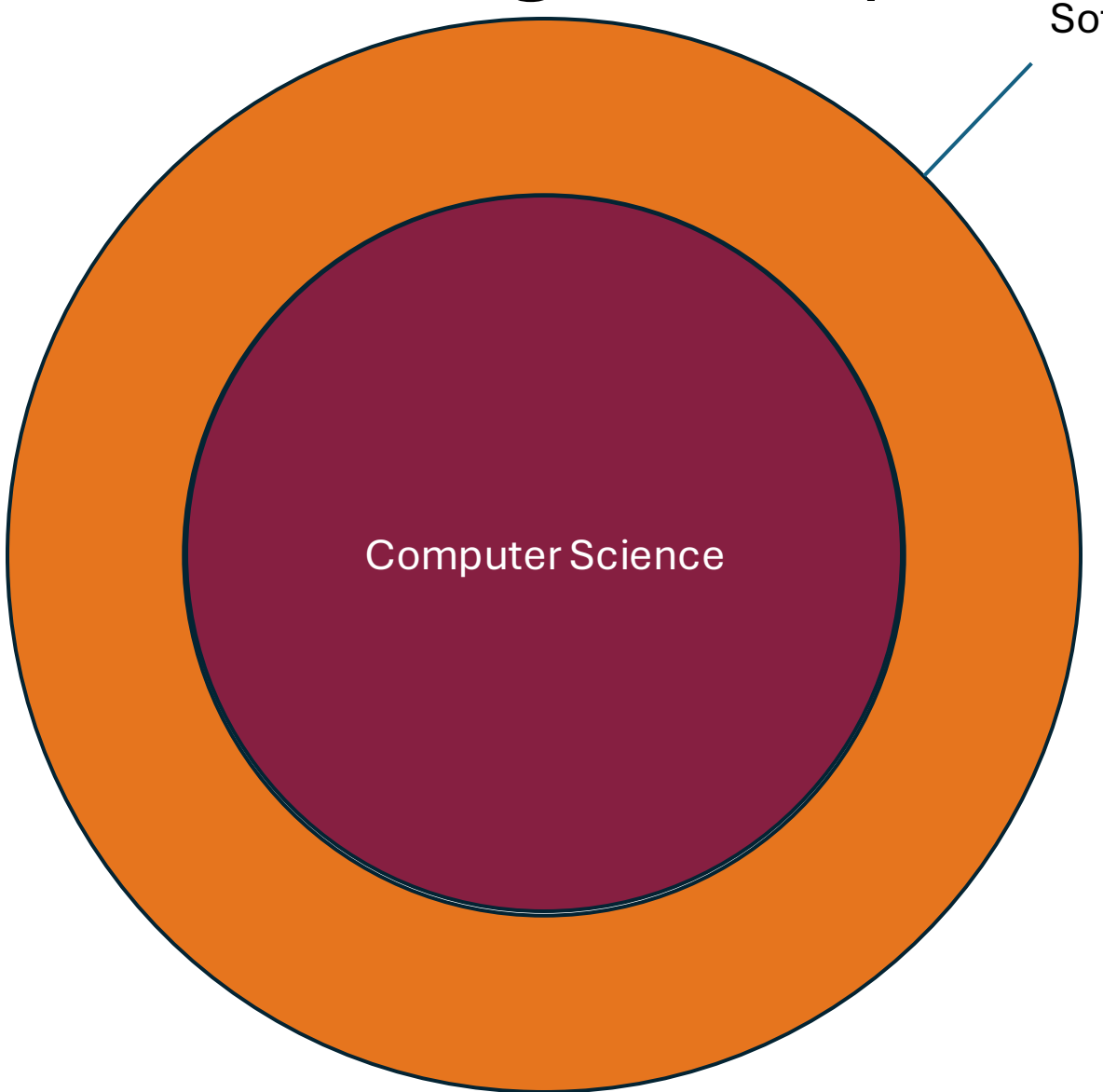
Designing Tools to Enhance Best Practices in Research Software Engineering

Minhyuk Ko, Chris Brown

ALFRED P. SLOAN
FOUNDATION

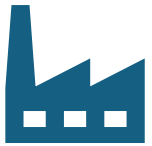
Research Software Engineers (RSEs)



Mathematician

Psychologists

Chemists



Industrial Engineers

Cole World, No Blanket

Cole World

- The coldness and pervasiveness of problems in life.

No Blanket

- No way to avoid or shield away from the harsh realities of life (yet).

Research Software Engineers (RSEs)

Code World

- More than 90% of researchers rely on software for their work. (Hettrick 2014)
- 67% of data scientists had non-computer science majors (Ravisankar 2016)

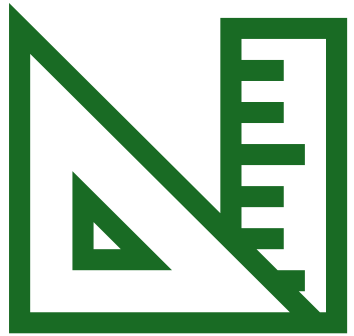
No Blanket

- Useful software development techniques are often overlooked by RSEs (Eisty 2019, Jay 2016, Kelly 2008, Smith 2016)
- This leads to problems such as security flaws or yielding inaccurate study findings (Milewicz et al. 2022, Soergal 2015)

“Designing a Blanket”

Research Questions

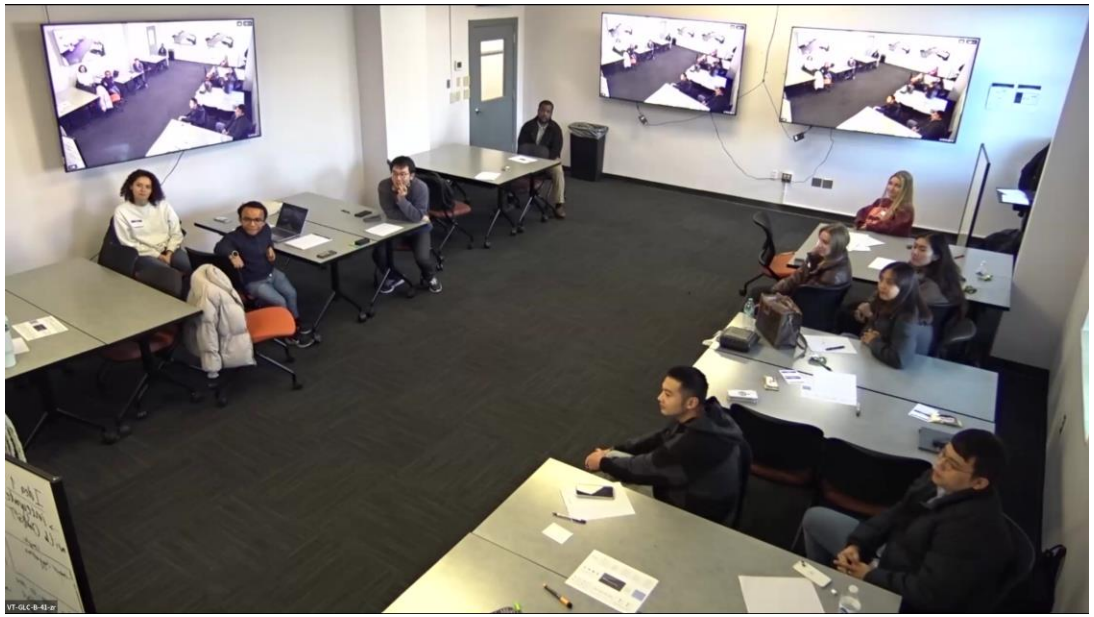
RQ1: What are the development challenges that RSEs face when developing research software?



