# USING SIX THINKING HATS METHOD AS CREATIVITY TRIGGER FOR REQUIREMENTS ELICITATION: A STUDY WITH ELICITATION WORKSHOP AND LARGE LANGUAGE MODELS

by

Şevval Konan

B.S., Computer Engineering, Gebze Technical University, 2019

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computer Engineering

Boğaziçi University

2025

# ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor, Atay Özgövde, and my co-supervisor, Fatma Başak Aydemir, for their invaluable guidance, support, and expertise throughout this journey. The support and feedback I received have played a crucial role in shaping this thesis.

I am deeply grateful to my mother Kezban, father Kadir, sister Dilara and brother Yusuf their endless encouragement, and belief in me.

I would also like to extend my heartfelt thanks to my husband, Ömer Konan, for his continuous love, understanding, patience, and motivation. His support has been invaluable and has kept me grounded throughout this research process.

# ABSTRACT

## USING SIX THINKING HATS METHOD AS CREATIVITY TRIGGER FOR REQUIREMENTS ELICITATION: A STUDY WITH ELICITATION WORKSHOP AND LARGE LANGUAGE MODELS

Requirement elicitation is a critical phase in the software development lifecycle, where effective communication and collaboration among stakeholders are essential for generating high-quality and creative requirements that align closely with user needs and project goals. A major challenge arises because stakeholders frequently struggle to clearly express their needs. This difficulty is increased when stakeholders come from diverse backgrounds, leading to communication barriers that hinder co-creation. Utilizing the Six Thinking Hats method is a beneficial approach to address stakeholder communication and collaboration. This work presents our experience conducting requirement elicitation workshops using the Six Thinking Hats method, highlighting the challenges faced and offering insights. Furthermore, we investigate applying the Six Thinking Hat method with LLMs for the requirement elicitation process by using various models and employing different prompting techniques and architectures. Our methods suggest a systematic and repeatable approach to applying the Six Thinking Hats technique in the requirements elicitation process. Our results indicate that LLMs can be used for six thinking hat requirements elicitation, as they generate creative and relevant requirements, highlight diverse perspectives, and help analysts identify gaps that may missed by stakeholders.

# ÖZET


## ALTI DÜŞÜNME ŞAPKASI YÖNTEMİNİN GEREKSİNİM BELİRLEMEK İÇİN YARATICILIK TETİKLEYİCİSİ OLARAK KULLANIMI: GEREKSİNİM BELİRLEME ÇALIŞTAYI VE BÜYÜK DİL MODELLERİYLE BİR ÇALIŞMA

Gereksinim belirleme, yazılım geliştirme yaşam döngüsünün kritik bir aşamasıdır ve bu süreçte, yüksek kaliteli ve yaratıcı gereksinimlerin kullanıcı ihtiyaçları ve proje hedefleriyle yakından uyumlu bir şekilde oluşturulabilmesi için paydaşlar arasında etkili iletişim ve iş birliği esastır. Ancak, paydaşlar genellikle ihtiyaçlarını net bir şekilde ifade etmekte zorlanırlar. Paydaşların farklı geçmişlerden gelmesi durumunda bu zorluk daha da artar ve iletişim engelleri, ortak yaratım sürecini zorlaştırır. Altı Şapkalı Düşünme yönteminin kullanılması, paydaş iletişimi ve iş birliği sorunlarını ele almak için faydalı bir yaklaşımdır. Bu çalışma, Altı Şapkalı Düşünme yöntemini kullanarak gerçekleştirilen gereksinim belirleme atölyelerindeki deneyimlerimizi sunmakta, karşılaşılan zorluklara ve elde edilen içgörülere vurgu yapmaktadır. Ayrıca, Altı Şapkalı Düşünme yöntemini LLM'lerle uygulamayı, farklı modeller kullanarak ve çeşitli yönlendirme teknikleri ve mimarilerden faydalanarak araştırıyoruz. Yöntemimiz, Altı Şapkalı Düşünme tekniğinin gereksinim belirleme sürecine sistematik ve tekrarlanabilir bir şekilde uygulanmasını önermektedir. Bulgularımız, Büyük Dil Modellerinin Altı Şapkalı Düşünme yaklaşımıyla gereksinim belirleme amacıyla kullanılabileceğini, yaratıcı ve bağlama uygun gereksinimler üretebildiklerini, farklı bakış açılarını ortaya koyabildiklerini ve paydaşlar tarafından gözden kaçırılabilecek boşlukların analistler tarafından tespit edilmesine yardımcı olabileceklerini göstermektedir.

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# LIST OF ACRONYMS/ABBREVIATIONS

| | |
|---|---|
| CoT | chain-of-thought |
| CT | Creativity Trigger |
| GPT | Generative Pre-trained Transformer |
| LLM | Large Language Model |
| NLP | Natural Language Processing |
| RE | Requirements Engineering |
| RESCUE | Requirements Engineering with Scenarios for User-Centered Engineering |
| TTCT | Torrance Tests of Creative Thinking |
| UI | User Interface |
| UX | User Experience |
| WKCT | Wallach-Kogan Creativity Test |

# 1. INTRODUCTION

Requirements elicitation is a significant Requirements Engineering (RE) activity for it provides the basic information to build upon in the later stages by uncovering, collecting, and gathering requirements of the stakeholders [1]. Requirements elicitation techniques encompass a variety of activities, each with differing levels of human interaction. Some methods, like interviews, require extensive personal engagement, while others, such as surveys, demand less direct interaction. Additionally, some techniques focus on automated approaches, such as extracting features by analyzing app reviews [2].

A requirements workshop is a structured meeting in which a strategically chosen group of stakeholders and experts collaborate to define, create, refine, and reach closure on deliverables that capture requirements [3]. Majority of the problems arising in group elicitation processes are generally linked to insufficient stakeholder involvement. Stakeholders often face difficulties expressing their needs due to a lack of technical understanding. The stakeholders can create imprecise language in the requirements when they come from different domains and bring diverse perspectives. These challenges often result in inefficient workshops, necessitating additional sessions to clarify the requirements and uncover critical requirements that were initially overlooked.

Traditional elicitation methods often focus on explicitly stated requirements, yet many needs remain undiscovered.By using creative thinking methods, stakeholders can better express their expectations, making it easier to uncover hidden needs, resolve complex problems and ambiguities, and ultimately develop innovative solutions. Studies conducted in the field of requirements engineering (RE) have focused on generating creative requirements by organizing creativity workshops [4], identifying creativity triggers [5] and automating the process of generating creative requirements [6]. Except for the studies that worked on automating the process, solutions succeed through the effective and collaborative efforts of individuals participating in the workshop. A variety of

creativity techniques have been successfully applied also to requirements modeling [7], in collaborative settings [8]. Saha et al. [9] survey the creativity techniques used in requirements engineering.

The Six Thinking Hats technique by de Bono [10] is useful for fostering creativity and enhancing collaboration among group members. It is a structured thinking method that individuals or groups can apply. The method provides a structured approach to thinking by assigning specific roles and perspectives to each "hat.". The White Hat focuses on facts, figures, and objective data, encouraging neutrality and unbiased analysis. The Yellow Hat represents optimism and explores value and benefits while adopting a positive attitude. In contrast, the Black Hat takes a critical perspective, considering risks, dangers, and potential downsides with a cautious and negative approach. The Red Hat prioritizes emotions, feelings, and intuitive reactions, allowing subjective insights to guide the process. Green Hat is responsible for creative thinking, generating alternatives, and fostering out-of-the-box thinking and innovation. Lastly, Blue Hat oversees the entire process, managing the thinking framework and ensuring a balanced and structured approach. Together, these hats enable comprehensive and effective problem-solving and decision-making.

This technique has been widely used in the education domain [11, 12], economy [13] and business [14]. Although there are a few studies [15, 16] implementing this technique for requirements engineering tasks in the literature, it has received limited attention. In this study, we describe a requirements workshop protocol implementing the Six Thinking Hats technique and the challenges, lessons learned, and implications for practice as a result of conducting four requirements workshops following it.

LLMs have great capabilities in addressing a variety of traditional Natural Language Processing (NLP) tasks, such as reasoning and natural language understanding. Researchers create LLM-based system for various domains like finance [17], healthcare [18] and education [19] for text classification [20], sentiment analysis [21], and summarization [22]. Besides their generative power, they also shine out for the creativ-

ity that they exhibited. Guzik et al. [23] apply Torrance Tests of Creative Thinking (TTCT) to the GPT-4 and indicate that the model generates novel and unexpected ideas.

This study aims to explore the application of the Six Thinking Hats method as a creativity trigger in requirements elicitation and how LLMs can support this process by simulating diverse stakeholder perspectives. To achieve our study objective, we have specified the following research questions (RQs):

RQ1: How can the Six Thinking Hats method be applied in a requirements elicitation workshop?

RQ2: How can LLMs support a Six Thinking Hats requirements elicitation workshop?

To answer these questions, I constructed a superset of human-generated requirements through a similarity search, conducted both single-agent and multi-agent LLM experiments, and assessed creativity in terms of fluency, originality, and model efficiency. To summarize, our contributions are three-fold:

(i) We present a protocol that can be used for each topic to apply the Six Thinking Hats method in requirement elicitation workshops.
(ii) By implementing the protocol in four different workshops, we report the challenges encountered and the insights gained.
(iii) We present the implementation of different methods for applying the Six Thinking Hats method on Large Language Models (LLMs), and assess and report the impact of these methods on creativity.

The thesis is structured as follows. Chapter 2 examines the related work in requirement elicitation, the Six Thinking Hats method, and LLM-based systems. Chapter 3 summarizes the methods, including language models, prompting techniques, and creativity evaluation. Chapter 4 describes the Six Thinking Hats method, the con-

ducted human workshops, and the insights derived from them. Chapter 5 details the experimental setups for large language models. Chapter 6 evaluates the results of the LLM experiments. Chapter 7 discusses the findings and threats to validity. Finally, Chapter 8 concludes the thesis with key findings and future work.

# 2. RELATED WORK

This chapter presents relevant research and provides detailed insights into studies that explore the intersection of requirement elicitation, creativity, the Six Thinking Hats method, and Large Language Models (LLMs).

## 2.1. Requirements Elicitation

Requirements elicitation plays a pivotal role in software development, laying the groundwork for comprehending stakeholder needs. The research and practice in requirements engineering focus on improving the elicitation process through various techniques. Zowghi and Coulin [1] conducted a comprehensive survey of requirements elicitation techniques by analyzing each technique's strengths and limitations. According to their survey, group workshops are among the most widely used and successful practices alongside interviews, goal-based approaches, and scenarios. Similarly, Palomares et al. [24] find that group interaction techniques are the favorite technique in their survey with practitioners.

Building on these techniques, recent approaches to requirements elicitation have started mixing supplementary sources of information to enhance the process. For instance, Karras et al. [25] explored the application of video data to support requirements elicitation, demonstrating the potential of visual stimuli in capturing user insights. Similarly, Spijkman et al. [26] employed note-taking and a full transcript of voice recordings. They create a REConSum prototype, which can extract relevant questions from the given transcript content. Another notable process is feature extraction from user reviews of applications to elicit requirements. Jha and Mahmoud [27] worked on detecting and classifying non-functional requirements from app reviews, and Dalpiaz and Parente [28] presented RE-SWOT to extract features from the reviews using Natural Language Processing (NLP) algorithms.

## 2.2. Novelty

Creativity has been considered a human attribute, so it has been widely studied in human-centric domains, especially psychology. And researchers applied their findings and newly introduced method mostly in education domain. However, with increasing recognition of creativity's product-oriented nature, research has recently expanded into technical fields like engineering, focusing on supporting creative thinking and measuring creativity.

Numerous methods and tools have been developed to support creative thinking and idea generation. Osborn [29] emphasizes that creativity is not an exclusive talent but a skill that can be cultivated through structured techniques, laying the foundation for modern brainstorming and creative problem-solving methods. Osborn states that group brainstorming enhances creativity output compared to solitary brainstorming. Researchers widely use brainstorming techniques to stimulate creativity, and they have investigated its application in various domains [30].

Another famous method to provoke creative thinking is SCAMPER [31]. SCAMPER proposes different methods to generate new ideas from existing ones by applying some changes. It is an acronym for each word representing the methods: substitution, combination, adaptation, modification, putting to other uses, elimination, and reverse. Researchers adopted this method mainly in the education domain and conducted experiments on students of all levels, from young children [32] to engineering students [33].

### 2.2.1. Six Thinking Hats

Six thinking hats [10] is another widely recognized strategy which introduced as a method that helps creative and innovative thinking and enhances collaboration in group discussions by thinking with different styles while analyzing or solving a problem. Practitioners employ this method in different areas to accomplish various goals. Vernon

and Hocking's [34] study aims to compare two different techniques, the Six Thinking Hats and the Six Good Men, by asking 100 participants to reinterpret a given problem. They report that the Six Thinking Hats technique delivers better results in terms of originality, while the Six Good Men technique is dominant for fluency in problem construction. Göçmen and Coşkun [35] conclude that the Six Thinking Hats technique can enhance an individual's creative thinking, and its effect is improved under time pressure. Chen et al. [36] use the Six Thinking Hats technique better to understand the service design practices of experts and novices.

In addition to its impact on group discussion, the Six Thinking Hats method also facilitates independent personal thinking, as it enables individuals to see from multiple perspectives while solving a problem. Researchers have explored the impact of the Six Thinking Hats method on various solo activities. Chone and Hand [11] applied the method to encourage diverse perspectives in oral discussions among electrical engineering students, while Phuntsho and Wangdi [12] investigated its effect on the writing skills of 8th-grade EFL students.

### 2.2.2. Novelty In Requirements Engineering

Creativity methods have gained attention in requirement engineering for their ability to foster innovative thinking and enrich the quality of requirements beyond enhanced elicitation techniques. The majority of the prior research used creativity workshops in order to encourage creative thinking among stakeholders to construct innovative requirements. Traditionally, these workshops expand over multiple days and require experienced analysts and stakeholders. Maiden et al. apply the Requirements Engineering with Scenarios for User-Centered Engineering (RESCUE) method by conducting three workshops to explore different types of creative thinking: exploratory, combinatorial, and transformational [4]. They highlight the necessity for new and more applied creativity models in their experience report. Maiden et al. also report the result of a workshop to investigate the impact of the creativity workshop on the system specification [37].

After these workshops, Maiden and Robertsons [38] propose a tutorial that defines the application of creativity techniques. The aim of their work is to fill the creativity gap in current requirements engineering and software design practices. Schlosser et al. [39] demonstrated the application of these triggers through workshop sessions. The subsequent studies develop more Creativity Triggers (CTs). Burnay et al. [5] develop six new triggers through brainstorming then Giunda et al. [6] generate 22 CTs by replicating Burnay's work [5] with more data. Horkoff et al. [7] examine how goal modeling can be enhanced with creativity triggers to support the elicitation of more innovative requirements by using creativity triggers to provoke ideas that stakeholders might not consider in a standard goal modeling exercise.

As the years advance, driven by the escalating volume of data and advancements in various Natural Language Processing (NLP) techniques, recent research in Requirements Engineering has been directed toward automating innovative and creative requirements processes. Do and Bhowmik [40] propose employing the Hidden Markov Model to generate innovative requirements from existing software product requirements, resulting in a system that produces creative yet incomplete requirements. Later, Do et al. [41] created a framework with a similar approach by leveraging machine learning and natural language processing techniques such as doc2vec for training and BIRCH for clustering. Their evaluation of creativity in terms of clarity, novelty, and usefulness concludes that the framework generally creates requirements with medium to high creativity levels. Later, Gudaparthi et al. [42] created a framework that creates new requirements by applying tiny modifications (perturbations) to the original requirements descriptions. According to the human subject evaluation, their method generates clearer, more novel, and more useful requirements than Do et al.'s method [41].

The Six Thinking Hats method received limited attention in the requirements engineering domain. Ribeiro et al. [15] use this method with some adjustments to gamify requirements elicitation. Macdonald et al. [16] uses this technique to co-design and co-create a learning platform.

## 2.3. LLM-Based Systems

In recent years, the use of Large Language Models (LLMs) has remarkably increased. With the capabilities to understand and generate human language, they demonstrate great success in traditional natural language processing (NLP) tasks such as text classification [20], sentiment analysis [21], and summarization [22]. Researchers from different domains use LLMs for a variety of downstream tasks. For the finance domain, Wu et al. [17] pre-traine the BloombergGPT model, and Shah et al. [43] pre-traine the Financial LANGuage model (FLANG) using both financial and general-purpose datasets. For healthcare, Li et al. [18] finetune the LLaMA model with the real-world patient-doctor dialogues dataset. Kasneci et al.'s [19] study discusses the opportunities to use large language models in education by creating educational content.

Software engineering is one of the key domains benefiting from advancements in Large Language Models (LLMs). These models have primarily been applied to code intelligence tasks, leveraging plenty of publicly available code, tests, and documentation. Modern approaches to code intelligence often follow a two-step process: pre-training on large-scale text data, followed by fine-tuning on specific downstream tasks. For instance, Wang et al. [44] developed CodeT5 through pre-training on large-scale unlabeled data and fine-tuning it for tasks such as code summarization and clone translation. Similarly, Chen et al. [45] fine-tuned GPT on code and code documentation texts from GitHub to enable natural language-based code generation.

