultimate goal of this research is to convey valuable knowledge about prompt engineering methods and patterns employed in the interaction of Large Language Models (LLMs). The main objective of the insights is to improve the proposed version of the prompt book in addition to providing a practical approach that can effectively be used to implement a comprehensive conversational agent for startups.

this master's thesis research will address the following research question to discover the ultimate solution for designing digital assistants for startups:

- **RQ**, How can prompt engineering be effectively implemented to enhance the utilization of large language models (LLMs) in supporting startup teams?

1.3 Approach

In this work, we thoroughly investigate the practical approach for turning the ChatGPT as a Large Language model (LLM) into a digital assistant particularly for the startup ecosystem, a chatbot, by utilizing prompt engineering techniques to increase the effectiveness and accuracy of the ChatGPT responses for startup related questions. A review of the relevant literature will be conducted to establish a theoretical foundation and identify existing prompt engineering approaches and patterns. Through comprehensive research, we attempt to classify the users' startup-related queries, identify their intent, and develop the relevant prompt templates, which are essential for utilizing the prompt engineering techniques that improve response efficiency, coherency, and accuracy by restructuring and empowering the user queries. Following the results of the desired investigation, we provide the technical configuration and implementation approach for implementing the exclusive chatbot for startups. By conducting the creative experiment, we evaluate the proposed chatbot functionality and assess its responses to research requirements.

1.4 Structure of the thesis

This master thesis will consist of six chapters, beginning with Chapter 1, which discusses the overview of the thesis, the motivation and objectives behind the research, as well as the approach and goal of the thesis. The background and related work section in Chapter 2 offers a summary of the relevant literature essential for accomplishing the research goals, as well as a review of the related endeavors undertaken so far. Research design Chapter 3 provides an overview of the research and relevant investigation necessary to design the artifact, implement it, and evaluate its process which is crucial for achieving the research goals. Afterward, in chapter 4, the research results chapter, by presenting the design of the prompt book and its corresponding belonging, its implementation as a chatbot, and the used evaluation process, we provide comprehensive results of the exploration process as well as the conclusions derived from the design phase. In Chapter 5, the discussion provides an interpretation and analysis of the results, study contribution, and its potential limitations. Lastly, chapter 6 summarizes the main findings, outlines research objectives, and expresses recommendations for future research.

Chapter 2

Background and related work

2.1 Large Language Models (LLMs)

Language modeling (LM) is an essential component of machine language intelligence. It forecasts the probability of future word sequences using LM models. LM development involves four stages (Statistical Language Models (SLMs), Neural Language Models (NLMs), and Pretrained Language Models (PLMs)) followed by the creation of large-scale pre-training language models called Large Language Models (LLMs), which are capable of performing a wide range of NLP tasks[Had+23; Sha22]. LLMs are trained on a large number of tokens to utilize the extensive information contained in their training data for different tasks and this allows such models to comprehend the patterns and structure of natural language to produce the appropriate and logical contextual responses to the prompts posed by users[Cla+23].

the performance of LLMs improves as both the size of the model and the training data rise, following intricate scaling principles. However, LLMs do not always facilitate seamless interaction and collaboration. Furthermore, real-life activities can sometimes be sophisticated, occasionally posing difficulties for existing LLMs to address using a solitary model[Liu+23].

In recent years, LLMs (Large Language Models) have been more popular and widely used in many different contexts. Translation, content creation, conversational AI, etc, are all within the capabilities of these LLMs because of their ability to process and produce language in a human-like manner. ChatGPT is an LLM model that has been extensively trained on large datasets. It is capable of producing coherent responses on various subjects regarding the user's requests[Liu+].

2.2 Prompt Engineering

A prompt is a query or input submitted to an LLM such as ChatGPT to guide its output generation. A prompt can be a combination of many forms such as a question, a statement, or a keyword, and such forms can be used to customize the output and interactions with LLMs. With the usage of a prompt, the users will have the facility to specify the context and constraints for the generated output, and such a facility will allow them to produce more structured and refined responses [Whi+23]. Users can improve the outputs of LLM by incorporating prompts, which are written instructions and examples of the intended interactions. The cues have a direct impact on the model's output, enhancing the possibilities of conversational user experiences [SYG23].

Prompt engineering refers to the engineering process of designing and implementing a prompt that will help in the customization of the desired output from an AI Large language model such as ChatGPT. Essentially, prompt engineering is the process of learning how to effectively interact with AI in a manner that produces the desired outcome by letting it do what is needed [Eki23]. Prompt engineering has been suggested as a collection of strategies, optimal methods, recommendations, and accompanying tools for generating effective prompts. The method involves creating, evaluating, and improving prompts that can yield more precise responses generated by LLMs regarding the composition of the input prompt [CC23].

A key component of dealing with LLMs like ChatGPT has been AI prompt engineering in recent years. Several common threads emerge from a review of the research on AI prompt engineering that shed light on its potential uses and consequences in many fields. The continuous endeavor to create and enhance prompt engineering methods stands out in the literature. To resolve frequent issues in LLM discussions. [RM21] proposed the idea of a 'metaprompt,' an advanced prompt that seeks to utilize the language model's fundamental capabilities to provide more efficient and sophisticated prompts. The metaprompt method enables the model to produce suitable prompts for a desired activity, rather than simply instructing it with a specific goal, [Whi+23] provided a database of prompt engineering patterns that can be utilized to address typical issues that arise in LLM conversations was presented. [Lin+23] investigated reasoning chains as a method of prompting, which enhances the model's capability to solve open-domain commonsense reasoning tasks by utilizing organized sequences of connected knowledge statements retrieved from external knowledge bases. [Eki23] gives a complete guide to ChatGPT, a strong NLP tool with applicability across industry sectors, including prompt engineering methodologies, recommendations, and best practices. This guide explains prompt engineering basics, best practices, advanced strategies, content generation, domain-specific knowledge retrieval, and interactive storytelling. Also, It was suggested by [Cha23] that the Socratic method be used to create prompt templates that are based on inductive, deductive, and abductive reasoning and can be used to improve results. By applying rigorous reasoning during the question-answering process and leveraging the vast knowledge embedded in LLMs, a rigorous approach can be developed to work with LLMs to create prompt templates that interact with language models.

Various prompt engineering applications serve diverse sectors. The study conducted by [SS23] showcased the capacity of AI prompt engineering to completely change content production and communication tactics inside the context of entrepreneurship, [BH07]conducted a thorough examination of prompt engineering methodologies and their applications in the healthcare field, [OLS23] explored prompt engineering as a creative ability for producing AI art, highlighting the importance of expertise and practice in mastering this skill, and [PM23] introduced the ChatExtract technique, which utilizes carefully designed questions on a conversational LLM .

There are suggested principles of prompting to get the result related to a prompt that would be accurate and relevant to the context [Atl23], including Choose your words precisely, Define your prompt with focus and purpose, Try to be concise and specific, Try to provide context and Try to ask for more. These principles are also reflected in prompt patterns [Whi+23]. Prompt patterns are reusable solutions to the raised problems while interacting with LLMs. The concept and inspiration behind prompt patterns are derived from software patterns.

Just as software patterns offer a structured method to address challenges in software development, prompt patterns offer a systematic approach to customize the output and interaction with LLMs. The main motivation behind prompt patterns is that such patterns can be used to enhance the discipline of prompt engineering. These patterns can be grouped into five categories: input semantics, output customization, prompt improvement, error identification, interaction, and context control. What remains unclear is how to decide which specific pattern is the most appropriate one to be applied to a user's query.

2.3 Startup challenges

Startups are an expanding segment of the economy, with worldwide startup figures increasing annually across several industries. A startup is a tiny business that investigates new market potential by creating a solution to a problem in an unstable market[GWA14]. The annual global creation of startups amounts to 305 million as of 2022. Out of that quantity, only a limited number ultimately achieve profitability, and the majority require several years of careful planning and experimentation before attaining it. In 2019, the startup failure rate was at 90%. The failure rate among technology startups was 63%, the highest among all industries. The poor technological choices and procedures used by these companies are one reason for the high failure rate of technology startups[Ric].

In the context of early-stage startups, particularly those in the software industry, the primary obstacles include the unpredictability of emerging markets, securing the first client who pays for the product or service, insufficient knowledge in key areas, inadequate resources and their inefficient allocation, and several other issues[Gia+15].

Furthermore, due to their initial modest size, these organizations have a limited number of individuals available within the organization that entrepreneurs can seek assistance from[CK15]. In order to address these obstacles and mitigate the causes of startup failure, many sorts of help have been provided to entrepreneurs, including mentorship, incubation, and digital tools. With the growth of generative artificial intelligence (AI), namely large language models (LLMs), startups now have more prospects for obtaining support and developing their creative concepts. Nevertheless, due to the development of LLMs, further research is required to explore their potential applications in assisting startups[Gia+15].

As a result of the fact that startups, and particularly those that are in the early stages of their formation, experience a significant amount of difficulty during the process of brainstorming and ideation, it is essential to have the appropriate and accurate assistance available. Even though the vast majority of startups are seen employing this technology to assist their operations in data analysis, chatbots, and process automation, they are still not aware of how to incorporate or make use of it[Duc+].

By combining LLMs with prompt engineering techniques, I aim to provide an approach that can be applied to different domains based on their needs and requirements. Particularly, my observations showcase that there was a significant gap in research and implementation in the field of conversational assistants for startups, so i am focusing on validating this approach by creating and implementing a conversational Digital Assistant-like chatbot for startups. As a result of the digital assistant, Startups will have the opportunity to gain valuable knowledge by having their questions and doubts answered, improving their productivity, and understanding, as well as developing their creative ideas and receiving assistance.

2.4 Digital Assistant for Startups

I was inspired by a master thesis of a student who proposed the idea of a digital assistant chatbot for startups that utilizes prompt engineering to maximize LLMs' capacity for producing the desired results to provide a solution to startups[Att23]. As a result, he identified three important phases. First, the user submits a query to the system. A second step in the process is to determine the purpose of the query. In the end, the system assigns the appropriate pattern to the query, enabling it to provide an appropriate response to the user within interaction with the LLM. Specifically, ChatGPT was used in his thesis, which was also considered my target LLM. Hence, i explain briefly what has been done so far and use his attempt as the V1.0 in the rest of the thesis to illustrate the changes that were made during my investigation 2.1.

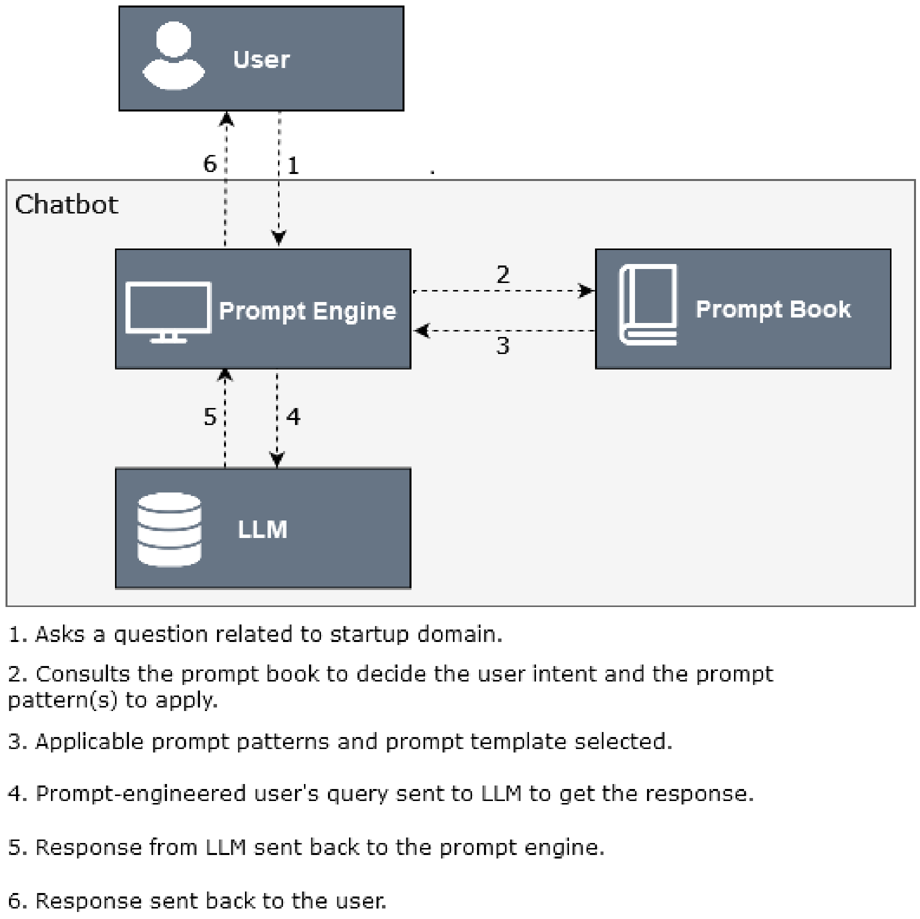

User
6
1
Chatbot
Prompt Engine
2
3
Prompt Book
5
4
LLM
1. Asks a question related to startup domain.
2. Consults the prompt book to decide the user intent and the prompt pattern(s) to apply.
3. Applicable prompt patterns and prompt template selected.
4. Prompt-engineered user's query sent to LLM to get the response.
5. Response from LLM sent back to the prompt engine.
6. Response sent back to the user.

