UNIVERSITY OF CALIFORNIA,
IRVINE

Tabbie: Guiding Novice Developers in Whiteboard Software Design

THESIS

submitted in partial satisfaction of the requirements
for the degree of

MASTER OF SCIENCE

in Software Engineering

by

Kahye Ahn

Thesis Committee:
Professor André van der Hoek, Chair
Professor David Redmiles
Professor James Jones

2024

DEDICATION

To my dearest Mom and Dad,
From across the miles, 5,956.34 to be exact, my heart reaches out to you with a warmth that knows no bounds. Despite the vast distance that physically separates us, your love has been a constant, enveloping presence in my life, as if you were never far away.
Your endless love has shaped me into the person I am today. Thank you for everything.
Your love is my guiding light, illuminating my path no matter where I am in the world.
Here is to the love that knows no distance, to the bond that strengthens with every passing day, and to you, Mom and Dad, my eternal sources of strength and inspiration.
With all my love,
Kahye as known as your beautiful flower.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I extend my deepest gratitude to Professor André van der Hoek, whose guidance and patience have been the cornerstone of my research. Your unwavering support and insightful feedback have not only shaped my academic journey but also bolstered my mental resilience amidst challenges. Your mentorship has been invaluable, and for that, I am eternally grateful.

I am also immensely thankful to the members of my committee, Professor David Redmiles and Professor James A. Jones, who have generously accepted my invitation to participate in my advancement. Your willingness to contribute your expertise and insights has significantly enriched my work.

To my lab friends, your camaraderie and support have been a source of joy and inspiration. Thank you for sharing your profound knowledge of the tool I implemented, and for always being there to lend a helping hand. Your willingness to share insights has not only facilitated my research but also made the lab a nurturing and enjoyable environment. Your friendship and assistance have been invaluable to me.

This journey would not have been the same without the collective support, wisdom, and encouragement of each of you. My heartfelt thanks to all of you for being an integral part of my academic and personal growth.

ABSTRACT OF THE THESIS

Tabbie: Guiding Novice Developers in Whiteboard Software Design

By

Kahye Ahn

Master of Science in Software Engineering

University of California, Irvine, 2024

Professor André van der Hoek, Chair

In software engineering, a whiteboard is a commonly used tool. One particular setting is that of software developers using a whiteboard when they hold software design meetings. They do so because whiteboards offer a fluid and flexible drawing experience. At the same time, the simplicity of the interaction on the physical whiteboard (only drawing and erasing) can be a weakness. Researchers in the past have proposed different variations of whiteboards to make up for this weakness, but to date they have not addressed the fact that novices are prone to becoming lost in their design work precisely because of the fluidity and flexibility. This thesis presents Tabbie, an electronic whiteboard tool with preloaded tabs that are meant to guide novices when they engage in design work. To assess Tabbie, I performed a preliminary experiment with 15 university students designing a ride-sharing service. I present preliminary results that the existence of tabs can be helpful to novices when they work on software design.

Chapter 1

Introduction

Whiteboards are used daily and for many different kinds of tasks by software developers, serving not only as a tool for jotting down quick ideas and listing to do items but also as a collaborative platform for more complicated activities such as system design and debugging. Whiteboards enable developers to visualize complex software architectures, algorithms, and user flows, fostering a shared understanding among developers.

The activity I focus on in this thesis is software design. Software design is a collaborative endeavor. It requires collective expertise of diverse team members to conceptualize, iterate, and refine a software design. This cooperative approach is essential for addressing the multifaceted challenges of software development.

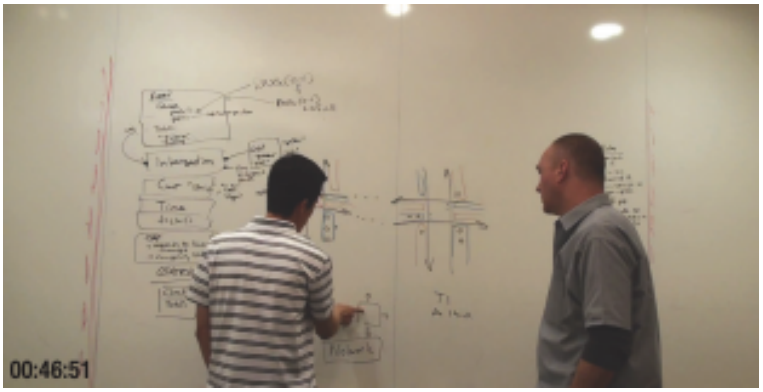


Figure 1.1: Developers working at the whiteboard [28]

The collaborative nature of the software design process involves brainstorming, idea sharing, and informal sketching on whiteboards [8] [13] [17] [24]. Developers draw abstract concepts, assess different alternatives, and converge on the design decisions. Through this process, whiteboards have become an important tool in software design.

There are multiple reasons why software developers turn to a whiteboard when they design. First and foremost is the user-friendly nature. Whiteboards provide simple functionality in that people do not need extensive training to learn how to draw and erase. Whiteboards allow individuals with varying levels of technical expertise to engage effortlessly in the creative process. Second, whiteboards are physically present in most workspaces. Anyone can readily pick up a marker and start sharing ideas nearly instantly. Unlike digital tools that may need user authentication and authorization, whiteboards have no such barriers. Third, whiteboards are flexible in terms of the contents. There is neither rule nor format to follow on whiteboards. Software developers can freely illustrate concepts such as pseudo code and diagrams, accommodating the diverse range of software design notations.

Besides these benefits, researchers have also identified some drawbacks of whiteboards that potentially impede productivity when designing. First, whiteboards have limited space, which can restrict the free expression of ideas [12]. Once it starts being full, people need to erase what they drew to make room for additional information. Second, the fact that

whiteboards involve arbitrary handwriting can lead to illegible content, which can hinder effective communication and recall [30]. Each person has their own handwriting and it sometimes makes others not able to understand what is drawn. Third, whiteboards also lack support for formal notations essential for some software design tasks [6]. While informal sketching is important [23], sometimes more formal representations are needed. For example, UML diagrams need accurate shapes and arrows, but it is cumbersome to manually draw at the kind of precision needed.

In attempts to address these challenges, researchers have proposed different features that leverage electronic or virtual whiteboards to help software developers produce more productive and meaningful outcomes. Some research, to make up for limited space, introduced a whiteboard which lets users add multiple canvases [20] [14]. As another example, VisionSynaptics uses pattern recognition to convert hand-drawn sketches into polished computer graphics for addressing issues related to poor handwriting [7]. As a final example, some research suggest gesture-capturing to integrate formality with informality of whiteboards [12]. This approach recognizes the user's sketches and turns them into objects in UML.

To date, however, an area that has been overlooked is that of novices: novices have varying levels of expertise and work in ways different from experts [25].

This thesis addresses this gap by focusing on novice software developers who may feel the difficulties of having meetings at a whiteboard. By novice software developers I refer to both university students pursuing software development related degrees and software practitioners who design software systems but do not have much experience in the actual practice of software engineering [31].

For this population, the lack of structure or guidance can be a hindrance to understand how to get started or learn how to proceed. I hypothesize that these developers can be helped by providing them with reminders of important aspects to think about when designing.

Particularly, I developed a tool that reminds developers of the need to consider stakeholders, design goals, ideas, alternatives, and detailed design, as well as not to forget next steps. I believe that a tool that suggests these design aspects can guide novices to move through design challenges with direction and confidence, preventing them from feeling lost.

This thesis presents Tabbie, an electronic whiteboard with six preloaded tabs of predefined design activities termed: stakeholders, design goals, ideas, alternatives, detailed design, and next steps.

The idea is that by having these tabs present, they implicitly invite the developers to use them. Doing so is not mandatory, however, so not to overly constrain the developers in their work and approach. To provide a preliminary assessment of the idea, I conducted an experiment involving 15 university students total. Participants were asked to design a ride-sharing service. After the design task, I interviewed each pair of participants to understand how they perceived Tabbie while they undertook the design task.

Overall, I found that the tab feature of Tabbie was well-received, although there was variability in how participants used the tabs. Preferences for specific tabs emerged, which are consistent in usage patterns and interview feedback. There are areas for improvement and additional features that could enhance the tool's functionality.

The remainder of the thesis is structured as follows. Chapter 2 reviews related literature, establishing the context of the thesis. Chapter 3 outlines the high level design of Tabbie. Chapter 4 provides the details of assessment design. Chapter 5 presents the findings from the assessments, including user interactions and feedback.

Chapter 2

Background

This chapter introduces relevant research concerning the role of whiteboards in software development, existing whiteboard research, some online whiteboard tools, and research about novices.