Besides training or fine-tuning, another LLM adoption technique is prompting, which allows querying the trained LLMs to generate responses. Prompting plays an essential role in navigating the model as indicated by recent research [46, 47]. By applying the prompting technique, Sridhara et al. [48] experimented with chatGPT on 15 software engineering tasks such as code summarization, vulnerability detection, and ambiguity resolution. They conclude that ChatGPT is viable for anaphora detection and code summarization but performs poorly in detecting code vulnerabilities.

Prompting is also applied in the requirements engineering field. Fantechi et al. tried to use chatGPT for ambiguity and inconsistency detection in requirements [49,50]. They concluded that chatGPT can detect ambiguities in requirements. For inconsistency detection, their results indicated that chatGPT offers potential benefits in expediting human annotation processes; however, it cannot replace expert judgment. Görer and Aydemir [51] used various prompting techniques, such as few-shot and chain-of-thought, to generate interview scripts for requirement elicitation with LLMs, ChatGPT, and Bard.

## 2.4. Novelty In LLM-Based Systems

LLMs are extensively utilized for content generation tasks, which also represent the most common usage aim of publicly available LLMs. Recent studies explore the impact of LLM assistance on tasks requiring creativity, such as story writing [52, 53], and the generation of coherent scripts and screenplays [54], including titles, characters, and dialogue. These advancements provide an opportunity to compare the creativity of human-generated and AI-generated texts, as well as to develop methods for evaluating creative output.

Chakrabarty et al. [55] introduced an evaluation framework called Torrance Tests of Writing, comprising 14 binary tests developed in collaboration with experts. They assessed 48 short fictional stories equally distributed among outputs from three LLM models and human professionals. The findings revealed that human-generated stories achieved a higher pass rate compared to those produced by LLMs. Padmakumar [56] conducted a comparative analysis between InstructGPT, a feedback-tuned Large Language Model (LLM), and GPT-3, focusing on content diversity. The study concluded that InstructGPT-generated content tends to exhibit higher similarity and reduced diversity compared to outputs produced by GPT-3.

Studies use the Torrance Tests of Creative Thinking (TTCT) [57] method, which is the most widely used test of creativity. Although a trained psychologist should

administer this test, studies conduct the test using LLMs instead. Zhao et al. [58] evaluate the creativity of six different LLMs by posing 700 distinct questions to each model. They assess the responses using GPT-4, scoring them based on four key criteria of TTCT: fluency, flexibility, originality, and elaboration. Chang and Li [59] also modify the TTCT in their research, which reveals how prompt engineering can improve creativity, especially in brainstorming sessions.

Also, DeLorenzo et al. [60] study for assessing creativity using four key metrics of TTCT. The difference is that they worked on hardware code outputs. They introduce a framework for evaluating creativity and conduct a comparative analysis across six language models. This study assesses the creativity of LLMs in generating hardware code, focusing on four key metrics: fluency, flexibility, originality, and elaboration.

Besides measuring creativity, researchers try to enhance the creativity of LLM models. Chang et al. [59] try to enhance creativity by conducting brainstorming sessions with LLMs. Also, the Six Thinking Hats method is gaining attention to achieve this objective. Two studies, in particular, apply this method to enhancethe creativity of LLMs. Rachmann [61] develops an LLM-based prototype of the Six Thinking Hats method, showing its potential for application in creative processes by supporting human creativity. Lu et al. [62] aim to enhance creativity by establishing a discussion framework using LLMs. They conduct experiments where they assign specific roles to agents within a discussion platform, including defining the hats in the Six Thinking Hats method as part of the role structure. Their findings suggest that the Green Hat role scores higher in originality, while the Yellow Hat role receives a lower score. Additionally, they identify other roles, such as Startup Founder and Futurist, which outperform the Green Hat in terms of originality.

# 3. METHOD

This study pursues multiple objectives. Primarily, it aims to apply the Six Thinking Hats method within a requirement elicitation scenario, documenting the experience and presenting a structured workshop outline that can serve as a reusable resource for the broader research and practitioner community. Additionally, the study seeks to investigate the effectiveness of this method in enhancing the creativity of Large Language Models (LLMs) through a systematic assessment of creativity. To achieve these aims, this section provides an overview of the Six Thinking Hats methodology, its adaptation within the context of a requirement elicitation workshop, and the integration of LLMs as participants in this creative process.

## 3.1. Six Thinking Hats Method

The Six Thinking Hats method [10] is a collaborative thinking method by Edward de Bono where the participants collectively wear a "hat" as a thinking strategy [10]. It is critical that all participants in a session simultaneously adopt the same thinking strategy, referred to as "parallel thinking" by de Bono. There are six hats in this method as described in Table 3.1.

Table 3.1. de Bono's Six Thinking Hats and Their Descriptions.

| Hat | Description |
|---|---|
| White | Thinks facts and figures, be neutral and objective |
| Yellow | Thinks the value and the benefits, be optimist |
| Black | Thinks the risks and dangers, be negative |
| Red | Thinks emotions, feelings and gut reactions |
| Green | Thinks out-of-the-box ideas and alternatives, be creative |
| Blue | Thinks about thinking, be the manager of the process |

The Six Thinking Hats method can be applied to individual and group discussions. The group discussions are not bound by rigid standards with the flexibility of group size, the group dynamics, or the occupation of the members. The number of sessions required depends on the group's needs, as there is no defined session limit to achieve problem resolution. In a group setting, a facilitator or chairperson manages the thinking sequence and decides when to transition from one hat to another. The role of the facilitator is a permanent and unalterable position throughout the session.

During a session, the hats can be worn in any order, there is no mandatory sequence of the colors. There is also no obligation to incorporate each hat in each session. De Bono proposes two alternative hat-wearing sequences. The initial step involves a sequence that develops through iteration, where the facilitator picks the first hat, and then participants select subsequent hats as the session continues. De Bono advises against this approach, as it can lead to debates and potential manipulation of the selection process. The second is the pre-set sequence, in which the facilitator determines the order of the colors beforehand and follows this order during the session. The hats can also be used multiple times if needed, so, for example, participants may revisit the black hat two or three times during a session. Regardless of the sequencing choice, de Bono states that there is no single correct sequence for the hats, except the blue hat, which is always used at the start and end of the session to frame and conclude the process. De Bono states that in any meeting, the facilitator automatically functions as the blue hat and is responsible for overseeing the process.

The method suggests laying down ideas in parallel, without immediate discussion or negation. Participants add new ideas without directly engaging with the previous speaker's statement, even if some suggestions seem contradictory. When contradictory ideas arise, they are placed alongside each other. The participants then decide which idea to follow when it is necessary to choose a clear direction. The discussion may incorporate alternatives if consensus cannot be reached. Thus, the method helps to include different opinions.

De Bono suggests starting with a short period for each hat to encourage focus and avoid unnecessary discussion. A rough guide is to allow about one minute per person involved, extending the time only if valuable ideas continue to emerge. This approach keeps the pace brisk, and the participants engaged. Under a specific hat, except the red hat, not everyone is required to speak; someone may not have a relevant idea at that moment, and that is acceptable; others can continue the discussion. However, it can be helpful to invite each individual to share their perspective under a hat.

## 3.2. Requirement Elicitation Workshop

This study introduces a requirements elicitation protocol adapting the guidelines of de Bono for the Six Thinking Hats methods. The protocol is applied in four requirements elicitation workshops conducted for a real-life project concerning the organization of a table tennis tournament twice a year for an amateur group of players from a community. Notably, the proposed protocol is fully reusable and can be readily adopted by individuals or organizations seeking to apply the same methodology in similar contexts.

In this section, we lay out the protocol we adapted to gather requirements in a requirements workshop following the Six Thinking Hats method. The protocol has five main steps: i. preparation, ii. introduction, iii. discussion, iv. conclusion, and v. follow-up. In the discussion step, the protocol visits each hat in the order described below. The discussion step establishes the hat order we have adopted for our workshops. However, Section 4.2 discusses the hat order-related challenges. The order of the hats can be adjusted according to the dynamics of each conducted workshop, as there is no fixed or universally correct sequence.

### 3.2.1. Preparation

The facilitator is responsible for arranging a room that can accommodate the workshop, setting the date and sending invitations to the participants. Before each

session, the seating plan in the room should be arranged as a circle or half-circle, to facilitate collaboration and open communication. The facilitator also sets up a voice recorder, and the devices are connected for the introductory presentation slides for the session. The facilitator also prepares and prints the cheat sheet that summarizes the function of each thinking hat and a page outlining the protocol of the workshop and ensures that the participants have access to pens and paper for note-taking.

### 3.2.2. Introduction

Each workshop starts with a brief introduction session where the facilitator explains the objective of the workshop as well as the Six Thinking Hats method including the description for each hat. The objective of the workshop is formulated in terms of requirements elicitation for the project at hand. The participants are encouraged to ask any questions they might have about the workshop. After the questions and answers, the facilitator distributes the workshop materials and collects the informed consent of the participants regarding the workshop and audio recording. The facilitator puts a visual indicator for the blue hat, and updates the indicator to remind the participants of the thinking approach required and reinforce their focus throughout the workshop. The facilitator also takes notes visible to the participants for their reference.

### 3.2.3. The Blue Hat

The facilitator initiates the discussion by announcing, "I am talking under the Blue Hat" and clarifies the objective of the workshop is to gather requirements for the given project. The participants collectively spent one minute to think about adopting the strategy for each hat, followed by approximately one minute for each individual to share their insights. The facilitator sets the order of the hats as white, yellow, black, red, green, and blue.

### 3.2.4. The White Hat

The participants are expected to seek factual information, drawing on their past experiences regarding the project. This phase is designed to help identify essential features that should be included in the application. After one minute of thinking time ends, participants talk about the essential features they require for the app, adopting the mindset of stakeholders to ensure their needs and expectations.

### 3.2.5. The Yellow Hat

At this phase, the participants adopt a positive, value-seeking perspective. The participants collaborate to come up with new features for the project in addition to the core requirements identified under the white hat or to enhance them. The key question they answer is "What features would make this given project more desirable to use?" to highlight the opportunities for improvement.

### 3.2.6. The Black Hat

In contrast to the yellow hat, the participants focus on identifying weaknesses, risks, and challenges in the identified points of discussion up to this point. They critically evaluate each proposed feature's potential drawbacks or difficulties, considering feasibility, usability, and more aspects. The Black Hat session encourages participants to ask probing questions like, "What could go wrong?", "What might be difficult to implement or use?" and "Are there any legal or ethical concerns?". This critical thinking session thoroughly discusses all potential dangers and obstacles before moving forward.

### 3.2.7. The Red Hat

The participants openly share their emotions and instinctive reactions under the red hat about the discussion and the workshop. After this reflection, they imagine their emotional states whilst interacting with the system-to-be and their affective responses

to particular interactions. The objective of this phase is to realize the emotional reactions that may impact the user's experience and their level of engagement with the system-to-be.

### 3.2.8. The Green Hat

This hat represents creativity and innovation. Under it, the participants engage in creative thinking, brainstorming innovative features or novel solutions to address the concerns raised during the black hat session. When participants encounter difficulties ideating novel concepts, the facilitator employs creativity triggers such as random words or personas to break away from traditional thinking patterns and foster an environment of creative exploration.

### 3.2.9. The Blue Hat

At the end of the discussion, all participants wear the blue hat and adopt a holistic view. They reflect on the entire process and discuss the final features. If a participant feels that certain aspects warrant further exploration, they can suggest revisiting a specific hat to delve deeper into that topic. After revisiting other hats, there should be another final blue hat to discuss again. Once all concerns have been addressed and participants express satisfaction with the final solution, the facilitator summarizes the key requirements identified throughout the process and concludes the workshop.

### 3.2.10. Concluding The Workshop

The facilitator ends the recording and starts a question-and-answer session with the participants. This segment helps participants provide verbal feedback on their experience and on the workshop process. This dialogue is instrumental in shaping the protocols for the subsequent workshops. The facilitator expresses gratitude towards all the participants for their contribution, acknowledging their essential role in the success of the workshops.

#### 3.2.11. Workshop Follow-Up

After the workshop, the facilitator promptly sends a thank-you email to the participants, expressing gratitude for their involvement. This email also includes links to two follow-up forms: collecting demographic information and gathering feedback on the workshop experience. The facilitator reviews the audio recordings and notes to capture the requirements. The facilitator asks for clarification via follow-up e-mails or meetings if any topic appears unclear or ambiguous. Additional follow-up workshops can be arranged to address any remaining questions or provide necessary details.

## 3.3. Large Language Models

### 3.3.1. The Framework

This study investigates the success of large language models (LLMs) in replicating the requirement elicitation process using the Six Thinking Hats method. To achieve this, we investigate various LLMs, prompting techniques, and software architectures specially designed for the task. Our experimental framework contains both single-agent and multi-agent systems to optimize task efficiency. We conduct LLM-based experiments utilizing CrewAI[1] , an open-source Python framework designed to coordinate AI agents with role-playing settings and build automated workflows.

The decision to select CrewAI is based on its comprehensive feature set, ease of use, and growing popularity. Also, the AI research community started to adopt the framework for orchestration [63] and to compare AI-based agents [64,65]. The primary strength of the framework lies in its capability to improve team collaboration by managing agents through intuitive configuration files. Making multi-agent communication feasible is crucial because it is key to the success of the requirement elicitation process.

[1] `https://www.crewai.com/`

### 3.3.2. The Models

The selected models are provided by industry leaders such as OpenAI and Google, alongside emerging competitors like Mistral AI, which is gaining adoption in both academic and industrial domains. All chosen models benefit from strong community support and are well-documented, facilitating seamless workflow integration. Furthermore, the selection encompasses general-purpose models with diverse architectural designs (e.g., dense transformers and a Sparse mixture of experts) and varying parameter scales, enabling a systematic analysis of the relationship between model size, architecture, and performance outcomes.

This study uses four large language models (LLMs) for the experiments, each representing a different approach to natural language understanding and generation. The GPT-4o Mini model is a variant of OpenAI's GPT-4 [66], designed for efficiency and smaller-scale deployment while maintaining a high level of textual intelligence. It is low-cost and can handle a large context volume with a context window of 128K tokens. The Gemini 1.5 Flash-8B, developed by Google DeepMind, is the smaller and faster Flash variant known for its high performance in tasks requiring reasoning, complex problem-solving, and language generation. Another model used in our experiments is the Mistral Mixtral-8x7B, a dense and efficient model that balances performance with computational efficiency. Finally, the Mixtral-8x22B is a larger variant, offering enhanced capabilities with 22 billion parameters and optimized for reasoning. 3.2 shows the summary of these models.

### 3.3.3. Prompting

Prompting refers to crafting and refining instructions to use the capabilities of Language Models for intended tasks and applications. This study examines the impact of different prompting strategies on generating requirement lists by using three widely recognized techniques: zero-shot, few-shot, and chain-of-thought (CoT) [67] prompting.

Table 3.2. Summary of Large Language Models.

| Model | Architecture | Number of Parameters | Provider | Key Features | Release Date |
|---|---|---|---|---|---|
| **GPT-4o Mini** | Transformer (Decoder-only) | Unknown | OpenAI | Small variant of GPT-4. Optimized for efficiency. General-purpose language tasks. | July 18th, 2024 |
| **Gemini 1.5 Flash-8B** | Transformer (Multi-modal) | 8 billion | Google DeepMind | Enhanced reasoning and language generation. | May 14th, 2024 |
| **Mistral Mixtral-8x7B** | Transformer (Sparse Mixture-of-Experts) | 8 billion (7B per expert) | Mistral AI | Mixture of Experts (MoE) model. Optimized for efficiency, scalability, and high performance on NLP tasks. | Dec 9th, 2023 |
| **Open-Mixtral-8x22B** | Transformer (Sparse Mixture-of-Experts ) | 22 billion | Mistral AI | Larger MoE model with 22 billion parameters. More powerful for complex tasks, with efficient computation. | Apr 10, 2024 |

In zero-shot prompting, the prompt includes no examples or contextual clues. The answer relies solely on the model's pre-trained knowledge and internal reasoning capabilities. For few-shot prompting, the prompt includes examples or context that navigates the model through the most relevant output. In the prompt, we provide examples of requirements from two different projects, e-commerce and meal-cooking applications. This prompting method can also be called two-shot prompting. In the chain-of-thought (CoT) prompting method, the reasoning process is explicitly structured into intermediate logical steps. In our implementation, we combine CoT prompting with a few-shot by providing one example from a different application, followed by the reasoning steps. This study uses these prompting strategies to analyze their relative effectiveness in generating comprehensive and accurate requirement lists across diverse scenarios. Appendix A shows the example prompts of these three prompting strategies.

## 3.4. Evaluation Metrics

Prior research explores various scoring methods like the Wallach-Kogan Creativity Test (WKCT) [68] and Scientific Creativity Test [69] to evaluate divergent thinking, which is the ability to generate alternative ideas and solutions. Research groups have primarily used the Torrance Tests of Creative Thinking (TTCT) [57] in studies conducted over the years. The TTCT framework contains four distinct metrics—fluency, flexibility, originality and elaboration.