Figure 2.1: Process map of the digital assistant for startups

In the proposed chatbot design, the prompt book and prompt engine play key roles. The prompt engine functions as a bridge between the user and an LLM, converting the user's initial questions into useful prompts by utilizing the prompt book. The study of the prompt patterns suggested in the work [Whi+23] served as the model for the indicated prompt book that can be divided into three sections: 1) The list of purposes for which a query posed by a startup team can be categorized, table 2.1, 2) The selected prompt patterns and corresponding templates for generating prompts2.2, and 3) Matching the purpose of the questions with the prompt patterns selected.

Table 2.1: The purposes of the questions

Question Purpose	**Definition**
Seeking information	Seeking information in the startup context involves the act of acquiring factual information related to startups and the startup ecosystem or startup experience of other people. The accuracy and trustworthiness of the answers matter most for these types of questions.
Seeking advice	Seeking advice in the startup context refers to the process of seeking and acquiring guidance and advice from experienced individuals such as entrepreneurs or experts in the industry. Taking into consideration the specifics of a startup team can help produce advice that suits better the team.
Making decision	In the context of startups, decision-making involves the process of selecting options and implementing actions that will have a significant influence on the startup's development. Balanced responses that consider both the pros and cons of an option are most useful for startup teams.
Reflecting on experience	Reflecting on the experience in the startup context refers to the process of analyzing a startup team's past experiences in order to learn from them to enhance future actions and decision-making.

The following procedure was used to validate his hypothesis, he used a simulated conversation with ChatGPT to test patterns related to startups. A group of students studying entrepreneurship applied these principles in real-life conversations. This helped them to better understand the requirements for designing the prompt book and prompt engine.

Table 2.2: Prompt patterns and corresponding templates for the categories

Purpose Category	**Prompt Pattern**	**Generated Prompt**
Seeking information	Persona,Context Manager,Question Refinement	“Please act as a startup mentor. Within the scope of startups, please consider only the early stages of a startup. When I ask a question, please suggest a better version of the question to use, incorporating information specific to the question that I am using, and check with me if I would like to use the suggested version of the question. The question is: < **Question** >”
Seeking advice	Persona,Cognitive Verifier	“Please act as a startup mentor, and answer my question:< **Question** >, Please generate three additional questions that would help you give a more accurate answer to this question. When I have answered the three questions, combine the answers to produce the final answers to my original question.”
Making decision	Persona,Alternative Approaches	“< **Question** >, Please generate alternative answers to this question, and then compare the pros and cons of each option.”
Reflection on experience	Persona,Flipped Interaction	“Please act as a startup mentor. Please ask me questions to answer my following question. When you have enough information to answer my question, create an answer to my question with consideration of all information provided to you. My question is: < **Question** >”

Furthermore, to understand contextual understanding, clarity, completeness, and coherency, he classified startup-related questions according to core startup pillars and applied the related prompt patterns. Finally, he suggested the approach to classifying startup-related questions to determine the most suitable prompt pattern for interacting with digital assistants for startups. This approach has contained the understanding of the purpose behind each question as well as the generated prompt that can be used for optimizing the user query before sending it to ChatGPT to ensure generating an accurate response for the user request.2.2.

Regarding the valuable findings and the proposed mechanism for implementing their idea, i considered this thesis as a practical approach to provide a better answer to startup-related questions On the other hand, this thesis was initiated to refine this concept in both its research and implementation aspects. This decision was prompted by the absence of an implementation phase and several technical issues within the proposed approach. Furthermore, we identified opportunities for enhancing the suggested purpose categories, purpose checking approach, and prompt templates.

Chapter 3

Research design

The purpose of this chapter is to develop an effective design for the creation of a digital assistant for startups by turning large language models into digital assistants. This study uses Design Science Research (DSR), a research framework that develops innovative solutions to real-world problems. A DSR approach involves six steps during its lifecycle, which include identifying and understanding the problem, setting clear objectives for its solution, designing and developing the solution, demonstrating its functionality, evaluating its effectiveness, and communicating its findings[DJ18].

As a result of this approach, this chapter will provide information about the research process to design and develop a comprehensive prompt book through finding the better version of the question categories, identifying the practical approach to find the purpose behind the user questions, and using the prompt patterns as the foundation for designing the corresponding prompt templates as well as implementing this prompt book as a chatbot. Also, As an important element in DSR, i will design a creative experiment to evaluate the generated results of my research.

3.1 Design the prompt book

In this section, by using an iterative experimental approach, i tried to identify a list of user intents and purposes and their selection process, as well as prompt patterns and templates that correspond to those intents and purposes, along with the selection process that was used to design and implement the chatbot logic for startups.

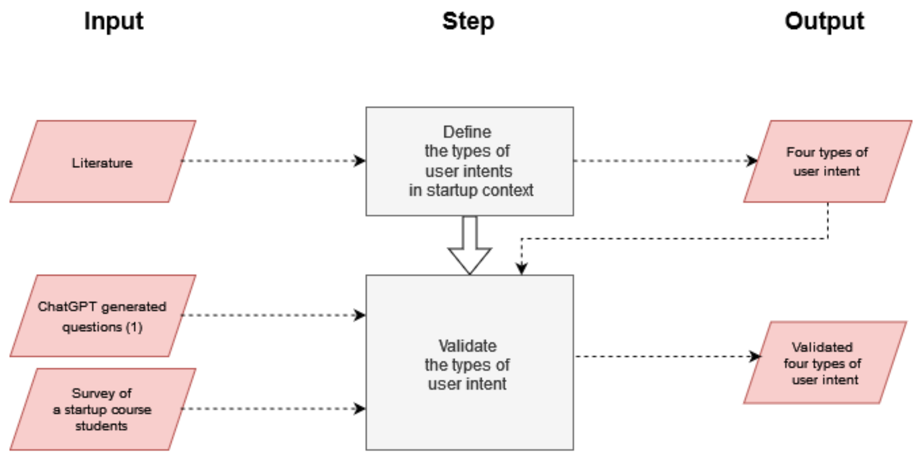

Input
Step
Output
Literature
Define the types of user intents in startup context
Four types of user intent
ChatGPT generated questions (1)
Survey of a startup course students
Validate the types of user intent
Validated four types of user intent

Figure 3.1: The process of definition and validation of user intents

3.1.1 User Intent classification and validation

i began investigating from scratch to finalize the chatbot's purposes to improve the list of user intents included in the prompt book. Since the chatbot was designed to support startup teams, i used the four pillars of startups defined in[Gia+15]: product, team, finance, and market. I quickly realized, however, that this classification scheme was irrelevant to the choice of appropriate prompt patterns. Rather, the purpose, or user intent, behind an asked question, seems to be more relevant when choosing a prompt pattern. A classification scheme was needed to categorize the intent of startup teams when interacting with LLMs. In order to categorize startup team queries when interacting with mentors or domain experts, i carefully reviewed the literature as well as analyzed the question categories in prompt book V1.0[Att23]. As a result of this step, i identified four types of intent that could be used to classify the queries from startup teams.

To evaluate the identified types of user intent and validate them, i followed the following steps based on the process shown in figure3.1. As a first step, i asked ChatGPT to generate 100 questions using a generic prompt without specifying any specific intent. The prompt is: *"Please act as a knowledge expert in the domain of startups, and could you please provide me with 100 common questions that startup teams ask to startup mentors, investors, other entrepreneurs or to themselves?"*. Then, i classified these questions manually according to the four types of user intent under the supervision of the expert in this field, to determine if they were relevant, and if any new types emerged.

Next, i conducted a survey with a class of graduate students who attended a startup course at a Finnish university. The survey was conducted at the beginning of the course. The questionnaire is completely anonymous. It includes two questions related to user intent: 1) “For what purposes would you like to use a chatbot?” and 2) “What questions would you like to ask a chatbot?” By analyzing the responses to these two questions, i validated the types of user intent identified previously.

3.1.2 Implement user intent classification

In light of the comprehensive investigation of user intent categories, which led us to create a unique template for each of these intents 4.4, by Using the research process that is depicted in figure 3.2 we began to create the logical mechanism for the prompt book and chatbot called ***Porpuse Checker*** that would extract the intent and purpose of user questions based on the nature of the questions. The goal was to develop this mechanism with the highest functionality and the lowest fault rate.

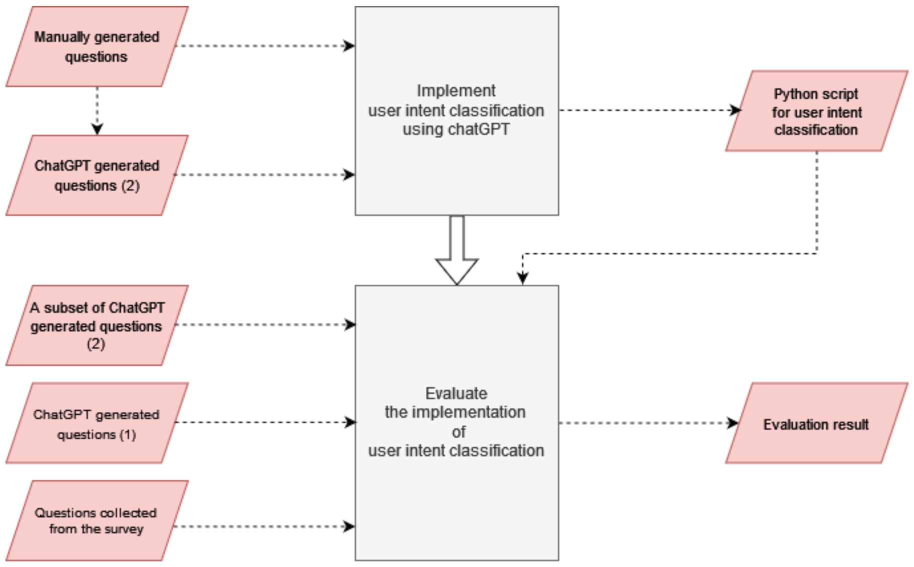

Manually generated questions
ChatGPT generated questions (2)
Implement user intent classification using chatGPT
Python script for user intent classification
A subset of ChatGPT generated questions (2)
ChatGPT generated questions (1)
Questions collected from the survey
Evaluate the implementation of user intent classification
Evaluation result

Figure 3.2: Purpose checker research process

As a result of the definition of user intent categories and the validation process i conducted before starting the research process to develop the chatbot’s purpose checker, i used OpenAI API for ChatGPT to automatically classify user intent, which minimized the stack of technology required for the chatbot to be implemented. In recent studies on prompt engineering and intent classification using ChatGPT, synthetic data generated by ChatGPT can be used, and few-shot prompting can perform reasonably well when real data is scarce[Tan+23; UPN23; Bou23]. With few-shot prompts, a language model can address an entirely new task with only a small number of training examples. Moreover, prompting allows the model to leverage its pre-existing knowledge in scenarios with limited task-specific training examples or a shortage of training data in general. When few-shot prompting is used, this capability is referred to as few-shot learning[Wan+20].