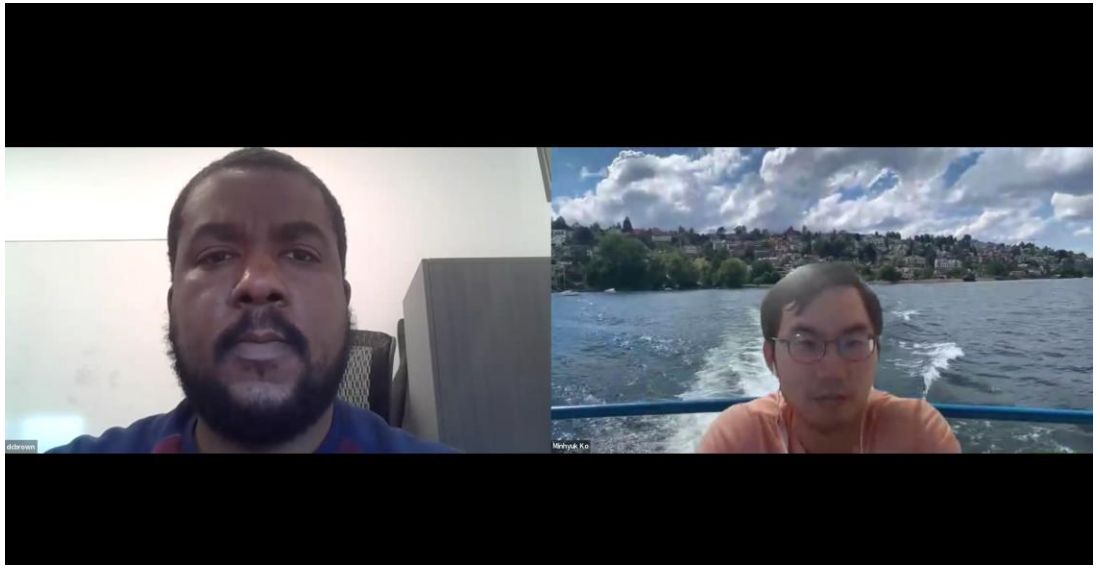
RQ2: What design affordances are needed to overcome development challenges from the perspective of RSEs?

Participatory Design Workshops

In-person

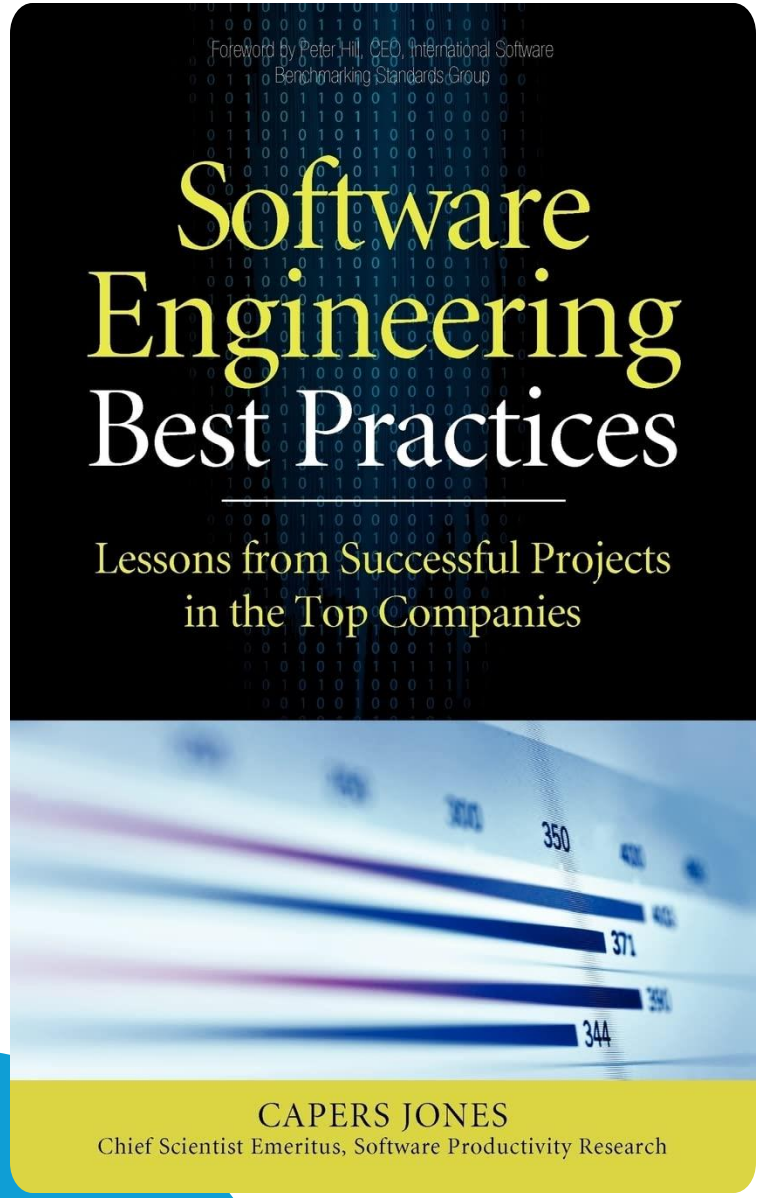


Virtual



Participant Eligibility

- Must be 18 years or older
- Have experience in using programming for research
- Do not major in Computer Science
- Physically located in the US

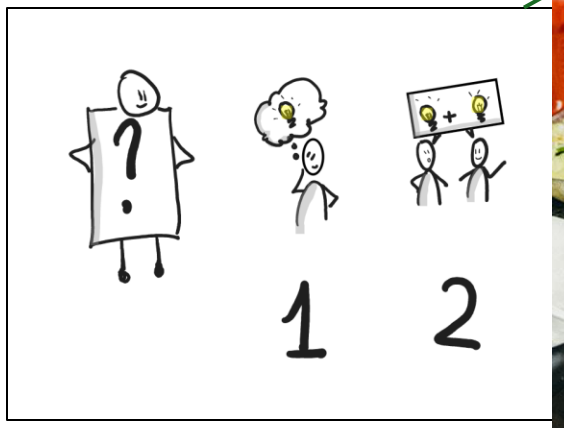


- **User Involvement in Software Projects**
- **Requirements of Software Applications**
- **Software Quality Assurance**
- **Software Reusability**
- **Software Project Measurements and Metrics**
- **Software Project Cost**
- **Communication During Software Projects**
- **Training Software Technical Personnel**
- **Software Maintenance and Enhancement**