This study embraces Sternberg's [70] definition of creativity: the ability to produce novel and useful. This definition aligns with earlier contributions in requirements engineering [5, 6]. Novelty indicates the same meaning as originality. Also, fluency is directly related to generating a variety of viable options, which is essential for exploring different possibilities in requirement development. As a result, our study adopts novelty, usefulness, and fluency as the primary criteria for evaluating creativity:

- Originality: the degree of novelty or uniqueness of the responses,
- Usefulness: applicable and relevant to the problem at hand,
- Fluency: the number of responses generated,

In addition, we incorporate a parameter known as "Tokens per Requirement" in our LLM experiments. This metric is significant as it reflects the efficiency of the LLM models. Fewer tokens per requirement indicating higher efficiency in generating the desired output.

### 3.4.1. Fluency

For each prompt, we request the LLMs to generate a list of requirements and collect n responses to estimate the average performance. In this context, fluency is measured by the number of requirements generated. The presentation of this metric differs based on whether the experiment is conducted with a single agent or multiple

agents. For single-agent experiments, fluency is computed as the mean of the number of requirements generated over n iterations. For multi-agent experiments, fluency is calculated as the sum of the number of requirements generated across the three iterations. These are expressed as:

$$\text{fluency}_{\text{single-agent}} = \frac{1}{n}\sum_{i=1}^{n} \text{num. reqs.}_i \tag{3.1}$$

$$\text{fluency}_{\text{multi-agent}} = \sum_{i=1}^{n} \text{num. reqs.}_i \tag{3.2}$$

where num. reqs.$_i$ is the number of requirements generated in the $i$-th iteration, and $n$ denotes the number of iterations.

### 3.4.2. Tokens Per Requirement

This metric quantifies the average number of tokens generated by the LLM for each requirement in the output list. It can be calculated as follows:

$$\text{Tokens per Requirement} = \frac{\text{LLM Output Tokens}}{\text{Number of Generated Requirements}} \tag{3.3}$$

CrewAI offers a function that returns the number of tokens in the output generated by the language model. We obtained the token count from this function.

### 3.4.3. Originality

After we get the generated requirements, then we calculate the number of unique requirements, representing the count of requirements that differ from those generated during the human workshop. The value is computed as the mean of the unique requirements for a single agent and the sum of the unique requirements, just like the number of requirements generated.

This number of unique requirements helps to calculate originality. Originality is quantified as the novelty ratio, which is computed as follows:

$$\text{originality} = \frac{\text{number of unique requirements}}{\text{number of generated requirements}} \times 100. \tag{3.4}$$

## 3.5. Similarity Detection

In the initial attempts of our LLM experiments, we utilized the Mixtral model hosted on the Groq server. Upon analyzing the results, we observed a pattern where the number of generated requirements was quite high, but the differences between some pairs of requirements were minimal. For example:

- "The app shall allow users to view and manage tournament settings for match broadcast and media coverage."
- "The app shall allow users to view and manage tournament settings for match security and integrity."
- "The app shall allow users to view and manage tournament settings for match hospitality and services."
- "The app shall allow users to view and manage tournament settings for match sponsorship and partnerships."

To detect such redundancy, we applied the Jaccard similarity between the generated requirements. The Jaccard similarity for two requirements is calculated as the ratio of the number of common words between the two requirements to the number of total unique words across both requirements. Mathematically, it is expressed as:

$$J(R_1, R_2) = \frac{\text{len(common words in } R_1 \text{ and } R_2)}{\text{len(all unique words in } R_1 \text{ and } R_2)} \tag{3.5}$$

where $R_1$ and $R_2$ are two requirements. If their Jaccard similarity exceeds 80%, only the first requirement is kept.

In our study, we also employ a vector-based approach for similarity search by using the cosine similarity metric. The details about the aim of this operation are detailed under Section 4.3. We utilize the text embedding model to represent requirements as vectors and calculate cosine similarity between them. Cosine similarity is calculated using the following formula:

$$\text{Cosine Similarity} = \frac{A \cdot B}{\|A\|\|B\|} \tag{3.6}$$

where $A$ and $B$ are two vectors representing the textual data, $A \cdot B$ is the dot product of the vectors, and $\|A\|$ and $\|B\|$ are the magnitudes (or norms) of the respective vectors.

# 4. SIX THINKING HATS REQUIREMENTS ELICITATION WORKSHOPS

This section provides a detailed explanation of the four requirements elicitation workshops conducted using the workshop protocol 3.2. Section 4.1 provides information about the workshop conditions and participants. Section 4.2 discusses the challenges encountered, lessons learned, and the potential industry implications of the workshop experience. Finally, Section 4.3 describes the preparing a baseline dataset, which serves as a comprehensive superset of the requirements generated across the four workshops.

## 4.1. Workshop Conditions and Participants

Four requirements workshops were conducted to gather insights for a table tennis tournament application designed for amateur players from a small community. All workshops were conducted face-to-face, with participants seated in a semicircle where they could see each other, as well as the table where the notes and a picture of the hat with the current color are displayed. This setup fostered open communication and collaboration, as all participants had a clear view of each other and the shared materials. Each participant contributed to the discussions for each hat; the order was determined by the facilitator, who picked a participant, and the others took their turns one by one clockwise or counterclockwise direction. The workshops are conducted in Turkish which is the native tongue of the participants and the facilitator.

### 4.1.1. Participant Selection

The project we explored is grounded in an actual need and holds potential for development within the senior-year coursework based on the elicited requirements. The participants are selected from a pool of candidates that are either the problem owners as tournament organizers, players, or fans, or IT experts. Apart from the facilitator, a total of 16 participants participated in the workshops. 15 out of 16 participants com-

pleted the demographic questionnaire. The participants came from a wide variety of professional backgrounds, with 40% having prior experience in requirements elicitation as an elicitor. 33.3% of the participants have been stakeholders of other projects. The diversity extended to academic levels, with representation from undergraduate students (four participants), graduate students (two participants), academics (two participants), software developers (four participants), product managers (two participants), and medical doctors (one participant). In the current study focused on the table tennis domain, 40% of the participants reported no experience, while 33.3% indicated they had personal experience with the game, and 26.7% had professional experience in the field. Table 4.1 provides a detailed summary of participant demographics.

Five participants, including one facilitator, were present for each workshop. Apart from the facilitator, a total of 16 participants participated in the workshops. I acted as the facilitator for all workshops. For each workshop, we determined the number of participants based on availability, adopting a pragmatic approach for the original method does not specify a set number for the participants.

Each session lasted approximately two hours. As detailed in the workshop protocol in 3.2, following the workshops, the audio recordings and notes were reviewed to capture the requirements in a manner similar to business analysis practices. Consequently, a list of requirements was created, including the corresponding hat color associated with each requirement during the workshop discussions.

## 4.2. Discussion Of Workshop Results

This section presents the challenges encountered before, during, and after the workshop, the lessons learned, and the implications for the industry, based on observations made during the workshop and feedback from participants.

Table 4.1. Participant Demographics.

| Category | Count | Percentage |
|---|---|---|
| **Total Participants** | 16 | 100% |
| **Completed Demographic Questionnaire** | 15 | 94% |
| **Elicitation Experience** | | |
| No experience | 4 | 26.7% |
| Elicitor | 6 | 40% |
| Stakeholder | 5 | 33.3% |
| **Occupation Profile** | | |
| Undergraduate Students | 4 | 26.7% |
| Graduate Students | 2 | 13.3% |
| Academics | 2 | 13.3% |
| Software Developers | 4 | 26.7% |
| Product Managers | 2 | 13.3% |
| Medical Doctors | 1 | 7% |
| **Domain Knowledge** | | |
| No Knowledge | 6 | 40% |
| Personal Interest | 5 | 33.3% |
| Domain Expert | 4 | 26.7% |

### 4.2.1. Challenges

Keeping the focus: Although the facilitator describes each hat and the order to be followed at the beginning, and constantly reminds the participants with a visual aid to indicate the hat color, some participants had difficulty staying focused on the thinking strategy indicated by the current hat and needed assistance to align their thinking strategy with the current hat. As a result, there have been some misaligned and scattered contributions during the workshops. For example, during the white hat discussions, in which participants focused on facts and information, a participant

would propose ideas unrelated to the app's core functionality, such as suggesting that the app should be fun. In such cases, the facilitator had to intervene, guiding the participants back to the factual nature of the white hat and reminding them that these suggestions could be captured for later discussion under a more suitable hat. This challenge underscored the need for active facilitation to ensure that participants remained aligned with the structured thinking the Six Thinking Hats method requires.

Adopting the thinking strategy: In some cases, even though the participants grasped the description of a hat and were aware of the current hat, they had difficulty implementing the thinking strategy required by that hat. Across all workshops, at least one participant expressed difficulty thinking under the yellow hat. The yellow hat required participants to focus on non-fundamental yet value-adding aspects. However, the distinction between fundamental (discussed under the white hat) and supplementary features (discussed under the yellow hat) is subjective and leads to confusion. Also, the order of the hats may have contributed to the challenge since the yellow hat followed the white hat in the protocol which might have made it more challenging for the participants to maintain a clear separation between them.

Order of the hats: Another challenge related to the order of the hats arose between the black hat and green hat thinking sessions. The participants typically focused on narrowing the application's scope at the black hat session while considering the risks and problems. In this phase, they often opted for more basic features and sometimes eliminated requirements created under a yellow hat. After this reduction in scope, they needed to think under a green hat which encourages creative and expansive thinking. As a result, the participants found it challenging to shift from a limitation mindset to one that fosters innovation and exploration of new ideas.

Availability of the participants: Although many members of the community expressed interest in participating in the workshops because they cared about the project, finding a suitable time slot for a group of participants was more challenging than organizing an interview with a single interviewee. In our experience, the participants were

willing to compromise due to their willingness to benefit from the output of the workshops. For projects where the participants do not show strong support, persuading the participants may require extrinsic motivation.

Limitations of the Method: Using the Six Thinking Hats method in requirements elicitation requires prior knowledge of this method. The protocol demands a structure different than a typical elicitation workshop. Another limitation is related to group size. As the number of participants increases, assuring everyone contributes from each perspective by using each hat becomes challenging. In such cases, the facilitator may struggle to maintain balance, and workshops require much more time. The protocol does not address any problems that may arise due to the group dynamics, which is another limitation.

### 4.2.2. Lessons Learned

The skills of the facilitator: A key lesson learned is that the facilitator's skills greatly influence the effectiveness of the workshops. Given the frequent challenges the participants faced in adhering to the boundaries of each hat, careful guidance from the facilitator proved essential. The facilitator plays a critical role in steering the participants' thinking within the specific objectives of each hat while maintaining the motivation of participants and encouraging engagement. Observations indicated that immediate correction or pointing out unsuitability often led to decreased participant confidence and reduced engagement in subsequent discussions. To mitigate this, rather than issuing direct warnings, the facilitator focused on consistently clarifying and reiterating each hat's purpose throughout the session, fostering a supportive environment for participant contributions.

Increasing participation by structured turn-taking: The structured rotation of speaking order was a successful addition to the workshop. This approach ensured that each person had a chance to share their ideas. By implementing this strategy, we effectively mitigated the potential risk of unintentional manipulation within the

conversation. Participants responded positively to this adjustment, with one individual noting that the organized turn-taking encouraged more balanced engagement and made it easier to follow the flow of conversation.

Promoting note-taking: When participants did not take notes during their thinking period, they often struggled to recall their initial ideas. This meant that, instead of introducing fresh insights, they tended to respond mainly to the most recent speaker, leading to a loss of valuable ideas. To address this, the facilitator should emphasize the importance of note-taking, helping participants capture and preserve their thoughts for the discussion.

Providing visual cues: During the first workshop, we observed that participants had a tendency to forget the current hat color and often diverged from its strategy. The participants frequently inquired about which hat was in use and needed reminders about the suitability of their contributions within the hat's framework. To address this, we displayed an image of the current hat on the screen in the subsequent workshops to help participants stay focused. Additionally, the facilitator noticed that all participants consulted the cheat sheet during their thinking time. The cheat sheet and the visual cues for the thinking hats proved valuable for the effectiveness of the workshops.

Self-Reflection: One key lesson from the workshops is the significance of self-reflection, particularly during the blue hat phase. When participants wore the blue hat, they reviewed their performance and assessed how comprehensively they had addressed the essential aspects of the application. In half of the workshops, the participants expressed their desire to revisit the white hat stage, recognizing that they had not adequately considered certain fundamental elements. This reflective helped participants to identify gaps without external guidance.

Stakeholder expertise: The only workshop where privacy and other legal issues were discussed was the latest workshop. Although the workshops bring together a heterogeneous set of participants, it is not possible to ensure completeness without

additional consultations with subject matter experts.

### 4.2.3. Implications for The Industry

Participant selection: The contributions of the participants directly reflect their relations with the project at hand. We observed that the participants who have previously organized such tournaments commented on features regarding the organizations, while the participants who have only participated in tournaments as players discussed features related to player needs. Additionally, the participants who are experts in certain non-functional requirements such as performance and compliance raised issues regarding these. The participants should be selected from a heterogeneous base to increase the coverage of functional and non-functional requirements.

Order of participant groups: For practitioners conducting multiple requirements elicitation sessions, beginning with stakeholders who are directly involved in or reliant on the system-to-be's core functionalities can be especially beneficial. Involving such stakeholders early on can reveal critical operational requirements and provide a clear picture of the core features of the system to be addressed. This approach allows the establishment of a solid foundation based on practical, essential features. Subsequent sessions with a broader set of participants can explore additional perspectives, offering insights into user engagement, usability enhancements, and supplementary features, ultimately leading to a well-rounded and comprehensive requirements set.

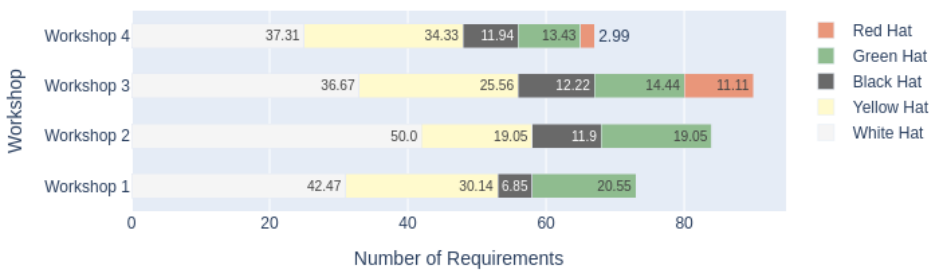


Figure 4.1. Number of elicited requirements per hat across workshops.

Co-creation: The workshops foster the co-creation process of the heterogeneous stakeholder groups by allowing them to express their ideas from various perspectives. We observed that the workshops encouraged the participants to critically approach an idea from various angles rather than simply accepting or dismissing it. As a result, the participants embraced conflicting ideas, allowing for the natural elimination of an idea through further discussion without resorting to conflict or blame. The participants stated their feelings and negative evaluations on topics without the influence or pressure of others. By employing this method, organizations can reduce stress and create an environment where individuals feel empowered to express their genuine ideas.

## 4.3. Creating Baseline Dataset

The primary objective is to assess the creativity of the LLMs by determining how their outputs differ from those of the human experiments. To achieve this, we construct a superset of the requirements derived from the four interviews that were conducted. Figure 4.1 presents the number of requirements elicited per hat across the workshops.

Naturally, the workshops include overlapping requirements, as we discuss around similar topics related to the table tennis tournament application. Furthermore, since the workshops were conducted at different times, the extracted requirements may express similar concepts using different wording. This variation necessitates the use of semantic similarity search to identify and group requirements that convey the same meaning despite differences in phrasing.

Table 4.2. Comparison of Human Requirements and Filtered Subset.

| **Index Name** | **Number of Elements** | **Percentage of Total** |
|---|---|---|
| `human-requirements` | 314 | 100% |
| `human-requirements-subset` | 281 | 89.5% |

We constructed a Pinecone vector database to enable a similarity search between requirements. We converted the text data into vector embeddings using the text-embedding-3-small model, a widely recognized model for generating high-quality embeddings. I can create embeddings with 1536 dimensions. Two indexes were created on the Pinecone vector database: human-requirements and human-requirements-subset. The dimensionality of the indexes was set as 1536 accordingly. Additionally, cosine similarity was chosen as the metric for querying the vector indexes.