Thus, i manually generated 4 questions per user intent type before asking ChatGPT to generate 26 additional questions. There were 120 questions prepared in total. A Python script is used to categorize user intent based on these 120 questions. Code snippets below showcase the used function in the Python script, the 120 questions, the prompts used to generate 104 of them, and the Python script have shared online[1].

```
        # Used function to find the purpose of the question
def get_question_purpose(question):
    st.session_state.example_selector =
    SemanticSimilarityExampleSelector.from_examples(
        examples,
        OpenAIEmbeddings(),
        Chroma,
        k=1
    )

    st.session_state.selected_examples =
    st.session_state.example_selector.select_examples(
    {"question": question}
    )
    st.session_state.purpose =
    st.session_state.selected_examples[0]["purpose"]
    print(st.session_state.purpose)
    return st.session_state.purpose
```

3.1.3 Evaluate the implementation of user intent classification

As a next step, i used three data sets to evaluate whether the Python script works. The first step was to select 28 questions from the 120 examples used in the Python script. Seven questions were selected randomly per user intent type. I ran the Python script on these 28 questions again to classify them. The Python script was run five times to ensure consistency across these rounds. Next, i ran the Python script on the 100 questions generated by ChatGPT in 3.1.1 five times. As the last dataset, the collected questions from the survey were used, which were provided by students from the startup course. As with

[1]https://figshare.com/s/feef2d27953be1188093

the other two data sets, we classified them manually using the types of user intent, then ran the Python script five times on these questions. All the data used and generated in this step is shared in the same online[2].

3.1.4 Selecting Prompt patterns and templates

The next step towards creating the prompt book used in the chatbot involved selecting the prompt patterns and designing the templates that were appropriate for each user's intent. The inspiration for selecting these prompt patterns was derived from the exploration of prompt patterns proposed in [Whi+23], as well as prompt book V1.0, which was based on the seven prompt patterns considered most relevant for the startup context.

Taking these seven prompts and their templates as a starting point, we created a prompt book using these seven prompts and their templates. From there, we sought out the best way to use these templates for each purpose to provide a more comprehensive and accurate response to user queries. Iteratively, we worked this way through the prompt book in different versions before finding the final version that was used in the project's implementation.

The descriptions of the seven patterns as well as the prompting templates are provided in A.

3.2 implement prompt book

Considering that the prompt book is a key part of this thesis, and it is taken into consideration as the core of the chatbot, comprehensive investigation, and experimentation were conducted to implement the prompt book. To provide an accurate, effective, and comprehensive response to startup-related questions, we implemented the prompt book as a chatbot called DAS. For the design of DAS, i conducted successful research during the implementation process of the research design. In order to choose the appropriate programming language and framework, an iterative process was needed involving various phases of design that required accurate effort. This enabled us to use user interface technology more effectively in conjunction with the logic part for us. In addition, we chose the powerful APIs and deployment environment for the chatbot.

3.3 evaluation

Without evaluation results, the purpose of DSR is to represent unsubstantiated assertions that if the designed artifacts are deployed and implemented they will

[2]`https://figshare.com/s/feef2d27953be1188093`

achieve their objectives[VPB12]. The thesis evaluation phase is important to assess the developed artifact's effectiveness. It will be evaluated to ensure that the artifact meets the requirements and objectives set in the analysis and design phases. I will evaluate the artifact after it has been completely developed in the post-evaluation approach[PBV08].

The evaluation criteria were carefully designed to provide a comprehensive analysis of the usefulness of the contents. In this evaluation, we will gain valuable insights into the artifact's practical benefits and user experience, enhancing its potential to help startups gain information, brainstorm, solve problems, and improve operational efficiency.

3.3.1 Content Usefulness

The purpose of usefulness evaluation is to assess whether DAS can contribute to the acquisition of knowledge, the solving of problems, and the brainstorming of startups in an efficient manner. To ensure that DAS's output is clear, complete, and coherent, we examine it to ensure it meets the requirements. By evaluating the effectiveness of the prompt book and the classification of user intent as a result of leveraging prompt engineering, we will determine how effective it is at providing significant benefits, empowering startups to overcome obstacles, make informed decisions, and leverage their collective experience to help them succeed and grow.

To acquire this evaluation, my initial plan involved interviewing students from the University of Bozen-Bolzano and a university in Finland. The interview protocol was crafted carefully to extract the required information. A pilot interview was conducted to validate the interview protocol, and its results led us to change the evaluation approach from interviewing to experimenting in the designed environment. Specifically, i found significant differences between ChatGPT and DAS in terms of the user interface, response time, and other technical configurations in the pilot interview that caused biases in user evaluations. To mitigate these biases, i decided to design the experiment so that the human subjects and the chatbots interact indirectly. I attempted in this experiment to respond to users without revealing which chatbot generated them.

According to the experiment protocol described in appendixB, we invited two start-up teams to take part and ask startup-related questions. Without revealing the name of the chatbot, we directed them to ask four startup questions one by one. I provided the shared Google document as an alternative to the chatbot interface, which hid the used chatbot for the users. The interactive part of the experiment used the following approach. Users first submitted their questions and waited for the bot's response. Then i retrieved the user question from the shared document and asked it via ChatGPT or DAS to generate the corresponding response, i decided to use ChatGPT for answering the first and third questions and DAS for answering the second and fourth.

the next step was to add the generated response to the shared document for the user to read it and write the potential answer, which created a conversational interaction between the user and the system. We repeated this process for all four questions sequentially. In the next phase, first, i explained the evaluation criteria we considered for this part including the ***Contextual Understanding*** of the chatbot which means to what extent the chatbot understood the context of the user question, ***Relevancy*** of the response which means to what extent the generated response was relevant to the user question, ***Completeness*** of the response which shows that to what extent the generated response was comprehensive and complete from the user perspective, ***Accuracy*** of the generated response which means to what extent the generated response addressed the user's needs accurately, and ***Coherency*** criterion which show to what extent the generated response was coherent to provide the clear and understandable conversation. Then i asked the users to read again the four generated conversations that corresponded to their questions and evaluate them based on the defined evaluation criteria. I provided the evaluation table containing the five evaluation criteria exactly at the end of each conversation and asked the users to rate them from 1 to 5 which indicates to what extent each generated conversation satisfied the evaluation criteria. Users' conversations with chatbots, as well as their evaluation results, are available online in shared Google Docs.[3] and [4].

[3]https://docs.google.com/document/d/1u_-SGcOw22C6z1rYqun_hej-8kCHAbcwIui4HX0aCWc/edit

[4]https://docs.google.com/document/d/1GJ7BiHUGoHMmVHcTkuvN-M9uDsfHVwXLnhSE9vf2nO8/edit

Chapter 4

Research Results

As part of this master's thesis, the finding chapter provides a detailed explanation of the results achieved by the research which was conducted using design science research (DSR)[DJ18]methodology for the project Using prompt engineering to transform LLM into a digital assistant for startups.

The results chapter provides an overview of the findings that have been achieved. A detailed report of findings will be provided in this paper that describes the final version of the prompt book, the implementation process of this prompt book as a chatbot, and the findings and outcomes as a result of the evaluation phase of the research design chapter, which examined the defined research question and determined the findings and outcomes.

4.1 Prompt Book

At the heart of the chatbot lies what we term the "prompt book," which serves as the fundamental framework of our chatbot system. The inspiration for designing our version of the prompt book originates from the exploration of prompt patterns proposed in the paper[Whi+23] and the proposed prompt book in V1.0 [Att23]. The prompt book is composed of three parts: 1) the list of user intent types that can be used to classify a user's query; 2) a set of prompt patterns and the corresponding templates for generating prompts; and 3) the matching between the types of user intent and the prompt patterns. In this part, based on the findings in the research part, each part of the prompt book is updated to develop the final version of the prompt book that was used to implement our digital assistance which performs more efficiently and accurately than the suggested versions.

4.1.1 The types of user intent and validation

Following this step, i identified four types of intent that could be used to classify startup queries. As shown in Table 4.1, you can find the list of intents and their descriptions.

Table 4.1: The types of user intent in the startup context

User Intent	**Definition**
Seeking information	Seeking information in the startup context involves the act of acquiring factual information related to startups and the startup ecosystem or startup experience of other people. The accuracy and trustworthiness of the answers matter most for these types of questions. Typically, questions with such intent are asked in the third person or an impersonal manner.
Seeking advice	Seeking advice in the startup context refers to the process of seeking and acquiring guidance and advice from experienced individuals such as entrepreneurs or experts in the fields. Taking into consideration the specifics of a startup team and making the reasoning explicit can help produce advice that suits better the team. Typically, questions with such intent are asked in the first person.
Brainstorming	Brainstorming in the startup context refers to the process of creative thinking to obtain good ideas related to the development of a startup, e.g., having multiple options for the business models of a startup idea. Questions with such intent typically require divergent thinking. These questions are typically asked in the first person.
Reflecting on own experience	Reflecting on own experience in the startup context refers to the process of analyzing a startup team's own past experiences in order to learn from them to enhance future actions and decision-making. Questions with such intent are always asked in the first person.

In comparison with the user intents used in V1.0, i found that we can consider making decision intent as part of the seeking information category, and the important intent that is called brainstorming is also added to the list of user purposes that was used in our final version of the prompt book. The brainstorming category refers to the process of creative thinking to obtain good ideas related to the development of a startup that hasn't already been mentioned. Also, Our research revealed that the name Reflection on Experience use in the V1.0 prompt book had been misinterpreted because the idea behind this intent is that rather than finding a reflection on other startup teams or someone else's experiences, the users are looking for Reflecting on their startup team's experience. This process involves analyzing a team's past experiences in order to learn from them to improve their future actions. We decided to rename this category Reflection on Own Experience, which is a more meaningful title.

Following the results of the conducted literature review, i surveyed graduate students in a Finnish university at the beginning of their startup course in October 2023. These students came from different faculties including IT, business and management, and finance. The class has a good balance of gender and nationality, and most of them are in their 20s to early 30s. Figure 4.1 shows the levels of their knowledge of startups at the beginning of the course, from 1 (very little) to 9 (very knowledgeable). I will focus on and explain in detail the results of this survey in the research part.

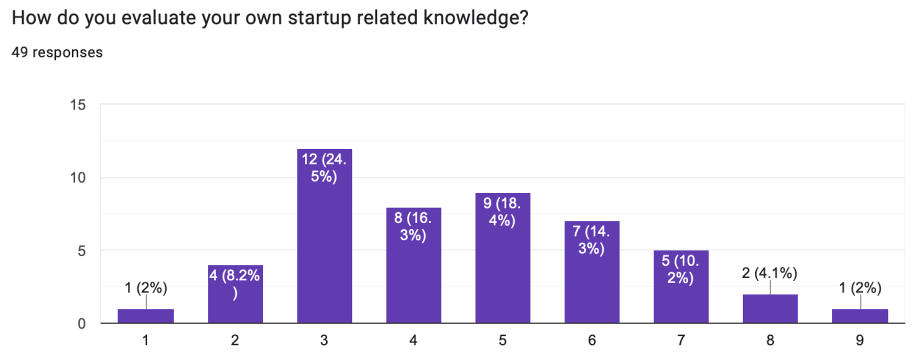


Figure 4.1: Startup related knowledge of the respondents

As shown in Figure 4.1, the majority of the class believes that they already have certain levels of startup knowledge, with a few of them (8) considering themselves knowledgeable or very knowledgeable on the topic. The median self-assessed startup knowledge level of the respondents is 4.0. The mean score is approximately 4.57. Therefore, the collective self-assessed knowledge level is below the mid-point on the scale used in the survey.

Figure 4.2 shows their attitudes towards using a chatbot to support their startup-building processes, which are mostly positive, with 51% of the respondents answering "Yes" and 46.9% saying "Maybe". Interestingly, for the respondents who answered "Yes," their mean startup knowledge level is approximately 4.96, higher than that of the "Maybe" group (approximately 4.26). This suggests that the students who are more open to using a chatbot for startup support ("Yes") tend to rate their startup knowledge slightly higher than those who are uncertain ("Maybe"). However, the difference in the levels of self-assessed startup knowledge between the two groups ("Yes" vs. "Maybe") is not statistically significant, according to the independent t-test (approximately 1.35 with a p-value of about 0.183, higher than the common threshold of 0.05).