2.1 Role of the Whiteboard in Software Design

Software design is a collaborated activity and the whiteboard serves as an important conduit in this context. In this thesis, the term "whiteboard" encompasses various forms, including physical, electronic, and online whiteboards. Whiteboards are invaluable tools for both in-person and remote design meetings. Whether physical or electronic, they support real-time communication and facilitate idea-sharing [28] [9]. Remotely, electronic whiteboards enable working together on a design in the first place: without them, remote design collaboration that is synchronous is very difficult [15].

Whiteboards play a crucial role particularly in the ideation and conceptualization phases of software design. During these phases, designers typically produce and interact with sketches

on whiteboards, since sketches can support a wide range of reasoning activities and allow rapid iterations of design concepts [19]. For example, designers can talk through the execution of a design idea to evaluate its feasibility. This activity transform their mental representation into the design at hand.

Whiteboards are also central to problem-solving and decision-making processes in software development. The informal sketching facilitates a free-flowing exchange of problems and ideas without the constraints of formal notation [8]. An example includes brainstorming sessions where developers sketch out different design artifacts or user flows, having collaborative refinement of ideas in real-time. Simplicity and immediacy of whiteboard sketches enables quick adjustments based on feedback or new insights.

While whiteboard usage is often associated with "new" design, they are also used for maintenance meetings, where developers address changes to existing software [1]. In these settings, whiteboards can be used to draw the current state of the system and overlay any modifications. Whiteboards can even be used to address bugs with visually drawing issues, brainstorming solutions, and debugging.

2.2 Existing Research About Whiteboard

Despite of wide use of whiteboards, numerous studies identified weaknesses in the traditional appearances of whiteboards and sought solutions to overcome these limitations.

As one direction of research, in whiteboard meetings, where content is fleeting and prone to loss, research has sought methods to capture and preserve whiteboard content effectively. Tools such as ReBoard automate the capture of whiteboard content as images [4], while KNOCAP focuses on capturing the important audio snippets of critical moments [29]. Related to saving content, LivelySektches addresses the iterative nature of the software design [2].

Specifically, it allows for linking sketches to related documents and managing version control for evolving sketches.

A second area of research counters the lack of formality on whiteboards. Some research, for instance, has developed electronic whiteboard software, including formal notational elements that can easily be dragged and dropped onto the canvas [11]. Another approach turns hand-drawn sketches into their more formal equivalents through sketch recognition. Skemo, for instance, is a web-based model editor where hand-drawn sketches are converted to formal model components [5]. SUMLOW takes a different approach. It preserves the hand-drawn appearance, but internally formalizes the elements into computer-recognized diagrams, so it can perform a range of checks for correctness of the diagrams [6]. SILK recognizes hand drawing and transforms into interface [18].

2.3 Online Whiteboard Software

There are several whiteboard tools easily accessible online, each with unique strengths, alongside supporting essential draw and erase functionalities. ClickUp Interactive Whiteboard[1] excels in document integration and linking, making project management more efficient. Miro[2] stands out with its extensive collection of templates and the capability to import a wide range of content including Google Images. Lastly, Jamboard[3] allows users to create multiple boards and switch among them smoothly, making it ideal for teams needing to segment their work.

[1] https://app.clickup.com/
[2] https://miro.com/
[3] https://jamboard.google.com/

2.4 Novices

Some research identifies differences in problem-solving strategies, design thinking, and project planning capabilities between novices and experts [26] [16].

First, an observational study of new software developers reveals their initial challenges. This study identifies common difficulties faced by novices including issues with communication, collaboration, and technical skills [3]. For example, they struggle with handling legacy code, understanding user needs, applying professional practices like Agile, and utilizing tools like version control [10] [27].

To address these kinds of difficulties, various tools and frameworks have been proposed. One example is a framework called Principles Patterns and Process Framework (P3F) [32]. This framework is designed to help novices adopt expert design strategies and improve their design skills by suggesting guidelines and a step-by-step approach, and chunking information into manageable pieces. Another example introduces a tool designed to support novice developers by providing metacognitive scaffolding around a problem-solving framework [22]. Through a structured process, it guides users to understand problem and evaluate solution. There is another tool to assist novice developers with planning and executing open-ended programming projects by breaking down factors of coding to begin a programming task [21]. These interventions aim to equip novices with a structured approach to tackle software development challenges. However, such structured interventions have not been explicitly applied to whiteboard software design, highlighting a potential area for research to enhance novice capabilities in this specific context.

Chapter 3

Tabbie

This chapter introduces Tabbie, an electronic whiteboard prototype tailored to assist novice software developers in software design meetings. The following sections will unveil the objective of Tabbie, its high-level design, how its features aim to achieve its objective, and an overview of the tool's interface.

3.1 Objective

Tabbie is an electronic whiteboard prototype designed to guide and support novice developers with a tab-based approach. The primary goal of Tabbie is to guide users into having a comprehensive design discussion by providing them with a clear direction, from requirements discovery to planning future tasks. Research show that novice developers tend to focus on isolated aspects of design, potentially overlooking the wider range of necessary elements for a robust design [32]. Tabbie tackles this problem by structuring the design conversation through predefined tabs, each representing an important area of focus in software design discussions.

3.2 High-level design

The foundational design principles underlying Tabbie are captured by a straightforward virtual whiteboard user interface that is extended with a set of tabs integrated into the basic functionalities.

3.2.1 Predefined Tabs

In Tabbie, there are six tabs preloaded with contents: (1) stakeholders, (2) design goals, (3) ideas, (4) alternatives, (5) detailed design, and (6) next steps. These tabs were selected to align with key aspects that typically need to be considered. The order of the tabs in Tabbie is also a deliberate design choice, reflecting a logical progression in thinking through design problem.

In software design, the nature of design problems can vary significantly. The design problems range from those that are user-facing to system challenges that deal with the underlying architecture. In addition, the scope of design may vary. Some design problems involve developing entirely new software, while others revolve around modifying or maintaining existing systems.

With Tabbie, I specifically target user-facing problems in designing new software. User-facing problems are more approachable to beginners compared to the intricate and hidden intricacies of system architecture. Designing user-facing features offers tangible outcomes that novice developers can directly observe and interact with. Furthermore, while existing software usually comes with comprehensive documentation, and a pre-existing and well-defined problem scope, new software projects are likely to require a broader scope of innovation but present more ambiguity in requirements. This ambiguity can be particularly challenging for novice developers who may feel difficulties with finding a starting point in the design process.

Below, I describe each of the six tabs that Tabbie provides in more detail. To illustrate the intent of each tab, an example of developing an e-commerce service will be used throughout.

Stakeholders

This tab focuses on identifying primary people, groups, and organizations involved in or affected by software. It is crucial to understand stakeholders and their needs and concerns, because they can significantly influence and shape the final design. This tab encourages Tabbie's users to consider possible stakeholders and how they may impact the design project. It is important to start with this tab because missing important stakeholders' needs and concerns could lead to a design addressing the wrong issue.

In the context of the example of e-commerce development, stakeholders can be vendors who list their products on the platform, customers who navigate the platform to buy these products, IT and sales teams who manage and operate the platform's administrative aspects, and more. Under this tab, recognizing and addressing these diverse stakeholders is encouraged. Sometimes, it might be okay to just list them, as a reminder. Other times it might be important to also work out the roles, goals, and needs of each stakeholder. Regardless, this tab provides the space to do so.

Design goals

Following stakeholder analysis, this tab allows Tabbie's users to articulate the goals for the software based on the needs and concerns of the various stakeholders. This tab also helps the users in creating a shared understanding of the overall system goals and what they are working towards throughout the meeting.

Designing an e-commerce service, the needs and concerns of stakeholders such as customers

and vendors are to be considered. For example, the software should incorporate an intuitive and seamless shopping experience for customers, with easy navigation and quick product discovery. For vendors, it equally should be easy to use, but it also should enable them to track their sales. Often conflict exists among different stakeholders. This tab is where those are reconciled, trade offs are made, and an overall direction is formed.

Ideas

The third tab is the place for brainstorming and discussing potential ideas for addressing the design problem. Developers can explore different ways to achieve the design goals. Since the previous tab establishes the design goals, they can bring more goal-oriented and relevant ideas. This tab encourages them to have innovative and creative thinking and it also records the process of brainstorming which will be a good future reference.

For an e-commerce service aiming for a seamless shopping experience, numerous ideas can emerge. In this example, developing systems for recommendations, customer support, and advertising can be considered. A recommendation system can utilize AI to analyze shopping behaviors, incorporate expert opinions, and curate choices based on customer preferences. Customer support can include AI chatbots for immediate assistance, human-operated live chats, and traditional helplines. Advertising can be powered by AI leveraging customer history for more relevant advertisements or user input on preferences. The purpose of this tab is to collect ideas like this, whether or not they are related and whether or not they are compatible with one another. Note that these ideas do not necessarily align, but result from brainstorming candidates that may be useful later.