**Algorithm 1** The algorithm belongs to subset creation with the output of workshops.

**Input:** List of human-generated requirements
**Output:** Filtered subset of unique requirements
Initialize Pinecone index `human-requirements`
**for** each requirement in human-generated list **do**
    `req_vector` ← `vectorized(requirement)`
    Insert `req_vector` into `human-requirements`
**end for**
Initialize Pinecone index `human-requirements-subset`
**for** each new requirement in human-generated list **do**
    `new_req_vector` ← `vectorized(new_requirement)`
    `similar_results` ← similarity_search(`new_req_vector`, `human-requirements`, top=3)
    **if** no similar requirements found **then**
        Add `new_requirement` to `human-requirements-subset`
    **else**
        **for** each result in `similar_results` **do**
            `similarity_score` ← cosine_similarity(`new_req_vector`, result)
            **if** `similarity_score` $< 0.80$ **then**
                Add `new_requirement` to `human-requirements-subset`
            **end if**
        **end for**
    **end if**
**end for**

The pseudo-code of the operation is given at Algorithm 1. First, we build the human-requirements index by adding the embeddings of all human-generated requirements. Each human-generated requirement is converted into its corresponding vector and stored individually in the human-requirements index. Next, we build the second index called human-requirements-subset. For each requirement, we convert it into a vector and perform a similarity search in the human-requirements index, retrieving the top three most similar results with their similarity scores. If no similar results are found, the requirement is added directly to the human-requirements-subset index. In cases where similar requirements are detected, we check the cosine similarity score between the new requirement and the existing results. The requirement is added to the human-requirements-subset index if the cosine similarity score is below 0.80. This process ensures that unique or sufficiently distinct requirements are retained while eliminating duplicate or highly similar entries. The table 4.2 shows the number of elements in both indexes after these iterations.

Using similarity search, we can also identify which hats were used to express the same requirement across different workshops. Figure 4.3 illustrates the overlapping requirements and their corresponding hats. While requirements that stem from a single source are often attributed to the green hat—commonly associated with creativity—other hats are also capable of producing innovative ideas. For instance, the idea of creating an avatar for a profile photo emerged uniquely under the red hat.

## 4.4. Topic Analysis

We employed topic modeling to analyze the stakeholders' main concerns. We started with applying text preprocessing to the requirements involved in converting the text to lowercase, removing special characters and numbers, and applying tokenization and lemmatization to exclude stop words and predefined terms like "system" and "user." We use the KMeans model with a random state of 42 and set the number of clusters to 8 as the clustering algorithm for BERTopic [71], ensuring that every data point is assigned to one of the eight clusters.Then, we compared the results to

understand the impact of each thinking strategy on the creativity of the results.

Table 4.3. The table for the list of requirements and the information related to the hat discussion in which each requirement emerged.

| **Requirement** | **Workshop 1** | **Workshop 2** | **Workshop 3** | **Workshop 4** |
|---|---|---|---|---|
| The system shall detect and filter inappropriate content, hate speech, and racist remarks on comment-enabled pages | green hat | green hat | black hat | black hat |
| The system shall provide a comment section on the match page where viewers can leave comments and interact with other users | yellow hat | white hat | black hat | yellow hat |
| The system shall generate a heatmap for each match, and this heatmap shall be saved at the end of the match | X | green hat | X | green hat |
| The system shall provide synchronization with calendar applications | X | white hat | X | yellow hat |
| The system shall provide a one-time reentry opportunity for players who belong to the upper levels when they already eliminated from tournament | green hat | X | X | X |
| The system shall provide a page that allows users to create an avatar for their profile picture | X | X | red hat | X |
| The system shall support integration with a virtual table tennis game within the app such as Nintendo | X | X | X | green hat |

We applied topic analysis to the requirements from all workshops and conducted separate analyses of the topics discussed in each workshop and the five thinking hats.

### 4.4.1. Whole Dataset

Figure 4.2 illustrates the results belonging to the whole dataset. Some topics in the figure were discussed in all four workshops, while others did not appear in every workshop.

According to the topic analysis, the first subject that everyone produced requirements in every workshop was organizing and managing a tournament, as expected. This includes creating a league, adding players to the league, generating initial matches, and having a tournament fixture. The workshop participants agree that there needs to be an organizer for the tournament who may also participate as a player in the event. Statistics was another topic mentioned in every workshop, although it was expressed in different words. While some workshops only mentioned basic metrics such as match duration, number of matches played, and the success rate, others stated that the system should produce different statistics for the match, the player, and the tournament. Additionally, there was a discussion about who should take responsibility for recording the match scores due to security concerns.

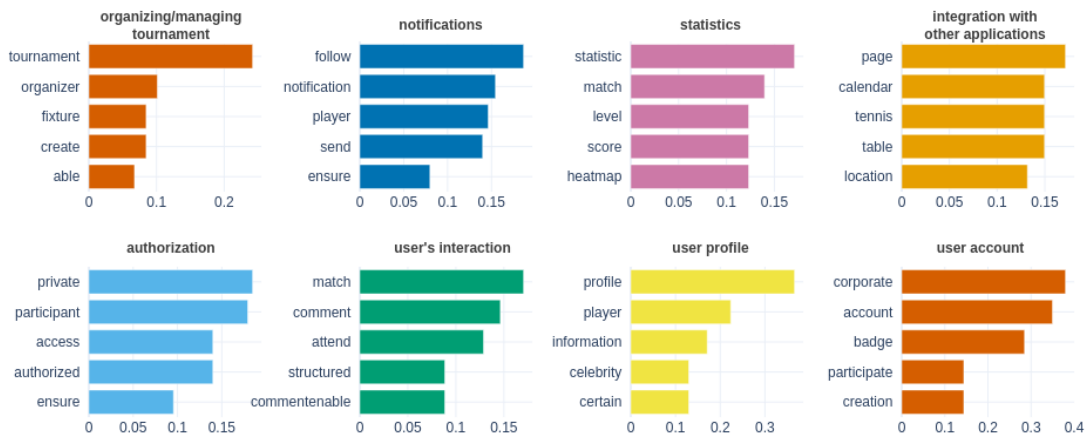


Figure 4.2. The topics gathered through all workshops.

Topics regarding user accounts were discussed. Regarding user account types, each workshop suggested a different structure. In some, workshop user accounts are divided as just a public/private account, sometimes player/spectator, and sometimes even the tournament organizer was designed as a different account. Another distinct idea is to establish a corporate account. It was also discussed whether it would be possible to access the application without opening an account. This topic was followed by features regarding a profile page where users could enter their information. The

discussion also covered what would be displayed on the profile.

Notifications of the application are also discussed, sometimes in one sentence and sometimes in more detail. The discussion revolved around several topics, including notification sound, timing for sending notifications, and options for turning off or muting notifications. At some workshops, it was sufficient to receive notifications solely about our own matches, while in some others, it was preferred to receive updates regarding matches involving our friends or followed tennis players.

User interaction was another topic that participants approached differently. Each group proposed distinct ideas regarding user communication and the ability to comment on a match or tournament. They talked about various communication options, including interactions between players, communication between spectators, and message channels between organizers and players, such as via direct message or e-mail. While some participants supported all kinds of interactions, others argued that they should be restricted.

The integration with other applications also took place in the workshops. Participants mentioned calendar integration to get notified and using location services to search nearby players. One of the workshops talked about importing data from outside, such as previous tournaments' history kept externally in CSV format, and another talked about the feature of integrating with virtual reality table tennis applications to keep track of statistics. One workshop even put forward the idea of creating an API to open application data to the externals.

### 4.4.2. Per Workshop

We provide an overview of the key points discussed during the workshops. This section highlights how the requirements developed from one workshop to the next.

4.4.2.1. The First Workshop. The participants are our potential end users with prior experience using tournament management applications for different sports but follow table tennis matches mainly as spectators. Similar to the other workshops, discussions also covered features related to tournament management, user and match statistics, and user accounts. However, distinct requirements emerged in this session, such as adding a license ID for professional players on the login page, having a level for players, and the dynamic adjustment of these player levels based on performance. A unique feature was representing eliminated players through animations on the tournament page.

The topics specified in this workshop contributed to developing features tailored for users who wish to follow tournaments but do not play in matches. During the session, participants discussed enabling live streaming of matches within the application. Unlike the other workshops, they explored how spectators could chat during a live match or on the match page after the game. As a result, specific requirements emerged, such as sending emojis through chat, muting the chat page, and including predefined template messages.

4.4.2.2. The Second Workshop. This group comprises end-users regularly participating in the annual university table tennis tournament. Setting the match dates and times was the most extensively discussed topic in this workshop because these are the most time-consuming tasks while managing a tournament for an amateur community. In this context, they talked about several points. It was suggested that the tournament organizer set hard or soft deadlines for match completion once the tournament fixtures were determined. It was obvious that participants tried to solve issues that they had previously encountered, such as charging penalties on players who did not complete their matches before the hard deadline. Additionally, the participants aimed to replace the methods they had previously used. They requested a feature in the application that allows proposing match times to their opponents instead of messaging or sending calendar invitations to coordinate match times.

This group had been organizing tournaments for several years and kept records of past events. Because of that, they considered potential features for importing previous tournament data into the application. They were the only group to discuss how to handle tiebreaker scenarios despite it being a fundamental question for a tournament application. They proposed utilizing statistics and historical tournament data to resolve ties. Another critical topic was determining who would enter match scores into the application and how the score would be verified. Additionally, security concerns, particularly regarding user permissions and access control, were also addressed within this group.

4.4.2.3. The Third Workshop. It was conducted with undergraduate and graduate students without prior knowledge of table tennis. Like the first workshop group, they contributed valuable requirements for potential tournament spectators. One word exists as a notable aspect in the topic analysis, which is the word "owl". This word originates from the random word chosen during the green hat exercise for the random word association game. Unlike other participants, this group embraced the randomly assigned word and explored ways to integrate it into the application as a mascot, animation, or other visual elements. This engagement facilitated the generation of creative requirements under the green hat framework.

4.4.2.4. The Fourth Workshop. Workshop 4 consists of software developers, project managers, and managers from the same company who are also amateur table tennis players. While the topics discussed in this workshop did not differ significantly from the others, the group provided insights from their professional experiences. For instance, the idea of a corporate account only emerged from this group. Participants suggested earning a commission from these accounts by letting them conduct a national tournament.

The group also suggested that the application should synchronize with calendar apps to send match reminders. They proposed a feature based on their professional

habit of managing meetings through calendar applications. Additionally, unique ideas such as integrating with virtual reality-based table tennis games and displaying table tennis venues on a map emerged as some of the most creative requirements from this group.

### 4.4.3. Per Hat

Analyzing separate topics to understand what participants discuss under different thinking hats yielded both expected and unexpected results. The unexpected findings primarily indicate misalignment with the intended direction of the hat. In general, features related to playing the game were discussed under the white and yellow hats, while features related to watching games, such as live streaming and spectator accounts, were explored under the green hat. Under the white hat, participants focused on tournament management, scheduling, leagues, match statistics, user authorization, chat functionality, and user avatars. As anticipated, most of these topics pertain to the core features of the application. However, the classification of the chat feature as a core or supplementary function remains debatable. This may suggest that extensive discussions under the white hat could create features that are not necessarily fundamental to the system.

Under the green hat, participants suggested innovative ideas, including setting up cameras to track players' movements and analyze gameplay and implementing features like heatmaps for statistical analysis. Sometimes, a proposed creative feature is supported with further additions. For instance, after suggesting a new feature called "making magic," they add that it can be used anonymously to ensure privacy. In addition, they talked about mentoring, screen animations, application mascots, and donation features.

Under the black hat, participants were expected to debate potential risks and propose mitigation methods, and they did so correctly. They addressed the risk of getting inappropriate content, such as racist comments, and discussed how to prevent it.

Also, they discussed assigning a referee to mitigate any possible problem and disabling some features, like direct messaging, to protect the user from unwanted situations. Additionally, several non-functional requirements related to the application's reliability, performance, and security emerged. The results show that participants engaged with the black hat thinking style and focused on relevant topics within this framework.

Features enhancing user experience were discussed under the red and yellow hats. Under the red hat, participants proposed customizable features, such as modifying the application background and notification sounds. Besides, under the yellow hat, discussions focused on photo-uploading features, including setting a victory pose for matches and creating collage photos for tournaments.

# 5. LARGE LANGUAGE MODEL EXPERIMENTS

We wanted to observe how LLMs, which can assist with various topics in daily life, could contribute to the requirement elicitation process and how successful they would be in applying a method that people can use. This section provides a comprehensive overview of the experiments conducted with Large Language Models (LLMs) during the research process. These experiments were initiated following the conclusion of four requirement elicitation workshops. The objective is to enhance the creativity on the requirement elicitation process by employing the Six Thinking Hats methodology, facilitated by LLM agents. The study explored both single-agent 5.1 and multi-agent 5.2 architectures to evaluate their effectiveness in supporting this approach.

## 5.1. Single-Agent Experiments

The single-agent experiments aim to leverage LLMs in simulating the Six Thinking Hats methodology for requirements elicitation tasks. These experiments concentrate on the five content-generating hats: White, Red, Black, Yellow, and Green. The Blue Hat, primarily responsible for managing and moderating discussions rather than generating requirements, has been intentionally excluded from the experimental scope. For each of the five hats, distinct LLM tasks were designed to reflect their respective cognitive strategies. The experimental framework, illustrated in 5.1, involves a stakeholder agent who can use four different LLMs, utilizing five distinct task definitions tailored to align with the unique perspective of each hat by using three different prompt techniques.

The crewAI framework has two core concepts named agent and task. The agent is an autonomous unit that can perform specific tasks. Eeach agent has a role, a goal, and a backstory definition. Also, the language model that drives its behavior and the specific task to execute are members of the agents. Each task has a description and an expected output. Agents can maintain a long-term or short-term memory. The crewAI

system generates prompts by combining three provided elements: a role, a goal, and a backstory for the agents. The agent prompt is constructed using the following format:

```
"You are {role}. {backstory}. Your personal goal is: {goal}."
```

The task prompt is created using this format:

```
"{task_description}. This is the expect criteria for your final
answer: {task_output} you MUST return the actual complete content
as the final answer, not a summary."
```

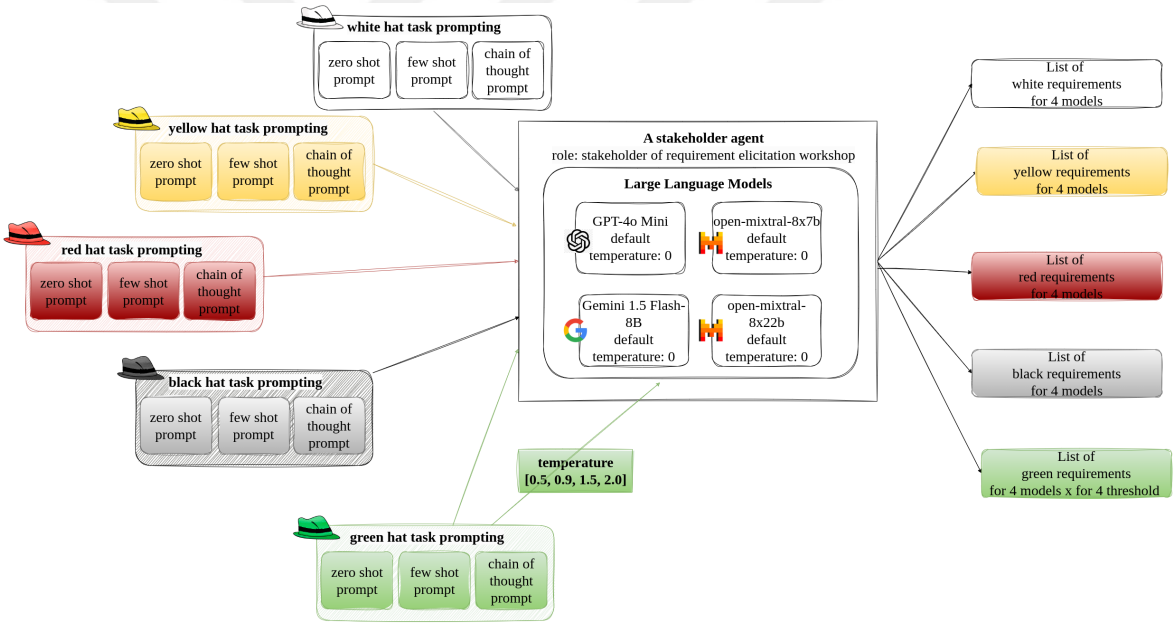


Figure 5.1. The framework for single agent experiments.

In our single-agent experiments, we kept the role, goal, and backstory consistent across all agents. The task descriptions incorporated thinking strategies derived from the Six Thinking Hats framework to guide agent behavior. The only variable parameter among the agents was the choice of language model. Four relatively small yet widely recognized LLMs were utilized. Notably, the agents operated without long-term or short-term memory, ensuring they did not retain outputs from previous tasks when transitioning to new ones. Furthermore, three prominent prompting techniques were

employed to define the task descriptions. The collection of sample prompts used for agents and tasks is provided in Appendix A. The output of the tasks is a list of requirements. Consequently, the agent does not provide any reasoning or explanations regarding the requirements. We ran the same setup 25 times.

For the Green Hat, a distinct parameter configuration is applied by adjusting the temperature setting of the language models. Four different temperature values—0.5, 0.9, 1.5 and 2.0—are utilized for the Green Hat, while the other hats use the default temperature values. This variation in temperature allows for an exploration of how the degree of randomness in the model response influences the generation of requirements within the Green Hat strategy.

## 5.2. Multi-Agent Experiments

In a multi-agent system, multiple agents interact with each other to accomplish tasks. These experiments are operated by CrewAI's sequential task execution process, confirming that tasks are completed in a given order. Under the default CrewAI settings, each task uses only the output of the preceding task as context. However, this experiment aims to investigate how each task influences the others. To achieve this, we adjusted that agents utilize the outputs of all previous agents as context throughout the process. This setup simulates a scenario described in 5.2 where five stakeholders, each putting on a different hat of different colors, which denotes their tasks and thinking styles, are seated around a table, engaging in a sequential discussion, with each participant hearing and building upon the contributions of others.