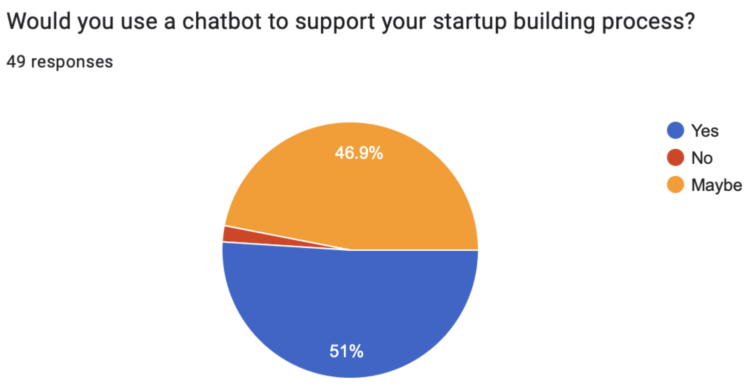


Figure 4.2: Intention of using chatbot

All information provided in the questionnaire is completely anonymous. It includes two questions related to user intent: 1)*"for what purposes would you like to use a chatbot?"* and 2)*"What questions would you like to ask a chatbot?"*. We received 49 responses from the survey and Based on the responses to these two questions, we validated the types of user intent previously identified.

Regarding the evaluation of the types of user intent, 47 responses were received to the first evaluation question, "*for what purposes would you like to use a chatbot?*" which is a multiple-choice one based on the four types of user intent as defined in Table 4.1. As shown in Figure 4.3, among the four types of user intent, *brainstorming* is the most frequently chosen type, followed by *seeking information* and *seeking advice*. *Reflecting on own experience* is less often in comparison to the other types, but still considered relevant by some respondents.

The other evaluation question in the survey is an open-ended question: "*what questions would you like to ask a chatbot?*" 29 responses were received. Following are some exemplar questions provided by the students:

- **Seeking information**: "*Tell me about the recent start-ups on my field of work*", "*Which companies are currently already on the market doing this idea?*"

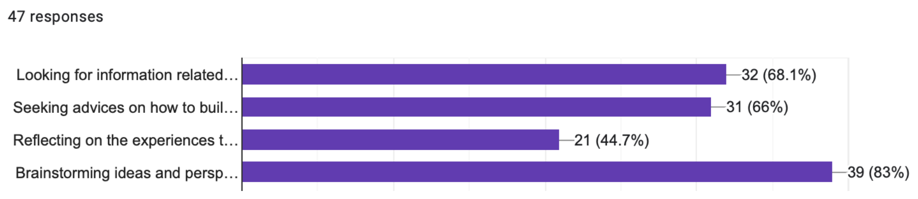


Figure 4.3: Intent validation

- **Seeking advice**: "*How good is my idea?*", "*Is my idea feasible in current enterprise situation?*"
- **Brainstorming**: "*Give me 10 ideas for a business model for a business operating in xxxx xxxx....*", "*Give me some startup ideas related to... (industry)*"

The 100 ChatGPT generated questions and the questionnaire and responses can be found online[1].

4.1.2 Evaluation of the user intent classification using ChatGPT

As described in Section 3.1.3, we used three data-sets to evaluate the implementation of user intent classification using ChatGPT. Data-set 1 is the reuse of the examples, to make sure the classification implementation works technically.

It is not surprising that ChatGPT classifies the questions in the data set 1 most correctly (except 1 out of 28 questions), and consistently across 5 runs. Data-set 2 and Data-set 3 are used to evaluate how well ChatGPT classifies new questions using the 120 examples provided. Data-set 2 contains synthetic questions generated by ChatGPT, and Data-set 3 contains real questions collected from the survey. Table 4.2 shows the agreement levels (Cohen's Kappa) between the manual classification and those from ChatGPT, as well as between various rounds of ChatGPT classifications. The results from both Data-set 2 and Data-set 3 is reported in the same table.

As shown in Table4.2, ChatGPT performed better on the real questions collected from the survey than on the synthetic questions generated by itself. The agreement levels between the manual classification of the real questions and ChatGPT classifications range from 0.47 to 0.67, indicating moderate to substantial agreement. The agreement levels between various rounds of ChatGPT classification range from 0.711 to 1, indicating substantial to almost perfect agreement. This indicates that ChatGPT performs consistently across the runs.

[1] `https://figshare.com/s/feef2d27953be1188093`

Table 4.2: Agreement levels of various classification rounds

		Manual Classification	ChatGPT Classification				
			1st run	2nd run	3rd run	4th run	5th run
Manual Classification		-	0.479	0.536	0.471	0.67	0.536
ChatGPT Classification	1st run	0.261	-	0.801	0.865	0.802	0.801
	2nd run	0.272	0.767	-	0.796	0.867	1
	3rd run	0.215	0.725	0.728	-	0.799	0.796
	4th run	0.167	0.732	0.867	0.746	-	0.867
	5th run	0.174	0.711	0.715	0.811	0.768	-

Note*: gray cells contain the cohen's kappa values from Data-set 2 (n=100); white cells contain the cohen's kappa values from Data-set 3 (n=23); all values have p-value lower than 0.05.*

Results have also shown that it is possible to effectively classify user queries using ChatGPT.

4.1.3 Selecting Prompt patterns and templates

In this section, i will explain in detail the used findings and the iterative approach, which led us to finalize the patterns and the templates used in our chatbot's prompt book.

To determine which pattern or combination of patterns provides a more efficient and accurate final response to a user query, we started by evaluating the prompt patterns and templates used in V1.0 described in table 2.2 for a constant question for each purpose of the user intent. Next, during an iterative approach, we called ***"Patterns selection process"***, we redesigned the templates to improve the effectiveness of the responses, improving the communication between the chatbot and the user, and providing a more user-friendly conversation that allows the user to provide the whole information and needs for the chatbot.

Our pattern selection process led to three versions of the prompt book containing different prompt patterns and templates that were updated continuously, we labeled them from V2.1 to V2.3, which shows the evolution of our template and patterns. In order to evaluate and validate the responses for each version, we worked with an expert in the startup field, helping us to find the best combination of patterns and templates. We considered V2.3 as our final version of patterns and templates, Table 4.3, that were implemented in the prompt book so that our chatbot could use them in its logic part.

Table 4.3: Prompt patterns selection process

User Intent	Prompt Book Versions		
Categories	PB V2.1	PB V2.2	PB V2.3
Seeking information	Persona,Context Manager	Context Manager,Question Refinemen	Context Manager,Question Refinement
Seeking advice	Persona, Cognitive Verifier	Persona, Cognitive Verifier,Context Manager	Cognitive Verifier,Context Manager
Brainstorming	Persona	Cognitive Verifier	Persona,Cognitive Verifier
Reflection on own experience	Persona,Flipped interaction,Context Manager	Flipped interaction,Question Refinement	Context Manager,Flipped interaction,Question Refinement

* PB : Prompt Book

By using the final startup-related questions classification, selected prompt patterns, and defined prompt template for each question category derived from our research findings, we finalized our prompt book to use in our digital assistance. You can find this final version of the prompt book contains the best prompt patterns as well as the prompt template for each user intent in table 4.4.

4.2 Prompt book implementation

According to the research findings, we implemented the prompt book as a chatbot called DAS to assist startups. The prompt book is an essential part of this thesis and forms the basis of the chatbot's functionality. This implementation

Table 4.4: Final prompt patterns and corresponding templates for the categories

Purpose Category	**Prompt Pattern**	**Prompt Template**
Seeking information	Context Manager, Question Refinement	"When I ask a question, please answer to the best of your knowledge. But at the end of your response, also suggest a better version of my original question that can result in a more focused and comprehensive answer. Also, check with me if I would like to use the suggested version of the question. My original question is: < **Question** >"
Seeking advice	Cognitive Verifier, Context Manager	"Please think of three most important questions but ask one question in one response that would help you give more clear and concrete advice to my question. You should ask one question first and wait for my reply before asking the next question. There is no need for the user to know how many questions are there so don't mention any such statement. When I have answered the three questions, combine the answers to produce the final answer to my original question. My original question is:< **Question** >"
Brainstorming	Persona,Cognitive Verifier	"Act as a startup co-founder, and following the best possible brainstorm process, please initiate the brainstorming steps based on my original question. ask enough questions for each steps one by one, You should ask one question first and wait for my reply before asking next question.There is no need for user to know how many questions are there so don't mention any such statement. whenever you have enough information provide the answer to my original question, My original question is:< **Question** >"
Reflection on own experience	Context Manager,Flipped interaction,Question Refinement	"Generate a reflective conversation by posing query related to my question that encourages me to reflect on my own experiences. Upon my response, check if i would like another question.If affirmative , please continue with new reflective question. if I answer 'no' or if you don't have any more questions for me, please combine all the answers to provide my final response as a summarized learning points.If i forget to answer your question at any time, give me a gentle reminder before asking any new question. Let's start with my question which is: < **Question** >"

was intended to figure out the prompt book's functionality. The prompt engine relies on prompt patterns defined in the prompt book, as we explained in section 2.4. Users interact with ChatGPT based on these patterns which we used to create the prompt template for each user intent. Prompt templates by reconstructing the user's prompt enable the chatbot to provide the most efficient and accurate responses for startup-related queries.

Several components are involved in the design process of the prompt engine, such as the selection of the programming language and framework, the integration of APIs to get the needed response from ChatGPT, the use of interactive user interface technology, and the incorporation of logic derived from the prompt book. The following section describes these concepts and processes of implementing the DAS in greater detail, along with how the final response is generated within DAS.

4.2.1 DAS Implementation

For the design of DAS as a digital assistant for startups, i conducted comprehensive and successful research during the implementation process of the research design. To choose the appropriate programming language and framework to implement the prompt engine as the logic part of DAS, a comprehensive iterative process was needed involving various phases of design that required accurate effort. This enabled us to use user interface technology more effectively in conjunction with the logic part for us. In addition, we chose the powerful APIs and deployment environment for DAS. In this section, i will outline the steps that took in order to implement the DAS.

Technical Configuration

We have decided to use Python as the programming language for the development of the prompt engine, while LangChain has been used as the framework that we are going to be using. With LangChain, you can build language-powered applications that are easy to implement and maintain. A context-aware application connects a language model to sources of context like prompt instructions, few-shot examples, etc. Taking advantage of these options, it will be possible to effectively implement and utilize the prompt engine as its core code base has already been established. There is a code snippet below that illustrates the most important elements involved in the design of the prompt engine for the langchain framework: chains, memory, prompt template, schema, and embedding.

```python
# python 3.9
from langchain.chat_models import ChatOpenAI
```

```
 from langchain import LLMChain
 from langchain.prompts import (
     ChatPromptTemplate,
     MessagesPlaceholder,
     SystemMessagePromptTemplate,
     HumanMessagePromptTemplate
 )
 from langchain.chains import ConversationChain
 from langchain.memory import ConversationBufferMemory

 from langchain.schema import (
     SystemMessage,
     HumanMessage,
     AIMessage
 )
from langchain.prompts.example_selector import
    SemanticSimilarityExampleSelector
from langchain.vectorstores import Chroma
from langchain.embeddings import OpenAIEmbeddings
```

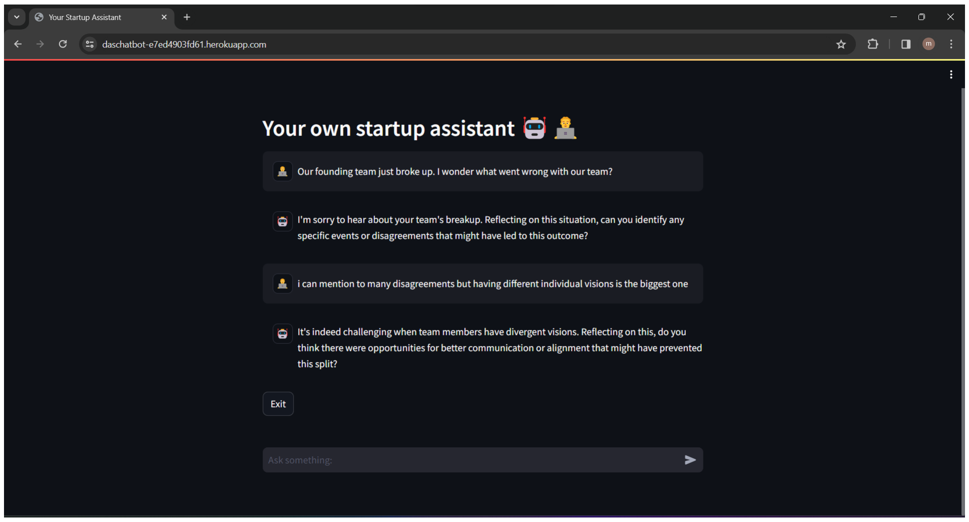
Streamlit was used to design the user interface for the prompt engine. Streamlit is an open-source Python library that offers an easy way to create interactive dashboards and visualizations without the need for extensive web development skills. As a result of its simplicity and efficiency in prototyping and deploying applications[str23], this library gained widespread popularity within the data science community. As Streamlit offers a user-friendly interface and ease of use, i deemed it the best choice for the prompt engine solution. The following code snippet highlights the most important elements of the prompt engine from the Streamlit library and the libraries it requires:

```
import streamlit as st
from streamlit_chat import message
import os
import json
import datetime
from dotenv import load_dotenv
```