Three-s y Design

RQ1

Ice Breaker



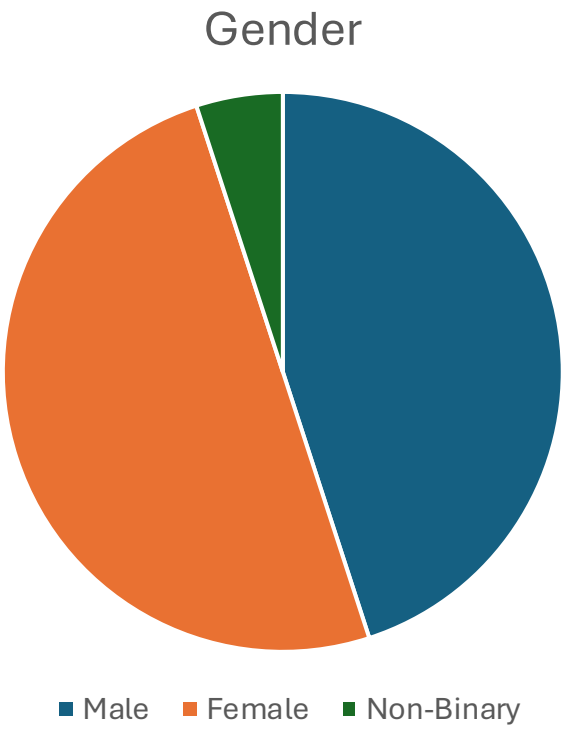
RQ2

Discovery Process

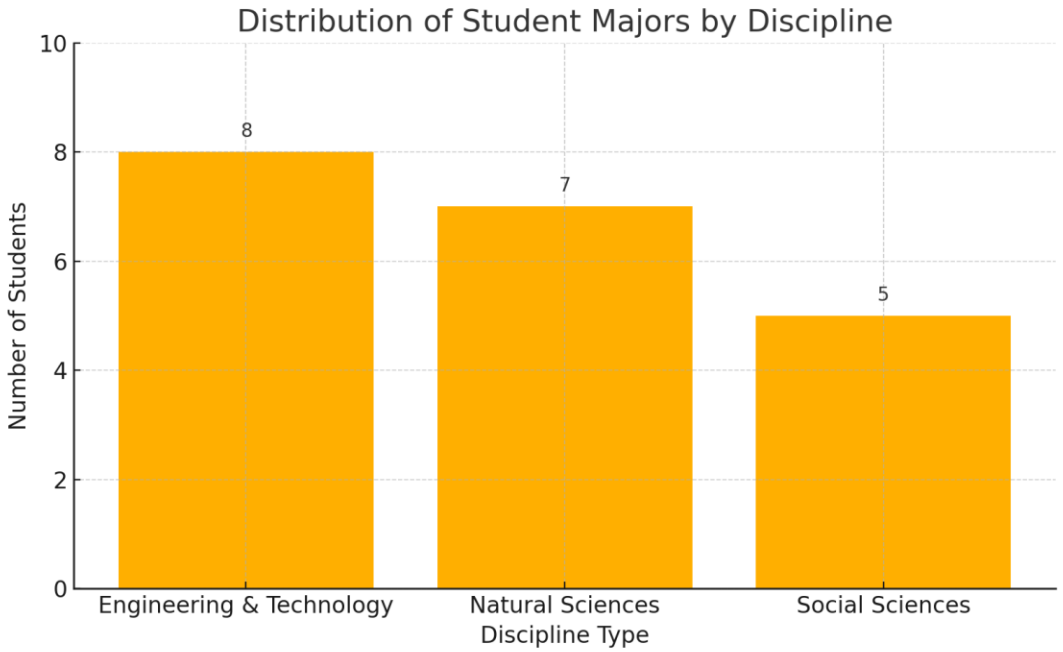
Prototyping

Participant Demographics

Gender

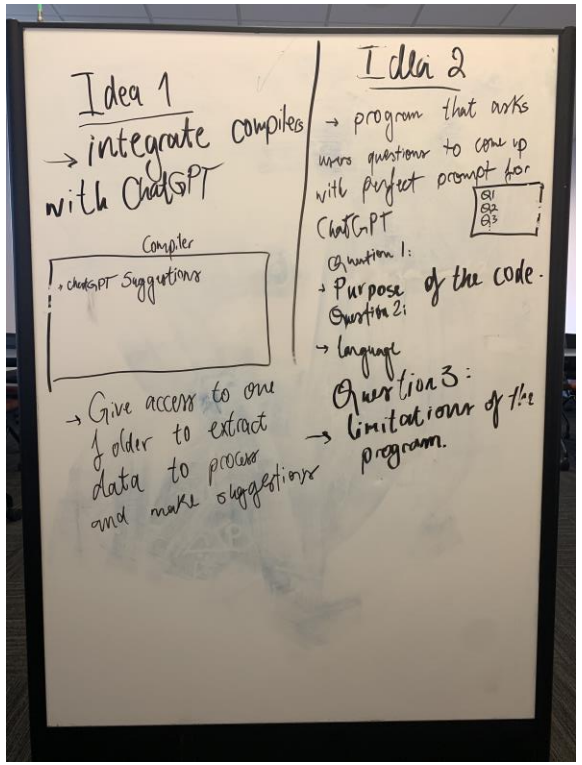


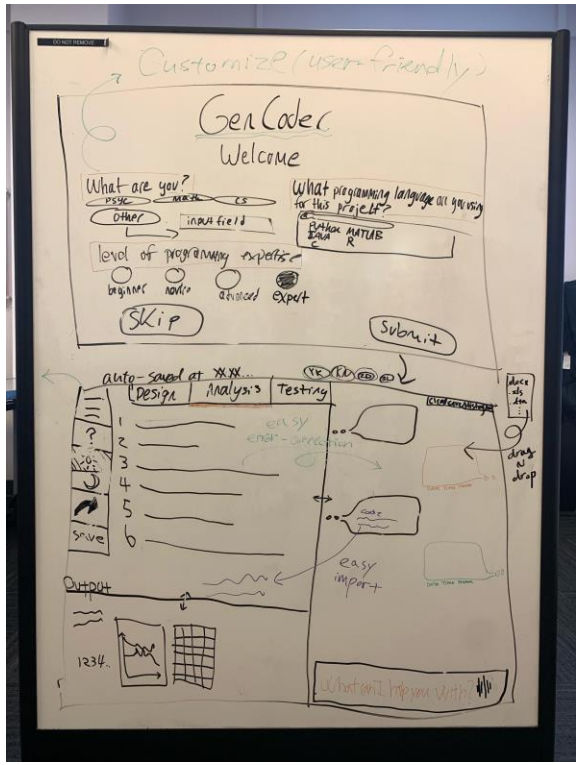
Majors



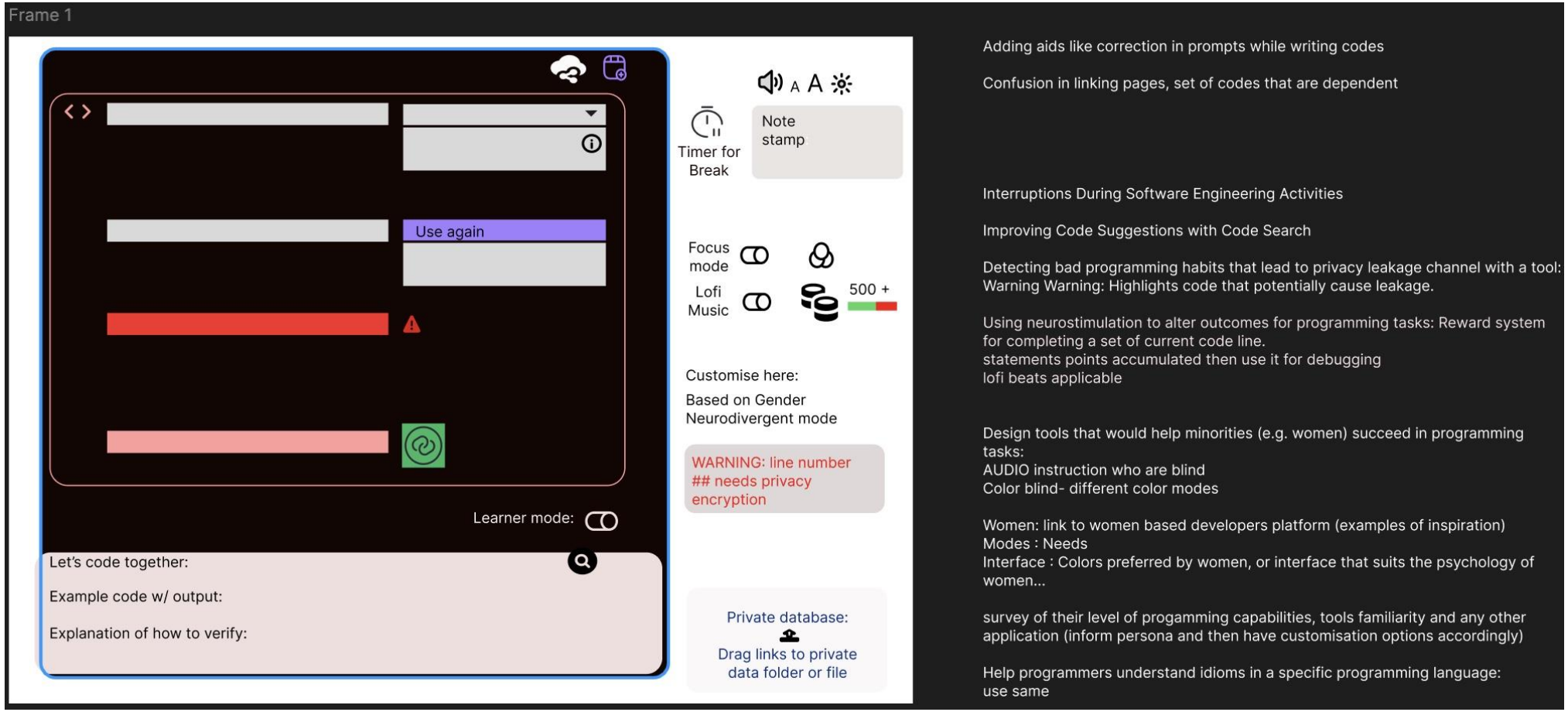
13 Different Majors Represented!