The discussion spanned three rounds, with agents receiving the previous round's requirements as part of their prompt. Unlike single-agent experiments, agents were instructed to engage in a multi-agent discussion with five participants, avoiding repetition of prior requirements unless presenting a new perspective. The final output is created by collecting the requirements generated as a result of each iteration.

Based on feedback received from participants in the elicitation workshop, we evaluate the impact of different hat orders by incorporating two additional sequences. The feedback identified two main concerns with the original hat order. First, participants struggled to engage with the Green Hat following the Black Hat, as the Black Hat tends to reduce scope and mitigate risks, which limits the creative potential of the Green Hat. Second, placing the Yellow Hat directly after the White Hat was seen as ambiguous, making it harder to differentiate the perspectives. We carefully redesigned the hat orders considering these two challenge. We ensured that the Yellow Hat would not follow the White Hat directly, and the Green Hat would not come immediately after the Black Hat. A significant change was placing the Green Hat at the beginning of one of the sequences. The finalized hat orders are presented in Figure 5.1.

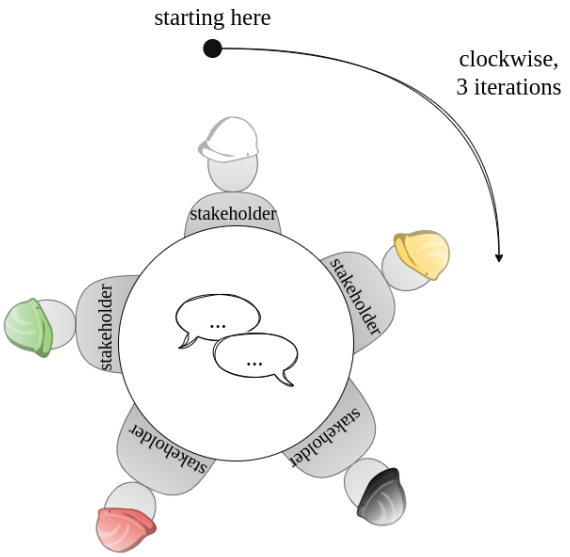


Figure 5.2. The framework for multi-agent experiments.

In another experiment, we adopt the role-playing technique to investigate the effect of assigning different roles to the agents. Rather than using the same stakeholders, we explore how the agents perform when each is assigned a distinct role. The role assignments are illustrated in Figure 5.3. This setup simulates a scenario where individuals from diverse domains, each putting on a different hat of different colors are seated around a table. These participants engage in a sequential discussion, with

each individual hearing and building upon the contributions of others. We aim to select balanced roles to mimic the requirements elicitation process for the table tennis tournament app.

Table 5.1. Hat Orders Used in LLM Workshop Experiments.

| Order Code | Hat Sequence |
|---|---|
| **wybrg (default)** | White → Yellow → Black → Red → Green |
| **wgyrb** | White → Green → Yellow → Red → Black |
| **gwryb** | Green → White → Red → Yellow → Black |
| **brygw** | Black → Red → Yellow → Green → White |
| **ywbgr** | Yellow → White → Black → Green → Red |

To achieve this, we incorporate several key roles. First, we introduce an end user: an amateur table tennis tournament manager who participates in workplace table tennis and is responsible for coordinating the tournaments. Additionally, we include a technical expert, specifically a cybersecurity and data protection specialist. A domain expert is also added—a professional table tennis player. Furthermore, we involve a technical stakeholder, namely a UI/UX designer with expertise in user-centered design. Finally, we include a project sponsor, represented by a startup founder. We chose the role of the startup founder because it has the highest originality score in the work of Lu et al. [62].

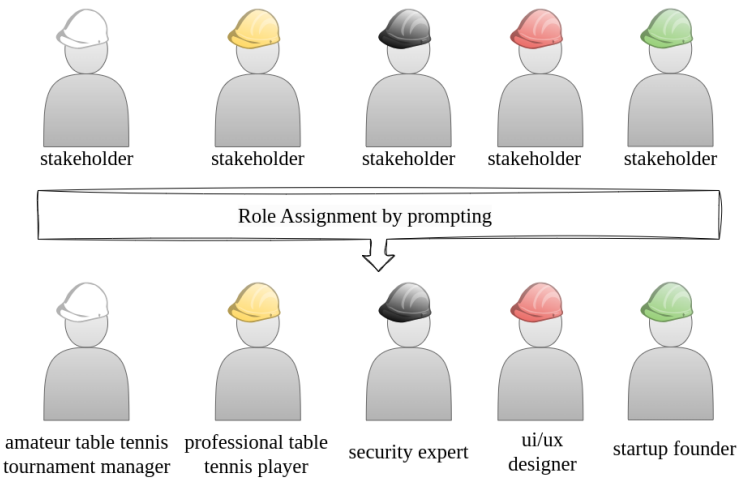


Figure 5.3. Role Assignments to the Agents.

# 6. EVALUATION

The result tables, which present the outcomes of the LLM experiments, will include the same set of columns. The three selected metrics for investigation are:

- Number of Raw Requirements (abbreviated as Raw Req. in the table) to represent fluency.
- Tokens per Requirement (abbreviated as tokens/req) to evaluate model efficiency.
- Novelty, displayed as a percentage, to measure originality.

Additionally, the metric Requirements After Jaccard (Req. after Jaccard) represents the number of requirements remaining after applying Jaccard similarity, providing insights into the potential for generating duplicate requirements in the responses. Lastly, Unique Requirements(unique req.) indicates the number of requirements that differ from those produced in the human workshops, highlighting their distinctiveness.

## 6.1. Statistical Analysis

In this study, I applied each set of experiments 25 times. I use these repeated results to apply statistical tests to understand whether the data reflects an actual relationship or results from randomness. The initial step involved testing the data distribution to determine whether the data followed a normal distribution. The test results revealed that the data did not follow a normal distribution, so I applied non-parametric methods. The following sections outline the procedures employed and the rationale for selecting each test.

### 6.1.1. Normality Assessment

Shapiro-Wilk test [72] is applied to assess the distributional characteristics of the data. The null hypothesis is set for the results of three metrics: Number of Raw Re-

quirements, Novelty, and Requirements per Token. I set the hypothesis as the data follows a normal distribution. The resulting p-values were below 0.05, the conventional significance threshold in all cases. These results led to the rejection of the null hypothesis. These findings indicated that the data significantly deviated from normality.

Although we assess three metrics, the analysis focuses on each metric individually without considering their interrelationships. Therefore, normality was assessed separately for each variable using univariate methods. Multivariate normality testing was not deemed necessary for this study.

### 6.1.2. Comparative Analysis Of Groups

The Friedman test is applied to evaluate whether there were significant differences in the distributions of the Number of Raw Requirements, Novelty, and Requirements per Token. The Friedman test was appropriate for this data because the experiment involved repeated measures, and we had more than two groups. We employed separate setups to assess model performance, prompting technique performance and temperature performance.

### 6.1.3. Post-Hoc Pairwise Comparisons

The Friedman test tells whether there are significant differences between groups but does not show which specific groups differ. So, where the Friedman test indicated statistically significant differences, I conducted post-hoc pairwise comparisons using The Wilcoxon signed rank [73] to specify which conditions differ.

## 6.2. Evaluation Of Single-Agent Experiments

The single-agent experiments involved evaluations of five distinct hats, four different large language models (LLMs), and three diverse prompting methods. For analysis, we calculated the mean values of the number of generated requirements and the nov-

elty percentage by aggregating the results. As the Green Hat introduces an additional variable—temperature—its results are presented separately.

### 6.2.1. Results Of Hats except Green Hat

Table 6.1 presents the results for each model by excluding green hat results, aggregated by their mean values. According to the Wilcoxon signed-rank test, Gemini 1.5 Flash 8b outperforms other models regarding fluency, and GPT-4o-mini outperforms other models regarding token efficiency. However, for the novelty rate, the null hypothesis of the Friedman Test is accepted since the p-value is 0.50, so there is no significant difference in the distribution of novelty rate across the models.

Table 6.1. Aggregated Model Performance Metrics (Mean and Std Scores, Except Green Hat).

| **Model** | **Raw Req.** | | **Req. after Jaccard** | **Unique Req.** | **Novelty (%)** | **Tokens/Req.** | |
|---|---|---|---|---|---|---|---|
| | **Mean** | **Std** | **Mean** | **Mean** | **Mean** | **Mean** | **Std** |
| **gemini-1.5-flash-8b** | **33.59** | 7.12 | 33.47 | 7.52 | 22.40 | 24.22 | 4.91 |
| **gpt-4o-mini** | 22.75 | 4.25 | 22.74 | 5.53 | 24.32 | **22.10** | 3.26 |
| **open-mixtral-8x22b** | 17.13 | 8.16 | 16.18 | 4.25 | 24.81 | 34.82 | 18.31 |
| **open-mixtral-8x7b** | 19.77 | 40.1 | 17.51 | 5.72 | **28.92** | 28.14 | 26.02 |

The GPT-4o-mini and Gemini 1.5 Flash 8b models show reliable performance by achieving a lower standard deviation for tokens per requirement. Mistral models produced fewer requirements by consuming relatively more tokens. This situation indicates that they produce more detailed requirements, meaning less concise outputs.

Table 6.2 summarizes the aggregated performance metrics of different prompt techniques by excluding green hat results. According to the Wilcoxon signed-rank test, zero-shot prompting statistically significantly outperforms other methods regarding fluency and token efficiency. While there is no significant difference between few-shot and chain-of-thought regarding fluency, few-shot performs better than chain-of-thought regarding token efficiency. For the novelty rate, the null hypothesis of the Friedman

Test is accepted since the p-value is 0.38, so there is no significant difference in the distribution of novelty rate across the prompting techniques.

Table 6.2. Aggregated Prompt Performance Metrics (Mean Scores, Except Green Hat).

| **Prompt Name** | **Raw Req.** | | **Req. after Jaccard** | **Unique Req.** | **Novelty (%)** | **Tokens/Req.** | |
|---|---|---|---|---|---|---|---|
| | **Mean** | **Std** | **Mean** | **Mean** | **Mean** | **Mean** | **Std** |
| **zero-shot** | **27.79** | 35.45 | 25.54 | 7.02 | **25.28** | **24.82** | 19.50 |
| **few-shot** | 21.41 | 7.78 | 21.31 | 5.09 | 23.77 | 26.03 | 6.59 |
| **chain-of-thought** | 20.72 | 8.67 | 20.57 | 5.15 | 24.86 | 31.10 | 20.23 |

Table 6.3. Comparison of Requirements Generated Under Different Hats and Prompts.

| **Hat** | **Chain of Thought Examples** | **Zero-Shot Examples** | **Human Workshop Requirements** |
|---|---|---|---|
| **Red Hat** | The app shall offer personalized welcome messages and greetings, tailored to the user's name, potentially incorporating a congratulatory message for past tournament victories to foster a sense of recognition and encouragement. | The app shall present a personalized welcome message upon login to enhance user engagement. | None |
| **Yellow Hat** | The application shall offer integrations with social media platforms for users to share their achievements and invite friends. | The app shall have a social feature for users to share their achievements and progress on social media. | None |
| **Black Hat** | The application shall provide a tutorial or guide for new users to facilitate onboarding and encourage engagement. | None | None |

The few-shot prompt emerges as the most reliable method, exhibiting the lowest standard deviation in both token usage and the number of raw requirements gen-

erated.The Chain-of-Thought (CoT) prompting method has higher token usage and variability, indicating more verbose and inconsistent outputs. As CoT prioritizes logical reasoning, it is usually expected to generate fewer alternative solutions. Table 6.3 shows some unique examples belonging to CoT by comparing them with zero-shot examples.

Table 6.4. Aggregated Model and Temperature Performance Metrics (Mean Scores, Green Hat).

| Model | Temperature | Raw Req. | | Req. after Jaccard | Unique Req. | Novelty (%) | Tokens/Req. | |
|---|---|---|---|---|---|---|---|---|
| | | Mean | Std | Mean | Mean | | Mean | Std |
| **gemini-1.5-flash-8b** | 0.5 | 30.85 | 3.56 | 30.16 | 8.00 | 25.93 | 25.23 | 2.14 |
| | 0.9 | 28.09 | 4.19 | 28.08 | 7.75 | 27.57 | 26.71 | 2.89 |
| | 1.5 | 22.99 | 4.50 | 22.99 | 7.16 | 31.15 | 31.08 | 5.30 |
| | 2.0 | 20.80 | 5.00 | 20.80 | 7.35 | 35.32 | 35.20 | 8.48 |
| **gpt-4o-mini** | 0.5 | 20.95 | 2.85 | 20.95 | 6.07 | 28.96 | 24.37 | 2.35 |
| | 0.9 | 20.55 | 2.87 | 20.55 | 6.48 | 31.54 | 24.73 | 2.39 |
| | 1.5 | 10.31 | 3.44 | 10.31 | 5.35 | 51.88 | 152.55 | 228.59 |
| | 2.0 | 2.84 | 2.57 | 2.84 | 1.89 | 66.67 | 4186.22 | 4358.93 |
| **open-mixtral-8x22b** | 0.5 | 14.40 | 6.07 | 14.05 | 4.00 | 27.78 | 48.69 | 40.88 |
| | 0.9 | 14.61 | 5.77 | 14.52 | 4.59 | 31.39 | 49.16 | 40.80 |
| | 1.5 | 7.79 | 3.19 | 7.76 | 5.49 | 70.55 | 1353.02 | 2668.77 |
| | 2.0 | 14.75 | 7.5 | 14.37 | 3.47 | 23.51 | 52.45 | 47.07 |
| **open-mixtral-8x7b** | 0.5 | 21.04 | 15.32 | 19.93 | 5.39 | 25.60 | 33.5 | 17.27 |
| | 0.9 | 15.83 | 5.33 | 15.76 | 3.81 | 24.09 | 38.26 | 38.90 |
| | 1.5 | 9.67 | 4.78 | 9.67 | 4.00 | 41.38 | 441.97 | 1088.04 |
| | 2.0 | 16.31 | 5.57 | 16.20 | 4.24 | 26.00 | 33.22 | 16.61 |

### 6.2.2. Results Of Green Hat

For green hat, both model and prompt performance metrics must be interpreted in relation to the temperature values. Before delving into analysis, we observe a notable misbehavior in the agent's output when temperatures 1.5 and 2 are used. The problem arises because the agent's response initially includes valid requirements but subsequently generates unrelated symbols, words from various languages, and nonsensical characters. When this issue is detected, only the first valid requirement produced by the model is considered in the analysis. While this adjustment inflates the novelty

scores, these results are unreliable due to the irregular behavior.

Table 6.4 presents the results of the green hat evaluation by different models. This table provides evidence of this issue, as models except for gemini-1.5-flash-8b exhibit problematic tokens-per-requirement scores with very high standard deviation. That means the problem arises here. In contrast, gemini-1.5-flash-8b emerges as the most reliable model, even when evaluated at higher temperatures. Other models require careful temperature tuning to manage token usage.

Table 6.5. Aggregated Prompt and Temperature Performance Metrics (Mean Scores, Green Hat).

| Prompt Name | Temperature | Raw Req. | | Req. after Jaccard | Unique Req. | Novelty (%) | Tokens/Req. | |
|---|---|---|---|---|---|---|---|---|
| | | Mean | Std | Mean | Mean | | Mean | Std |
| **zero-shot** | 0.5 | 26.53 | 12.81 | 25.45 | 6.68 | 25.18 | 27.43 | 14.49 |
| | 0.9 | 23.45 | 6.14 | 23.41 | 5.97 | 25.46 | 26.17 | 4.40 |
| | 1.5 | 13.87 | 7.81 | 13.87 | 5.63 | 40.59 | 309.03 | 1154.8 |
| | 2.0 | 16.99 | 9.71 | 16.38 | 4.89 | 28.78 | 1160.80 | 3041.38 |
| **few-shot** | 0.5 | 20.24 | 7.29 | 19.89 | 5.21 | 25.74 | 27.42 | 3.78 |
| | 0.9 | 19.01 | 5.42 | 18.92 | 5.13 | 26.99 | 28.47 | 5.29 |
| | 1.5 | 12.93 | 6.59 | 12.93 | 5.11 | 39.52 | 225.32 | 508.48 |
| | 2.0 | 12.82 | 7.29 | 12.82 | 4.12 | 32.14 | 935.18 | 2601.41 |
| **chain-of-thought** | 0.5 | 18.66 | 8.43 | 18.48 | 5.70 | 30.55 | 43.99 | 36.87 |
| | 0.9 | 16.85 | 7.77 | 16.85 | 5.87 | 34.84 | 49.52 | 47.91 |
| | 1.5 | 11.26 | 7.10 | 11.24 | 5.76 | 51.15 | 949.61 | 2269.41 |
| | 2.0 | 11.21 | 7.54 | 5.76 | 3.70 | 33.01 | 1134.35 | 2814.71 |

The increase in temperature directly affects randomness, which in turn boosts creativity. The results for the novelty ratio indicate that setting the temperature as 2.0 significantly outperforms other temperature values for GPT-4o-mini and Gemini models. However, for Mixtral models, temperature 1.5 outperforms other temperature values.