Alternatives

Software design often involve evaluating and combining multiple ideas into high level alternative approaches, instead of a single idea to reach a conclusion. As derived from the 'Ideas' tab, at the 'Alternatives' tab, users can compare different combinations of ideas and options as alternative high level approaches. While working through these, developers can identify the best approach to take forward.

To continue the example, by combining AI recommendation, advertising, and chatbot customer support, one alternative would be a high level concept centered on AI. This integration makes real-time updates of recommendations and advertisements based on customer data, enhancing the shopping experience. A curation-based approach would be another alternative, which leverages customer interests, preferences, and demographic information to tailor recommendations and advertisements. This strategy aims to enhance the shopping experience by presenting the most relevant products and content to each user, positioning it as another competitive approach. The curation-based approach excels in deeply personalized experiences based on explicit customer inputs but might not scale as quickly as AI in real-time personalization. The choice between them depends on prioritizing scalability versus depth of personalization.

Detailed design

After the selection of the alternative with which to move forward, this tab is where the developers can focus on the details of the specific solution. The abstract ideas and concepts are made into a concrete and detailed form of design. By drawing diagrams, workflows, user interfaces, or parts of software architectures, and other items as needed. As they delve into the design in detail, they can discuss technical and functional details.

This tab, then, is where the detailed solution takes shape and many decisions are made that then later can be taken forward away from the whiteboard to realize the design in the software.

For instance, for the AI-driven e-commerce website, detailed design can involve specifying the architecture for the AI recommendation engine, such as the data models and machine learning algorithms. Additionally, the user interface for displaying personalized advertisements and the user flow for chatbot interaction with customers could be addressed at this tab.

Next steps

In software design, a single meeting is typically not sufficient to address all the necessary requirements. This tab is designed to capture the information needed to later remember the next steps to be taken, allowing users to document what has been accomplished and what still needs to be done. In other words, through this tab, users can document current progress and future plans. Proper usage of this tab can help a software design project stay on track.

For the ongoing development of the e-commerce service, the 'Next steps' tab might list items such as optimizing the AI algorithms reflecting the algorithms users discussed in the 'Detailed design' tab, exploring additional AI capabilities such as predictive analytics for inventory management, or planning for AI system scalability to manage traffic peak.

3.2.2 Adding tabs

Tabbie includes a feature for adding tabs, which accommodates the diverse and evolving needs of software design meetings. The predefined tabs cover fundamental factors involved in typical design discussions, however, they cannot always encompass every aspect of software design because software design can be highly varied. Therefore, Tabbie enables users to add a

tab with their own title. This feature can be helpful for two reasons. First, adding tab feature enhances Tabbie's versatility to the varying spectrum of software design meetings. While the predefined tabs offer a structured approach, the spectrum of software design is too broad for a one-size-fits-all approach. Among the predefined tabs, assumptions and constraints are not directly addressed, however, they can sometimes be crucial considerations for some software design projects. And, in the example of e-commerce website, it is important to consider regulatory restrictions on selling certain products and security measures to protect customer data. Being able to add tabs to address some other needs is important.

A second reason for the ability to add tabs is the practical aspect of additional space. While people discuss the complex software design, it is easy to fill up the space in a single screen. By allowing users to add extra tabs, Tabbie gives them to opportunity to expand their workspace as needed. This is beneficial for a situation where users need more extensive visual representation. For example, the 'Detailed design' tab can be out of space quickly, because detailed design includes various design representations. Users can add tabs to cover detailed design as much as they need. Users can continue their work without erasing the contents or reorganizing the contents.

3.3 User Interface

This section introduces the palette, an important tool set for drawing and annotation, and tab switching, a feature that enhances the navigation of the design workspace.

3.3.1 Palette

Users are provided with a palette on the left of the screen(A in Figure 3.1). This palette includes a pen, eraser, line tool, circle, rectangle, text box, and sticky note. Users can

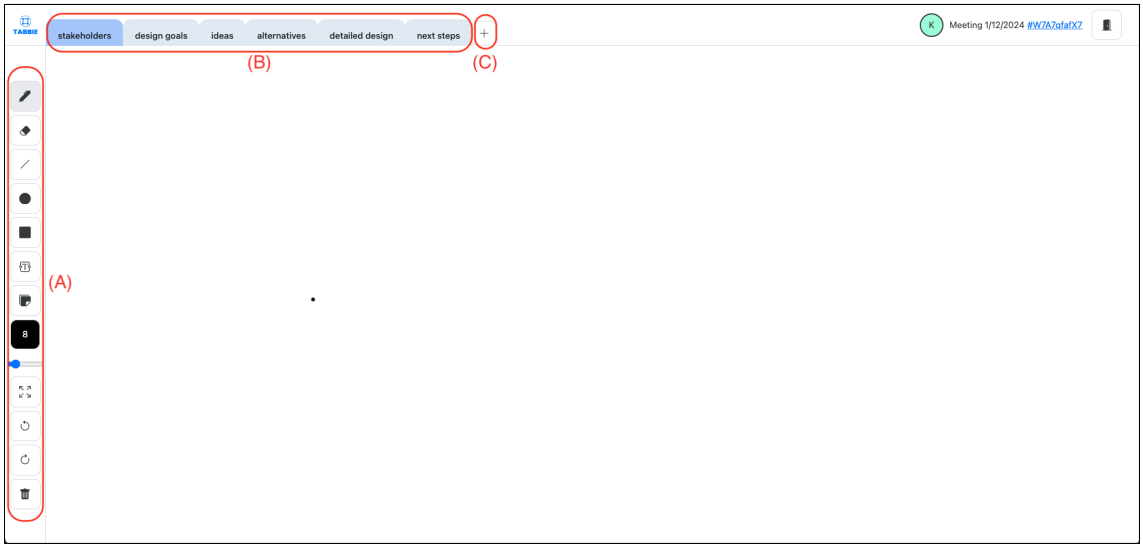


Figure 3.1: A screenshot of Tabbie

customize these tools for their needs: the pen, eraser, and line tool come with options to adjust color and thickness. For text boxes and sticky notes, users can adjust font sizes, and a sticky note's color can be changed for visual differentiation. Users can use a full-screen mode, maximizing the workspace for focusing their efforts without other distractions. Tabbie also includes the typical redo and undo functionalities, providing an ease in the design process. In addition, users can clear the entire whiteboard with a single click to give them a fresh start.

3.3.2 Tab Switching

At the top of the whiteboard, there are six distinct tabs(B in Figure 3.1). Users can navigate to any of the tabs at their convenience by clicking a tab. Upon selecting a tab, Tabbie offers initially a fresh and blank whiteboard workspace specifically assigned to that tab, independent of any work done in other tabs. Tabbie preserves content and no data is lost in the transition when switching among tabs. This ensures that the content created in one tab remains intact and undisturbed. By adding a new tab(C in Figure 3.1), users are able to create another blank whiteboard space and switch to that tab.

3.4 Implementation

Tabbie is a web application and it is accessible on various devices such as mobile phones, tablets, and desktops. The project was forked from a GitHub repository: `https://github.com/ameza13/knocap`. Several features were modified. A feature of capturing voice was removed and tabs were added at the top. The front-end was developed using Vue.js and TypeScript, while the back-end employed Node.js, Express.js, Peer.js, and Socket.io. Docker was used for deployment.

Chapter 4

Preliminary Assessment Design

4.1 Objective

I performed a preliminary assessment of Tabbie. The primary goal of the assessment was to examine how Tabbie impacts the design practice of novice software developers and investigate how novices perceive Tabbie.

The assessment was designed to provide initial observations on how Tabbie can be utilized by novice developers in software design meetings. It aimed to gain a first insight into how novice developers perceive Tabbie as a tool for software design. This involved observing their interactions with the tool as well as interviews to uncover initial impressions and the level of satisfaction with the features, together with any challenges the novice developers encountered in using Tabbie.

4.2 Participants

The assessment was conducted with 15 participants, comprising both three graduate and twelve undergraduate students. These students have academic backgrounds in computer science and software engineering, but have limited practical experience in industry, either none at all or less than a year. Their educational background lets them equip them with a basic foundation in software design and most of them engaged in at least one school project or side project. Their experience level places them in the category of novices.