In-Person Workshop Prototype Examples





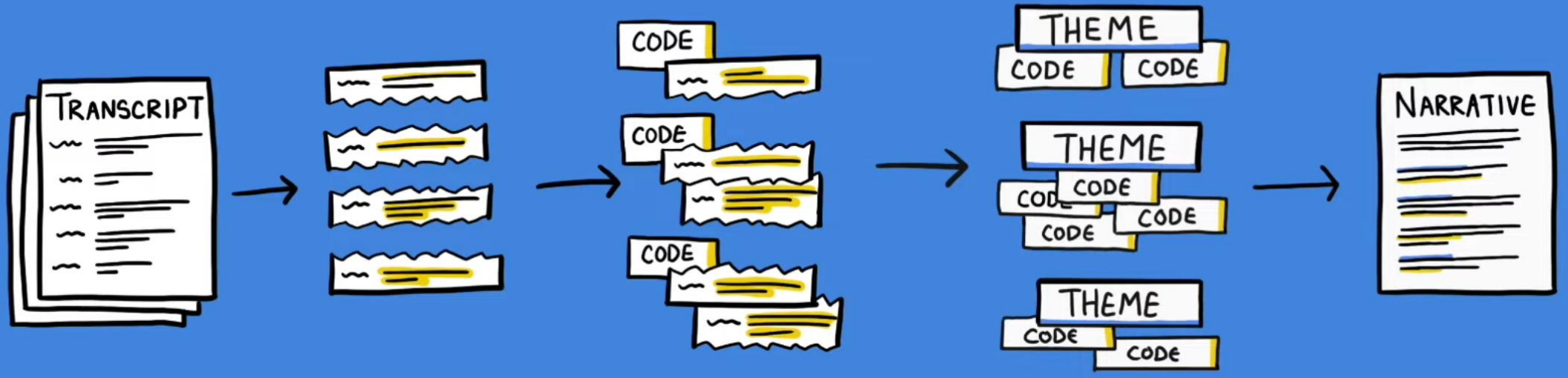
Virtual Workshop Prototype Example



Data analysis?

Thematic analysis

A data analysis process that involves delving through a data set, identifying patterns, systematically coding, deriving themes, and creating a narrative.



RQ1: Development Challenges

Code	Definition	Grounded
Implementation	It is difficult to think of a programmer's way of solving coding problems to write the code.	51
Debugging	Debugging the code written by others, including generative AI, is difficult.	25
Understanding	Understanding the architecture of the code is difficult.	20
Syntax	It is difficult to remember all the minor syntax.	19
Selection	Choosing the best programming language that would best suit the programming needs is difficult.	16
Verification	It is difficult to verify if the code that I wrote is correct or not.	12

RQ2: Design Affordances to Overcome Challenges

Code	Definition	Grounded
Code Translation	Translating between different coding languages.	177
Code Understanding	Helping humans understand the code or idioms better.	133
Communication	Researchers want to communicate with researchers with expertise or use different programming languages.	94
Auto-Correction	Developers desire a live editing feature where simple errors are easily fixed.	57
AI for Development	Researchers want to use **AI** to accomplish certain tasks.	43
Help Use **AI**	Researchers desire a tool that would help them use **AI** efficiently.	41

Key Findings

RQ1: RSEs struggle with implementing, debugging, and understanding code due to lacking developers' mental models, which limits their ability to choose appropriate tools, languages, and testing strategies.

RQ2: RSEs want interactive tools to aid in translating, understanding, correcting, and discussing code, and see **AI** as a valuable aid that should be better leveraged through user-focused solutions.

So, What Can We Do?

INCREASING AWARENESS OF SE BEST PRACTICES

UTILIZING GENERATIVE **AI**

OVERCOMING PROGRAMMING LANGUAGE BARRIERS

Also, we can publish our work to a conference!

What’s Our Next Step?