Table 6.5 presents the results of the green hat evaluation by different prompts. The zero-shot prompting method emerges as the most efficient and fluent approach

with all temperatures. Table 6.6 demonstrates the influence of temperature settings on the green hat outputs. It is important to note that models often produce overlapping requirements with themselves. For instance, Gemini 1.5 Flash 8b generated a similar requirement related to Virtual Reality when using a temperature of 0.5. It also emerged during one of the human workshops, phrased as "the app shall support virtual reality games.".

Table 6.6. Comparison of Requirements Generated Under Green Hats.

| **Requirement Example** | **Model Name** | **Prompt Name** | **temperature** |
|---|---|---|---|
| The app shall offer a coaching feature, with AI-powered analysis of players' techniques and personalized training programs. | Mixtral 8x22b | Zero Shot | 0.5 |
| The system shall offer a virtual reality (VR) training mode where users can practice against virtual opponents with adjustable skill levels, providing immersive and interactive training. | Gemini 1.5 Flash 8b | Few Shot | 0.9 |
| The app shall include eco-actions that support carbon neutral tournaments to map players stemming-effects. | GPT-4o-mini | Chain of Thought | 1.5 |
| The app shall offer a 'Ghost Mode', enabling players to virtually practice against past matches, visualizing their opponents' moves. | Mixtral 8x7b | Chain of Thought | 2.0 |

## 6.3. Evaluation Of Multi-Agent Experiments

In the single-agent experiments, agents generated requirements independently from each other. However, when agents build upon each other's ideas, the potential to enhance the quality of the generated requirements arises. We created a new setup for the multi-agent LLM experiment to expose this potential and mimic the dynamics of human workshops.

The multi-agent experiments aim to evaluate the impact of simulating a real workshop environment on the generated requirements. Table 6.7 presents the number of requirements and other metrics generated based on the utilized hat order and models. It is important to note that requirements were collected after each iteration, and the final requirements list was compiled from the outputs of all three iterations. In this way, three iterations of requirement conversations were collected for 25 iterations. The results of the Wilcoxon signed-rank test for novelty rate indicated that the sequence gwryb significantly outperforms other hat orders for the GPT 4o mini model. There is no significant difference in the distribution of novelty rate across the hat orders for Mixtral models. There is no best hat sequence for Gemini 1.5 Flash 8b, but the sequence wgyrb performs significantly worse than other orders.

Table 6.7. Results for Different Models and Hat Orders.

| Model | Hat Order | Raw Req. | Req. after Jaccard | Unique Req. | Novelty (%) |
|---|---|---|---|---|---|
| **gemini-1.5-flash-8b** | **wybrg-(default)** | 329.92 | 318.72 | 37.48 | 11.36 |
| | wgyrb | 314.4 | 290.28 | 26.16 | 8.32 |
| | gwryb | 252.16 | 240.48 | 50.72 | 20.11 |
| | brygw | 271.8 | 264.56 | 41.96 | 15.44 |
| | ywbgr | 273.88 | 266.76 | 49.12 | 17.93 |
| **gpt-4o-mini** | **wybrg-(default)** | 271.96 | 271.96 | 123.92 | 45.57 |
| | wgyrb | 252.56 | 252.56 | 106.76 | 42.27 |
| | gwryb | 222.52 | 222.48 | 113.72 | 51.11 |
| | brygw | 219.36 | 219.36 | 99.20 | 45.22 |
| | ywbgr | 261.64 | 261.60 | 124.64 | 47.64 |
| **open-mixtral-8x22b** | **wybrg-(default)** | 167.20 | 143.8 | 54.04 | 32.32 |
| | wgyrb | 180.88 | 157.56 | 59.24 | 32.75 |
| | gwryb | 175.88 | 135.68 | 65.20 | 37.07 |
| | brygw | 203.12 | 158.52 | 78.80 | 38.79 |
| | ywbgr | 194.32 | 146.4 | 66.68 | 34.31 |
| **open-mixtral-8x7b** | **wybrg-(default)** | 302.84 | 260.56 | 90.44 | 29.86 |
| | wgyrb | 361.48 | 232.2 | 89.60 | 24.79 |
| | gwryb | 234.84 | 220.64 | 64.48 | 27.46 |
| | brygw | 288.0 | 238.24 | 87.24 | 30.29 |
| | ywbgr | 280.52 | 258.48 | 92.40 | 32.94 |

Figure 6.1 shows the number of raw and unique requirements generated under each hat. The most notable finding observed here is that, for each sequence, the LLMs generated a greater number of requirements for the hat that was encountered first. The position of the black and red hats within the sequence—whether at the end or in the middle—did not appear to significantly affect the number of requirements produced. However, when the white hat was placed at the end of the sequence, there was a substantial decrease in the number of requirements generated. The black and red hats exhibited similar novelty levels regardless of whether they appeared at the beginning or the end of the sequence.

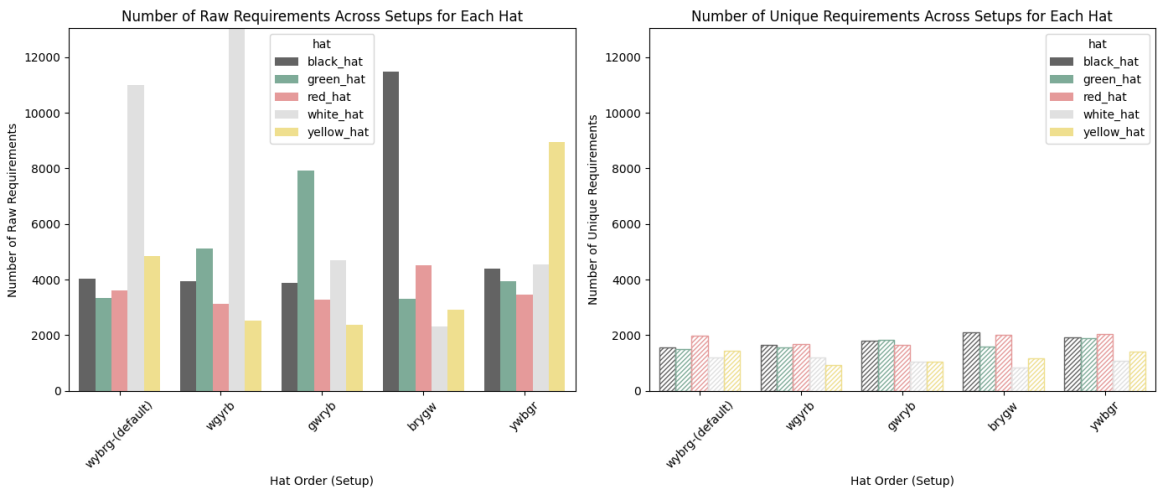


Figure 6.1. Table showing the number of raw and unique requirements generated by each hat color when applied in different orders. The left column represents raw requirements, while the right column represents unique requirements.

Figure 6.2 shows the breakdown of the final requirements obtained from discussions conducted with different hat. LLM workshops have proven to be effective and consistent in generating black hat requirements, with LLMs producing more black hat requirements compared to human workshops. Across all experiments, regardless of the order, LLM workshops consistently yield a higher proportion of black hat requirements than human workshops. This suggests that LLMs may be more adept at generating requirements that involve negative or oppositional thinking.

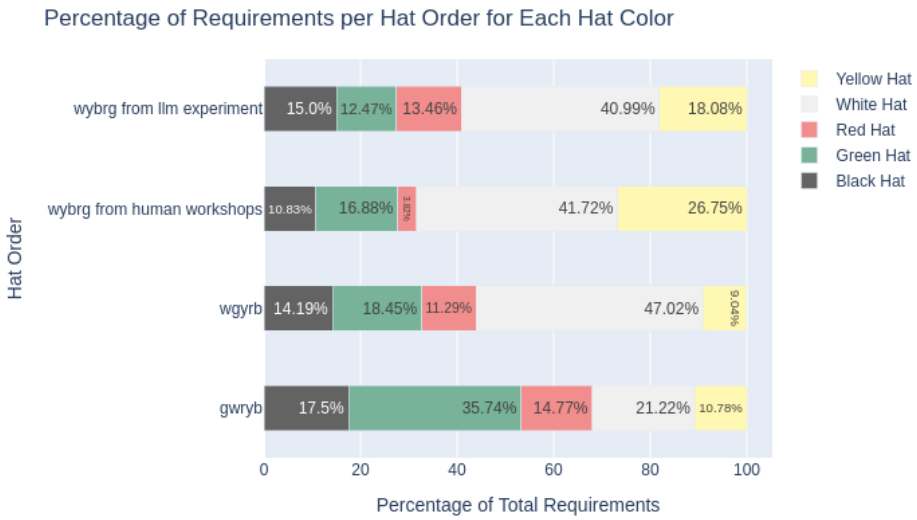


Figure 6.2. Number of generated raw requirement per hat across discussions using different hat orders.

Additionally, LLM workshops produce more red hat requirements than human workshops, which is expected since humans generally struggle to generate red hat requirements. In contrast, humans tend to generate more creative, positive, and factual requirements within the same order. Overall, LLM workshops demonstrate the most balanced distribution across the hat colors, with the highest proportion of requirements belonging to the green hat, which is associated with fostering creativity.

Table 6.8 compares the metrics of single-agent and multi-agent experiments at both the hat and model levels. Notably, only the Gemini model shows a decrease in creativity when working in a multi-agent setup, whereas for other models, the multi-agent setup results in an increase in both the number of generated requirements and creativity.

The novelty rates of black, red and green hats are relatively higher than other hats and consistently higher in the multi-agent setup than in the single-agent setup, particularly for GPT-4o-mini. This suggests that multi-agent collaboration enhances the generation of critical or oppositional thinking requirements. The largest boost in novelty is observed in the red hat, where the novelty increases from 39.70% to 72.60%, highlighting the impact of agents' collaboration on generating creative requirements.

Table 6.8. Single Agent and Multi Agent Novelty Metrics for Different Models and Hat.

| Hat | Model | Single Agent Novelty Rate (%) | Multi-Agent Novelty Rate (%) |
|---|---|---|---|
| **Black Hat** | **gemini-1.5-flash-8b** | 29.33 | 26.18 |
| | **gpt-4o-mini** | 26.41 | 46.95 |
| | **open-mixtral-8x22b** | 32.49 | 43.65 |
| | **open-mixtral-8x7b** | 33.52 | 42.89 |
| **Green Hat** | **gemini-1.5-flash-8b** | 29.45 | 25.99 |
| | **gpt-4o-mini** | 36.21 | 50.15 |
| | **open-mixtral-8x22b** | 34.04 | 41.76 |
| | **open-mixtral-8x7b** | 27.75 | 42.08 |
| **Red Hat** | **gemini-1.5-flash-8b** | 28.76 | 28.98 |
| | **gpt-4o-mini** | 39.70 | 72.60 |
| | **open-mixtral-8x22b** | 30.55 | 46.49 |
| | **open-mixtral-8x7b** | 29.25 | 53.83 |
| **White Hat** | **gemini-1.5-flash-8b** | 16.38 | 15.17 |
| | **gpt-4o-mini** | 13.15 | 25.77 |
| | **open-mixtral-8x22b** | 18.13 | 28.60 |
| | **open-mixtral-8x7b** | 14.11 | 28.81 |
| **Yellow Hat** | **gemini-1.5-flash-8b** | 16.63 | 17.84 |
| | **gpt-4o-mini** | 20.71 | 42.74 |
| | **open-mixtral-8x22b** | 17.41 | 33.92 |
| | **open-mixtral-8x7b** | 34.68 | 33.66 |

## 6.4. Evaluation Of Multi-Agent Experiments With Role Definitions

Another experiment was conducted in a multi-agent setup using roles for the agents. This role-playing setup was implemented using the default hat order, which was also used in human workshops, and was run with only gpt-4o-mini. The choice of GPT-4o-mini was made because, at the 0.5 temperature, it proves to be more consistent, reliable, and efficient.

Table 6.9 presents a comparison between role-based and non-role-based experiments. Since the role-based study was conducted exclusively using the default order and GPT 4o mini, the non-role-based experiments were evaluated using 25 runs con-

ducted with GPT 4o mini under the default hat order. This comparison aims to assess the impact of role assignment on performance. Role-playing significantly increased the novelty rate for yellow and black hats. While no significant change occurred for white and green hats, the novelty significantly decreased for the red hat.

We can argue about the role and hat relationships in the role-playing setup. For instance, the black hat had a particularly significant impact on the novelty rate. This indicates that the role of a security expert aligns well with the black hat, further enhancing the generation of critical or oppositional thinking requirements. When we consider the red hat, we claim that the UI/UX designer is not the most suitable role to play the red hat.

Table 6.9. Single Agent and Multi-Agent Metrics for Different Roles.

| **Role** | **Hat** | **With role playing** | | **Without role playing** | |
|---|---|---|---|---|---|
| | | **Num of Req.** | **Novelty (%)** | **Num of Req.** | **Novelty (%)** |
| **Amateur Table Tenis Tournament Manager** | **White Hat** | 63.64 | 17.52 | 69.80 | 29.68 |
| **Professional Table Tenis Player** | **Yellow Hat** | 52.80 | **57.99** | 49.96 | 54.97 |
| **Security Expert** | **Black Hat** | 56.28 | **78.72** | 52.28 | 57.83 |
| **UI/UX Designer** | **Red Hat** | 51.48 | 45.91 | 50.96 | 76.1 |
| **Startup Founder** | **Green Hat** | 58.08 | **58.95** | 48.96 | 44.18 |

## 6.5. LLM Topic Analysis

We also applied topic analysis to a mix of selected LLM iterations, trying to map out which topics LLMs discussed under each hat.

The requirements generated under the white hat encompass essential elements such as the organization of tournaments, the configuration of match settings, and the schedule setting up. These are very similar topics to what people talk about. Except

for overlapping topics, LLMs discuss payment methods, building a community, and personalized training plans. These topics are mostly the values-driven requirements people discuss under the yellow hat, which are not counted as core features of the application.

LLMs have brought up a wider variety of topics under the red hat than humans. They suggested the expected things like customizable themes and backgrounds, but they also had some interesting ideas to increase engagement and reward. For instance, picking a "Player of the Week" or adding a journal page where people can share how they got into table tennis. They also pointed out the need for features related to mental health, meditation, and motivation.

The black hat requirements focused on solutions for potential risks, such as secure payment processing, protecting registered user information, and automatically blocking and reporting inappropriate content. There were also discussions about features like bug reporting and live support for issues users might encounter in the app. However, topics like statistics, training, and coaching were also mentioned, though they weren't directly related to any risks.

Under the green and yellow hats, LLMs discussed very similar topics. These included statistics-based training sessions, the ability to create tournaments with custom rules, syncing with calendars, and integrating virtual reality, among other features. While these are features that people also mentioned, the LLMs came up with unique ideas like virtual match parties and mystery tournaments.

# 7. DISCUSSION

This chapter presents an analysis of the findings from the workshop protocol and the workshops conducted following the developed protocol. It also discusses the results of evaluating the LLM experiments, offering insights into their performance and impact on the requirement elicitation process.

## 7.1. Findings

We explain the protocol we devised following the Six Thinking Hats creativity method to elicit requirements through requirements workshops and our experience from running four workshops with the devised protocol. We successfully followed the protocol for the workshops and introduced improvements as we faced issues. Similar to requirements elicitation interviews, the skills of the facilitator are a key factor for the success of the workshops. Another key factor is the variety of the stakeholders and the order of the workshops with certain stakeholder groups. Our verdict is that our protocol is an effective way to bring a heterogeneous set of stakeholders together to co-create a system-to-be by expressing their ideas freely, evaluating them from specific perspectives, and fostering creativity.

This study explores three approaches to implementing the Six Thinking Hat method for requirement elicitation through an LLM-based system. It also examines the performance of four language models, three commonly applied prompting methods, and four different temperature values. The results indicate that language models can generate more requirements than humans in a shorter time while also capturing a broader range of requirements on the same topic. These capabilities make LLMs a valuable tool for analysts in the requirement elicitation process.

The zero-shot prompting approach allows single agents to generate the highest number of requirements among the prompting methods. The Chain of Thought (CoT)

prompting reflects the model's internal reasoning process and adds more details about the requirements it generates. In RE, generally, simple and precise requirements are preferred, as detailed requirements tend to be complex and ambiguous. However, CoT offers analysts valuable insights into how the model is formulated and detailing its requirements.

The multi-agent system enhances the novelty of the generated requirements. It helps uncover requirements that may be overlooked in human workshops and produces more reliable results than running a single agent multiple times. This is because the agents in a multi-agent setup are aware of the previous context, which enables them to build upon previous interactions. In contrast, repeatedly calling a single agent often results in similar outputs. However, this improved novelty and reliability come at a cost, as the multi-agent system requires more tokens to use. Additionally, input size restrictions are more likely to occur when context grows. It is a potential risk for the multi-agent system, which can occur in later iterations.