API integration refers to establishing connections between software systems or applications through APIs. Using API integration with ChatGPT, a chatbot can interact with messaging platforms, allowing it to accept and respond to user inputs. Obtaining API credentials for a messaging platform and configuring the chatbot accordingly is required to integrate ChatGPT with that platform[ope23]. When designing the prompt engine, i generated my openAI API key and configured its credentials within the code base setup. I used the paid version of the openAI API key to allow us to use the GPT-4 that is capable of producing texts across many domains and tasks[Raf+20] in the prompt engine along with a temperature of zero, which provides a more deterministic answer to the requests. The following code snippet shows how ChatGPT configurations API keys:

```
#openai api key settings
OPENAI_API_KEY=
        "sk-F4n0kn4YyQ6CZpUQYjBOT3BlbkFJfegdUKyNDzCTzvib****"
# Creating the model and its configuration
chat = ChatOpenAI(temperature=0, model="gpt-4")
```

The ***Heroku*** platform was selected for the final stages of implementing DAS to deploy it on a cloud service. This choice enables the chatbot to be accessible to the public for use and facilitates the evaluation process outlined in the thesis evaluation part 3.3.1. Heroku is a cloud service platform that is built on a managed container system with integrated data services and a robust ecosystem for deploying and running applications[sol23]. It also includes powerful log files, which were critical for the evaluation phase. Figures4.4 shows an overview of the user interface of the deployed DAS in the Heroku platform:



Your own startup assistant 🤖 🧑‍💻

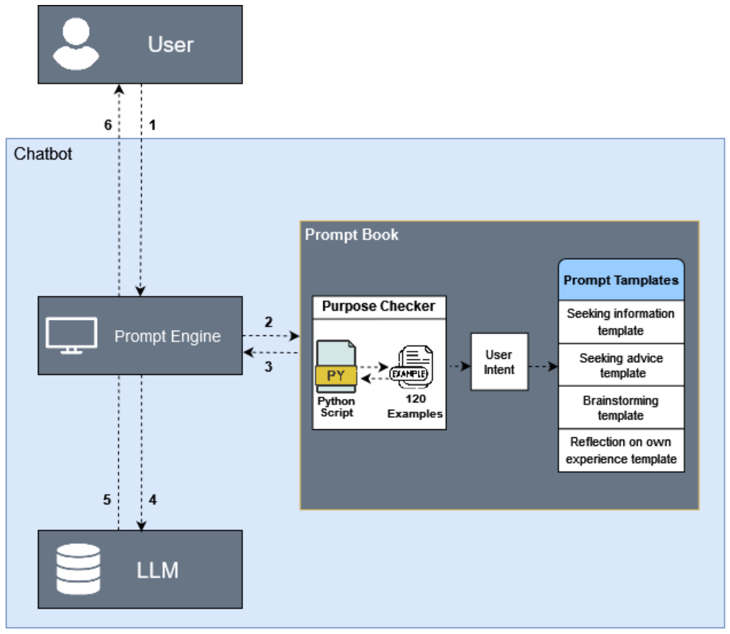
Our founding team just broke up. I wonder what went wrong with our team?

I'm sorry to hear about your team's breakup. Reflecting on this situation, can you identify any specific events or disagreements that might have led to this outcome?

i can mention to many disagreements but having different individual visions is the biggest one

It's indeed challenging when team members have divergent visions. Reflecting on this, do you think there were opportunities for better communication or alignment that might have prevented this split?

Figure 4.4: Overview of the DAS in dark theme



1. Asks a question related to startup domain.
2. Consults the prompt book to decide the user intent and the prompt pattern(s) to apply.
3. Applicable prompt patterns and prompt template selected.
4. Prompt-engineered user's query sent to LLM to get the response.
5. Response from LLM sent back to the prompt engine.
6. Response sent back to the user.

Figure 4.5: Prompt book Architecture

Response generation process

As we have explained, the technical configuration of DAS contains libraries, frameworks, and technical aspects that have been used for the implementation of the chatbot. This section will explain the process that prompt book uses to reconstruct the user's startup question and optimize it to provide the most accurate and efficient answers to their questions, figure 4.5.

In the first step, the user submits the desired startup-related question which the chatbot considers as the input. Regarding the process of ***Purpose Checking*** described in 3.1.2, the prompt engine consults with the prompt book to extract the purpose of the question based on the defined question categories. In fact, the prompt book compares the input with the defined 120-example within the semantic similarity process to label it with the category that has more similarity with the structure of the input.

Next, the user prompt based on its purpose is reconstructed using the template defined in the prompt book 4.4 with the intent of sending it to ChatGPT. Accordingly, the user question is incorporated into the corresponding template selected in the previous step and this template is used to generate an accurate and appropriate answer to the user question using ChatGPT. The code snippet detailing these steps, from submitting a question by the user to prompt engineered response from ChatGPT is available in the appendix C.

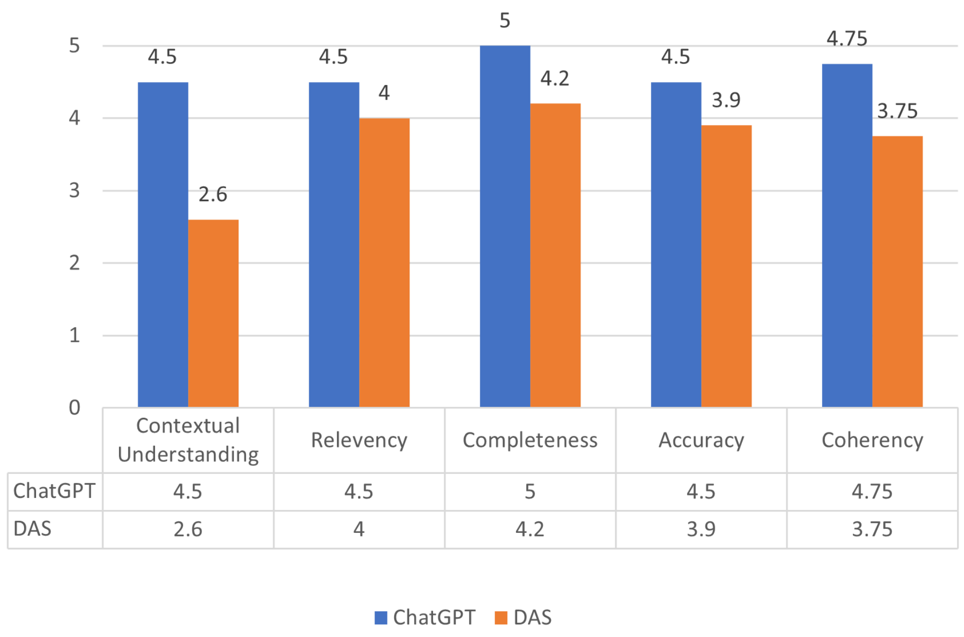
4.3 Evaluation

A phase of evaluation is carried out as part of the thesis to determine the effectiveness of the application that was developed. As part of the evaluation phase, the artifact is tested to determine whether it satisfies the requirements and objectives that were established during the phases of analysis and design. In this phase, i will evaluate the prompt engineering part of the DAS in order to ascertain how well i did in researching and implementing it. So taking a look at the effectiveness of the prompt book and the classification of user intent will help us to determine this.

this evaluation was performed by comparing the users' interactions with ChatGPT and DAS based on the experiment protocol explained in chapter3.3. The results illustrate the extent to which the chatbot was able to utilize prompt engineering techniques to reply to startup-related questions by comparing the generated prompt-engineered responses from DAS with the standard generated responses from ChatGPT. The purpose of this part is to describe the obtained results to evaluate the effectiveness of the prompt book along with the mechanism, and prompt engineering techniques that were examined during my research.

4.3.1 Content Usefulness

Considering the description and evaluation results provided in the figure4.6, it was not surprising that ChatGPT outperformed DAS on all the evaluation criteria in the context of startup-related questions evaluated by two startup teams. Despite the necessity of working more on the logic part of DAS to enable it to provide more accurate responses, it gained an acceptable score of 4.2 and 4 in the two factors of completeness and relevancy of responses, respectively. As a result, it can provide relevant, comprehensive, and complete answers to user queries, making it possible for users and entrepreneurs alike to take advantage of this digital assistance at any stage of their startup's development. The accuracy and coherency criteria for DAS, however, scored 3.9 and 3.75 respectively, which indicates that the responses were less accurate than those from ChatGPT.



	Contextual Understanding	Relevency	Completeness	Accuracy	Coherency
ChatGPT	4.5	4.5	5	4.5	4.75
DAS	2.6	4	4.2	3.9	3.75

Figure 4.6: Content usefulness evaluation result ChatGPT Vs. DAS

It also indicates that the responses were coherent, but perhaps not as clear or easy to understand for users as ChatGPT's responses. With a score of 2.6 in the contextual understanding criterion, we need to work more on the prompt book and DAS which allow the chatbot to understand the user's questions and provide accurate responses. As a reason to reflect on the evaluation findings, It is important to remember that during the research process, all the results were evaluated and validated by a startup expert, especially during the creation process of prompt templates. She approved the results based on the startup's principles, which was the primary objective of this study. As a result, the difference in responses to the experiment's results can be explained by the fact that the expert and users of DAS had different perspectives and knowledge regarding the startup's principles. It has been observed that users usually get their results with fewer interactions with the chatbot, which differs from the nature of the DAS, which asks questions based on defined startup structures to provide more accurate, comprehensive, and relevant responses to user inquiries. In light of this, we can conclude that the low scores achieved by DAS, especially in contextual understanding, were a consequence of the differing levels of knowledge and expectations of users. As an additional reason for the low scores in contextual understanding and accuracy criteria, we conducted a time-consuming manual experiment in the evaluation part, which in some cases frustrated the users in attempting to continue the conversation.

As a result of this evaluation, despite four of five evaluation criteria, DAS managed to obtain some acceptable scores, but the logic component of the DAS needs to be improved further in the next phase so that it can provide the user with more accurate, coherent, and relevant responses.

Chapter 5

Discussion

This master thesis provided valuable insight into how prompt engineering can be effectively implemented to improve the utilization of large language models (LLMs) for startup teams. Regarding the research findings that led me to design the prompt book containing prompt engineering patterns, i was able to design DAS that brings the benefit of using ChatGPT to answer startup-related questions from brainstorming, problem-solving, and decision-making to the knowledge acquisition processes for startups and entrepreneurs. Moreover, i introduced a practical mechanism to categorize user questions according to their purpose, which helped me to understand the reasonable connection between the user intention and the proposed prompt templates to finalize my prompt book. Regarding the conducted experiment and analyzing its results in the evaluation part of the research, despite gaining a low score in some evaluation criteria i can opine that we successfully leveraged prompt engineering techniques to provide more accurate and efficient responses by optimizing the performance of ChatGPT as the large language model(LLM) for the different needs of startups. It was the discrepancy between the insight of the product designer, who worked with a startup expert during the design phase, and the two DAS evaluators who had less prior experience with startup principles that led to the unexpected results of the experiment during the evaluation phase. In fact, as a result of this discrepancy, different reactions were observed to responses.

5.1 Research contribution

Research conducted for this master's thesis has made a significant contribution to the field of supporting startups, particularly by utilizing chatbots. Through the application of prompt engineering techniques in the startup domain, we aimed to turn large language models (LLM) into digital assistants to maximize the potential of artificial intelligence (AI). In this research, significant findings such as prompt patterns and user intent classifier have been identified through a comprehensive literature review, experiments, interviews, and studies which contribute to the design of a prompt book that can be used to assist startups in various ways.