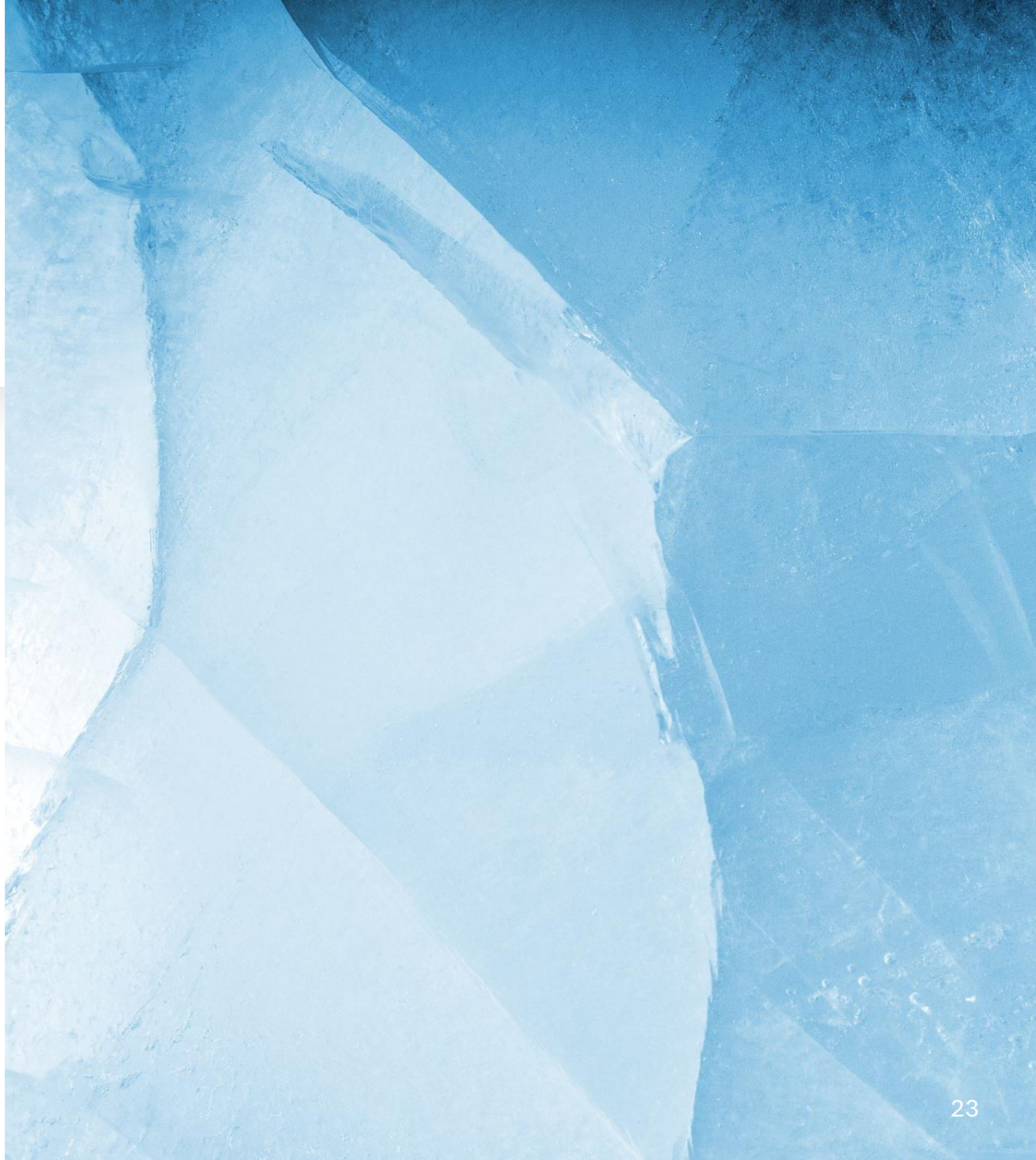
VT Library Workshop

- Friday, April 25th 4~6pm
- Newman Library (Room TBD)
- Dinner will be provided.
- More details coming soon, so keep an eye on the HCI listserv!

Attention!

Ice Breaker

- What is the funniest prompt you have ever given to ChatGPT?



How Do You Use Generative AI for Programming?

How can we improve prompts?

Prompt engineering guidelines for LLMs in Requirements Engineering

1st Simon Arvidsson
dept. of Computer Science and Engineering
University of Gothenburg
Gothenburg, Sweden
gussimoar@student.gu.se

2nd Johan Axell
dept. of Computer Science and Engineering
University of Gothenburg
Gothenburg, Sweden
gusaxelljo@student.gu.se

***Abstract*—The rapid emergence of large generative AI models has demonstrated their utility across a multitude of tasks. Ensuring the quality and accuracy of the models' output is done in different ways. In this study, we focused on prompt engineering. Prompt engineering guidelines for how to utilize large generative AI models in the field of requirements engineering are limited in the literature. The objective of this study was to explore the potential advantages and limitations of the possible application of existing prompt engineering guidelines from the literature in requirements engineering. To achieve this goal, we conducted a systematic literature review on prompt engineering guidelines to gather guidelines which could be applicable to various tasks. Subsequently, we considered different requirements engineering activities and their characteristics before proposing a mapping of our gathered guidelines to requirements engineering activities. Furthermore, we conducted interviews with three requirements engineering experts to gain further perspectives on our findings and mapping suggestions. Through thematic analysis, we ex-**

in order to help mitigate unnecessary expensive corrections of the errors at later phases [7], [8]. Often, the errors arise due to different interpretations or terminology among stakeholders and the written requirements being ambiguous [9]. These requirements are written using natural language (NL), and one sub-fields of AI4RE addresses AI for NL and include natural language processing (NLP). NLP incorporates different computational techniques in order to enable interaction between AI and humans through the use of natural language.

This described subfield of AI4RE, known as NLP4RE, revolves around the application of NLP techniques in activities related to RE like requirement elicitation and classification. The 2018 NLP4RE workshop highlighted the growing significance of NLP as a fundamental component in domains including RE [10].

Context

Adding context to examples in prompts produce more efficient and informative output.

Provide context to all prompts to avoid output hallucinations.

Provide context of the prompt to ensure a closely related output.

Use open-ended prompts to generate context before providing the intended question(s).

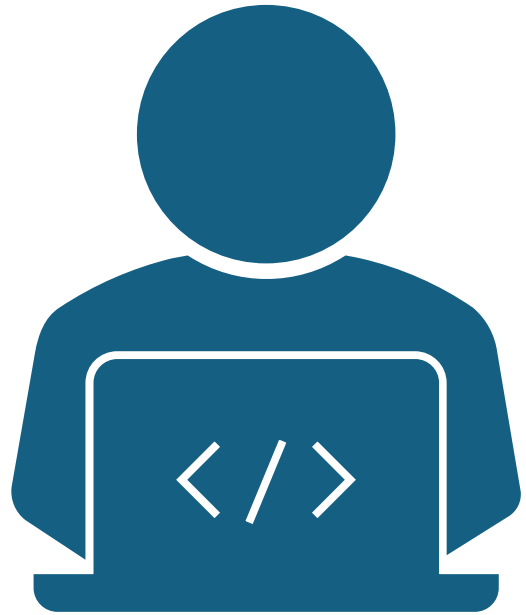
Provide context to the topic of the prompt before describing a task.

Adding context tokens to enhance the prompt, improves the related output.

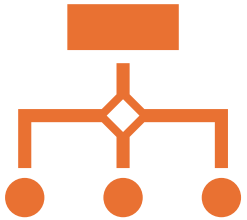
The more context tokens pre-appended to prompts, the more fine-grained output.

Persona

- Improves the generation quality by conditioning the prompt with an identity, such as “Python programmer” or “Math tutor”
- To explore the requirements of a software-reliant system, include:
  - “I want you to act as the system”,
  - “Use the requirements to guide your behavior”,
  - “I will ask you to do X, and you will tell me if X is possible given the requirements.”,
  - “If X is possible, explain why using the requirements.”,
  - “If I can’t do X based on the requirements, write the missing requirements needed in the format Y.”

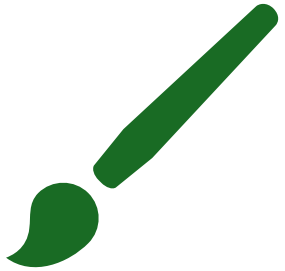


Templates



To improve reasoning and common sense in output, follow a template such as:

“Reason step-by-step for the following problem. [Original prompt inserted here]”



The following prompt template has shown an impressive quality of AI art:

“[Medium] [Subject] [Artist(s)] [Details] [Image repository support]”

Disambiguation

Ensure any areas of potential miscommunication or ambiguity are caught, by providing a detailed scope:

"Within this scope",

"Consider these requirements or specifications"

To find points of weakness in a requirements specification, consider including:

"Point out any areas of ambiguity or potentially unintended outcomes"