Considering the experiment with the hat orders, when the process begins with the Green Hat and ends with the Black Hat, the GPT 4o mini model gets a significantly higher novelty rate. This finding aligns with the principles of the Double Diamond framework [74]. The LLM workshop increases the volume of generated requirements, starting with the Green Hat, which emphasizes generative and imaginative thinking. Also, the discovery and develop phases of the Double Diamond mirror the same behavior. These approaches prioritize broad exploration and generating diverse ideas, resulting in more novel and innovative requirements. This hat order has also been finalized with the Black Hat. Moreover, this part corresponds to the define and deliver phases of the Double Diamond. The objective is to evaluate, select, and finalize the most viable and impactful solutions by focusing on critical evaluation, which facilitates the narrowing and refinement of ideas.

The experiment investigating optimal hat order revealed that there is no one hat sequence that works best for all. Models generally perform differently in different

sequences. They tend to create more requirements for the first hat.

Ultimately, the impact of role-playing is more significant for certain hats, possibly because the roles align well with the specific thinking strategies associated with those hats. The role of a security expert aligns well with the Black Hat, as the expert's perspective complements the focus on risks and challenges.

LLM experiments demonstrate that LLMs operating under the Black and Red Hats generate more requirements than humans and often uncover creative ideas. We suggest leveraging the outputs of the Black and Red Hats to enhance the creativity of human-generated requirements. Mainly, they perform better than humans in identifying risks and pinpointing requirements that foster users' emotional attachment to the application. This approach helps to enrich the overall quality of the elicitation process and may reduce the need for extra workshops.

## 7.2. Topic Analysis Comparison

Table 7.1 shows the topics discussed under each hat by both humans and language models. Under White Hat thinking, both human participants and LLMs expressed not only core functional requirements but also value-driven requirements that would typically fall under the Yellow Hat. While facilitators and participants in the workshops partially guide and limit such deviations, LLMs were not subjected to similar constraints. Likewise, the topics covered under the Yellow and Green Hats by LLMs were highly similar. However, in the workshops, facilitators and participants occasionally classified certain requirements as creative in nature and redirected them to be discussed under the Green Hat. Regarding the Black and Red Hats, LLMs demonstrated a broader and more diverse range of topics than human participants.

Table 7.1. Topic Comparison of Human and LLM Talks for Different Hats.

| Hat | Human Talks | LLM Talks |
|---|---|---|
| **White Hat** | Tournament management, scheduling, leagues, match statistics, user authorization, chat functionality, user avatars. | Tournament and team creation, match scheduling, match statistics, secure payment, training strategies, community building |
| **Yellow Hat** | Player following and fan accounts, corporate accounts, setting referee, photo uploading features, earning badges | Mental health, feature creating and management, coaching skills, sharing community achievements, friend engagements, local clubs, performance analyzing |
| **Black Hat** | Inappropriate content, assigning a referee, disabling features, non-functional requirements for reliability, performance, and security. | Inappropriate content, secure payment, data security, bug resolving, training progress, sharing content |
| **Red Hat** | Changing application background, notification sounds | Customizable themes, player engagement, rewards. "Player of the Week" recognition, journal page for personal table tennis stories, mental health, meditation, and motivation features. |
| **Green Hat** | Screen animations & mascot, mentoring, generating heatmap, donation features | Realtime match, social reward, spiritual gameplay, community sharing, skill training |

## 7.3. Issues In LLM Output Quality

LLMs are capable of generating requirements significantly faster than human participants. While this speed offers certain advantages, it also introduces a number of potential drawbacks. While the LLMs were able to generate requirements within the framework of the hat they were discussing, they often suggested requirements on various other topics. As a result, very similar requirements may have appeared under different hats. This scenario may be regarded as typical in a single-agent system; however, it was also noted in multi-agent experiments. Even though the LLMs generated more ideas than humans, they performed worse than humans when it came to sticking to the specific thinking framework of each hat.

As another drawback, there were instances where the LLMs generated completely irrelevant and unusable requirements. In particular, the mixtral models produced absurd statements like "The app shall provide a feature to display the tournament's

emotional user volunteer and job opportunities," or even meaningless expressions like "The app shall provide a feature to display the tournament's emotional user user user terms and conditions." The LLMs also showed a tendency to form sentences with different objects linked to the same verb. For example, outputs that begin with "The app shall display" and then present different requirements for displaying policies, statistics, results, and player information.

Another observation is that LLMs tend to be overly verbose. Requirements should be clear and focused on a single system characteristic at a time, but LLMs often try to elaborate as much as possible. Such phrases that LLMs generally use before ending the sentence, like "ensuring participants' safety," "enhancing overall satisfaction," or "mitigating privacy risks," are vague and need to be refined. They are too broad and lack precision; however, these requirements can be made applicable by translating them into actionable specifications.

Another point to consider in LLM outputs is that some requirements are too general. For instance, phrases like "Shall ensure the app is compatible and optimized for various devices and operating systems" or "Shall ensure the app is compliant with relevant data protection regulations, such as GDPR and CCPA" are broad enough to apply to any other application. While these requirements may not be domain-specific, they can still be useful to the analyst as they highlight a potential need for the application.

## 7.4. Limitations

Using the Six Thinking Hats method in requirements elicitation requires prior knowledge of this method. The protocol demands a structure different than a typical elicitation workshop. Another limitation is related to group size. As the number of participants increases, assuring everyone contributes from each perspective by using each hat becomes challenging. In such cases, the facilitator may struggle to maintain balance, and workshops require much more time. The protocol does not address any

problems that may rise due to the group dynamics, which is another limitation.

The study utilized a small subset of available Large Language Models (LLMs) and prompting strategies, focusing on only two commercial and two non-commercial models and the three most common prompting techniques. Given the rapid development of LLMs and the growing range of advanced prompting methodologies, this limited selection may not capture the full spectrum of capabilities or limitations across the field.

Furthermore, this study seeks solutions for a table tennis tournament application. We apply the elicitation process for this application in both user workshops and LLM-based experiments. While this domain provided a concrete and manageable context for evaluation, it represents only one type of application scenario. Applications in other domains like healthcare, finance, or e-commerce may involve different user needs, constraints, and system complexities. Therefore, the findings and conclusions derived from this study may not fully generalize to systems with fundamentally different functionalities, target users, or domain-specific requirements. Consequently, we encouraged the community to apply our methodologies across various domain-specific solutions.

## 7.5. Threats To Validity

This section outlines potential threats that can affect the reliability and generalizability of the study's conclusions. The threats are categorized into three primary types: internal, external, and construct validity.

### 7.5.1. Internal Validity

For all elicitation workshops conducted with humans, introduce I take the business analyst role, which has a great potential to introduce personal bias into the workshop process. Furthermore, I provided ideas regarding the requirements during some of the workshops. The potential for personal bias could affect the internal validity of

this study. However, I adopt a management style and use it for all workshops to be consistent. I tried to be neutral and fair.

Additionally, the prompt strategy for LLM-based research can significantly impact the internal validity of the results. To mitigate this risk, I employed various prompts and analyzed their effects to understand their impact on the outcomes better. To further reduce potential bias, I also involved LLMs in the design of some of the prompts, thereby diversifying the sources and perspectives used in their creation.

### 7.5.2. External Validity

External validity concerns the generalizability of the study's findings. While both the method employed, namely the Six Thinking Hats method and the technique of using large language models (LLMs), are widely applicable and have been tested across various domains, the workshops conducted in this study and the resulting baseline requirements are specific to a single domain—table tennis tournament management. So, we do not claim generality.

### 7.5.3. Construct validity

Construct validity is how well a measurement reflects what it's supposed to measure. In the context of assessing creativity, I have employed several well-known metrics that are commonly used across various domains. Creativity is complex and subjective. Because of this, I have used proxy metrics that may not fully reflect its true depth. I have sought to mitigate this limitation by incorporating multiple metrics rather than relying on a single measure. However, the potential construct validity threats of using proxy metrics must be acknowledged.

# 8. CONCLUSIONS

Requirement elicitation is a significant process of requirements engineering. As the techniques of Natural Language Processing (NLP) advanced, the researchers are actively seeking methods to simplify the requirements elicitation process and to improve the outcomes. This study aims to utilize the Six Thinking Hats methodology, a well-established method for fostering group collaboration. This method helps mitigate potential conflicts and communication barriers among diverse stakeholders.

Upon internalizing the Six Thinking Hats methodology, we devised a plan for how to effectively utilize the hats within the framework to elicit the requirements. When the use of each hat and the hat orders were clarified, we prepared a workshop protocol. While designing this protocol, we ensured that each step was explained in detail and that the protocol could be applicable across various domains as well.

We conducted four engaging workshops, successfully implementing this protocol with 16 participants from various professions and diverse fields. During the workshop, we identified the requirements for an application that would manage table tennis tournaments for individuals who organize these events twice a year. We collect demographic information and workshop feedback from the users at the end of these workshops.

Considering our observations during the workshop and the feedback we received from the participants, we reported the protocol's applicability, the important steps, and the problems that would be experienced. We observed that facilitator skills play a vital role in managing the process. Taking notes throughout the process proved essential in helping participants retain and articulate the requirements that came to mind. Moreover, including a visual representation for each hat proved helpful, as it served as a tangible reminder of the hat's function and significance, making the abstract concept of the framework more accessible.

The four workshop requirements list are collected together to create a baseline dataset. One of two semantically similar requirements was removed from the list. This semantic similarity search was applied to the Pinecone vector database with vector embeddings by calculating semantic distance with Cosine similarity.

This study also leverages Large Language Models (LLMs) to harness their ability to generate contextual text and their inherent creativity, making them valuable assistance in elicitation. GPT-4o-mini, Gemini 1.5 Flash 8b, Open Mixtral 8x7b, and Open Mixtral 8 x 22b models prompted by zero-shot prompting, few-shot prompting, and chain of thoughts prompting. Four different temperatures are also used for LLM configuration for the green hat setup.

Two metrics are created to assess the creativity of the models. The novelty percentage is calculated as the ratio of unique requirements, different from the baseline dataset, to the total number of generated requirements. The tokens per requirement metric are calculated as the rate of output tokens over the total number of generated requirements. The number of generated requirements is also used to represent the model's fluency.

In addition to the single-agent experiments, multi-agent experiments are conducted to evaluate the performance of the agents when they influence each other. This setup allows us to examine how collaboration and interaction impact the overall creativity of the process. In this setup, three different hat usage orders were experienced. Additionally, five different roles were assigned to the agents to assess the effect of role-playing on the creativity of the generated requirements.

Additionally, our Large Language Model experiments indicate that using multi-agent systems and appropriately defining roles aligned with each hat's responsibility can enhance creativity. Specifically, the Black Hat, responsible for managing risks, and the Red Hat, focused on emotions, have the potential to generate more creative requirements compared to humans. Role-playing further amplifies creativity, enabling

these hats to produce even more innovative ideas during the elicitation process.

By using Large Language Models as an assistant in the analysis, we can quickly cover any uncovered parts of the requirements, eliminating the need for additional workshops. This helps reduce the time spent in the elicitation process, making it more efficient.

In future work, adopting the workshop protocol for different topics and applying it to various domains, such as for legal application, could provide additional insights. Implementing the Six Thinking Hat method during requirement elicitation in new domains will give us a clue in assessing the adaptability of the protocol to various contexts, as well as in identifying unique challenges or opportunities.

In addition to expanding upon the outcomes of our work, there is potential for the use of Blue Hat. Our approach indicates that Blue Hat is primarily responsible for managing the session rather than generating requirements. However, the Blue Hat could be integrated into the system as the workshop analyst. In this role, the Blue Hat would listen to the generated requirements, ask for clarification when needed, and pose follow-up questions to further elaborate on the claims made by other hats. Incorporating the Blue Hat as an active participant in the sequential process, much like an "LLM-as-judge," could provide a valuable use case for ensuring the complete usage of the Six Thinking Hats Method.

In conclusion, our study explored the integration of Large Language Models (LLMs) in the Six Thinking Hat method for requirements elicitation. The study adopts a method frequently used in other domains and uses it successfully in the RE domain. It creates a reproducible protocol for the community. Through both single-agent and multi-agent experiments, the study demonstrated how the creativity of LLMs can be enhanced and leveraged in the elicitation process.

# REFERENCES


1. Zowghi, D. and C. Coulin, "Requirements Elicitation: A Survey of Techniques, Approaches, and Tools", A. Aurum and C. Wohlin (Editors), *Engineering and Managing Software Requirements*, pp. 19–46, Springer, Berlin, Heidelberg, 2005.

2. Bakar, N. H., Z. M. Kasirun and N. Salleh, "Feature Extraction Approaches From Natural Language Requirements for Reuse in Software Product Lines: A Systematic Literature Review", *Journal of Systems and Software*, Vol. 106, pp. 132–149, 2015.

3. Gottesdiener, E., *Requirements by Collaboration: Workshops for Defining Needs*, Addison-Wesley Professional, USA, 2002.

4. Maiden, N., A. Gizikis and S. Robertson, "Provoking Creativity: Imagine What Your Requirements Could Be Like", *IEEE software*, Vol. 21, No. 5, pp. 68–75, 2004.

5. Burnay, C., J. Horkoff and N. Maiden, "Stimulating Stakeholders' Imagination: New Creativity Triggers for Eliciting Novel Requirements", *2016 IEEE 24th International Requirements Engineering Conference (RE)*, pp. 36–45, IEEE, Beijing, China, 2016.

6. Giunta, B., C. Burnay, N. Maiden and S. Faulkner, "Creativity Triggers: Extension and Empirical Evaluation of Their Effectiveness During Requirements Elicitation", *Journal of Systems and Software*, Vol. 191, p. 111365, 2022.

7. Horkoff, J., N. Maiden and D. Asboth, "Creative Goal Modeling for Innovative Requirements", *Information and Software Technology*, Vol. 106, pp. 85–100, 2019.

8. Mahaux, M., L. Nguyen, O. Gotel, L. Mich, A. Mavin and K. Schmid, "Collaborative Creativity in Requirements Engineering: Analysis and Practical Advice",

*IEEE 7th International Conference on Research Challenges in Information Science (RCIS)*, pp. 1–10, IEEE, Paris, France, 2013.

9. Saha, S. K., M. Selvi, G. Büyükcan and M. Mohymen, "A Systematic Review on Creativity Techniques for Requirements Engineering", *2012 International Conference on Informatics, Electronics & Vision (ICIEV)*, pp. 34–39, IEEE, Dhaka, Bangladesh, 2012.

10. De Bono, E., *Six Thinking Hats*, Little, Brown, Boston, 1985.

11. Chone, L. S. and L. T. Heng, "Engaging Electrical Engineering Students in Oral Discussions through Six Thinking Hats in Problem Based Learning", *2010 2nd International Congress on Engineering Education*, pp. 240–244, IEEE, Kuala Lumpur, Malaysia, 2010.

12. Phuntsho, U. and D. Wangdi, "The Effect of Using Six Thinking Hats Strategy on the Development of Writing Skills and Creativity of Seventh Grade EFL Students", *i-Manager's Journal on English Language Teaching*, Vol. 10, No. 2, p. 27, 2020.

13. Kivunja, C., "Using De Bono's Six Thinking Hats Model to Teach Critical Thinking and Problem Solving Skills Essential for Success in the 21st Century Economy", *Creative Education*, Vol. 6, No. 3, pp. 380–391, 2015.

14. Aithal, P. and P. Kumar, "Ideal Analysis for Decision Making in Critical Situations Through Six Thinking Hats Method", *International Journal of Applied Engineering and Management Letters (IJAEML)*, Vol. 1, No. 2, pp. 1–9, 2017.

15. Ribeiro, C., C. Farinha, J. Pereira and M. M. da Silva, "Gamifying Requirement Elicitation: Practical Implications and Outcomes in Improving Stakeholders Collaboration", *Entertainment Computing*, Vol. 5, No. 4, pp. 335–345, 2014.

16. MacDonald, K., I. Oberski, K. Christie and A. Stears, "Innovative Co-evaluation and Co-creation of an Online Learning Programme with de Bono's Six Thinking

Hats", *The Journal of Educational Innovation, Partnership and Change*, Vol. 3, No. 2, 2017.

17. Wu, S., O. Irsoy, S. Lu, V. Dabravolski, M. Dredze, S. Gehrmann, P. Kambadur, D. Rosenberg and G. Mann, "BloombergGPT: A Large Language Model for Finance", , 2023, abs/2303.17564.

18. Li, Y., Z. Li, K. Zhang, R. Dan, S. Jiang and Y. Zhang, "ChatDoctor: A Medical Chat Model Fine-Tuned on a Large Language Model Meta-AI (LLaMA) Using Medical Domain Knowledge", *Cureus*, Vol. 15, No. 6, 2023.

19. Kasneci, E., K. Seßler, S. Küchemann, M. Bannert, D. Dementieva, F. Fischer, U. Gasser, G. Groh, S. Günnemann, E. Hüllermeier *et al.*, "ChatGPT for Good? On Opportunities and Challenges of Large Language Models for Education", *Learning and Individual Differences*, Vol. 103, p. 102274, 2023.

20. Peña, A., A. Morales, J. Fierrez, I. Serna, J. Ortega-Garcia, I. Puente, J. Cordova and G. Cordova, "Leveraging Large Language Models for Topic Classification in the Domain of Public Affairs", *International Conference on Document Analysis and Recognition*, pp. 20–33, Springer, Cham, 2023.