A digital assistant developed by the LLM not only offers startups a user-friendly and efficient interface that provides them with more accurate and practical responses to their startup needs but also offers personalized and customized services by leveraging the vast knowledge and capabilities of the LLM. According to this research, AI-driven chatbots can support startups throughout their growth journeys, from the validation of ideas to developing growth strategies. This study provides valuable insights for entrepreneurs, startup founders, and students interested in leveraging the power of artificial intelligence and learning management systems in the startup ecosystem and contributes to the existing body of knowledge on AI applications in startups.

5.2 Research limitation

There is no doubt that this master's thesis has made considerable progress by proposing and implementing a comprehensive approach to transform ChatGPT into digital assistance for startups by leveraging prompt engineering techniques in order to develop it as a chatbot (DAS). However, it is important to highlight the limitations of this research as well.

Firstly, it primarily focused on the startup environment, which may not fully represent the diverse range of scenarios where LLMs like ChatGPT are applied. However, i do believe that the methodology and insights gained from this study can be adapted to various other domains where LLMs are utilized. Future work could explore the application of this user intent classification framework in different sectors, such as education, healthcare, or customer service. Furthermore, the evaluation of the user intent categories and content usefulness used a small sample of students in one course at one university as well as a small sample of startup teams that participated in the experiment, which limits the generalizability of the evaluation results. Therefore, a larger sample of student and startup teams would provide more valid evaluation results. Furthermore, i firmly believe that training LLMs to interpret and respond accurately to nuanced questions raised by humans is one of the most challenging tasks that will require a great deal of content refinement and adaptation in the long run.

Finally, the study relied heavily on research focused on the accuracy and reliability of ChatGPT, which may limit its generalizability despite its state-of-the-art capabilities. This study, despite its limitations, provides valuable insight into the integration of AI and LLM technologies in the startup ecosystem along with the use of prompt engineering techniques to create a chatbot that can act as a reliable assistant for startups and lay the foundation for further research.

Chapter 6

Conclusion and future work

In conclusion, this master thesis explored and implemented the development of a digital assistant tailored specifically for startups, utilizing prompt engineering methods and techniques in order to maximize the use of language models (LLMs) in order to address the defined research question. The proposed solution consists of a prompt book, which employs accurate prompt patterns and their corresponding templates and the result of the purpose classifier mechanism. The purpose checker as an important part of the proposed prompt book identifies the purpose of the question which is used for selecting the corresponding template to answer the user question.

We refined, evolved, and validated the prompt book and their corresponding templates through the iterative defined research process by using various sources including generating questions using ChatGPT, surveying graduate students, and inputs from an expert in the field of startup. I also evaluated whether the Purpose Checker element was effective in classifying user intents. A substantial to almost perfect level of agreement was observed across several datasets, which enabled us to implement the chatbot more effectively by relying on its functionality which led the chatbot to reconstruct user questions, apply relevant templates, and interact with ChatGPT to generate accurate responses.

As the main goal of this project was to provide an exclusive digital assistant for startups, i implemented the proposed prompt book and suggested mechanisms as a chatbot called DAS. This implementation needed to choose an effective programming language, flexible frameworks, and versatile APIs. As a result of my research for selecting the best configuration, i reached the combination of Python, LangChain, and Streamlit which were selected for the prompt engine's design and user interface, while the ChatGPT API facilitated interactions with OpenAI's language mode. To show my awareness and curiosity regarding the new technology that was released by openAI in the middle of the implementation phase i also created the customized version of the GPTs called Startup Mentor by using the prompt book to express the idea and logic behind it D.1. Furthermore, i compared the implementation process of two chatbots that followed the same logic and prompt book, to determine the pros and cons of each chatbot during the implementation phase D.2

Additionally, i conducted experiments involving two startup teams in the prompt engineering phase and evaluated the content usefulness of the DAS based on five criteria: context understanding, completeness, accuracy, coherency, and relevancy of the generated responses. These assessments aimed to measure the effectiveness of both the research and implementation phases of the thesis. Fortunately, DAS achieved acceptable scores in both completeness and relevancy of answers, which means that the software can answer user queries in a comprehensive, relevant, and complete manner. As shown by the accuracy and coherency criteria scores, the responses were less accurate than those from ChatGPT. According to the contextual understanding criterion, the responses were coherent, but perhaps not as clear or easy to understand as ChatGPT's. A summary of the results indicates that although DAS obtained acceptable scores on four of the five evaluation criteria, it is necessary to improve the logic component of the DAS further in the next phase so that it can provide more accurate, coherent, and relevant responses to the users.

In terms of future work, there are several potential directions. For the evaluation phase, we introduced a creative manual experiment that was time-consuming and lacked the background knowledge of the participants. Therefore, creating an automated experiment that captures the level of experience and knowledge of the user would mitigate biases and optimize the validity of the results. Additionally, automated experiments allow for long-term data collection, which can be beneficial for the problem-finding of the research hypothesis.

Additionally, A comparable approach to the one taken in this study could be applied to other Large Language Models (LLMs) in addition to ChatGPT. The versatility and applicability of the digital assistant can be increased by adapting the prompt book, prompt patterns, and their corresponding templates to different language models. A key objective of this master thesis is to demonstrate the potential of using AI and LLMs for maximizing their use in startups by implementing a chatbot that follows prompt engineering techniques, so it can serve as a good recommendation and approach for future studies in this field to utilize prompt patterns and the proposed mechanism used in DAS in other domains. Although comprehensive research and study will be required to provide prompt books and patterns for the specific domain, it will offer the wonderful benefit of combining prompt engineering techniques with the capabilities of LLM for those domains.

Bibliography

[DG02] Thomas H Davenport and John Glaser. “Just-in-time delivery comes to knowledge management.” In: *Harvard business review* 80.7 (2002), pp. 107–11.

[All+04] Tammy D Allen et al. “Career benefits associated with mentoring for protégés: A meta-analysis.” In: *Journal of applied psychology* 89.1 (2004), p. 127.

[BH07] Anol Bhattacherjee and Neset Hikmet. “Physicians’ resistance toward healthcare information technology: a theoretical model and empirical test”. In: *European Journal of Information Systems* 16.6 (2007), pp. 725–737.

[CJH08] Clayton M Christensen, Curtis W Johnson, and Michael B Horn. *Disrupting class: How disruptive innovation will change the way the world learns.* McGraw-Hill, 2008.

[PBV08] Jan Pries-Heje, Richard Baskerville, and John R Venable. “Strategies for design science research evaluation”. In: (2008).

[VPB12] John Venable, Jan Pries-Heje, and Richard Baskerville. “A comprehensive framework for evaluation in design science research”. In: *Design Science Research in Information Systems. Advances in Theory and Practice: 7th International Conference, DESRIST 2012, Las Vegas, NV, USA, May 14-15, 2012. Proceedings 7.* Springer. 2012, pp. 423–438.

[GWA14] Carmine Giardino, Xiaofeng Wang, and Pekka Abrahamsson. “Why early-stage software startups fail: a behavioral framework”. In: *Software Business. Towards Continuous Value Delivery: 5th International Conference, ICSOB 2014, Paphos, Cyprus, June 16-18, 2014. Proceedings 5.* Springer. 2014, pp. 27–41.

[CK15] Daniel Cukier and Fabio Kon. “Early-stage software startup patterns strategies to building high-tech software companies from scratch”. In: *Proceedings of the 22nd Conference on Pattern Languages of Programs.* 2015, pp. 1–11.

[Gia+15] Carmine Giardino et al. “Key challenges in early-stage software startups”. In: *Agile Processes in Software Engineering and Extreme Programming: 16th International Conference, XP 2015, Helsinki, Finland, May 25-29, 2015, Proceedings 16*. Springer. 2015, pp. 52–63.

[DJ18] Qi Deng and Shaobo Ji. “A review of design science research in information systems: concept, process, outcome, and evaluation”. In: *Pacific Asia journal of the association for information systems* 10.1 (2018), p. 2.

[Raf+20] Colin Raffel et al. “Exploring the limits of transfer learning with a unified text-to-text transformer”. In: *The Journal of Machine Learning Research* 21.1 (2020), pp. 5485–5551.

[Wan+20] Yaqing Wang et al. “Generalizing from a few examples: A survey on few-shot learning”. In: *ACM computing surveys (csur)* 53.3 (2020), pp. 1–34.

[RM21] Laria Reynolds and Kyle McDonell. “Prompt programming for large language models: Beyond the few-shot paradigm”. In: *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*. 2021, pp. 1–7.

[Mis22] Ahmed Mahmoud Misbah. “6 Technical Tips for Tech Startups”. In: *Proceedings of the Federated Africa and Middle East Conference on Software Engineering*. 2022, pp. 83–84.

[Sha22] Murray Shanahan. “Talking about large language models”. In: *arXiv preprint arXiv:2212.03551* (2022).

[Zho+22] Denny Zhou et al. “Least-to-most prompting enables complex reasoning in large language models”. In: *arXiv preprint arXiv:2205.10625* (2022).

[Atl23] Stephen Atlas. “ChatGPT for higher education and professional development: A guide to conversational AI”. In: (2023).

[Att23] Mohammad Idris Attal. “Digital Assistant for Startups based on Large Language Models”. 2023. unpublished master thesis.

[Bou23] Alkistis G Bouzaki. “Enhancing Intent Classification via Zero-shot and Few-shot ChatGPT Prompting Engineering: Generating training data or directly detecting intents?” 2023. unpublished master thesis.

[Cha23] Edward Y Chang. “Prompting large language models with the socratic method”. In: *2023 IEEE 13th Annual Computing and Communication Workshop and Conference (CCWC)*. IEEE. 2023, pp. 0351–0360.

[CC23] R. Clariso and J. Cabot. “Model-Driven Prompt Engineering”. In: *2023 ACM/IEEE 26th International Conference on Model Driven Engineering Languages and Systems (MODELS)*. Los Alamitos, CA, USA: IEEE Computer Society, Oct. 2023, pp. 47–54. DOI: `10.1109/MODELS58315.2023.00020`. URL: `https://doi.ieeecomputersociety.org/10.1109/MODELS58315.2023.00020`.

[Cla+23] Benjamin Clavié et al. “Large Language Models in the Workplace: A Case Study on Prompt Engineering for Job Type Classification”. In: *arXiv preprint arXiv:2303.07142* (2023).

[Eki23] Sabit Ekin. “Prompt Engineering For ChatGPT: A Quick Guide To Techniques, Tips, And Best Practices”. In: (2023).

[Gir23] Louie Giray. “Prompt Engineering with ChatGPT: A Guide for Academic Writers”. In: *Annals of Biomedical Engineering* (2023), pp. 1–5.

[Had+23] Muhammad Usman Hadi et al. “A survey on large language models: Applications, challenges, limitations, and practical usage”. In: *TechRxiv* (2023).

[HB23] Rawad Hammad and Mohammed Bahja. “Opportunities and Challenges in Educational Chatbots”. In: *Trends, Applications, and Challenges of Chatbot Technology* (2023), pp. 119–136.

[Kor+23] Pawel Korzynski et al. “Artificial intelligence prompt engineering as a new digital competence: Analysis of generative AI technologies such as ChatGPT”. In: *Entrepreneurial Business and Economics Review* 11.3 (2023), pp. 25–37.

[Lin+23] Chen Ling et al. “Knowledge-enhanced Prompt for Open-domain Commonsense Reasoning”. In: (2023).

[Liu+23] Pengfei Liu et al. “Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing”. In: *ACM Computing Surveys* 55.9 (2023), pp. 1–35.

[Lun+23] Brady D Lund et al. “ChatGPT and a new academic reality: Artificial Intelligence-written research papers and the ethics of the large language models in scholarly publishing”. In: *Journal of the Association for Information Science and Technology* (2023).

[ope23] openai.com. *ChatGPT API Integration*. OpenAI. 2023. URL: `https://platform.openai.com/docs/api-reference`.

[OLS23] Jonas Oppenlaender, Rhema Linder, and Johanna Silvennoinen. “Prompting ai art: An investigation into the creative skill of prompt engineering”. In: *arXiv preprint arXiv:2303.13534* (2023).