3. The persona prompt method can be used to consider potential ambiguities from different perspectives.

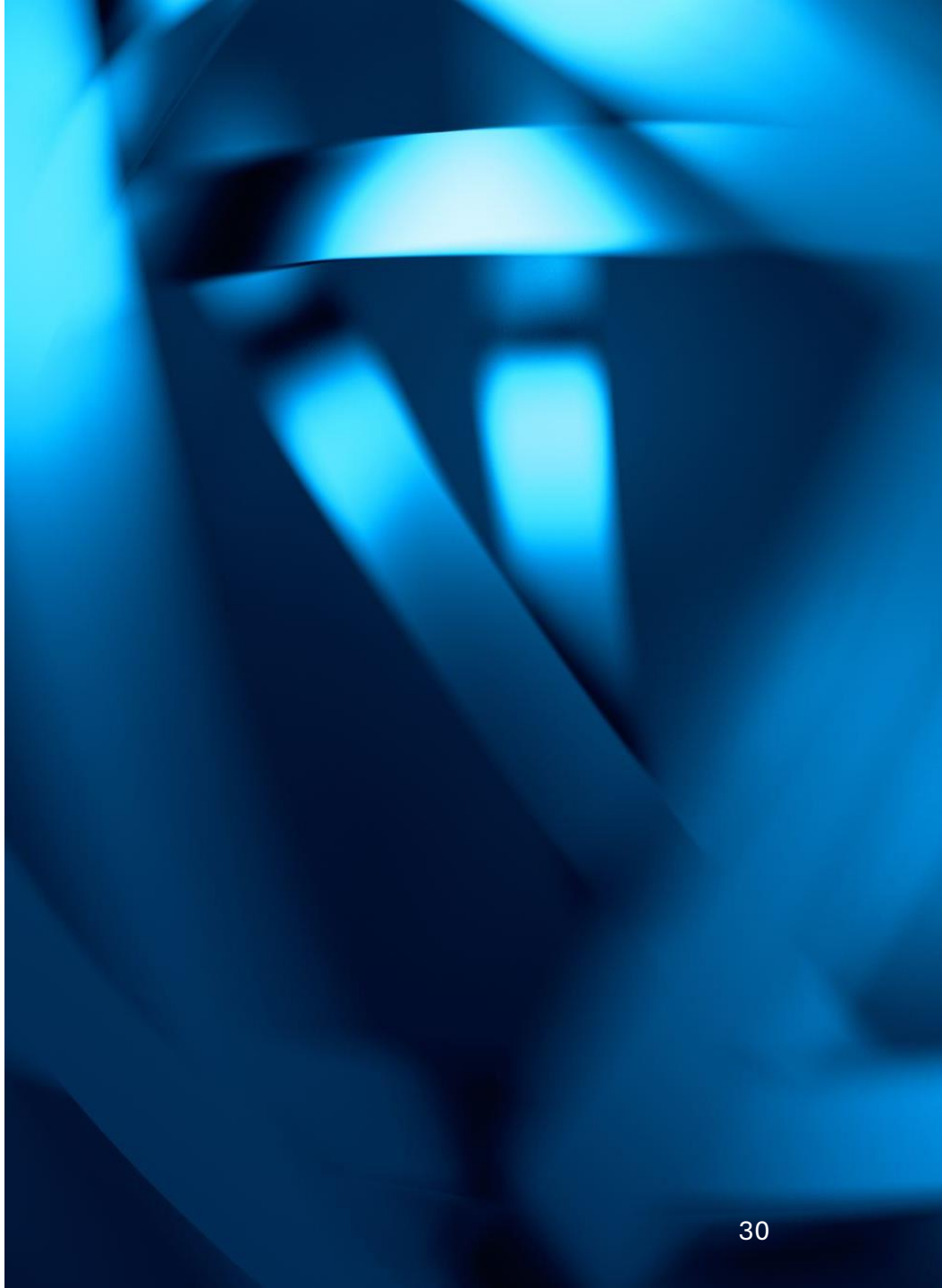
Reasoning

Prepending “Let’s think step by step” improves zero-shot performance.

Extending the previously known “Let’s think step by step”, with “to reach the right conclusion,” to highlight decision-making in the prompt.

Factual inconsistency evaluation can be significantly boosted using chain-of-thought prompting.

Chain Of Thought (CoT) prompting improves LLM performance compared to Zero-shot and without CoT.



Analysis

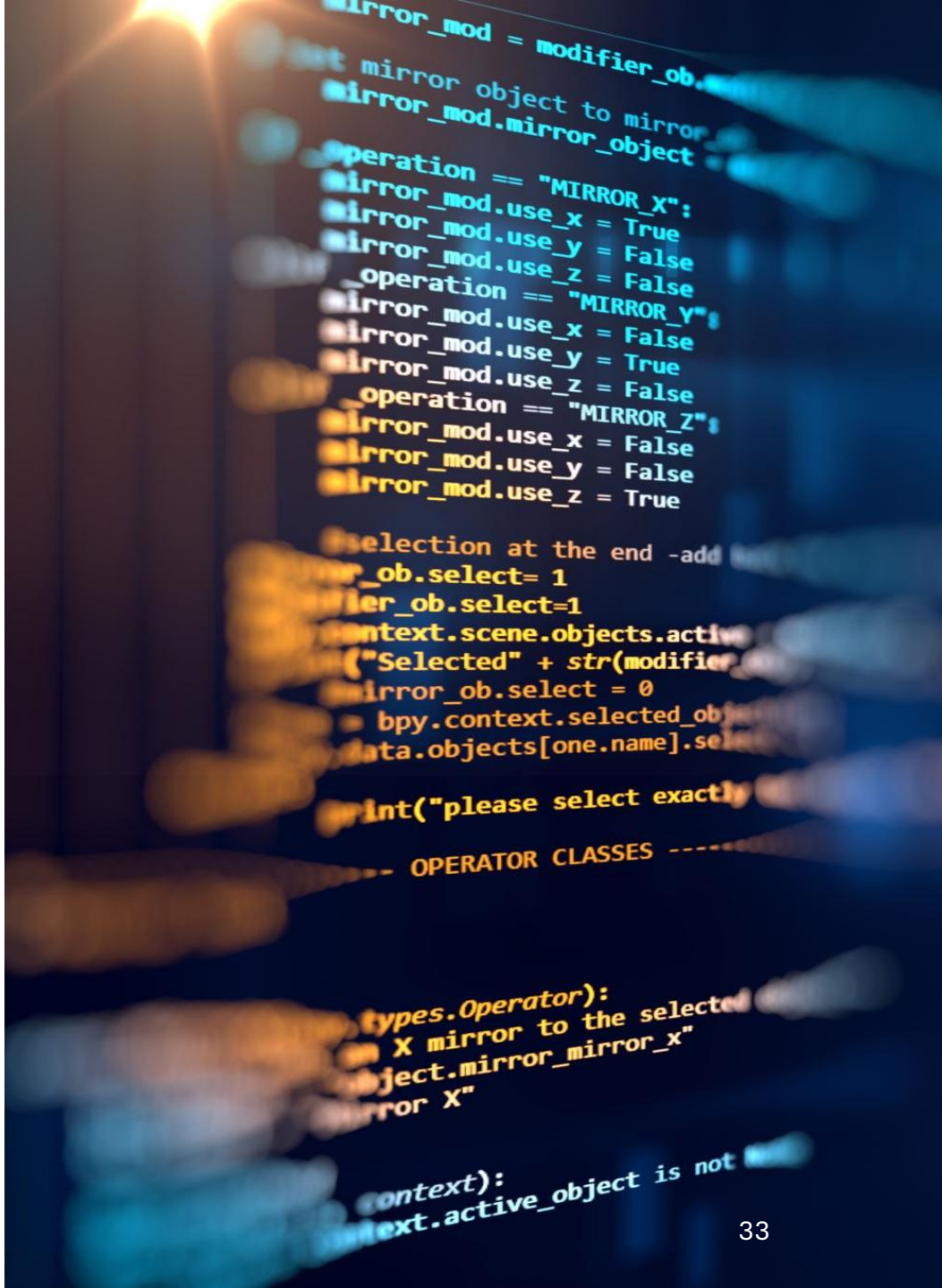
- Prepend a prompt in a Zero-shot setting: “Please analyze if the hypothesis is true or false” and use the following template for an analytical output: prompt + approach + premise + hypothesis + “True or False?”
- ChatGPT models are not “mature enough” for emotional evaluations.
- Emotion-enhanced CoT prompting is an effective method to leverage emotional cues to enhance the ability of ChatGPT on mental health analysis.

Keywords

- When picking the prompt, focus on the subject and style keywords instead of connecting words.
- Pre-appending keywords to prompts are shown to greatly improve performance by providing the language model with appropriate context.
- Modifiers/Keywords can be added to the details or image repository sections of a template such as:
  - “[Medium] [Subject] [Artist(s)] [Details] [Image repository support]”
- The inclusion of multiple descriptive keywords tends to align results closer to expectations.