21. Qin, C., A. Zhang, Z. Zhang, J. Chen, M. Yasunaga and D. Yang, "Is ChatGPT a General-Purpose Natural Language Processing Task Solver?", H. Bouamor, J. Pino and K. Bali (Editors), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 1339–1384, Association for Computational Linguistics, Singapore, Dec. 2023.

22. Liu, D. and V. Demberg, "ChatGPT vs Human-authored Text: Insights Into Controllable Text Summarization and Sentence Style Transfer", V. Padmakumar, G. Vallejo and Y. Fu (Editors), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, pp. 1–18, Association for Computational Linguistics, Toronto, Canada, Jul. 2023.

23. Guzik, E. E., C. Byrge and C. Gilde, "The Originality of Machines: AI Takes the Torrance Test", *Journal of Creativity*, Vol. 33, No. 3, p. 100065, 2023.

24. Palomares, C., X. Franch, C. Quer, P. Chatzipetrou, L. López and T. Gorschek, "The State-of-Practice in Requirements Elicitation: An Extended Interview Study at 12 Companies", *Requirements Engineering*, Vol. 26, No. 2, pp. 273–299, 2021.

25. Karras, O., S. Kiesling and K. Schneider, "Supporting Requirements Elicitation by Tool-Supported Video Analysis", *2016 IEEE 24th International Requirements Engineering Conference (RE)*, pp. 146–155, IEEE, Beijing, China, 2016.

26. Spijkman, T., X. De Bondt, F. Dalpiaz and S. Brinkkemper, "Summarization of Elicitation Conversations to Locate Requirements-Relevant Information", A. Ferrari and B. Penzenstadler (Editors), *REFSQ*, Vol. 13975, pp. 122–139, Springer Nature Switzerland, Cham, 2023.

27. Jha, N. and A. Mahmoud, "Mining Non-Functional Requirements from App Store Reviews", *Empirical Software Engineering*, Vol. 24, No. 6, pp. 3659–3695, 2019.

28. Dalpiaz, F. and M. Parente, "RE-SWOT: From User Feedback to Requirements via Competitor Analysis", E. Knauss and M. Goedicke (Editors), *REFSQ*, Vol. 11412, pp. 55–70, Springer International Publishing, Cham, 2019, series Title: Lecture Notes in Computer Science.

29. Osborn, A., *Your Creative Power*, Read Books Ltd, New York, 2011.

30. Sosa, R. *et al.*, "Retrospective and Prospective of the Study of Design Creativity: 80 Years into the Past and the Future", *Proceedings of the Sixth International Conference on Design Creativity (ICDC 2020)*, pp. 001–010, Finland, 2020.

31. Eberle, B., *Scamper On: Games for Imagination Development*, Prufrock Press Inc., New York, 1996.

32. Gündoğan, A., "SCAMPER: Improving Creative Imagination of Young Children", *Creativity Studies*, Vol. 12, No. 2, pp. 315–326, 2019.

33. Wu, T.-T. and Y.-T. Wu, "Applying Project-Based Learning and SCAMPER Teaching Strategies in Engineering Education to Explore the Influence of Creativity on Cognition, Personal Motivation, and Personality Traits", *Thinking Skills and Creativity*, Vol. 35, p. 100631, 2020.

34. Vernon, D. and I. Hocking, "Thinking Hats and Good Men: Structured Techniques in a Problem Construction Task", *Thinking Skills and Creativity*, Vol. 14, pp. 41–46, 2014.

35. Öznur Göçmen and H. Coşkun, "The Effects of the Six Thinking Hats and Speed on Creativity in Brainstorming", *Thinking Skills and Creativity*, Vol. 31, pp. 284–295, 2019.

36. Chen, D., F. Liu, Y. Hu, X. Du, Y. Bai, Z. Ren, L. Lan and W. Yu, "Service Design from the Perspective of "Six Thinking Hats": A Comparison of Storytelling Strategies of Experts and Novices", *Thinking Skills and Creativity*, Vol. 47, p. 101219, 2023.

37. Maiden, N., C. Ncube and S. Robertson, "Can Requirements Be Creative? Experiences with an Enhanced Air Space Management System", *29th International Conference on Software Engineering (ICSE'07)*, pp. 632–641, IEEE, Minneapolis, MN, USA, 2007.

38. Maiden, N., S. Robertson and J. Robertson, "Creative Requirements: Invention and Its Role in Requirements Engineering", *Proceedings of the 28th International Conference on Software Engineering*, pp. 1073–1074, Shanghai, China, 2006.

39. Schlosser, C., S. Jones and N. Maiden, "Using a Creativity Workshop to Generate Requirements for an Event Database Application", *Requirements Engineering:*

*Foundation for Software Quality: 14th International Working Conference, REFSQ 2008, June 16-17, 2008 Proceedings 14*, pp. 109–122, Springer, Berlin, Heidelberg, 2008.

40. Do, Q. A. and T. Bhowmik, "Automated Generation of Creative Software Requirements: A Data-Driven Approach", *Proceedings of the 1st ACM SIGSOFT International Workshop on Automated Specification Inference*, pp. 9–12, Lake Buena Vista, FL, USA, 2018.

41. Do, Q. A., T. Bhowmik and G. L. Bradshaw, "Capturing Creative Requirements via Requirements Reuse: A Machine Learning-Based Approach", *Journal of Systems and Software*, Vol. 170, p. 110730, 2020.

42. Gudaparthi, H., N. Niu, B. Wang, T. Bhowmik, H. Liu, J. Zhang, J. Savolainen, G. Horton, S. Crowe, T. Scherz *et al.*, "Prompting Creative Requirements via Traceable and Adversarial Examples in Deep Learning", *2023 IEEE 31st International Requirements Engineering Conference (RE)*, pp. 134–145, IEEE, Hannover, Germany, 2023.

43. Shah, R., K. Chawla, D. Eidnani, A. Shah, W. Du, S. Chava, N. Raman, C. Smiley, J. Chen and D. Yang, "When FLUE Meets FLANG: Benchmarks and Large Pretrained Language Model for Financial Domain", Y. Goldberg, Z. Kozareva and Y. Zhang (Editors), *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 2322–2335, Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, Dec. 2022.

44. Wang, Y., W. Wang, S. R. Joty and S. C. H. Hoi, "CodeT5: Identifier-Aware Unified Pre-trained Encoder-Decoder Models for Code Understanding and Generation", M.-F. Moens, X. Huang, L. Specia and S. W.-t. Yih (Editors), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 8696–8708, Online and Punta Cana, Dominican Republic, Nov. 2021.

45. Chen, M., J. Tworek, H. Jun, Q. Yuan, H. P. D. O. Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman *et al.*, "Evaluating Large Language Models Trained on Code", , 2021, abs/2107.03374.

46. Feng, S. and C. Chen, "Prompting Is All You Need: Automated Android Bug Replay with Large Language Models", *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*, ICSE '24, Association for Computing Machinery, New York, NY, USA, 2024.

47. Liu, P., W. Yuan, J. Fu, Z. Jiang, H. Hayashi and G. Neubig, "Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing", *ACM Computing Surveys*, Vol. 55, No. 9, pp. 1–35, 2023.

48. Sridhara, G., R. H. G. and S. Mazumdar, "ChatGPT: A Study on its Utility for Ubiquitous Software Engineering Tasks", , 2023, abs/2305.16837.

49. Fantechi, A., S. Gnesi, L. Semini *et al.*, "Rule-based NLP vs ChatGPT in Ambiguity Detection, a Preliminary Study", *REFSQ Workshops*, Barcelona, Spain, 2023.

50. Fantechi, A., S. Gnesi, L. Passaro and L. Semini, "Inconsistency Detection in Natural Language Requirements Using ChatGPT: A Preliminary Evaluation", *2023 IEEE 31st International Requirements Engineering Conference (RE)*, pp. 335–340, IEEE, Hannover, Germany, 2023.

51. Görer, B. and F. B. Aydemir, "Generating Requirements Elicitation Interview Scripts with Large Language Models", *2023 IEEE 31st International Requirements Engineering Conference Workshops (REW)*, pp. 44–51, IEEE, Hannover, Germany, 2023.

52. Ippolito, D., A. Yuan, A. Coenen and S. Burnam, "Creative Writing with an AI-Powered Writing Assistant: Perspectives from Professional Writers", , 2022,

abs/2211.05030.

53. Chen, Z., E. Zhou, K. Eaton, X. Peng and M. Riedl, "Ambient Adventures: Teaching ChatGPT on Developing Complex Stories", , 2023, abs/2308.01734.

54. Mirowski, P., K. W. Mathewson, J. Pittman and R. Evans, "Co-Writing Screenplays and Theatre Scripts with Language Models: Evaluation by Industry Professionals", *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, CHI '23, Association for Computing Machinery, New York, NY, USA, 2023.

55. Chakrabarty, T., P. Laban, D. Agarwal, S. Muresan and C.-S. Wu, "Art or Artifice? Large Language Models and the False Promise of Creativity", *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, CHI '24, Association for Computing Machinery, New York, NY, USA, 2024.

56. Padmakumar, V. and H. He, "Does Writing with Language Models Reduce Content Diversity?", , 2024, abs/2309.05196.

57. Torrance, E. P., *Torrance Tests of Creative Thinking*, Personnel Press, 1966., Princeton, N.J., 1966.

58. Zhao, Y., R. Zhang, W. Li, D. Huang, J. Guo, S. Peng, Y. Hao, Y. Wen, X. Hu, Z. Du, Q. Guo, L. Li and Y. Chen, "Assessing and Understanding Creativity in Large Language Models", , 2024, abs/2401.12491.

59. Chang, H.-F. and T. Li, "A Framework for Collaborating a Large Language Model Tool in Brainstorming for Triggering Creative Thoughts", *Thinking Skills and Creativity*, Vol. 56, p. 101755, 2025.

60. DeLorenzo, M., V. Gohil and J. Rajendran, "CreativEval: Evaluating Creativity of LLM-Based Hardware Code Generation", , 2024, abs/2404.08806.

61. Rachmann, A., "Six Thinking Chatbots: A Creativity Technique Deployed via a Large Language Model.", *REFSQ Workshops*, Winterthur, Switzerland, 2024.

62. Lu, L.-C., S.-J. Chen, T.-M. Pai, C.-H. Yu, H. yi Lee and S.-H. Sun, "LLM Discussion: Enhancing the Creativity of Large Language Models via Discussion Framework and Role-Play", , 2024, abs/2405.06373.

63. Han, S. and W. Choi, "Development of a Large Language Model-based Multi-Agent Clinical Decision Support System for Korean Triage and Acuity Scale (KTAS)-Based Triage and Treatment Planning in Emergency Departments", , 2024, abs/2408.07531.

64. Barbarroxa, R., L. Gomes and Z. Vale, "Benchmarking Large Language Models for Multi-agent Systems: A Comparative Analysis of AutoGen, CrewAI, and TaskWeaver", P. Mathieu and F. De la Prieta (Editors), *Advances in Practical Applications of Agents, Multi-Agent Systems, and Digital Twins: The PAAMS Collection*, pp. 39–48, Springer Nature Switzerland, Cham, 2025.

65. Duan, Z. and J. Wang, "Exploration of LLM Multi-Agent Application Implementation Based on LangGraph+CrewAI", , 2024, abs/2411.18241.

66. Achiam, J., S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat *et al.*, "Gpt-4 Technical Report", , 2023, arXiv preprint arXiv:2303.08774.

67. Wei, J., X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou *et al.*, "Chain-of-thought Prompting Elicits Reasoning in Large Language Models", *NIPS '22*, Vol. 35, pp. 24824–24837, New Orleans, LA, USA, 2022.

68. Wallach, M. and N. Kogan, *Modes of Thinking in Young Children: A Study of the Creativity-intelligence Distinction*, Holt, Rinehart and Winston, New York, 1965.

69. Hu, W. and P. Adey, "A Scientific Creativity Test for Secondary School Students",

*International Journal of Science Education*, Vol. 24, No. 4, pp. 389–403, 2002.

70. Sternberg, R. J., *Handbook of Creativity*, Cambridge University Press, Cambridge, 1999.

71. Grootendorst, M., "BERTopic: Neural Topic Modeling with a Class-Based TF-IDF Procedure", , 2022, arXiv preprint arXiv:2203.05794.

72. Shapiro, S. S. and M. B. Wilk, "An Analysis of Variance Test for Normality (Complete Samples)", *Biometrika*, Vol. 52, No. 3-4, pp. 591–611, 1965.

73. Woolson, R. F., *Wilcoxon Signed-Rank Test*, pp. 1–3, John Wiley & Sons, Ltd, 2008.

74. Council, D., "Framework for Innovation", , 2004, `https://www.designcouncil.org.uk/our-resources/framework-for-innovation/`, accessed on Jan 10, 2025.

# APPENDIX A: STUDY FOR PROMPT

The Appendix includes selected example prompts from the single-agent experiments, covering five hat styles and three prompting methods: zero-shot, few-shot, and chain-of-thought prompting.

The prompt belongs to the White Hat Task for zero-shot in single-agent experiments:

> "Create very detailed requirements for a table tennis tournament app by focusing objective facts, verifiable data and information needs without opinions, emotions, or speculation. Generate requirements which put core features that are universally expected and widely recognized in a given system or application. You need to create shall requirements from the given idea. Make sure you created most comprehensive set of requirements. Do not create duplicated requirements. The requirements must be atomic. This is the expect criteria for your final answer: List of shall requirements. Create as many requirements as possible. you MUST return the actual complete content as the final answer, not a summary."

The prompt belongs to the Yellow Hat Task for zero-shot in single-agent experiments:

> "Create very detailed requirements for a table tennis tournament app by focusing on the positive, beneficial aspects of an application. Generate requirements to delight users, drive engagement, and create an experience users will love and repeatedly use. You need to create shall requirements from the given idea. Make sure you created most comprehensive set of requirements. Do not create duplicated requirements. The requirements must be atomic. This is the expect criteria for your final answer: List of shall requirements. Create as many requirements as possible. you MUST return the actual complete content as the final answer, not a summary."

The prompt belongs to the Black Hat Task for zero-shot in single-agent experiments:

> "Create very detailed requirements for a table tennis tournament app by seeking risks and potential problems. Generate requirements which put the caution points and solutions. You need to create shall requirements from the given idea. Make sure you created most comprehensive set of requirements. Do not create duplicated requirements. The requirements must be atomic. This is the expect criteria for your final answer: List of shall requirements. Create as many requirements as possible. you MUST return the actual complete content as the final answer, not a summary."

The prompt belongs to the Red Hat Task for zero-shot in single-agent experiments:

"Create very detailed requirements for a table tennis tournament app by focusing focusing on the intuitive, emotional, and instinctive aspects of the user experience. Generate requirements which put emotional needs that helps emotional engagements. You need to create shall requirements from the given idea. Make sure you created most comprehensive set of requirements. Do not create duplicated requirements. The requirements must be atomic. This is the expect criteria for your final answer: List of shall requirements. Create as many requirements as possible. you MUST return the actual complete content as the final answer, not a summary."

The prompt belongs to the Green Hat Task for zero-shot in single-agent experiments:

"Generate creative and original requirements for table tennis tournament app by focusing creativity, innovation and unconventional ideas. Think new alternatives for the existed solutions. Focus on different options, fresh solutions. It is allowed to put forward crazy ideas. You need to create shall requirements from the given idea. Make sure you created most comprehensive set of requirements. Do not create duplicated requirements. The requirements must be atomic. This is the expect criteria for your final answer: List of shall requirements. Create as many requirements as possible. you MUST return the actual complete content as the final answer, not a summary."

Few-shot examples for the White Hat in single-agent experiments:

"For an e-commerce application example white hat requirements can be:
1. The application must allow users to create accounts, securely log in, and manage their profiles and order history.
2. The application must display a searchable and filterable product catalog with detailed product descriptions, images, and pricing.
For a meal cooking application example white hat requirements can be:
1. The application must provide a ingredient list with measurement unit.
2. The application must allow users to upload, share, and review recipes."

Few-shot examples for Yellow Hat in single-agent experiments:

"For an e-commerce application example yellow hat requirements can be:
1. The application must provide personalized product recommendations based on user browsing behavior, past purchases, and preferences.
2. The application must provide a loyalty rewards program that offers discounts or incentives based on user activity.
For a meal cooking application example yellow hat requirements can be:

1. The application must offer high-quality step-by-step cooking videos for every recipe.
2. The application must allow users to create and customize weekly meal plans."

Chain of Thoughts examples for Red Hat in single-agent experiments

"Step 1. Identify user emotions and intuitions.
Step 2. Think about user needs and preferences that can be met through personalization.
Step 3. Create requirements by converting user emotions and intuitions into features.
For example about meal cooking application you can follow these steps:
Step 1: Identify user emotions — ' Users may feel stressed when they don't have all the ingredients needed for a recipe.'
Step 2: Think about user needs — 'How can I ease their frustration and make it easy to substitute ingredients?'
Step 3: Create requirement statement — 'The application should personalized ingredient substitution suggestions based on user preferences.'
Identify all emotional aspect and personalized features in the system and create requirements to realm them."