Table 4.1: Demographics of the participants

	Degree	Major	Year	Professional background	Software design experience
P1	Undergraduate	Informatics	4th	No	School project
P2	Undergraduate	Informatics	4th	No	School project
P3	Undergraduate	Informatics	2nd	No	School project
P4	Undergraduate	Informatics	3rd	No	School project
P5	Graduate	Software Engineering	1st	No	School project
P6	Undergraduate	Computer Science	3rd	No	School project
P7	Graduate	Software Engineering	2nd	Yes, internship	School project
P8	Graduate	Software Engineering	2nd	Yes, internship	School project
P9	Undergraduate	Computer Science	3rd	No	School project
P10	Undergraduate	Computer Science	2nd	No	School project
P11	Undergraduate	Computer Science	3rd	No	No
P12	Undergraduate	Software Engineering	4th	Yes, internship	School project
P13	Undergraduate	Computer Engineering	2nd	No	Personal project
P14	Undergraduate	Computer Science	3rd	No	School project
P15	Undergraduate	Computer Science	3rd	No	School project

4.3 Procedure

In the assessment, participants were paired into teams comprising two individuals, with one exception where a team was formed with three participants because an extra person showed up spontaneously.

We arranged for participants to be in a room equipped with a large smart whiteboard and two electronic pens running Tabbie. Figure 4.1 shows an image capturing the participants

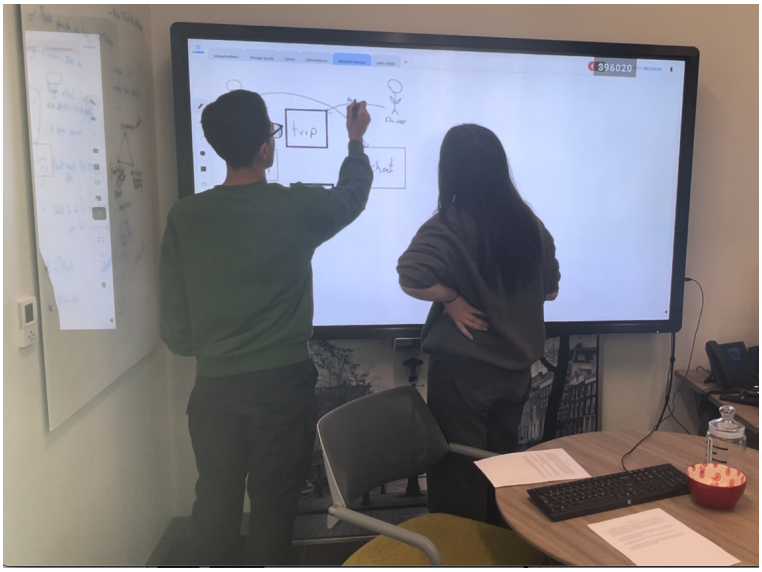


Figure 4.1: An image of a pair of participants during the assessment.

engaged in the assessment. Even though Tabbie is compatible with various devices, this setup was chosen because the board is the approximate size of a regular whiteboard and draws approximately as smoothly.

The session began with the introduction of Tabbie. I briefly explained the functionalities of the tool, including the use of pen, eraser, circle, rectangle, text box, and sticky note. Following this, I also explained the tabs, clarifying the meaning of each tab and the feature of adding a tab.

After this briefing, the team was given some time to freely explore to familiarize themselves with the features of Tabbie. This phase was important for them to become comfortable with the interface and functionalities before starting the design task.

Once the team felt ready, I gave them the software design task. This was the core of the experiment, where the team interacted with the tool for a practical design task. The total duration of the experiment was set at 1 hour and 30 minutes, with approximately an hour devoted to the briefing and the design task, and 30 minutes for the debrief of the work. Most teams spent about 40 to 50 minutes on the task. I recorded the interactions between the

users and the tool.

At the end of the experiment, I conducted an interview with the team. This interview aimed at gathering feedback, focusing on their experience using Tabbie.

4.4 Design Task

The team was given a design prompt. Appendix A.1 contains the detailed design prompt given to the participants. This design prompt asked the participants to design a ride-sharing service specifically for children and is modeled after an assignment in a software design class at University of California, Irvine. The design task is open-ended, which is a characteristic of new software design. The participants were asked to develop a high-level architecture ensuring a specific requirement of transporting children safely to various destinations.

4.5 Post Interview

The debriefing began by asking participants' background in software design. For a detailed list of the interview questions, please refer to Appendix A.2. We inquired about their experience in this field, asking the years of involvement and description of the nature of their work in software design. This helped us to understand their baseline expertise and perspective and ensured that they were good candidates for this study.

I inquired about the participants' experiences during the design exercise with Tabbie. Participants were asked to reflect on how they felt about their overall design work. An important point of the interview concerned the impact of Tabbie's tabs on their workflow. Participants were asked to explain the reasons for not using certain tabs, as well as adding or not adding new tabs.

We sought to understand the participants' perceptions of the predefined tabs' purpose and how these influenced their design process. Other questions were about their potential willingness to use Tabbie for future work and whether they would recommend Tabbie to others involved in design work.

Lastly, we asked for any suggestions or improvements they had for Tabbie, as well as any other feedback they had on the tool that had not been covered.

Chapter 5

Evaluation

This chapter is structured into two parts. The first part focuses on analyzing how participants engaged with Tabbie's features, including their usage patterns of the tabs, the time spent on each tab, the order of tab usage, and instances of adding new tabs. In the second part, I will examine the content of the interviews to understand participants' perceptions of Tabbie.

5.1 Feature Use

This section presents an analysis of feature use in Tabbie through a series of questions and answers. It provides an overview of how participants used various functionalities offered by Tabbie.

In Table 5.1, I detail the tabs used by each team. The first column lists the seven teams, while the first row identifies the specific tabs. Within the table, an 'O' marks the intersection of rows and columns to indicate what team used a particular tab.

Table 5.1: Tab usage table

	stakeholders	design goals	ideas	alternatives	detailed design	next steps	new tab
Team 1	O	O	O	O			
Team 2	O	O	O	O			O
Team 3	O	O			O	O	
Team 4	O	O	O		O		O
Team 5			O		O		
Team 6		O	O				
Team 7	O	O	O			O	

Q1. What tabs did they use? As the table shows, all of the participants actively used the tab feature in Tabbie. There is no tab that was not used at all. All teams used at least two different tabs during the design activity, and all the tabs were used by at least two teams.

Q2. Which tab was used most often? There are three ways to interpret the tabs that were used most often. The first is considering the tabs visited by most teams. The second takes into account which tabs were visited the most often, and the third is by looking at which tabs had the longest engagement time. The trend was consistent in terms of number of teams, visits, and the duration of time spent, with 'Ideas' and 'Design goals' being top-ranked.

Referring to Table 5.1, 'Design goals' and 'Ideas' stand out as the tabs visited by the most teams, with six out of seven teams interacting with them. 'Stakeholders' was next tab, being used by five teams. 'Detailed design' also saw notable usage, with three teams using this tab. On the other hand, 'Alternatives' and 'Next steps' were the least popular, with only two teams visiting these tabs.

Participants did not follow a linear workflow, instead, they moved back and forth between the tabs, making first time visits and revisits. This resulted in different frequencies of visiting to each tab. As Figure 5.1 shows, the most visited by the participants were 'Design goals' and 'Ideas', which is consistent with these two tabs being visited by most teams as well.

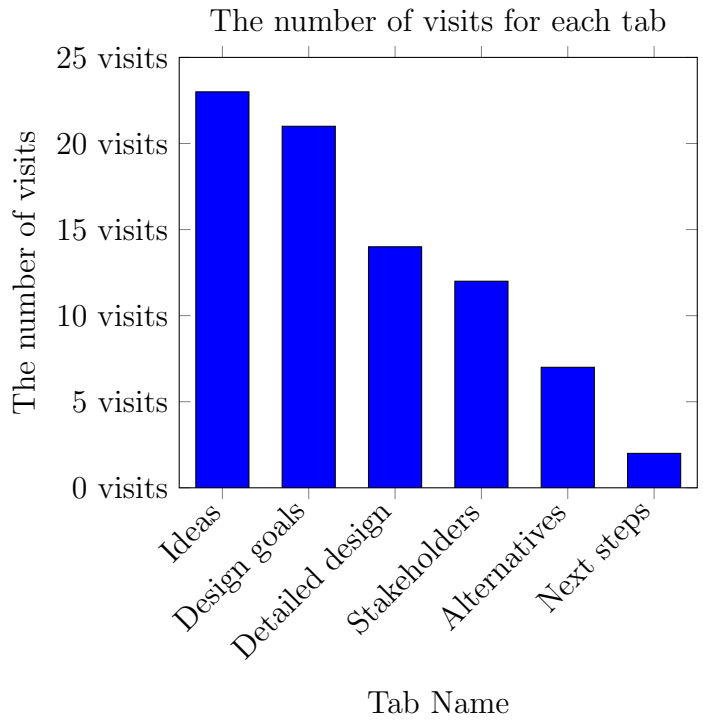


Figure 5.1: Bar graph showing the number of visits on each tab.