Wording

- In translation tasks, adding a newline before the phrase in a new language increases the odds that the output sentence is still English.
- A complete sentence definition with stop words performs better as a prompt than a set of core terms that were extracted from the complete sentence definition after removing the stop words.
- Words such as “well-known” and “often used to explain” are successful for analogy generation.
- Modifying prompts to resemble pseudocode tend to be the most successful in coding tasks.
- Prompts to contain explicit algorithmic hints in engineering tasks perform better.



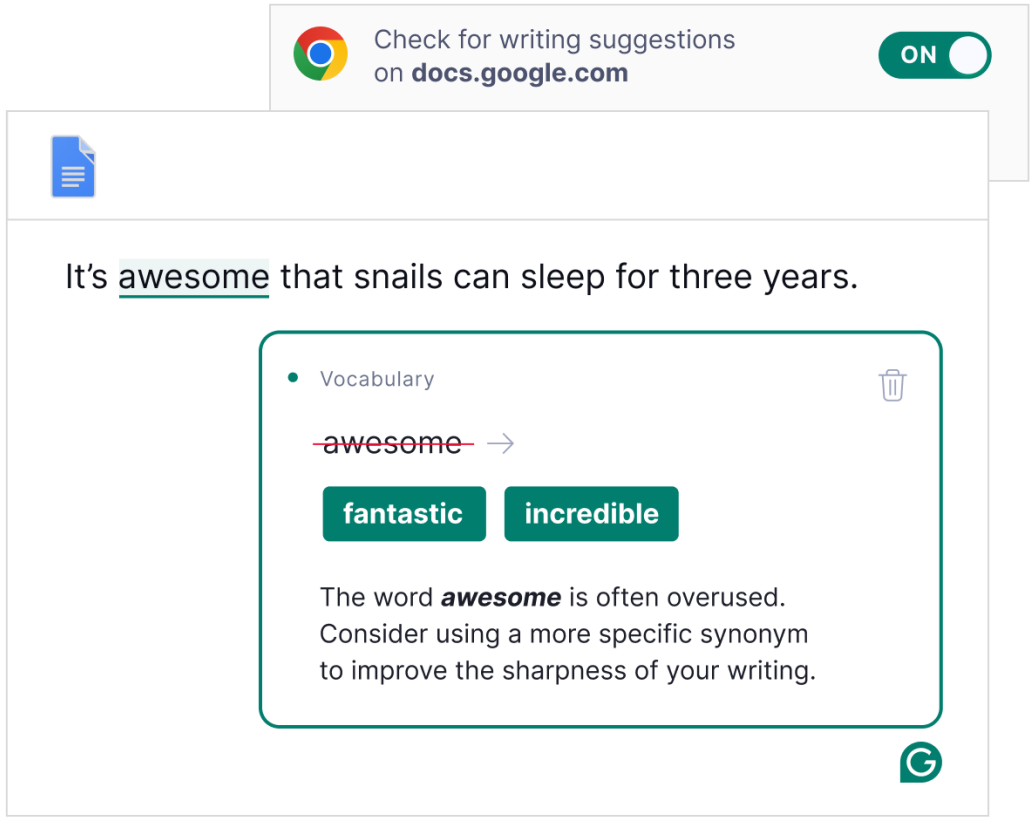
Shorten

- For summarization or text-shortening tasks, the prompt should be written results- and information-oriented, leaving out unnecessary elements.

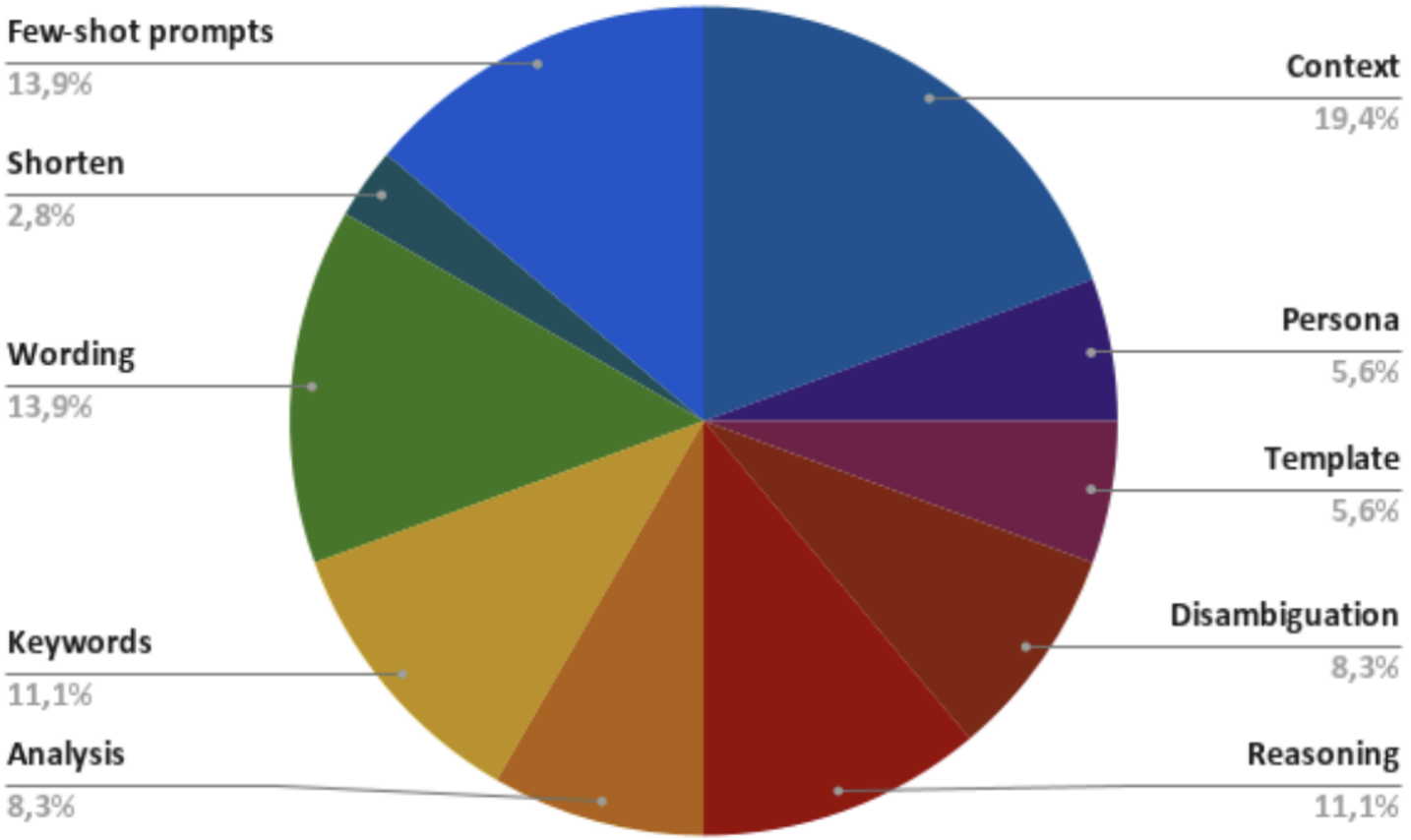
Few-shot Prompts

- Inclusion of “Question:” and “Answer:” improves the response, but rarely gives a binary answer.
- For easier understanding, number examples in few-shot prompting.
- The format of [INPUT] and [OUTPUT] should linguistically imply the relationship between them.
- Specifications can be added to each [INPUT] and [OUTPUT] pair to give extra insight into complicated problems.
- In Few-shot prompting include a rationale in each shot (Input-rationale-output).

Make a Grammerly-like extension?