[PM23] Maciej P Polak and Dane Morgan. “Extracting Accurate Materials Data from Research Papers with Conversational Language Models and Prompt Engineering–Example of ChatGPT”. In: *arXiv preprint arXiv:2303.05352* (2023).

[SS23] Cole E Short and Jeremy C Short. “The artificially intelligent entrepreneur: ChatGPT, prompt engineering, and entrepreneurial rhetoric creation”. In: *Journal of Business Venturing Insights* 19 (2023), e00388.

[sol23] solidstudio.io. *Heroku development company*. Heroku. 2023. URL: `https://solidstudio.io/technologies/heroku`.

[str23] streamlit.io. *Streamlit*. streamlit. 2023. URL: `https://docs.streamlit.io/`.

[SYG23] Guinan Su, Yanwu Yang, and Jie Guo. “Prompt Your Mind: Refine Personalized Text Prompts within Your Mind”. In: *arXiv preprint arXiv:2311.05114* (2023).

[Tan+23] Ruixiang Tang et al. “Does synthetic data generation of llms help clinical text mining?” In: *arXiv preprint arXiv:2303.04360* (2023).

[UPN23] Solomon Ubani, Suleyman Olcay Polat, and Rodney Nielsen. “ZeroShotDataAug: Generating and Augmenting Training Data with ChatGPT”. In: *arXiv preprint arXiv:2304.14334* (2023).

[Whi+23] Jules White et al. “A prompt pattern catalog to enhance prompt engineering with chatgpt”. In: *arXiv preprint arXiv:2302.11382* (2023).

[Duc+] Anh Nguyen Duc et al. “Understanding the Role of Artificial Intelligence in Digital Startups: A Conceptual Framework”. In: ().

[Hea] Alex Heath. *OpenAI is letting anyone create their own version of ChatGPT*. URL: `https://www.theverge.com/2023/11/6/23948957/openai-chatgpt-gpt-custom-developer-platform`. (accessed: 06.011.2023).

[Liu+] Y Liu et al. “Jailbreaking ChatGPT via Prompt Engineering: An Empirical Study. arXiv 2023”. In: *arXiv preprint arXiv:2305.13860* ().

[Ric] Audrey Rawnie Rico. *27 Startup Statistics Entrepreneurs Need to Know*. URL: `https://fitsmallbusiness.com/startup-statistics/`. (accessed: 08.06.2023).

Appendix A

Prompt patterns

The descriptions of the seven patterns as well as the prompting templates are provided below:

Persona: in this pattern, a user asks an LLM to play a particular role. In role-playing, the LLM is given a role without providing fine details of that role. This pattern is useful when users are unsure about the exact information needed to process their request, but they are aware of the role of a person responsible for that particular job. **Context**: When the startup team would like the LLM to assume a specific startup role, e.g., a co-founder, a mentor, or an investor, and provide a detailed answer accordingly. **Template**: This prompt pattern will be applied at the beginning of a conversation, to define the role that the user would like the LLM to assume when having the conversation.

Context Manager: This pattern helps users to either introduce or remove a specific context while having a conversation with an LLM. Therefore, users drive LLMs to consider or ignore a few aspects while producing the output. Otherwise, LLMs tend to provide broader or generic answers to a particular question asked by the user. **Context**: When the startup team would like the LLM to focus on (or exclude) a specific stage or aspect of the startup process. **Template**: This prompt pattern will be applied at the beginning of a conversation.

Now take a look at the purpose-specific prompt patterns which are :

Question Refinement: the purpose of this pattern is to prompt an LLM to produce a refined version of the user question so that a piece of accurate information can be produced. Therefore, the LLM needs to have a few interactions with a user to produce the refined question. Alongside this, the LLM also needs some context to produce a better version of the question. **Context**: When the startup team is unsure what is the right question to ask and would like to get help to rephrase the question. **Template**: This prompt pattern will be applied at the beginning of a question.

Template: this pattern is recommended when the user needs to restrict LLMs to follow a use-case and produce output accordingly. Therefore, LLMs deliver output in a format that the user specifies. In this scenario, LLMs do not know the specified template and the user has to specify it while asking a question. **Context**: When the startup team would like to ask the LLM to produce structured output applying a given template. **Template**: This prompt pattern will be applied at the end of a question.

Cognitive Verifier: this pattern is proposed to restrict LLMs to always decompose the original questions into a series of sub-questions automatically. Thereafter, by combining answers to sub-questions, an LLM produces the answer to the original question. The original thought for this pattern comes from a recent study [15]. According to the authors, the LLM can assert more reasonably if we divide the key problem into sub-problems and then the LLM can process them in a sequence. This strategy is referred to as least-to-most prompting[Zho+22]. **Context**: When the startup team asks a question that is too complex more specific and elaborated information is needed to answer the question. **Template**: This prompt pattern will be applied at the end of a question.

Alternative Approaches: it helps to overcome the problem of cognitive biases. Humans have the natural tendency to deviate from norm and/or rationality in judgment and filter information based on their personal opinions and experiences. Therefore, the rationale behind this pattern is to overcome cognitive errors so that users may ask for alternative ways of doing a particular task. A comparison of alternative practices, in terms of pros and cons, could also be asked by users. **Context**: When the startup team looks for alternative options and would like to compare them using certain criteria. **Template**: This prompt pattern will be applied at the end of a question.

Flipped Interaction: in this pattern, the interaction between the user and the LLM is flipped. It means that the LLM is supposed to lead the interaction and ask questions to accomplish the user's goals. Communicating a goal to the LLM is a prerequisite in this pattern. As the content of the interaction is produced by the LLM according to the specified goal, therefore, a more precise output is generated by the LLM using the knowledge that the user does not possess. **Context**: When the startup team asks about a topic that they have limited knowledge of. This pattern enables the LLM to guide the team to ask more questions to obtain more knowledge on the topic. **Template**: This prompt pattern will be applied at the beginning of a question.

Appendix B

Experiment protocol

Here is the experiment protocol for doing experiment as a part of my master thesis with startup teams with the intent of gathering information based on their experience in using DAS and ChatGPT for their startup-related questions. Below is a suggested structure will be used. This structure is divided into two sections: one for gathering background information and experiences in the startup context and their familiarity with chatbots, and the other for collecting their feedback indirectly based on the chatbot's generated responses to their startup questions b

B.1 Background and Chatbot familiarity

First, i start welcoming the participants and state the purpose of the experiment as a part of the chatbot's (DAS) user experience feedback and evaluation for my master's thesis. i also explain the importance of this evaluation for the thesis and emphasize that their responses will remain confidential. Second, as a first series of questions, i ask some demographic questions like Name and role within the startup.

In the next step, i start to ask questions about their experiences and familiarity with the chatbot:

- Have you ever used a chatbot for any purpose? If yes, please describe your experience.
- What questions do you ask in chatbots typically?
- Have you ever used any chatbots to support your startup activities?
- How Would you evaluate your experience with chatbots?

B.2 Method and materials

In this section, i start to introduce the chatbot (DAS) and its purposes, functionality, and the tasks it will help with during this experiment. I will direct them to ask four startup questions based on the current stage of their service or product development one by one. I provide the shared Google document as an alternative to the chatbot interface, which hides the used chatbot for the users. Based on the experiment approach, first, Users submit their questions and wait for the bot's response. Then i will retrieve the user question from the shared document and ask it via ChatGPT or DAS to generate the corresponding response, we decided to use ChatGPT for answering the first and third questions and DAS for answering the second and fourth. The next step is to add the generated response to the shared document for the user to read it and write the potential answer, which creates a conversational interaction between the user and the system. We repeated this process for all four questions sequentially.

At the end of the last conversation, i asked them the below question to provide feedback on the chatbot's performance they used by explaining the evaluation criteria below and asking them to fill out the considered tableB.1 at the end of each conversation by rating from 1 to 5 where 1 is weak and 5 is strong.

- How was your experience with the chatbot?
- **Contextual Understanding**: To what extent did the chatbot understand the context of your queries?
- **Relevancy**: How relevant were the chatbot's responses to your task or question?
- **Completeness**: Did the chatbot provide comprehensive and complete answers?
- **Accuracy**: How accurate were the responses in addressing your needs?
- **Coherency**: How coherent were the responses in providing a clear and understandable conversation?

Table B.1: Evaluation table that be filled out by the user in range 1 to 5

Contextual Understanding	**Relevancy**	**Completeness**	**Accuracy**	**Coherence**

And finally, ask the below question:

- Can you tell which one is coming from ChatGPT and is it distinguishable?

At the end of the experiment, allow participants to share any additional thoughts or comments about their experience and also Thank the participants for their time and valuable feedback.

Appendix C

Response generation code snippet

this code snippet demonstrates how all the steps used during the response generation of DAS from submitting a question to generating the engineered response by ChatGPT. In addition, a short description of the code is provided to assist with a better understanding of the code.

```
if st.session_state.user_input:
    if st.session_state.question:
       st.session_state.user_purpose =
            get_question_purpose(st.session_state.user_input)
            st.session_state.chat_prompt =
            get_chat_prompt_by_purpose(st.session_state.user_purpose)
              if st.session_state.user_purpose == "Reflecting on own experience":
                    st.session_state.chat_prompt =
                    f"{st.session_state.chat_prompt} {st.session_state.user_input}?"
                    st.session_state.messages.append
                    (HumanMessage(content=st.session_state.chat_prompt))
                    st.session_state.chat_prompt1=
                    f"ROLE:User\nPURPOSE:{st.session_state.user_purpose}
                    \nQUESTION:{st.session_state.chat_prompt}"
                    st.session_state.purposes.append
                    (HumanMessage(content=st.session_state.chat_prompt1))
           elif st.session_state.user_purpose == "Brainstorming":
                    st.session_state.chat_prompt =
                    f"{st.session_state.chat_prompt} {st.session_state.user_input}?"
                    st.session_state.messages.append
                    (HumanMessage(content=st.session_state.chat_prompt))
```

```
                  st.session_state.chat_prompt1=
                  f"ROLE:User\nPURPOSE:{st.session_state.user_purpose}
                  \nQUESTION:{st.session_state.chat_prompt}"
                  st.session_state.purposes.append
                  (HumanMessage(content=st.session_state.chat_prompt1))
         elif st.session_state.user_purpose == "Seeking advice":
                  st.session_state.chat_prompt =
                  f"{st.session_state.user_input}? {st.session_state.chat_prompt}"
                  st.session_state.messages.append
                  (HumanMessage(content=st.session_state.chat_prompt))
                  st.session_state.chat_prompt1=
                  f"ROLE:User\nPURPOSE:{st.session_state.user_purpose}
                  \nQUESTION:{st.session_state.chat_prompt}"
                  st.session_state.purposes.append
                  (HumanMessage(content=st.session_state.chat_prompt1))
         elif st.session_state.user_purpose == "Seeking information":
                  st.session_state.chat_prompt =
                  f"{st.session_state.chat_prompt} {st.session_state.user_input}?"
                  st.session_state.messages.append
                  (HumanMessage(content=st.session_state.chat_prompt))
                  st.session_state.chat_prompt1=
                  f"ROLE:User\nPURPOSE:{st.session_state.user_purpose}
                  \nQUESTION:{st.session_state.chat_prompt}"
                  st.session_state.purposes.append
                  (HumanMessage(content=st.session_state.chat_prompt1))
         st.session_state.conversation_history.append
          (HumanMessage(content=st.session_state.user_input))
else:
          st.session_state.user_input2=
          f"ROLE:User\nMESSAGE:{st.session_state.user_input}"
          st.session_state.messages.append
          (HumanMessage(content=st.session_state.user_input))
          st.session_state.purposes.append
          (HumanMessage(content=st.session_state.user_input2))
          st.session_state.conversation_history.append
          (HumanMessage(content=st.session_state.user_input))
 with st.spinner("Thinking..."):
          st.session_state.response =
          chat(st.session_state.messages)
          st.session_state.response1=
          "ROLE:Mentor\nMESSAGE:" + st.session_state.response.content
st.session_state.messages.append
(AIMessage(content=st.session_state.response.content))
st.session_state.conversation_history.append
(AIMessage(content=st.session_state.response.content))
st.session_state.purposes.append
```

```
  (AIMessage(content=st.session_state.response1))
  st.session_state.user_input = ""
```

A user's question will be taken into consideration by checking if the question has already been submitted. By using the *is_question* function, it checks whether the user input is a question, comment, or statement. In this case, if the input is a question, the *get_question_purpose* function is automatically used to identify its category among five defined categories. As a result of the previous step, the script uses the *get_chat_prompt_by_purpose* function to get the prompt of the relevant purpose. Following the selected purpose, the prompt is concatenated to the user question. An enhanced question is now sent to ChatGPT for response. The original conversation is appended to *session_state.conversation_history* and the enhanced is appended to *session_state.messages*. To generate the response, ChatGPT uses *session_state.messages* based on the past whole conversation. Additionally, *session_state.conversation_history* is used to display conversation history.