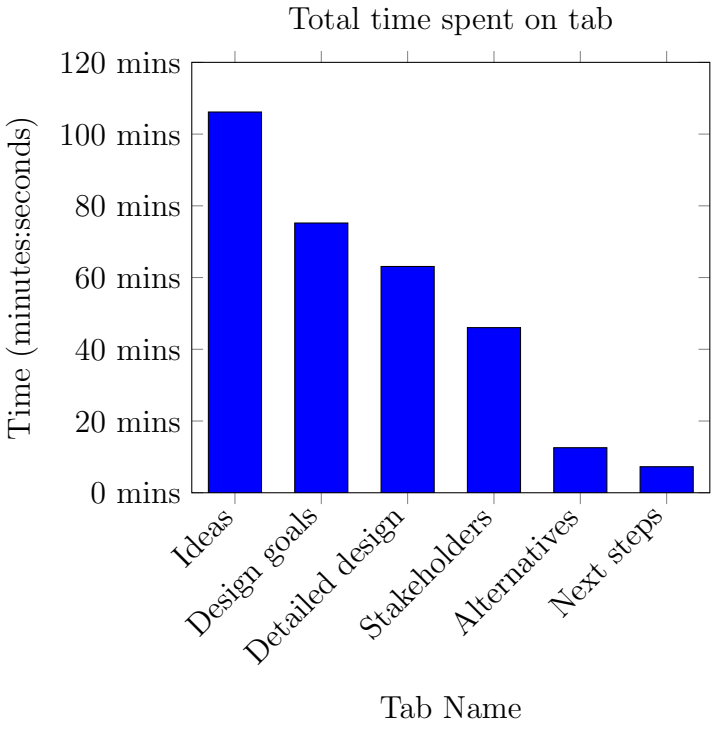


Figure 5.2: Bar graph showing total time spent on each tab.

When it came to the total amount of time spent on each tab, the 'Ideas' tab was visited the longest, followed by 'Design goals', and 'Detailed design'. For a comprehensive overview of the time usage across tabs, please see Figure 5.2. Four teams spent the majority of their time on the 'Ideas' tab. 'Design goals' received the most time from two teams, and 'Detailed design' was where one team dedicated most of their time. It indicates that participants dedicated the most time to brainstorming and developing their ideas.

Q3. What was the order of tab usage? From Figure 5.3 to Figure 5.9, an overview of tab usage is provided. The Y-axis displays the initial visitation order. The X-axis is the time and shows how participants moved among tabs as time passed.

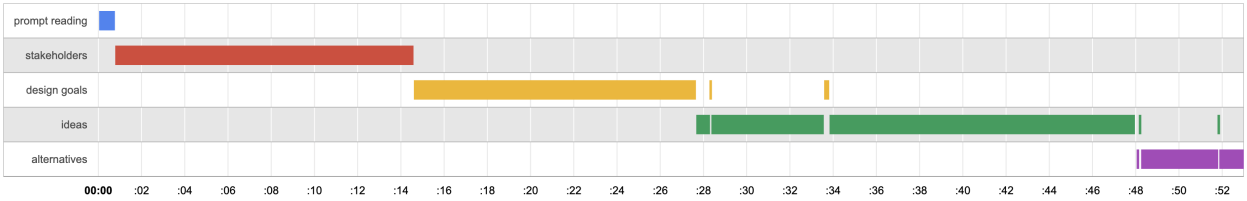


Figure 5.3: Team 1's Tab Usage Timeline.

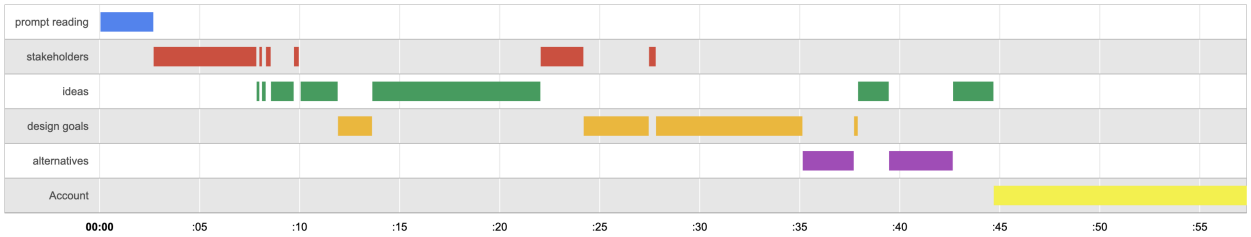


Figure 5.4: Team 2's Tab Usage Timeline.

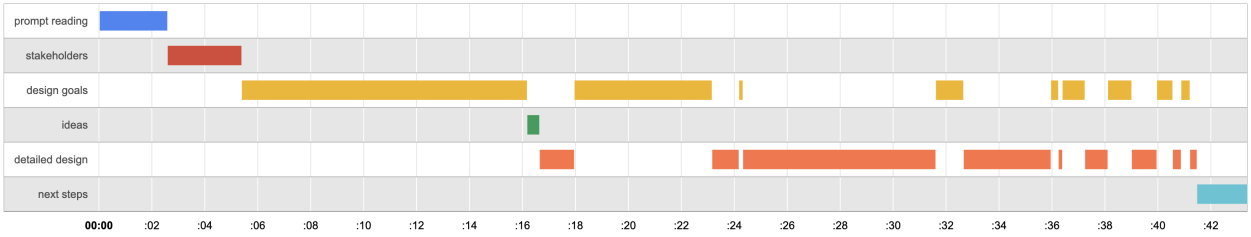


Figure 5.5: Team 3's Tab Usage Timeline.

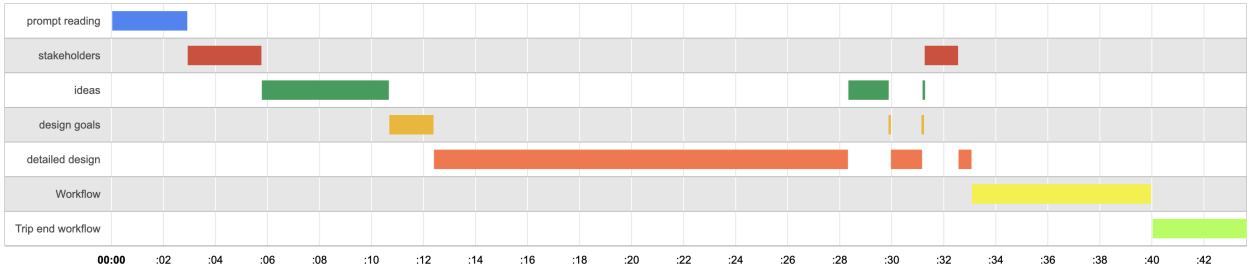


Figure 5.6: Team 4's Tab Usage Timeline.

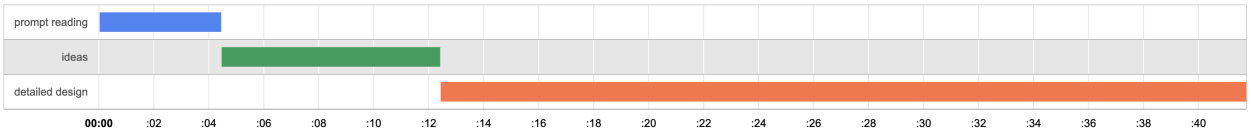


Figure 5.7: Team 5's Tab Usage Timeline.

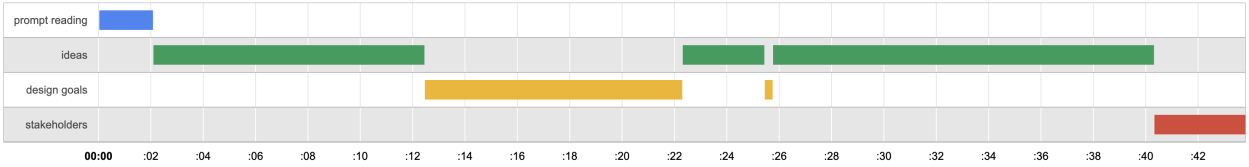


Figure 5.8: Team 6's Tab Usage Timeline.

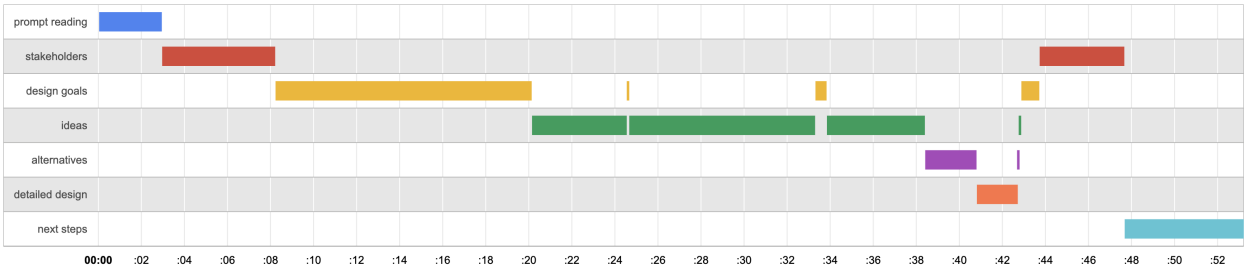


Figure 5.9: Team 7's Tab Usage Timeline.