Theme Distribution From the Studies



Structure Overview

- Project/
  - Interface.py
  - Themes/
    - analysis.py
    - shorten.py
    - context.py
    - keywords.py
    - template.py
    - disambiguation.py
    - persona.py
    - wording.py
    - fewshot.py
    - reasoning.py

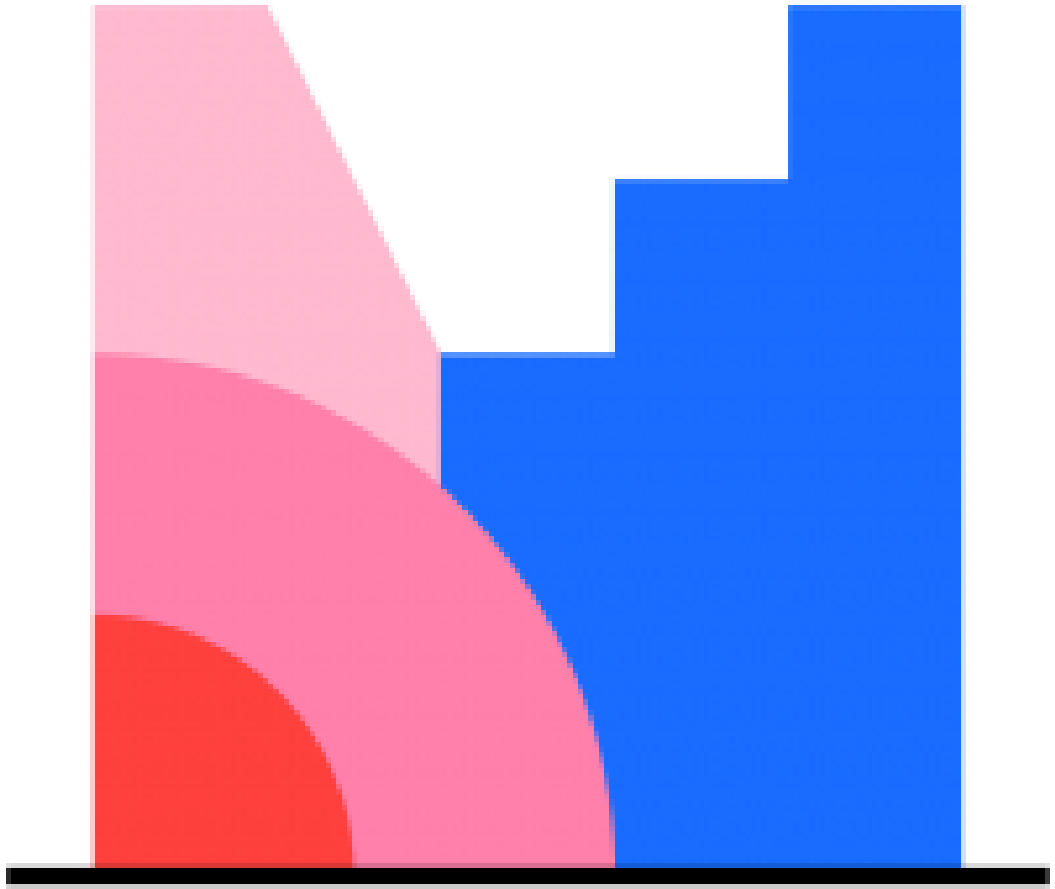
Features of our tool

- Uses Hugging Face to conduct sentiment analysis 🤗
- Uses the theme distribution percentage as a prompt relevance weight
- Uses the score from each theme as the relevance score.

  Rank_score = relevance_score x context_weight
- Returns suggestions provided by the theme file with the highest Rank_score
- Currently Supports Command-Line Interface – plans to integrate with the front-end to work as a Chrome extension.

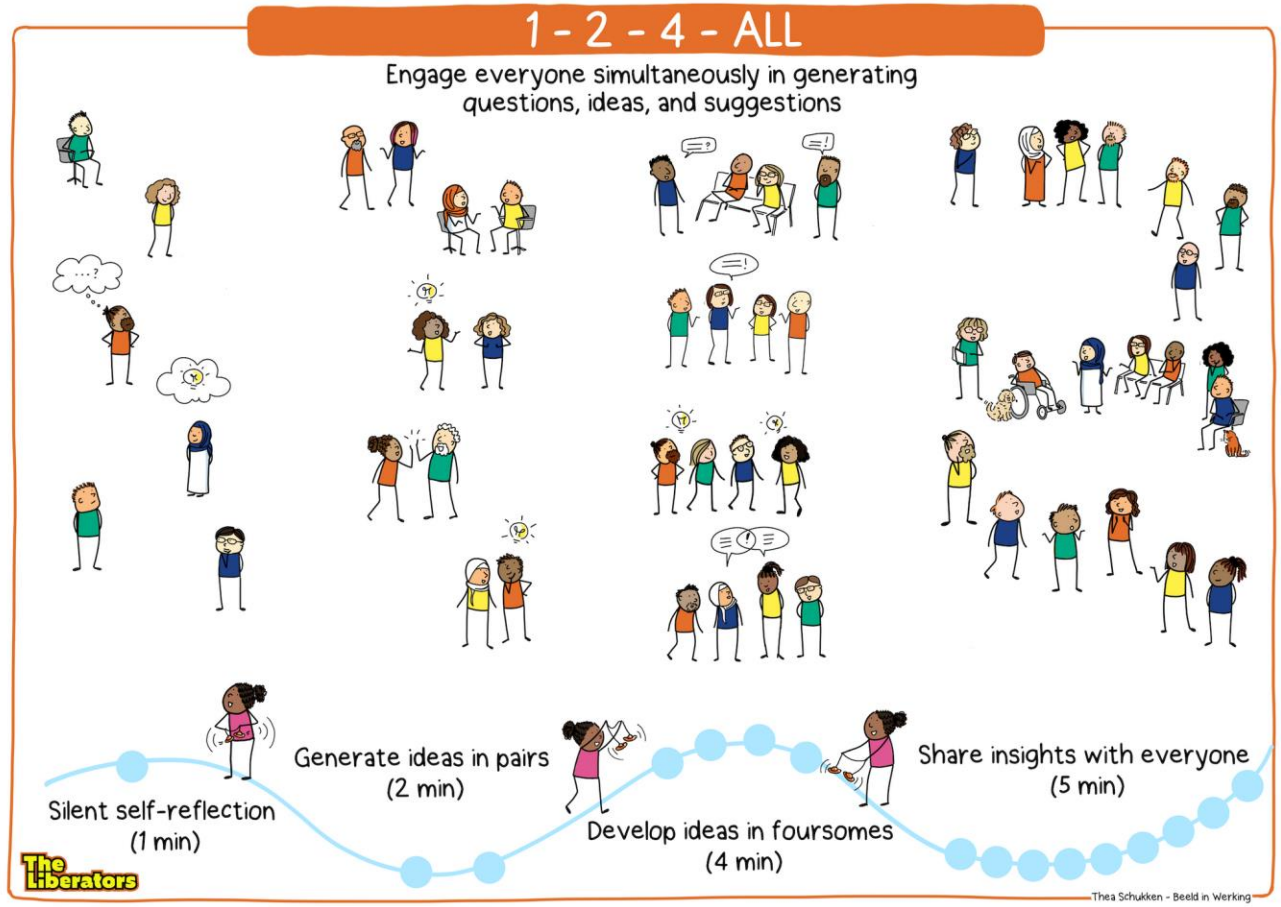
Live Demo Prompts

Access code: XXXX XXXX



Mentimeter

How can we improve our tool?



Our Future Work: User Study

- 1 hour Zoom interview

Time	Activity
0 ~ 15 minute	Generating code without our tool
15 ~ 30 minute	Generating code with our tool with random suggestions
30 ~ 45 minute	Generating code with out tool with ranked suggestions
45 ~ 60 minute	Feedback

- Please let us know in the exit survey if you are interested in participating in our study!

Exit Survey

- On the survey, ask if they would be interested in participating in our tool user study in the near future.

Dinner Time

- Feel free to enjoy the food while talking about RSEs in software engineering!