Appendix D

Comparing DAS and alternative chatbot

D.1 Startup Mentor

My thesis implementation part was underway when i encountered the newly released product from the openAI company, titled "GPTs". It's a revolutionary way to create customized ChatGPT versions for specific tasks and customize how a GPT interacts with people before it's published[Hea]. As a result of this useful opportunity, i developed the GPTs version of my chatbot for startups. The idea behind this implementation was to use the same prompt book i used in DAS to investigate whether my prompt book is working in a different environment with unknown technology in comparison to the technology used in DAS. Thus, i developed ***Startup Mentor***, a customized version of GPT. Throughout this section, i will explain how Startup Mentor was created and compare two implementation approaches to determine which one is easier to implement the idea of the prompt book by providing complete definitions of the positive and negative aspects of each approach.

The only thing you need to implement Startup Mentor is ChatGPT Plus, so i used the same ChatGPT Plus account used in DAS. Startup Mentor was implemented without the use of any coding, and the remaining technical configurations were carried out systematically during the conversation between the GPT builder and the founder. As part of this process, the system asked: "What would you like to create?". In response to the question, i explained the purpose of my thesis which is to create a chatbot to help startups gain the advantage of prompt engineering for their questions. During the next steps, the system gets some generic information to build up the GPT by asking a few questions such as the name of the GPT and the profile picture that others will see when they browse your GPT. Intending to refine the context and set the behavior of the Startup Mentor, the next step system asked questions regarding the logic part. I provided answers to all of these questions to help the system tailor its interactions to be more engaging and effective, as well as determine to what extent responses should be personalized.

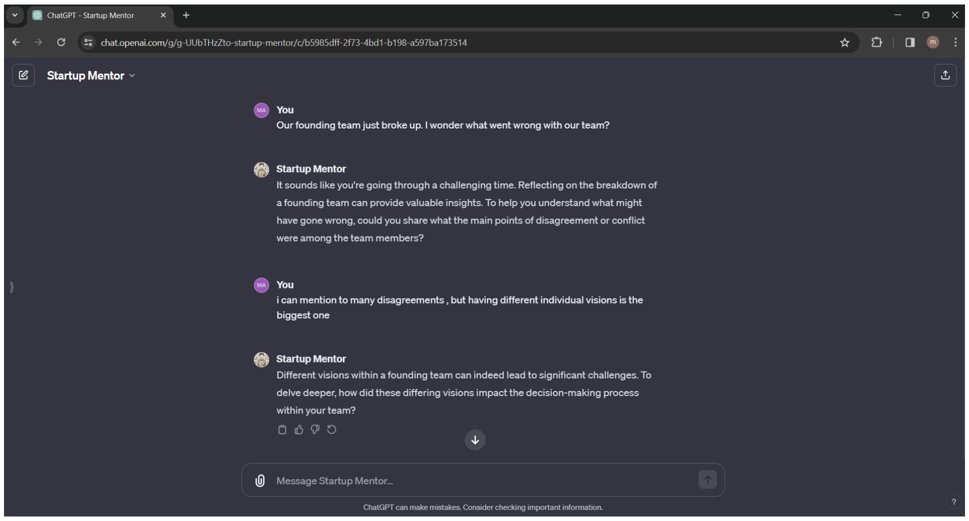
after setting up the Startup Mentor, in the next phase system asks to provide needed refinements or additional adjustments as an optional configuration. As a result, i started to explain the logic used in DAS and explain all the requirements that were needed to implement the prompt book. In fact, during the iterative process in this part, i started to explain the list of user intent categories and the examples used in DAS implementation which increased the ability of the Startup Mentor to understand the user's intent and provide more relevant and helpful responses. The challenging part of this phase was clarifying the interaction strategies or defined templates for each of these categories. It was necessary to conduct an iterative testing and development process to identify the idea of a prompt book and explain the templates and response strategies for the logic part of Startup Mentor in order to provide the best performance possible.

To find the best functionality close to the concept of the project and the nature of my prompt book, i defined the strategies for the responses and unexpected circumstances multiple times. This iterative process involves defining response strategies and response templates for each of the user intents separately in order to find the best interaction mechanism between startup mentors and users. Additionally, i explained them all together as part of the final stages of testing and making sure the system was understood correctly. As part of this sophisticated process, six iterations ire conducted, and due to unexpected behavior from the chatbot when i tested its knowledge to ensure that it understood the point, i had to explain all the steps from scratch the following day. Eventually, i arrived at the current format of the startup mentor, which follows the prompt book concept and acts according to the defined response strategies
.

Finally, Startup Mentor was created, a customized version of ChatGPT that follows the structure and foundation of the prompt book for answering startup-related questions accurately. An overview of Startup Mentor's user interface is shown in D.1:

D.2 Implementation comparison DAS Vs. Startup Mentor

As part of my comparison, I concentrated on three criteria to evaluate the strengths and weaknesses of both DAS and Startup Mentor implementations, drawing upon the insights acquired from both implementations. The compari-


ChatGPT - Startup Mentor
chat.openai.com/g/g-UUbTHzZto-startup-mentor/c/b5985dff-2f73-4bd1-b198-a597ba173514
Startup Mentor

You
Our founding team just broke up. I wonder what went wrong with our team?

Startup Mentor
It sounds like you're going through a challenging time. Reflecting on the breakdown of a founding team can provide valuable insights. To help you understand what might have gone wrong, could you share what the main points of disagreement or conflict were among the team members?

You
i can mention to many disagreements , but having different individual visions is the biggest one

Startup Mentor
Different visions within a founding team can indeed lead to significant challenges. To delve deeper, how did these differing visions impact the decision-making process within your team?

Message Startup Mentor...
ChatGPT can make mistakes. Consider checking important information.

Figure D.1: Overview of the Startup Mentor in Dark theme

son was based on the ease of implementation criterion, which shows how much effort is required for each implementation, the customization criterion, which evaluates how difficult it is to customize the chatbot's behavior and function, and adaptability to changes, which determines how much effort is needed to adjust to changes in requirements or technology updates. Here is a comparison based on the criteria, which i tried to provide in the tableD.1.

Due to predefined steps of GPTs implementation that happen in a conversation process between the founder and the system, Startup Mentor implementation is easier and requires less effort in terms of ease of implementation. In this case, the implementation does not require any coding and an additional environment or platform for deployment or documentation because all functionalities are handled internally, such as storing results and uploading additional documents. The DAS implementation, on the other hand, relies on lines of code and the correct use of APIs and also needs to choose the deployment environment for publishing and the external storage for storing results. Such a chatbot requires additional effort in terms of research, testing, and development, which are all necessary for implementing it.

Regarding the customization of the chatbot's behavior and functions, during the DAS implementation process, i found that due to having extensive documentation on the utilized frameworks, APIs, and technical elements, we could readily modify the chatbot's logic and appearance. This allowed for easier control over its behavior and functionality through code compared to Startup Mentor's configuration process, which involves interactions between founders and the system. While attainable, this customization requires testing and the development of a

Table D.1: DAS versus Startup Mentor implementation comparison

Comparison Criterion	**DAS**	**Startup Mentor**
Ease of implementation	- Needs to write codes and use APIs. - Considering the deployment environment and cloud platform to save the results.	- Implementation happens within the conversation between the founder and the system. - Implementation without writing code. - saving the results internally.
Customization	- Any behavioral customization is controllable and can be done within the code.	- A sophisticated and iterative process is needed to finalize the system behavior based on the concept of the project.
Adaptability to changes	- Adoptive to any changes in appearance and logic which can be done within the code.	- Unchangeable user interface, and unpredictable behavior against changes regarding its defined nature.

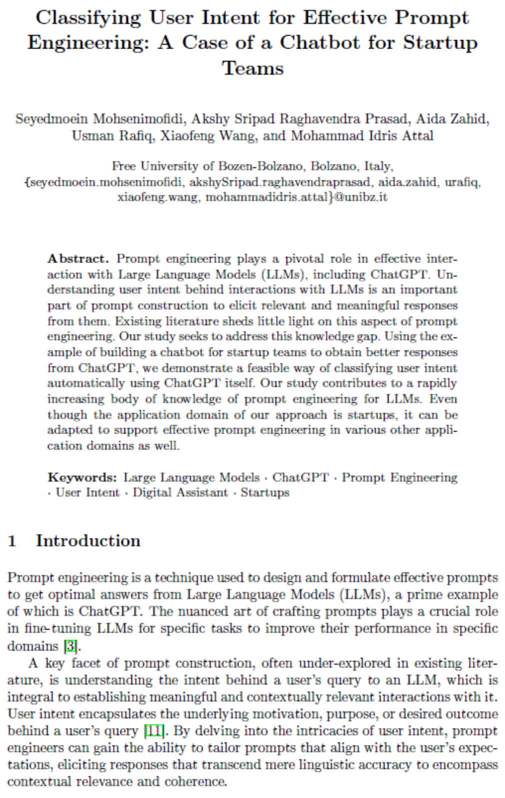
process to adjust any modifications.

Lastly, i found that DAS was more adaptable to the changes than its competitor, based on my experience with both implementation processes. Changing the definitions of templates in Startup Mentor, for example, led us to the conclusion that it would be better to rebuild the current bot and build it from scratch instead of getting stuck in an uncontrollable conversation with the system to apply the changes. When it came to the DAS implementation process, the same update or change was performed very easily within the code.

Appendix E

Published Book Chapter

A published book chapter from section 3.1 of my master's thesis is presented in this appendix, which emerged from an extensive investigation conducted during that phase.



Classifying User Intent for Effective Prompt Engineering: A Case of a Chatbot for Startup Teams

Seyedmoein Mohsenimofidi, Akshy Sripad Raghavendra Prasad, Aida Zahid, Usman Rafiq, Xiaofeng Wang, and Mohammad Idris Attal

Free University of Bozen-Bolzano, Bolzano, Italy,
{seyedmoein.mohsenimofidi, akshySripad.raghavendraprasad, aida.zahid, urafiq, xiaofeng.wang, mohammadidris.attal}@unibz.it

Abstract. Prompt engineering plays a pivotal role in effective interaction with Large Language Models (LLMs), including ChatGPT. Understanding user intent behind interactions with LLMs is an important part of prompt construction to elicit relevant and meaningful responses from them. Existing literature sheds little light on this aspect of prompt engineering. Our study seeks to address this knowledge gap. Using the example of building a chatbot for startup teams to obtain better responses from ChatGPT, we demonstrate a feasible way of classifying user intent automatically using ChatGPT itself. Our study contributes to a rapidly increasing body of knowledge of prompt engineering for LLMs. Even though the application domain of our approach is startups, it can be adapted to support effective prompt engineering in various other application domains as well.

Keywords: Large Language Models · ChatGPT · Prompt Engineering · User Intent · Digital Assistant · Startups

1 Introduction

Prompt engineering is a technique used to design and formulate effective prompts to get optimal answers from Large Language Models (LLMs), a prime example of which is ChatGPT. The nuanced art of crafting prompts plays a crucial role in fine-tuning LLMs for specific tasks to improve their performance in specific domains [3].

A key facet of prompt construction, often under-explored in existing literature, is understanding the intent behind a user's query to an LLM, which is integral to establishing meaningful and contextually relevant interactions with it. User intent encapsulates the underlying motivation, purpose, or desired outcome behind a user's query [11]. By delving into the intricacies of user intent, prompt engineers can gain the ability to tailor prompts that align with the user's expectations, eliciting responses that transcend mere linguistic accuracy to encompass contextual relevance and coherence.

Figure E.1: Overview of the title page of published book chpater