Discounting any tab revisits, the teams adhered to the sequential order of the tabs overall. Notably, three teams followed the tabs in their exact order, initiating the design process with the 'Stakeholders' tab and proceeded linearly. This initial step aligns with the intended design flow of the tool, because understanding stakeholders' needs and concerns is crucial for setting a solid foundation for software design. One team of these three teams chose to

skip the 'Alternatives' tab while still proceeding through the other tabs in sequence. This observation suggests that the order of predefined tabs was largely respected, and teams also tailored their use of tabs to fit their design needs and perspectives.

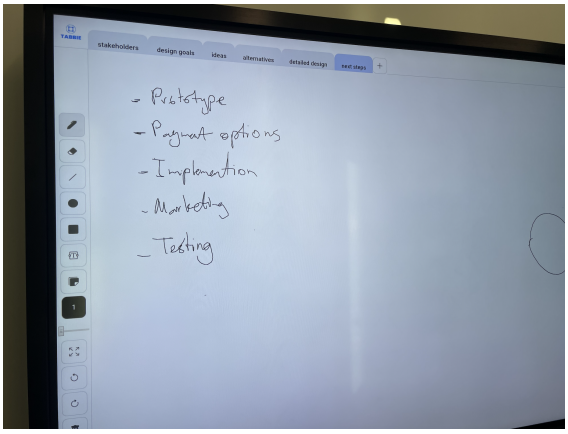
Q4. Did they revisit tabs and for what purpose? All teams, except one, revisited tabs. All these teams reviewed and reflected on the content they created. One team expanded the ideas by reviewing 'Design goals' and drawing under the 'Ideas' tab. Another action that these six teams showed is that they drew when they revisited. For example, discussions within the 'Detailed design' tab led to revisiting the 'Ideas' tab to refine concepts based on new insights. Two teams made significant revisions by erasing and redrawing. One team refined their 'Ideas' tab content after discussing 'Design goals', opting to erase their initial concept in favor of a more developed version. Similarly, the other team chose to erase the mention of 'Children' to concentrate on parents instead in the 'Stakeholder' tab.

Q5. Is the content under the tab related to that specific tab? The purpose of this inquiry was twofold. First, answers to this question can unveil if the users aligned their work with the tabs as I introduced. Second, it is a good way to observe if they leveraged the tabs to organize the content that was pertinent to each tab.

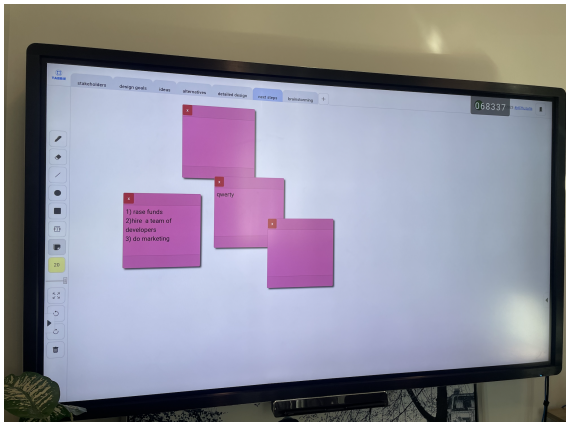
Analyzing the content of each tab, out of the 26 tabs used by all teams, content in 18 tabs was correctly aligned with the intended tab themes. However, 6 tabs contained a mix of related and unrelated content, and content in 2 tabs was entirely unrelated.

The tabs with mixed or unrelated content primarily revolved around features. When teams discussed under the 'Stakeholders' and 'Design goals' tab, there was a tendency that they also touched on features of the software. For instance, one team wrote 'customer support' under the 'Stakeholders' tab and another team listed out features under the 'Design goals' tab with the process of ride sharing from user side and payment.

Two teams used the 'Next steps' tab in a way that was not intended. Specifically for



(a) Team 3's next steps



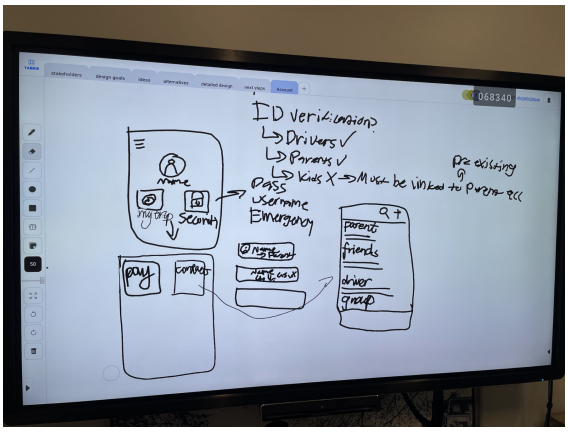
(b) Team 7's next steps

Figure 5.10: Next steps

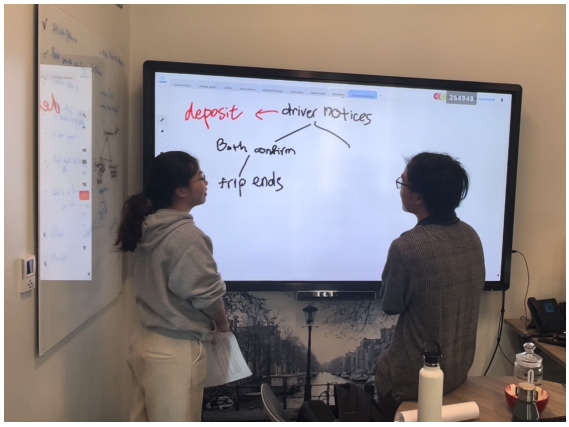
each team, the 'Next steps' tab contained completely unrelated content. Instead of listing action tasks required to further the design process, participants outlined broader project milestones, including testing, marketing, and financing(see Figure 5.10). This suggests a broader interpretation of the 'Next steps' tab, encompassing overall project development.

Q6. Did they add new tabs? Two teams among seven teams used the feature of adding tabs. The first team added an 'Account' tab. Figure 5.11 (a) displays what this team drew on the 'Account' tab they created. They designed the user interface of account management. During the post interview, they explained that they wanted to concentrate exclusively on the accounts user interface separately from the rest of the design, hence they created a new tab. This allowed them to also preserve and reference their prior design work on the other tabs.

The second team added two tabs: 'Workflow' and 'Trip end workflow'. The 'Workflow' tab emerged as an extension of their discussion under the 'Detailed design' tab, where they initially outlined the user interface. Under the 'Workflow' tab, the conversation started with the process of initiating a ride from the parents side. Subsequently, they talked about the driver side when the ride ends and introduced 'Trip end workflow tab'. As Figure 5.11 (b) shows, the focus of this tab was to address the procedures following a trip's conclusion. In



(a) Account



(b) Trip end workflow

Figure 5.11: Added tabs

their post-interview feedback, the team mentioned that their decision to add tabs was driven by the need for more space.

5.2 Perceptions

In this section, I will analyze the participants' perceptions of Tabbie by presenting questions and their responses in the post interview.

Q1. Did using Tabbie change from how you normally might have proceeded working at a whiteboard? The majority, 12 participants, reported that using Tabbie altered their approach compared to traditional whiteboards. Those who experienced a change in their approach mentioned that the design contents had become more organized and structured, thereby streamlining workflow. They expressed favorable feedback on having separate tabs of separate aspects of the design process.

"*[P1] I think...like the fact that I separate the stakeholders, ideas... and check it back. More organized*"

Conversely, the reasons among the three who did not perceive a change in their behavior

varied. One participant felt that Tabbie's functionality is similar with physical whiteboards that allow for multiple screens by pulling them down. Another cited the time constraint as a limiting factor in exploring tabs, adding that with more time, the use of additional tabs might have influenced her perception. The third participant, who seldom used whiteboards, felt that it is similar with other whiteboards which support an extra space.

"*[P14] I feel like it's pretty similar to any whiteboard like Jamboard. I don't usually use whiteboard that much. I don't have an iPad. Whenever I use whiteboards, it's just like I just do it with my mouse*"

Regardless of their differing views on how the presence of tabs influenced their design approach, all participants unanimously agreed that their overall design work with Tabbie was successful.

Q2. Did you find some tabs more or less useful than others? 'Design goals' and 'Ideas' were mentioned as useful tabs among participants, aligning with the finding of these tabs being most frequent and prolonged in use. 'Stakeholders' and 'Detailed design' were also recognized as more useful than other tabs, whereas the 'Alternatives' and 'Next steps' were not mentioned as beneficial by any participants, despite them being used by two teams.

In discussions about less useful tabs, 'Alternatives' and 'Next steps' were top ranked, being consistent with previous observations. Notably, 'Ideas' tab was considered as less useful by one participant despite its heavy use. The participant said that 'Ideas' tab was not applicable to this problem, because it was already implied in the prompt. The 'Detailed design' tab was cited twice as being less useful.

Q3. Did you feel like any tabs were missing? Three teams expressed satisfaction with the existing set of tabs, mentioning no tab is missing in Tabbie. Four participants did not identify any specific tab as missing. In contrast, three participants suggested the addition of specific tabs. One individual proposed a 'Prototype' tab, while another saw the necessity

for a 'Diagrams' tab. It highlights a demand for more dedicated space to accommodate detailed design components. Additionally, there was a suggestion for a 'Brainstorm' tab to be placed at the beginning even though that was the intent behind the 'Ideas' tab. One participant expressed a strong dissatisfaction with the existence of tabs saying that all the tabs are equally useless.

Q4. I noticed that you did not add more tabs. Why not? For the five teams that did not add additional tabs, I inquired about their reasons. Seven participants believed the existing tabs sufficiently supported the completion of their design task, seeing no necessity for an additional tab. Time constraints were cited three times as a limiting factor, suggesting they might have added new tabs with more time passing. One participant expressed a pressure to use the existing, predefined tabs fully. Another participant had no opinion on tab addition.

Q5. I noticed that you did not use some tabs. Why not? None of the teams used the full array of tabs available in Tabbie. The majority, consisting of nine participants, attributed their selective tab usage to time constraints. Three participants noted that their discussions naturally converged on other tabs. One individual whose team did not use the 'Detailed design' tab highlighted a preference for using a whiteboard as a brainstorming tool. Another participant felt that the design prompt did not necessitate the use of the 'Alternatives' tab. Additionally, there was feedback that a single 'Detailed design' tab does not provide sufficient space for various components for software design, and this team added an additional tab specifically for user interface.

Q6. What do you feel was the intention of the pre-made tabs? The responses were diverse, but there was consensus around some themes. Specifically, organization and guidance were prominent themes and two participants, in particular, emphasized the intention of tabs as a guiding tool for those new or returning to software design. Additionally, two responses pointed out that the tabs offer a solid starting point, and ease of use and ability to focus on tasks were each noted once.

Organization and Guidance: Many participants highlighted the structured organization, appreciating how it effectively categorized and grouped their design efforts. This organization provided them not only with a workflow but also an overview of the design aspects to consider.

"*[P9] I would say to get a set of organized information of detailed, because, every section has its own organized information so that way it makes easy for the testers and programmers to analyze the project*"

Guidance for New and Returning Designers: Closely linked to the concept of organization was the sense of guidance. Two participants pointed out that Tabbie would be helpful especially for those new to software design or returning after a hiatus because it gives clear direction.

"*[P6] I think it can also be like helpful for people who are like new to software design. It's like providing a guideline for you to follow if you like have no idea of what you are supposed to be doing*"

A Solid Starting Point: The tabs were recognized for providing a clear starting point. It clarifies the agenda at the onset of meetings and guides users to initiate the design process with a focused direction. This clarity encourages users to have purposeful discussions.

"*[P4] I think the intention of the pre-made tabs was to give users a sort starting point to guide their thoughts and to specific goal which I assume software design*"

Additional Feedback: One individual mentioned the ease of use and another individual touched on the ability of the tab to maintain focus on specific aspects of the design process. These comments highlighted the user-friendly interface of Tabbie and its impact on enabling designers to concentrate on one task at a time.

Q7. If you had to do another design exercise like this, how would you feel if the whiteboard again came with tabs? Eleven participants expressed a desire to use Tabbie for future design

projects. People in favor of revisiting Tabbie for design work appreciated its strengths in organizations and workflow which were identified as underlying intentions of the pre-made tabs. For instance, the participant who was favorable to Tabbie mentioned that the intention was to make ideas more organized and give users a sense of direction.

However, there were two participants who showed no interest in using Tabbie again. One participant preferred other software tools like Figma and Lucidchart where more advanced features for UI and diagram design are available:

"*[P8] I think I will use a whiteboard and another software. Another software. I prefer software. Famous software. Design software like Lucidchart and Figma*"

Another participant felt that the experience gained from this experiment in software design meeting diminished the necessity for Tabbie, believing he had acquired enough knowledge to proceed without it. Another individual agreed to some extent, still considered himself a novice in the field, and would like to use Tabbie again.

Q8. Would you recommend having tabs to others who might need to do some design work? The majority of participants, except for one participant, would recommend Tabbie to others, particularly for beginners. They valued Tabbie providing a starting point, with clear organization and a segmented approach beneficial in maintaining what otherwise might become cluttered content.

"*[P3] Tabs are really helpful to separate every task. Because it's... we are designing software. It's a huge task, we need to divide them into a lot of subtasks*"

"*[P4] If they don't have these fundamentals of design in their head like in grain, then yes, it will be useful to help people and even people who do to have starting point. If they want that sort of structured design*"

Yet, there was a dissenting voice among the feedback. One participant criticized that the

tab functionality Tabbie offers can be replicated through the use of separate document files, arguing that Tabbie does not offer unique advantages compared to other software solutions.

Q9. In terms of the implementation of the tabs, is there anything you would change? The responses to the question provide insight for improving tab implementation and whiteboard functionality.

Tab Management: Four individuals highlighted the need for renaming the tabs and two individuals suggested the ability to remove tabs, including the one they created. Three participants proposed the option for users to change tab colors to organize them.

Whiteboard Functionality: The ability of duplicating whiteboard contents was mentioned by one person. In addition, the functionality of viewing multiple whiteboards on one screen was mentioned once. In line with this, there was a suggestion for a split-view feature, allowing users to view multiple boards side by side. These comments indicate a need for more flexible viewing options.

Q10. Any overall feedback that we did not touch upon yet? There were some suggestions that provide a clear roadmap for tool improvement, focusing on usability and enhanced functionality.

Text Box and Sticky Note: For text boxes and sticky notes, users desired an advanced formatting option such as bullet points and the ability to change font sizes after the creation. Maintaining focus on these elements after switching tabs can enhance usability.

Drawing Features: Many participants mentioned the need for object manipulation such as moving objects and erasing by stroke which are provided in many other electronic whiteboards. Participants also requested the ability to save color and size settings for each tool so that the setting is neither lost when they switch the tool nor they have to set up the color and size again. Some participants pointed out the latency of the tools which should be

addressed.

Viewing Options: There was a need for improved zoom functionality. This can allow users to zoom in and out for better detail management and overall workspace navigation.

Collaboration: The whiteboard that Tabbie ran on was not able to detect two pens at a time, and one participant requested that there should be the ability for multiple users to draw simultaneously.

Additional Tools: One participant suggested incorporating a clock for an efficient time management during design sessions.

5.3 Discussion

Every team used multiple tabs. The feedback on tabs was generally positive, with participants recognizing their utility in structuring the design process. The importance of the 'Ideas' and 'Design goals' tabs was clear as these two tabs saw more frequent and prolonged use. In contrast, 'Alternatives' and 'Next steps' were perhaps less useful, reflected in their lesser and briefer usage. The option to add tabs was used by only two teams, yet there was an expressed interest in this feature, suggesting that extended time might naturally lead to use this feature more. The iterative use of tabs was observed with participants moving back and forth among them.

A significant number of participants expressed a desire to modify the predefined tabs through renaming, removal, and color changes. Additionally, some suggestions for enhancements such as multi-view and split-view capabilities were proposed. Areas for improvement were identified in addressing latency issues in drawing and erasing, as well as the inclusion of features found in other software, such as saving tool settings, stroke-based erasing, object moving, and

zoom functionalities. Incorporating these features could enhance user experience, suggesting a clear path for the refinement of Tabbie.

Given the findings in the study, Tabbie holds promise in aiding novice designers in navigating the software design activities. The participants noticed the intention of the tabs, which provided a structured approach to exploring different aspects of their design, and enabled users to segment their work into manageable and focused areas. I believe this structure can facilitate easier navigation through the complexities of software design, helping novices to maintain organization and clarity in their projects. After some enhancements as identified in this chapter, Tabbie could be a more valuable resource for novices in software design.

Chapter 6

Conclusion

While whiteboards play an important role in software design meetings, their conventional setup presents limitations. Considerable research has been conducted on the use of whiteboards in software design. However, to date these studies have overlooked the unique needs and challenges faced by novice software developers when they design.

This thesis introduced Tabbie to address this gap. Tabbie offers informal guidance and support to novice developers through a tabbed interface approach. Our preliminary assessment of Tabbie in use by 15 novice software developers revealed its potential to render software design meetings more organized to beginners. Participants of the assessment expressed an overall positive response to the predefined tabs and the functionality to add new tabs.

However, the varied used of tabs and feedback for additional features signal areas for enhancement. By deepening our understanding of novice developers' needs and further aligning Tabbie's features with those requirements, Tabbie can be further refined to better assist novices in their design work.

Bibliography

[1] A. Baker and A. van der Hoek. Understanding maintenance design: how developers work at the whiteboard to design solutions for software maintenance. *Institute for Software Research Technical Report, University of California, Irvine (to appear)*, 2010.

[2] S. Baltes, F. Hollerich, and S. Diehl. Round-trip sketches: Supporting the lifecycle of software development sketches from analog to digital and back. In *2017 IEEE Working Conference on Software Visualization (VISSOFT)*, pages 94–98. IEEE, 2017.

[3] A. Begel and B. Simon. Struggles of new college graduates in their first software development job. In *Proceedings of the 39th SIGCSE technical symposium on Computer science education*, pages 226–230, 2008.

[4] S. Branham, G. Golovchinsky, S. Carter, and J. T. Biehl. Let's go from the whiteboard: supporting transitions in work through whiteboard capture and reuse. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 75–84, 2010.

[5] A. S. Chapai and E. J. Rapos. Skemo: Sketch modeling for real-time model component generation. In *2023 ACM/IEEE 26th International Conference on Model Driven Engineering Languages and Systems (MODELS)*, pages 173–183. IEEE, 2023.

[6] Q. Chen, J. Grundy, and J. Hosking. Sumlow: early design-stage sketching of uml diagrams on an e-whiteboard. *Software: Practice and Experience*, 38(9):961–994, 2008.

[7] Y. Cheng, T. Zhang, and S. Chen. Visionsynaptics: a system convert hand-writing and image symbol into computer symbol. In *Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human*, pages 82–85, 2009.

[8] M. Cherubini, G. Venolia, R. DeLine, and A. J. Ko. Let's go to the whiteboard: How and why software developers use drawings. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, page 557–566, New York, NY, USA, 2007. Association for Computing Machinery.

[9] I. Chinangkulpiwat, S. Chaveesuk, and W. Chaiyasoonthorn. Collaboration platforms for organizational integration uses and benefits: A review. In *Proceedings of the 2023 14th International Conference on E-business, Management and Economics*, pages 166–171, 2023.

[10] M. Craig, P. Conrad, D. Lynch, N. Lee, and L. Anthony. Listening to early career software developers. *Journal of Computing Sciences in Colleges*, 33(4):138–149, 2018.

[11] R. Dachselt, M. Frisch, and E. Decker. Enhancing uml sketch tools with digital pens and paper. In *Proceedings of the 4th ACM symposium on Software visualization*, pages 203–204, 2008.

[12] C. H. Damm, K. M. Hansen, and M. Thomsen. Tool support for cooperative object-oriented design: Gesture based modelling on an electronic whiteboard. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '00, page 518–525, New York, NY, USA, 2000. Association for Computing Machinery.

[13] U. Dekel. Supporting distributed software design meetings: What can we learn from co-located meetings? In *Proceedings of the 2005 Workshop on Human and Social Factors of Software Engineering*, HSSE '05, page 1–7, New York, NY, USA, 2005. Association for Computing Machinery.

[14] R. DeLine, G. Venolia, and K. Rowan. Software development with code maps. *Communications of the ACM*, 53(8):48–54, 2010.

[15] V. Jackson, A. van der Hoek, R. Prikladnicki, and C. Ebert. Collaboration tools for developers. *IEEE Software*, 39(2):7–15, 2022.

[16] M. A. Jalil and S. A. M. Noah. The difficulties of using design patterns among novices: An exploratory study. In *2007 International Conference on Computational Science and its Applications (ICCSA 2007)*, pages 97–103. IEEE, 2007.

[17] M. Kleffmann, M. Hesenius, and V. Gruhn. Connecting ui and business processes in a collaborative sketching environment. In *Proceedings of the 7th ACM SIGCHI Symposium on Engineering Interactive Computing Systems*, EICS '15, page 200–205, New York, NY, USA, 2015. Association for Computing Machinery.

[18] J. A. Landay and B. A. Myers. Sketching interfaces: Toward more human interface design. *Computer*, 34(3):56–64, 2001.

[19] N. Mangano, T. D. LaToza, M. Petre, and A. van der Hoek. How software designers interact with sketches at the whiteboard. *IEEE Transactions on Software Engineering*, 41(2):135–156, 2014.

[20] N. Mangano, T. D. LaToza, M. Petre, and A. van der Hoek. Supporting informal design with interactive whiteboards. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 331–340, 2014.

[21] A. Milliken, W. Wang, V. Cateté, S. Martin, N. Gomes, Y. Dong, R. Harred, A. Isvik, T. Barnes, T. Price, et al. Planit! a new integrated tool to help novices design for open-ended projects. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*, pages 232–238, 2021.

[22] Y. Pechorina, K. Anderson, and P. Denny. Metacodenition: Scaffolding the problem-solving process for novice programmers. In *Proceedings of the 25th Australasian Computing Education Conference*, pages 59–68, 2023.

[23] M. Petre. Insights from expert software design practice. In *Proceedings of the 7th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering*, pages 233–242, 2009.

[24] M. Petre and A. van der Hoek. Beyond coding: Toward software development expertise. *XRDS: Crossroads, The ACM Magazine for Students*, 25(1):22–26, 2018.

[25] V. Popovic and B. Kraal. Expertise in software design: Novice and expert models. In *Proceedings of the Studying Professional Software Designers: An NSF (National Science Foundation)-Sponsored International Workshop*, pages 1–7. University of California, 2010.

[26] R. Porter and P. Calder. Applying patterns to novice programming problems. In *Proceedings of the 2002 conference on Pattern languages of programs-Volume 13*, pages 73–82, 2003.

[27] A. Radermacher, G. Walia, and D. Knudson. Investigating the skill gap between graduating students and industry expectations. In *Companion Proceedings of the 36th international conference on software engineering*, pages 291–300, 2014.

[28] J. Rooksby and N. Ikeya. Collaboration in formative design: Working together at a whiteboard. *IEEE software*, 29(1):56–60, 2011.

[29] A. M. Soria. Knocap: capturing and delivering important design bits in whiteboard design meetings. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: Companion Proceedings*, pages 194–197, 2020.

[30] M. Stefik, G. Foster, D. G. Bobrow, K. Kahn, S. Lanning, and L. Suchman. Beyond the chalkboard: Computer support for collaboration and problem solving in meetings. *Communications of the ACM*, 30(1):32–47, 1987.

[31] S. Sulaiman, N. A. Rashid, R. Abdullah, and S. Sulaiman. Supporting system development by novice software engineers using a tutor-based software visualization (tubvis) approach. In *2008 International Symposium on Information Technology*, volume 4, pages 1–8. IEEE, 2008.

[32] D. R. Wright. Inoculating novice software designers with expert design strategies. In *2012 ASEE Annual Conference & Exposition*, pages 25–784, 2012.

Appendix A

Appendix

A.1 Design Prompt

Suppose you are to design the software by which parents can have their kids transported to places in a way similar to Lyft or Uber, but without the parent or guardian necessarily going along on the ride. That is, you are to design a ride share service by which kids can be transported to school and picked up, go to birthday parties, visit relatives or friends, be brought to and picked up from the library or the park, and so on.

A particular feature that this software should have is for multiple kids—from the same or from different families—to ride together to the same or sometimes close by but different destinations.

You are free to add additional features and functionality that you feel is important for this app.

Your design should focus on the high level architecture of how the system may work together with the user experience (interface); both will serve as the starting point for further, refined

design.

A.2 Interview Questions

1. What degree are you pursuing?
2. What is your major?
3. What year are you in?
4. Any internship or industry experience?
5. Do you have any experience in software design?
   - **a.** If so, how many years?
   - **b.** Can you please describe your experience with software design?
6. How did you feel your design work went overall?
   - **a.** Did it change from how you normally might have proceeded working at a whiteboard?
   - **b.** Did you find some of the tabs more useful than others?
   - **c.** Did you find some of the tabs less useful than others?
   - **d.** Did you feel like any tabs were missing?
7. How did the presence of the tabs impact how you worked?
8. I noticed that you added more tabs. Why?
9. I noticed that you did not add more tabs. Why not?
10. I noticed that you did not use the 'x' tab. Why not?
11. What do you feel was the intention of the pre-made tabs?
12. If you had to do another design exercise like this, how would you feel if the whiteboard again came with tabs?

13. Would you recommend having tabs to others who might need to do some design work?

14. In terms of the implementation of the tabs, is there anything you would change?

15. Any overall feedback that we did not touch upon yet?